12-2015

Joint optimization of allocation and release policy decisions for surgical block time under uncertainty

Mina Loghavi
University of Tennessee - Knoxville, mloghavi@vols.utk.edu

To the Graduate Council:

I am submitting herewith a dissertation written by Mina Loghavi entitled "Joint optimization of allocation and release policy decisions for surgical block time under uncertainty." I have examined the final electronic copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Management Science.

Charles E. Noon, Major Professor

We have read this dissertation and recommend its acceptance:

Melissa Bowers, Bogdan C. Bichescu, Xueping Li

Accepted for the Council:

Carolyn R. Hodges

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

Joint optimization of allocation and release policy decisions for surgical block time under uncertainty

A Dissertation Presented for the
Doctor of Philosophy
Degree
The University of Tennessee, Knoxville

Mina Loghavi

December 2015

Dedication

To my inspiring Mom and Dad, Sousan and Mohammad,

It’s impossible to thank you adequately for everything you've done; from loving me unconditionally to teaching me everything is possible with hard work and discipline. I could not have asked for better parents or role-models.

To my husband, Armin,

who has been a constant source of support and encouragement during the challenges of graduate school and work.

Acknowledgements

I would like to express my deep gratitude to my advisor, Dr. Charles Noon, for his excellent guidance, caring, patience and letting me be with my husband while pursuing my degree. For the past 3 years, he has devoted countless hours to guiding me through this dissertation from far distance. His technical and editorial advices were not only essential to the completion of this dissertation but also have taught me innumerable lessons and insights for my work.

My special thanks go to my committee members, Dr. Melissa Bowers, Dr. Bogdan C. Bichescu, and Dr. Xueping Li for guiding my research for the past several years. Their comments and suggestions have been invaluable and, as a result added, significant value to my dissertation. My graduate study has always been enjoyable being with them and their profound attitude for life and research are the role models to design my life. I would also like to acknowledge Dr. Mee as the second reader of this thesis, and I am gratefully indebted to him for his very valuable comments on the design of experiment chapter.

Special Thanks goes to Dr. Jody Crane, who was willing to share Stafford Hospital information for this research. In addition, a thank you goes to Robin Clark, who introduced me to the ExtendSim simulation software and continually helped me throughout my dissertation. Many thanks to Jane Moser, who always willing to help and giving her best suggestions. Finally, I would like to thank my sister, Laleh, who without doubt believed in me and encourage me to come to US and my brother, Saeed, whose success inspired me every day, I couldn’t be more proud.

Abstract

The research presented in this dissertation contributes to the growing literature on applications of operations research methodology to healthcare problems through the development and analysis of mathematical models and simulation techniques to find practical solutions to fundamental problems facing nearly all hospitals.

In practice, surgical block schedule allocation is usually determined regardless of the stochastic nature of case demand and duration. Once allocated, associated block time release policies, if utilized, are often simple rules that may be far from optimal. Although previous research has examined these decisions individually, our model considers them jointly. A multi-objective model that characterizes financial, temporal, and clinical measures is utilized within a simulation optimization framework. The model is also used to test “conventional wisdom” solutions and to identify improved practical approaches.

Our result from scheduling multi-priority patients at the Stafford hospital highlights the importance of considering the joint optimization of block schedule and block release policy on quality of care and revenue, taking into account current resources and performance. The proposed model suggests a new approach for hospitals and OR managers to investigate the dynamic interaction of these decisions and to evaluate the impact of changes in the surgical schedule on operating room usage and patient waiting time, where patients have different sensitivities to waiting time.

This study also investigated the performance of multiple scheduling policies under multi-priority patients. Experiments were conducted to assess their impacts on the waiting time of patients and hospital profit. Our results confirmed that our proposed threshold-based reserve policy has superior performance over common scheduling policies by preserving a specific amount of OR time for late-arriving, high priority demand.

Table of Contents

List of Tables

List of Figures

Chapter 1 : Introduction

Overview of the Problem and Motivation

The competition among healthcare systems has escalated in recent years. Non-optimal decisions can lead to inefficiency, reduced profits and a loss of market share. According to BBC news on June 3rd 2014, the number of patients who died while waiting for heart surgery at two south Wales hospitals has risen in the past year. Between April 2013 and March 2014, 29 patients died waiting for surgery at Cardiff's University Hospital of Wales and Morriston Hospital in Swansea. According to the Los Angeles Times on April 2014, the chairman of the Senate Veterans Affairs Committee pledged to convene a hearing on allegations that excessive wait times at a Phoenix Veterans Administration facility led to the deaths of 40 veterans. These real cases are just examples of the challenge all health care systems face daily. Each patient type has different sensitivity to the waiting time and an optimal scheduling policy needs to find the right balance of individual waiting cost while maximizing the efficiency. The methods that hospitals use to schedule their patients greatly determine the ultimate throughput. With improved scheduling, hospitals can better utilize fixed assets and better control the costs for variable resources. Surgery departments represent the largest cost centers and the greatest sources of revenue for most hospitals. Operating Room (OR) planning and scheduling is a key tool which can be used to improve the productivity level of ORs and their downstream resources. Basically, there are three OR scheduling strategies commonly employed:

(1) Block scheduling strategy; (2) Open scheduling strategy; and (3) Modified scheduling strategy. Most hospitals schedule their OR suites using case or block surgery schedules in which OR time is assigned to surgical specialists/surgeons. In open scheduling, OR time is shared among all specialists or surgeons based on first-come-first-serve order and finally, the Modified scheduling is a combination of these two strategies. Some percentages of the blocks are devoted to each specialty, but there are still some open blocks that are shared among all.

Block schedules are concerned primarily with elective surgery. Elective surgery scheduling decisions consist of three stages; (1) determining the amount of OR time to allocate to each surgical specialty, individual surgeons or groups (or case mix planning), (2) creating a cyclic timetable, implementing the desired assignment of OR blocks to specialties (or surgery master schedule), and (3) scheduling individual patients into available time (or case scheduling). The first stage decision reflects the long-term strategic goals of hospital, such as achieving desired levels of patient throughput by service line or maximizing revenue (Strum et al., 1999; Dexter et al., 1999; Blake and Carter, 2002; Gupta, 2007; Santibañez et al., 2007; Testi et al., 2007, Aringhieri et al., 2015). The second stage is constructed on a medium-term horizon to build a specific cyclic schedule for specialties and is updated whenever the total amount of OR time changes or when the make-up of some specialties changes (Blake et al., 2002; McManus et al., 2003; Santibanez et al., 2005; Beliën and Demeulemeester, 2007; Testi et al., 2007; Chow et al., 2008; Van Ostrum et al., 2008; Price et al., 2011), and the third stage represents the short-term operational decision of assigning a specific surgery and OR appointment slot to each patient over the planning horizon, which can range from one week to one month (Dexter et al., 2000; Guinet and Chaabane, 2003; Denton et al., 2007; Patrick and Puterman, 2008; Erdelyi and Topaloglu, 2009; Sauré et al., 2012).

As evidenced by the references listed above, the vast majority of papers found in the literature only consider one decision level at a time although all of these decisions are interrelated and are sensitive to uncertainty with respect to case durations, arrival rates, patient and provider preferences, punctuality, cancellations, no-shows, etc. Hence, a fundamental objective of OR scheduling is to transition from the generality of the block schedule (stage 1) to the specificity of a detailed schedule for each day (stage 3) (Herring and Herrmann, 2011). Approaches dealing with more than one planning level simultaneously are indeed rare. Among these, Jebali et al. usees a two-phase approach to deal with both the case scheduling and allocation scheduling problems and proposes an integer programming model aimed at minimizing OR over-time and under-time costs as well as hospitalization costs related to the number of days patients are kept in the hospital waiting for an operation or procedure (Jebali et.al., 2006). Testi et al. presents a hierarchical three-phase approach to determine operating theater schedules. First, integer programming models are developed in order to divide the available OR time among the different surgical specialties. Then, they formulate a master surgery scheduling problem in order to assign a specific operating room and day of the planning horizon to the OR time blocks of each specialty. Finally, a discrete-event simulation model is used to evaluate the decisions concerning patients scheduled dates, OR and time assignments

(Testi et al., 2007). Tanfani and Testi propose a linear programming model to simultaneously address the decisions involved in the three-phases of the OR planning and scheduling problem described above, excluding only the most strategic ones dealing with the number and type of the ORs and their operating hours. The objective of the model consists of minimizing a cost function that combines the patients' waiting time since referral and urgency status. The solution approach is based on a sequential heuristic (Tanfani, and Testi, 2010). Aringhieri et al. adopt the idea proposed in Tanfani and Testi (2010) and extend it to incorporate both patient utility (by reducing waiting time costs) and hospital utility (by reducing production costs measured in terms of the number of weekend stay beds required by the surgery planning) (Aringhieri et al., 2015).

These three scheduling levels mostly consider the forward planning aspects of the scheduling process and sometimes fail to capture the available capacity when allocated OR block times are not fully utilized. Having a portion of OR block time released in advance of the day of surgery allows schedulers to add cases to blocks that otherwise would be underutilized. In this case, a finite resource (OR time) must be allocated to competing surgical demands. These demands for surgery arrive over time and the decision makers must decide at the time of arrival. So, the main questions would be, "when to release the unfilled blocks, how much time from these blocks would be offered to which specialties, and who within a specialty is the best recipient for them". Some hospitals have policies on "suggested block release times" based on experience but they may not be the optimal rules to follow. These questions represent fundamental problems facing nearly all hospitals to balance the costs of deferring waiting cases and blocking higher priority patients.

Our model considers the joint impact of block schedules (all three stages) and block release policies on quality of care and hospital revenue, taking into account current resources and performance. The proposed model suggests a new approach for hospitals and OR managers to investigate the dynamic interaction of these decisions and to evaluate the impact of changes in the surgical schedule on operating room usage and patient waiting time. Both mathematical programming and simulation have been used to answer all of the above scheduling questions and are used to recommend the improved strategic, operational and tactical decisions. Not having a strategic methodology leaves decision makers to make critical changes based on prior experiences, retrospective data, or even politics, without the benefit of a logical and systematic framework.

Outline of this Dissertation

The general content of this dissertation is presented in six chapters. Chapter 1 includes a brief introduction, an overview of the problem and the motivation for this research. Chapter 2 provides a literature review and background on the problem of scheduling surgical patients into operating rooms at all stages, from case mix planning to elective case scheduling and continues with the different strategies that have been introduced to block scheduling strategy to improve block utilization and flexibility. At the end, we highlight the main contribution of this study both in research and practice.

In Chapter 3, we address the model objective and its relation to yield management. We also devote this chapter to a case study of a hypothetical two OR-facility that demonstrates the practical aspects of implementation of the objective in joint optimization of block allocation decisions and block release policies.

In Chapter 4, we develop and evaluate case scheduling policies from simple heuristic policies to more complex policies driven from a Markov Decision Process (MDP) model. We then present our results and insights from the case study for stochastic multi-priority scheduling. Our mathematical and computational results show how multi-priority scheduling can be optimized using reserved-based policies.

Chapter 5 is devoted to the application of simulation/optimization at Stafford Hospital, a small-sized hospital with modified block scheduling and multi-priority patients. It continues with the application of the Design of Experiments method in reducing the dimensionality of the simulation model. We provide a summary of the results and contributions of this dissertation, discuss conclusions and highlight insights for the practitioner in Chapter 6.

Chapter 2 : Literature Review and Contribution of this Dissertation

While the problem of scheduling elective surgical procedures has received extensive treatment in the operations-research literature, most of the previous works have centered on the level of individual decisions (case mix planning, surgery master schedule or case scheduling). However, no decision is made in isolation: every decision is constrained by the effects of previous decisions and creates its own downstream effects on future decisions (Santibañez et al. 2007). Decisions about how to schedule elective surgeries are further complicated by the fact that elective surgeries are made on a different planning horizon than urgent non-elective surgeries. A survey of the literature on OR scheduling shows that there are very few studies that address this complexity in the decision-making process for assigning (OR) capacity.

This dissertation's research is aimed at filling this gap by improving surgical suite efficiency through a multi-level decision model. Improvement of the surgical suite's efficiency not only may lead to increased productivity, in terms of the number of surgeries undertaken, but also may contribute to a reduction in surgery waiting lists. Costs involved in keeping a patient on the waiting list for surgery are high, both at the prevention and the maintenance level, even more so considering the user's quality of life. Another major contribution of this study is that it integrates the costs of patients' waiting time into its model. Although most of the previous research in this area has centered on optimizing the use of OR capacity, few prior studies have explicitly addressed the costs generated when patients are forced to wait for needed or desired surgeries. Our model is able to optimize the yield of OR capacity by finding the best block release and allocation strategy to allocate limited OR capacity to the right patients at the right time with the lowest cost for patient, surgeon and hospital.

We begin with a conceptual model (Figure 1) of the scheduling process to draw insight into the type of decisions that must be made in scheduling surgeries and to portray their interaction. Finally, this research makes a valuable contribution to the literature on OR scheduling by developing and testing the model using real world data.

As noted above, within the scheduling process, all upstream decisions affect downstream decisions (Santibañez et al. 2007). The complex interrelationship among decisions must be understood and carefully managed in order to achieve optimal use of the limited resource of OR capacity. One way that hospitals have begun to attempt to manage this interrelationship is to use block release policies. Effective block release policies can help hospital administrators manage the effects of such sources of variation in the scheduling process as variability in demand (not only arrival rates, but also patient preferences) as well as in case durations, cancellations, and no shows. If resource managers make effective use of scheduling strategies, they can minimize both unused capacity and wait times.

Improvements in release policies, in coordination with sequential decisions, can enhance efficiency and increase profits. Our research contributes to this effort by offering insight into how hospitals can achieve not only optimal profit but also higher levels of patient satisfaction. Decreasing the waiting time for surgery may not require that hospitals increase their overall capacity, but rather that they simply make better use of existing OR blocks. Implementing optimal release rules can essentially generate more usable capacity at no additional cost.

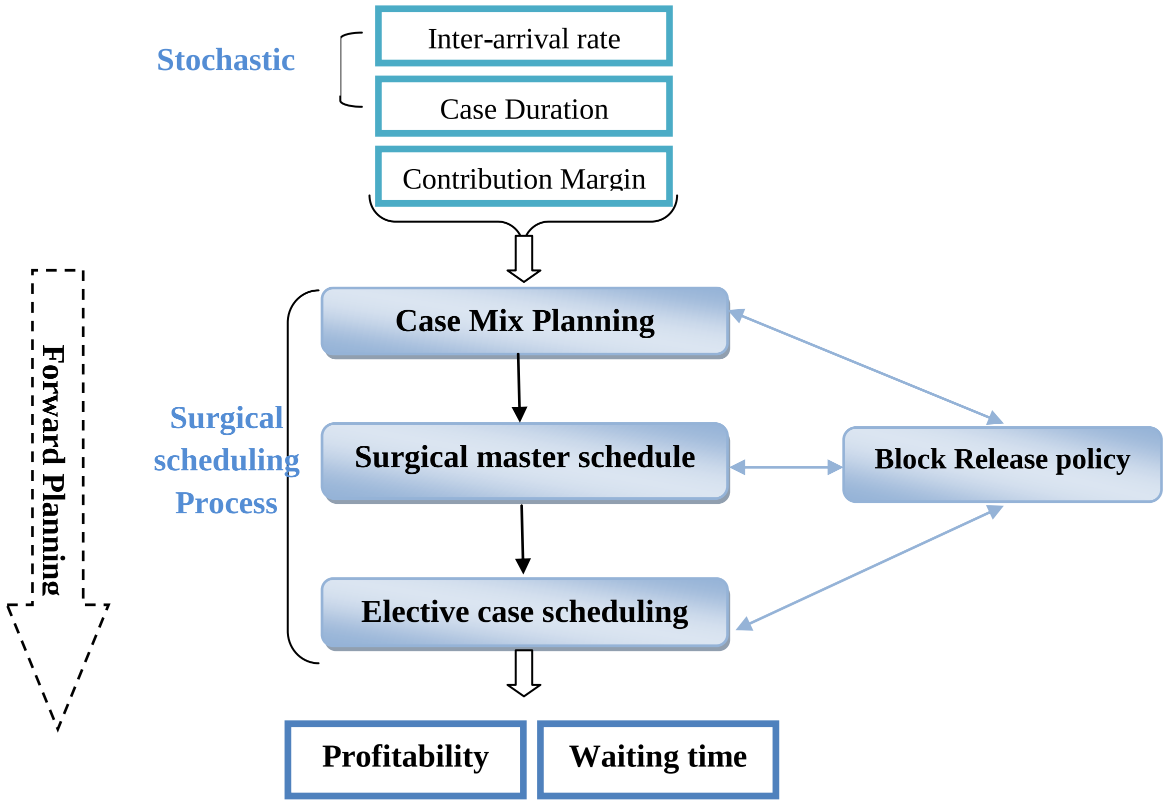


Figure 1: The Conceptual Model

First stage: Determining the amount of OR time allocated to each surgical specialty (*Case Mix Planning*)

Initially, the available total OR time is often determined by a hospital's budget (revised annually). This finite capacity (OR time) must then be allocated to competing surgical demands. Each specialty gets a piece of the pie (Figure 2) based on different criteria, such as total cases per allocated block (i.e. historical utilization and target throughput), hospital costs and financial gains per allocated block, and demand for services (i.e. waiting time), etc. The choice of schedules and resource availability (OR time) directly affects the number of patients treated, cancellations, waiting times and ultimately the overall profit of a hospital. Hospitals can be assumed to seek an optimal patient mix and volume that can yield the maximum overall financial contribution under the given resource capacity. However, there is not an easy answer to the question of how to achieve this optimization since all of the factors listed above play fundamental, interacting roles. On the other hand, a systematic approach to OR time allocation can improve the transparency and fairness in surgeons' time allocation.

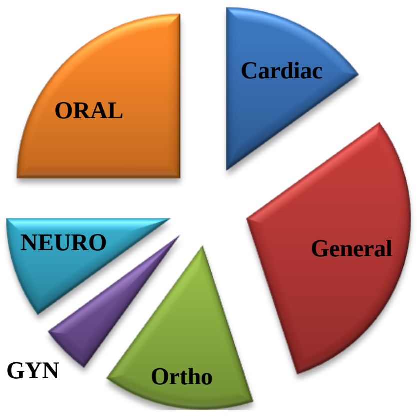


Figure 2: Capacity divided according to specialty/Surgeon

Case Mix Planning Literature Review

There is a growing pressure on health care providers to improve the financial contribution of their resources through efficient capacity allocation and management. Naturally, it is expected that the available resource capacity may match the stochastic patient demand as perfectly as possible and the utilizations of hospital resources (i.e., hospital beds, operating rooms (ORs), nursing staff, etc.) may be coordinated as well as possible. Nevertheless, due to the scarcity of resources and the variability (e.g., the random patient arrivals, the variable length of stays (LOSs), etc.) within the patient flow, the capacity management problem seems to be quite complicated. A number of studies have addressed this complex problem using operational research methods, such as mathematical programming, discrete-event simulation, and so on (Ma and Demeulemeester, 2012).

Strum et al. (1999) and Dexter et al. (1999) have employed statistical analyses of hospital historical data to predict the number of hours that should be allocated to surgical specialties. Hughes et al. (1985) and Robbins et al. (1989) applied linear programming to optimally and efficiently use the hospital's mix of services to maximize net contribution. Blake and Carter (2002) have proposed a methodology that uses two linear goal programming models to determine the trade-offs between service cost, mix volume and clinical necessity. Samanlioglu et al. (2010) have used a similar integer programming approach to determine block schedules that meet surgeons' demand levels. Vissers (2005) proposed two mathematical models, one supports long-term decisions about the resources required to match the future patient mix demand (choose the patient mix that can bring maximum profits), and another supports decision making at the medium-term level for balancing the resource requirements of various types. Dexter et al. (2005) have incorporated two levels of capacity decisions: tactical and operational. Tactical decisions for the selective expansion of operating room resources incorporate financial criteria and operational decisions for any adjustment and are influenced by the uncertainty in subspecialties' future workloads. Numerous reasons have been presented to explain why tactical planning for the expansion of OR capacity should not be based on current or past utilization but instead on total contributions, while meeting certain constraints (Gupta, 2007; Wachtel and Dexter, 2008).

A final group of papers has focused on finding block schedules that minimize the amount of time patients have to wait for surgery (Zhang et al., 2009; Tanfani and Testi, 2010). It has been shown mathematically that, when variation exists, buffer capacity is necessary to be certain of meeting demand. Thus, the optimal capacity is the result of a trade-off between excess capacity and patients' waiting time (Pandit et al., 2010). Santibañez et al. (2007) have developed a mixed integer programming model to explore the tradeoffs between OR availability, bed capacity, surgeons' booking privileges, and waiting time. The main focus of Testi et al. (2007) has been on improving the overall operating efficiency of OR time in terms of overtime and throughput as well as waiting list reduction. Vansteenkiste et al. (2012) introduced the concept of the individual patient deviation from the optimal due time (DT) (an acceptable time after the need for surgery is established) as a potential driver for OR (re-) allocation among surgical disciplines. They believe use of a DT-based model provides a transparent, acceptable system for regular reallocation of OR times between and within specialties.

Second stage: Creating a block schedule with desired allocations (*surgery master schedule*)

Once OR time has been allocated to each surgical group, the second stage of planning involves the development of a master surgery schedule (MSS). The time given to the surgeon/surgical group is named as allocated block time and it should be converted into a desired weekly scheduled time table (Figure 3). The master surgery schedule is a cyclic timetable that defines the number and type of operating rooms available, the hours that rooms will be open, and the surgical groups or surgeons who are to be given priority for the operating room time (Blake et al., 2002). Developing or adjusting a MSS is a complex problem that involves creating and allocating blocks of OR time to each specialty in such a way that it best satisfies some given objectives (such as, balancing patient queue lengths among different specialties, maximizing OR utilization and reducing overtime, maximizing profit, etc (Herring, 2011)) under various sets of realistic constraints (such as, recovery and downstream bed availability, limitations on patient waiting times, follow-ups, surgeons' preferences and different levels of stochasticity (with respect to case)) which have remained fairly consistent in all of the previous research. Also, the master surgery schedule is often preferred to be as simple and repetitive as possible, which entails as few changes as possible from week to week (Blake et al. 2002).

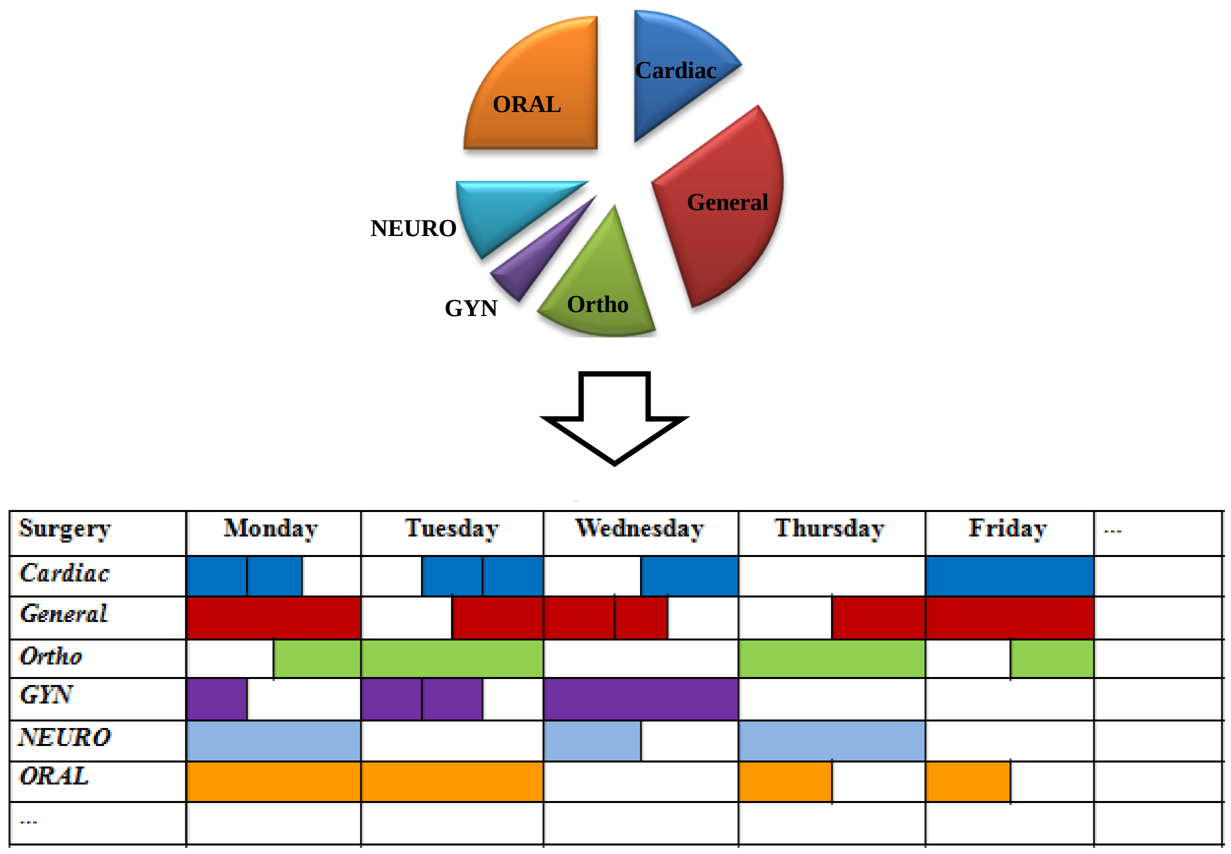


Figure 3: Desired weekly schedule timetable

Surgery Master Schedule Literature Review

Block scheduling models have traditionally focused on implementing desired allocation levels without fully considering the downstream effect. They are often solved by mixed integer linear programming or goal programming. Blake et al. (2002) propose an integer programming model that minimizes the weighted average undersupply of operating room hours that are allocated to each surgical group (a number of operating room hours as close as possible to its target hours). Current models focus more on leveling hospital bed occupancy and minimizing overcapacity in a stochastic approach. In 2003, Ogulata and Erol presented a set of hierarchical multiple criteria mathematical programming models to generate weekly operating room schedules. The objectives considered in this study are maximum utilization of operating room capacity, balanced distribution of operations among surgeon groups and minimization of patient waiting times. McManus et al. (2003) argued that much of the variability in hospital bed occupancy levels is caused by imbalances in the surgical schedule. Researchers have addressed these issues by incorporating patients' lengths of stay into mathematical programming models and heuristic procedures (Beliën and Demeulemeester, 2007; Testi et al., 2007; Chow et al., 2008; Van Ostrum et al., 2008; Price et al., 2011).

The research that is most relevant to our work is presented by Santibañez et al. (2007) who used a system-wide optimization model for block scheduling that enables managers to explore trade-offs between operating room availability, booking privileges by surgeons, bed capacity, and waiting lists for patients. Mannino et al. (2012) introduced a new mixed integer linear model to find a suitable allocation of operating resources to surgical groups in a trade-off of two major variants of balancing patient queue lengths among different specialties, while minimizing overtime. These studies tried to ensure that the patient gets his/her surgery in a reasonable time while surgeons' preference and hospital profit are satisfied.

Third stage: Scheduling individual patients into available time (elective case scheduling)

The third phase on individual patient scheduling is centered on daily decisions about the patient selection, room assignment, and the sequence of cases in each allocated block[1]. In general, patient scheduling studies can be summarized in three decision groups: choosing the right surgical cases to schedule, assigning cases to the right OR on the right day, and optimally sequencing cases within each OR. Usually, either the first two or last two of these decision groups are modeled, although some research focuses more narrowly on just one of these decisions (Herring, 2011). The first group, choosing the right surgical cases from a waiting list, is only applied in situations where patients are kept on a waiting list until an appropriate day is found for them. The next two decision groups are common in all online and waiting list scheduling. First, each surgical case is scheduled for a specific operating room and day (sometimes referred to as advance scheduling). Then, as it gets close to the day of surgery, either

[1] - This study excludes the sequencing of cases (or allocation scheduling) on the day of surgery since we are only interested in indirect waiting time and not in waiting time on the day of surgery

each surgery is scheduled for specific periods in the day or the surgeries scheduled for the same day are simply ordered (allocation scheduling) (Marques et al. 2012).

Elective Case Scheduling Literature Review

This stage of surgery scheduling has more robust literature than earlier stages, because the earlier stages are only applied in the block scheduling strategy. Regardless of whether or not they use block scheduling, they must solve the problem of scheduling individual patients into specific OR times. The main goal in this body of research typically is either to minimize patient delays (waiting time) or maximize OR utilization (Guinet and Chaabane, 2003; Denton et al., 2007). This type of decision is very sensitive to efficiency and variability in the operating room. Ozkarahan (2000) proposed a goal-programming model that can produce schedules that best serve the needs of the hospital, i.e., by minimizing idle time and overtime, and increasing satisfaction of surgeons, patients, and staff. The approach involves sorting the requests for a particular day on the basis of block restrictions, room utilization, surgeon preferences, and intensive care capabilities. Guinet and Chaabane (2003) modeled case scheduling as a general assignment problem aimed at reducing patient stay duration and overtime costs. Hans et al. (2005) addressed the problem of assigning elective surgeries to operating rooms in such a way that not only the utilization of the OR rooms is optimized but also the total overtime is minimized. Both Denton et al. (2007) and Cardoen et al. (2009) investigated the optimal sequencing of cases within an OR using stochastic linear programming and a branch-and-price approach, respectively. In Denton et al. (2010), cases are assigned to operating rooms using a stochastic programming model to incorporate uncertain case durations.

Sier et al. (1997) used simulated annealing to find improved solutions to surgical case scheduling. Simulation has been used in many studies to compare alternative scheduling policies to maximize the efficiency of use of operating room (OR) time, e.g. El-Darzi et al. (1998), Dexter et al. (2000), Dexter and Traub (2002), and Sciomachen et al. (2005). Most of the researches that focus on assigning patients to ORs and sequencing the cases within ORs are focused on the single day problem while the following studies develop their model over a longer horizon. Jebali et al. (2006) solve a series of integer programs for assigning surgery patients to operating rooms over a planning horizon while minimizing the costs of patient waiting times and over-/under- utilized operating rooms. Hans et al. (2008) use a heuristic model to create robust schedules using planned slack, and their work is the exception in that it schedules cases over the course of a week rather than a single day.

Strategies to improve the Utilization and Flexibility

As noted in the literature review, the main concern of all scheduling studies is to find the optimal combination of block size, allocation and case schedule which maximizes capacity usage. However, all of these decisions require forward-planning as much as a year before cases actually fill the blocks. Dealing with demand uncertainty is also an issue that must be addressed in surgical scheduling. To be able to adjust to variation in demands such as case duration, arrival rate and patient and surgeon preference, the following strategies have been applied to block scheduling strategies to improve block utilization and flexibility:

- Modified block scheduling policy
- Block release policy

In many cases, hospitals use a modified block scheduling policy, a mixture of open and block scheduling strategies. Similar to the block-scheduling policy, an MSS is constructed; however, similar to the open-scheduling strategy, certain slots in the MSS are left open for flexibility. Similar to modified block scheduling policy, block release policy consists of open and block scheduling in which surgical groups (or subspecialties) may share blocks, depending on the demand that arises for their scheduled block time. This sharing is achieved by setting a deadline (a particular number of days prior to the day of surgery), at which time the unutilized block time of a surgical group becomes available for use by other groups.

The main difference between a modified scheduling policy and a block release policy is that in modified scheduling, blocks are open from the beginning (no deadline) so that cases can be assigned based on first-come-first-serve basis, while in a block release policy the block will be shared after a deadline. These block times can be used to accommodate overflow and more urgent cases (Gupta, 2007). Thus, released blocks and open blocks offer more flexibility to surgeons to assign their cases based on a first-come-first-served rule. The main focus of this research is on block release policies which are discussed in more detail.

Modified scheduling policy

"Which is better for OR scheduling—block, modified block, or first-come-first-served (open)? It's a common question, but there is no simple answer and many issues must be weighed."

Modified block seems to be the most-used method based on a benchmarking study in 1996 by the University HealthSystems Consortium (OR Manager, Patterson, 1996). Each scheduling strategy has its own pros and cons shown in Table 1.

Table 1: Pros and Cons of different Scheduling strategies

	Advantage	Disadvantage
Open scheduling strategy	• Satisfies the expectations of surgeons and patient's about the day of surgery (Dexter, et al., 2003) • Very efficient planning method if appointments are being made in advance and required resources can be calculated precisely (Fei et al., 2009; Blake, et al., 2002)	• Frustrates surgeons, as they may not be able to schedule their cases back-to-back well in advance (Dawn Mclane-kinzie, 2005)
Block scheduling strategy	• More reliable OR time for surgeons	• Revenue loss or underutilization due to surgeries ending sooner than expected, or cancellations • Assigning and reallocating block time can raise difficult political issues (OR manager)
Modified scheduling strategy	• Offers maximum flexibility and since it combines the two main strategies, this strategy is more flexible to deal with different kinds of patients: elective and urgent	• There may be too many blocks unoccupied, resulting from reserved blocks that were released late

Due to the pros and cons outlined earlier, each strategy works best for a specific set of conditions. Also, as Hamilton and Breslawski (1994) argued, the factors considered by operating room administrators to be critical to operating room scheduling are dependent on the nature of the scheduling policy. The results of their large-scale survey indicated that in block strategy the number of operating rooms, the equipment limitations, the block times assigned and the hospital scheduling policy are considered to be important criteria. While, in open scheduling strategy the number of operating rooms, the estimated room set up duration, the estimated case duration and the equipment restrictions are considered to be essential in optimal scheduling.

Block scheduling is more predictable for surgeons. Specific surgeons or groups of surgeons are assigned one or more blocks of time each week in which to schedule their surgeries and no one else is allowed to assign a case in their block. In reality, pure block scheduling is rarely used because it is too rigid and it may result in highly underutilized OR blocks.

Open scheduling strategy is great for specialties that anticipate their schedules well in advance (specialties with less urgent cases), in this case; there is no time gap between consecutive cases. According to Dexter et al. (2003) this method of planning consists of surgeons and patients who together decide at which date the treatment should take place and the other staff will be adjusted to achieve maximum efficiency. Due to this certainty, every minute of OR time can be optimally used. Unlike the block scheduling strategy (Blake, et al., 2002; Fei et al., 2009), every minute of the operating room can be reserved separately, so there is a better chance of high utilization.

A modified scheduling strategy contains the benefit of both block and open scheduling strategies. More hospitals employ modified block scheduling, since it easily deals with diverse kinds of patients; urgent and elective. In addition, the modified scheduling strategy can gain additional benefit by combining it with block release policy to release unreserved block time at an agreed-upon point before surgery to be shared with other surgeons, the utilization can be high. Although in this strategy, block release policies should be well managed to keep the utilization on target.

As shown in Figure 4, there are multiple ways of creating a modified schedule. Any combination of open and block scheduling policy might be optimal for a given set of conditions. The optimal combination highly depends on hospital patients' combination of elective, urgent or semi-urgent and blocks release policies. The release policy maintains the highest fairness among different types of surgeons.

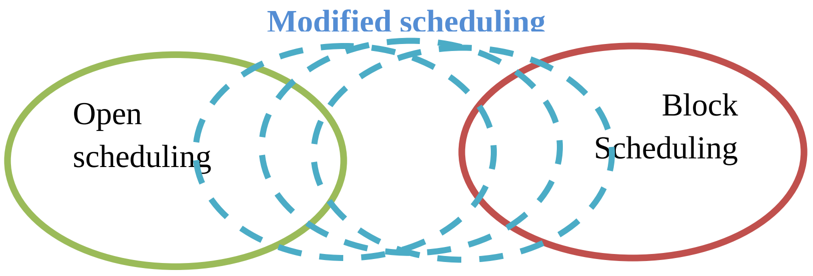


Figure 4 : Modified scheduling strategy

Block release policy

A choice of release of block time maximizes access to the elective schedule for block holders while maintaining sufficient lead time for other surgeons to take advantage of underutilized operating capacity. The intent is to increase access to the OR schedule for all users. The block release dates provide control capability for scheduling managers and allow decision makers to assign upcoming cases to the unfilled blocks based on their urgency. Hence, release times must be managed well to maximize OR utilization.

Two parameters are involved in the block release policy:

(1) The maximum time that a patient can wait for accessing his surgeon’s primary block before being considered for scheduling in an off block, and

(2) The minimum number of days before the day of surgery that the block can be released.

Parameter 1 depends on the expectation of the patients about the longest time to wait for surgery and the urgency of the case, but parameter 2 depends on the arrival rate of patients and how quickly blocks are filled. The maximum time that a patient can wait can be estimated with certainty for each type of surgery based on survey or statistical methods. Table 2 shows maximum time a patient can wait for accessing his surgeon’s primary block at Stafford. The minimum number of days before the day of surgery that the block can be released will be analyzed by Scenario Manager.

Table 2 : Maximum time that a patient can wait for accessing his surgeon's primary block

Specialty	Podiatry	Plastic	Otho	GYN	GV	ENT
Maximum wait time	43	33	18	29	16	49

Block release policy literature review

Most hospitals use a predefined block release time that is reevaluated every year and usually ranges between 3 to 7 days before the day of surgery. This fact motivates the very natural questions of how block release dates can be set optimally and who is the best recipient for the released block. Dexter et al. (2003) addressed the question of which services should release their blocks to minimize under/over utilization. According to Dexter et al., there are three possible rules for releasing OR time: (a) the most expected underutilized OR on the day of surgery, (b) the largest difference between allocated and scheduled OR time at the moment the new case arrives, or (c) the second largest difference between allocated and scheduled OR time. Dexter et al. conclude that the first option (a) is the best strategy. This finding aligns with Herring and Hermann (2011) who argue that a blocking penalty (the dissatisfaction cost that happens if a non-primary case is scheduled and another primary case arrives, then the OR manger has blocked the primary surgeon's access to its allocated time) incurred on a given day is a random variable that depends on the arrival rate of primary cases. Thus, the release policy is highly dependent on the arrival rate of cases.

Dexter and Macario (2004) have extended this study by assessing the effect of release time on efficiency. They claim that the timing of the block release has little impact on OR efficiency. However, this paper, which proposes adding a single case to existing schedules at different points in time, fails to consider the potential effect of decisions on the evolution of the schedule after cases have been added. In addition, they assume it is possible to hold a case on the waiting list until a block is released. Other papers that consider scheduling add-on cases have limited their analysis to the day before and the day of surgery, thus neglecting the role of the block release date (Gerchak et al., 1996; Dexter et al., 1999).

As stated by Herring and Herrmann (2011), one of the shortcomings of previous studies in the field of release blocks and block scheduling, in general, is that these models schedule patients all at once rather than sequentially over time. Although they cover all of the shortcomings of previous research in this field, they still assume the use of waiting lists for cases.

Challenges in Elective Case Scheduling

Although many researchers have conducted studies in the field of optimizing case scheduling, most of these studies have assumed that decision makers have access to full information about the pool of requests that are accumulated into a waiting list until the final decision is made about assigning cases into blocks. This type of research has mostly been conducted by using dynamic programming. However, in reality, surgical demands arrive over time and decision makers must assign a surgery date based on the current state of a system as soon as the patient's request arrives. Uncertainty in future state plays an important factor in the scheduling process. Uncertainty in case durations, patients' arrival rates, patient and provider preferences and probability of cancellations makes it difficult to plan properly. The request for elective surgery arrives over the span of multiple days before the day of surgery. These case requests arrive with different arrival rates and fill the blocks according to their level of urgency and the time that a procedure needs. Some patients request the earliest possible dates, while others are most interested in choosing the most convenient time well into the future. Some patients are sensitive to the surgery postponement and some are not. These natural differences among specialties explain why one specialty's blocks may be almost full fourteen days before the day of surgery (such as GYN in Figure 5), whereas others may have filled only 50% of their blocks less than two weeks before the day of surgery (such as ORAL in Figure 5).

It would be a simpler problem if there were no sharing of capacity allowed among specialties and no patients seeking the earliest possible dates. In reality, hospitals release excess surgical time to surgeons who have urgent patients, based on a first-come-first-served basis. In this situation, the decision of case scheduling becomes even more complex.

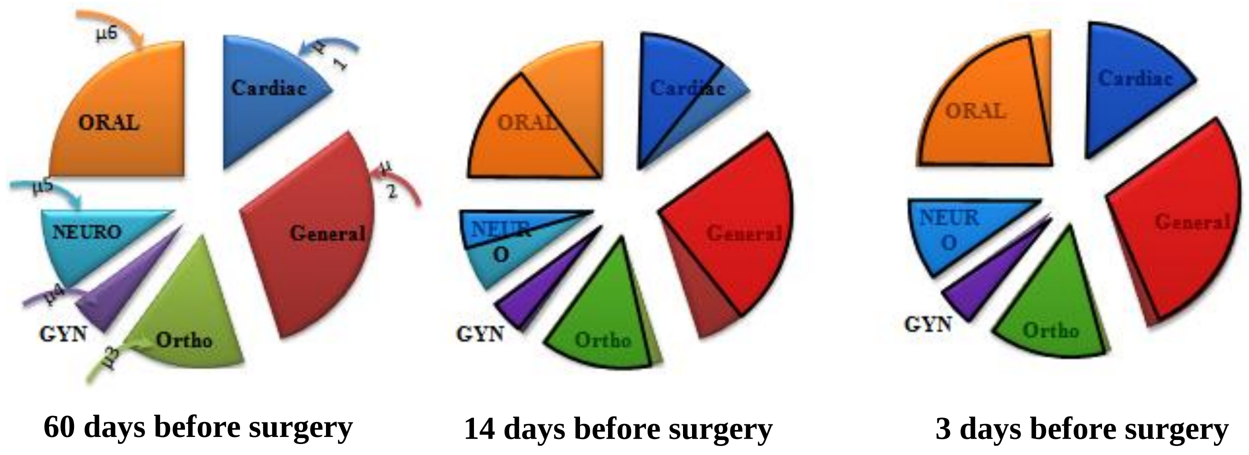


Figure 5: Block allocation and progressive fill up capacity (μ is the arrival rate)

Assume that, in an environment where blocks may be released some days before surgery, a new Cardio case (urgent) arrives (Figure 6). For this patient, the scheduling decision must be made based on the current state of the system. Assigning the case into the first available primary block (OR block associated with the case specialty) in four weeks or into the non-primary block in five days, given a block release policy set as five days prior to the day of surgery.

How to handle such complexities is not a straightforward question since any decision will affect the future state of the system. The potential effects of all decisions must be captured in the block release policy and the strategies must be assigned according to the primary objectives of minimizing patients' waiting times and maximizing overall profit. These challenges in surgical scheduling are a primary motivation for this study.

Although, in some cases, long waiting times may have little medical impact. In other cases, excessive wait times can potentially impact health outcomes and result in lost patients. In this study, we present a simulation model for scheduling surgeries within the Stafford Hospital, a small size hospital with two types of patients: semi-urgent who may require immediate treatment and non-urgent patients where it may be medically acceptable to wait up to several weeks. There is no cost associated with a delay in scheduling non-urgent patients (zero waiting cost). In contrast, a hospital will be penalized for postponing the scheduling of semi-urgent patients one more day. Based on the result of a logistic regression analysis on Stafford data, there is a statistically significant difference between semi-urgent patients across different specialties in terms of a waiting cost coefficient.

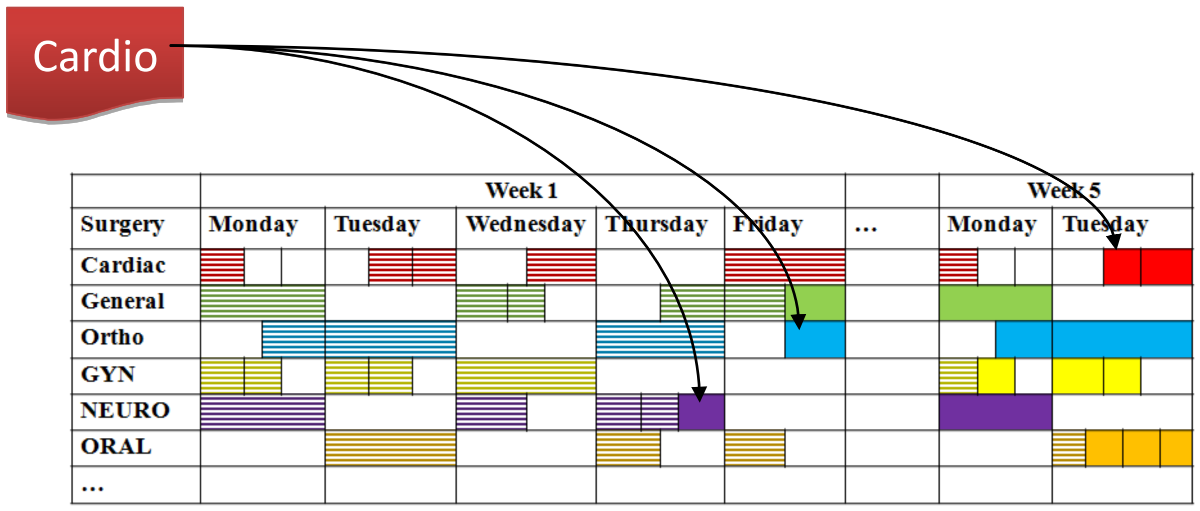


Figure 6: Case scheduling decision challenges

At Stafford, all semi-urgent patients are treated the same based on a FCFS strategy, even though their urgency levels will differ. But, what will be the best set of scheduling policies for optimal yield across multi-priority patients? If less-urgent patients are booked further into the future, this raises the question of how much resource capacity to reserve for later-arriving but higher-priority demand?

Access rules help clinics determine how much capacity to reserve for each type (or length) of appointment and for future callers with more urgent needs. These rules also determine planned appointment lengths for each diagnosis of the referring physicians. Thus, a request may not be scheduled on the first available date

The decision of when a patient should be scheduled is made based on the cost for surgery postponement. A numeric solution is formulated to address this problem and multiple strategies are conducted to understand the properties of an optimal schedule policy. We are looking for sets of superior strategies to better manage health-care resources in order to reduce wait times to acceptable levels without undue additional costs.

Related research and contribution of this study

The research presented in this dissertation contributes to the growing literature on applications of operations research methods to problems in healthcare through,

1. The development and analysis of elective surgery scheduling decisions considering the joint impact of case mix planning, block allocation, case allocation and block release policies on patient wait time and hospital profitability.

The limited existing literature on the joint optimization of block allocation decisions and release policies suggests it is advantageous to consider all level of decisions in this study. A case study of the scheduling system at a hypothetical two OR-room facility is introduced to illustrate the interaction among three surgical scheduling decisions and release block policies. The model illustrated in this research is closely related to the work of Herring and Hermann (2011) which addresses the problem of single-day surgery scheduling incorporating block schedules, block release policies and a surgical waiting list. They employed a stochastic dynamic programming (SDP) model to identify the optimal scheduling policy by continually minimizing utilization cost and customer (patient and surgeon) satisfaction cost. They introduce a threshold policy as the amount of space preserved each day for future primary cases (those that have allocated OR block time on that day) such that a balance is maintained between the differential cost of secondary cases and the blocking cost of higher-priority primary cases. Secondary cases are defined as specialties that do not have allocated OR time on that day, but still wish to perform a surgery. This threshold policy leads to a conventional block release threshold in which unused OR time is gradually released over the course of several days leading up to the day of surgery. What differentiates this proposed study from Herring and Hermann is that it does not assume the existence of any waiting list for patients (secondary cases). Instead, decisions are to be made at the time of the request for surgery. In addition, our study considers all the tactical and operational decisions (not only daily decisions), into one model, where the behavior of patients affects the profitability of the hospital.

Several papers study the joint impact of hierarchal block scheduling decisions. The most relevant study is that of Testi et al. (2007) which reports on a hierarchical three-phase approach to determine operating room schedules. In the first phase, which they refer to as session planning, the number of sessions to be scheduled weekly for each discipline is determined. Since they distribute the available operating room time over the set of disciplines, this problem can be regarded as a case mix planning problem. Phase 2 formulates a master surgery scheduling problem in which they assign an operating room and a day in the planning cycle to the sessions of each discipline. Both phases are solved by integer programming and are modeled at the discipline level. Phase 3, on the contrary, is formulated in terms of individual patients. A discrete-event simulation model is presented to evaluate decisions concerning date, room and time assignments (Cardoen et al., 2010). Although their model suggests an integrated way of facing surgical activity planning in order to improve overall operating theatre efficiency (in terms of overtime, throughput as well as waiting list reduction), their model ignores the interaction of these decisions and how the lower level decisions can affect the optimality of the previous decisions. Also, they assumed a pure block scheduling strategy which implies no block release policy is involved.

The second paper that is related to our work is Tanfani and Testi (2010), where the authors proposed a linear programming model to simultaneously address the decisions involved in the three-phases of the OR planning and scheduling problem described above, excluding only the most strategic ones dealing with the number and type of the ORs and their operating hours. A sequential heuristic algorithm is applied to solve an NP-hard combinatorial optimization problem intended to minimize a cost function based upon a priority score, as a function of waiting time and the urgency status of each patient. Aringhieri et al. (2015) adopt the idea proposed in Tanfani and Testi and extend it to incorporate both patient utility (by reducing waiting time costs) and hospital utility (by reducing production costs measured in terms of the number of weekend stay beds required by the surgery planning). The main contribution of these papers is to characterize the joint optimization of all three stages of scheduling; incorporating both patient and hospital societal benefits although their model focuses on waiting list management and does not capture the immediate scheduling challenges.

2. Development of a multi-objective model that characterizes financial, temporal and clinical measures to balance between competing classes of demand for surgery.

Rather than focus on traditional block scheduling and individual patient scheduling objectives (such as leveling hospital occupancy and maximizing OR utilization), our analysis will focus on how OR managers can make equitable scheduling decisions in the face of competing demands from various surgeons and surgical specialties. In order to consider all conflicting objectives, we have developed a multi-objective function that considers hospital profit, surgeon availability and multi-priority patients' sensitivity to waiting time. The details are provided in the next chapter.

3. New way of looking at waiting cost where the cost of excess waiting is defined as a function of leaving/health deterioration.

If access to elective surgical procedures is managed by scheduling patients from a surgery waiting list, the main question will be to decide how many of the patients in the waiting list can/should be assigned for the next available block time. This optimal strategy can be chosen based on the trade-offs between the cost for overtime work and the cost for surgery postponement. Stenevi et al. (2000) has focused on the productivity loss costs incurred by waiting such as income loss, community service such as home help, medical treatment at home and hospital stays. Bishai and Lang (2000) focused on utility loss and the willingness of patients to pay (bid) to reduce their waiting time.

While the literature on waiting list management is rich, the issue of immediate scheduling, where no waiting list exists, has received less attention from operations researchers. Dexter et al. (1999) showed, based on simulation results, that OR utilization depends greatly on the average length of time patients have to wait for surgery. The longer patients can wait, the greater is the percentage of OR block time that can be used, since more surgical dates can be evaluated for a good match between case duration and the remaining OR time in the block. Although, after conducting a survey to determine patients' perceptions of acceptable waiting times for elective surgery, they concluded that the optimal strategy would be to schedule patients in "overflow" time outside of block if there is no available time before the acceptable waiting time.

Several previous studies have used statistical tools to evaluate the relationship between surgical wait times and adverse events. Sobolev and Kuramoto (2008) introduced several statistical methods using descriptive and comparative statistics and regression models to demonstrate the correlation between waiting time and adverse events. Regression models have been used to quantify the effects of explanatory variables on wait list outcomes. Using logistic regression, Sobolev et al. (2008) showed there was a linear trend, approximately a 5% increase in the odds of in-hospital death for every additional month of delay before surgery. Applying logistic regression, Zamakhshary et al. (2008) found that a wait time for surgery of more than 14 days was associated with a doubling of the risk of hernia among infants and young children. Ahn et al. (2011) used logistic regression modeling to find the target access time by which the risk of additional surgical procedures and other adverse events increases.

In this study, the waiting cost function in derived using logistic regression based on multi-priority patient behavior in response to waiting time. This is consistent with existing research on capacity allocation and revenue management.

4. Introducing a special revenue management policy, "Reserve Policy", under modified block scheduling policy.

We borrow the idea of reserve scheduling policy from the revenue management literature. Littlewood (1972) developed the first static single resource model using protection levels to characterize the optimal airline booking policies for single flight leg revenue management problems. Since then, many allocation policies were developed by modifying existing models to fit the needs of the health care industry. Among them, we have the expected marginal seat revenue (EMSR) control for multiple classes (Belobaba 1987; Belobaba 1989), a sequential application of the two-fare class rule to the multiple-fare class situations, when requests arrive in increasing fare order. Despite the success of this body of work, most of the above-mentioned models make simplifying assumptions. Dynamic programming has been used in an effort to relax some of the assumptions incorporated into the policies reflected by Littlewood's rule, EMSR and the optimal policies.

While this study is motivated by a case study of the scheduling system at the Stafford Hospital, our analysis of the resulting model focuses on generating valuable insights for practitioners as well. This includes providing an answer to the following questions:

- When do we need to allocate more block time than average demand would suggest?
- Which mix of open/block scheduling strategy is best for each combination of patients?
- Who should access released hours?
- What are the rules of thumb for scheduling multi-priority patients?

Chapter 3 : Allocation and Release Policy Decisions: Model Objective and Case Study

Model Objective:

Any planned capacity that accommodates stochastic demand most of the time will inevitably involve considerable unused capacity. Conversely, a level of capacity chosen to minimize unused capacity will inevitably cause an increase in wait time for patients (Pandit et al., 2010). Thus the question of how to balance demand and capacity is closely related to the question of how to balance utilization and waiting time. To answer this question, a multi-objective model that characterizes financial, temporal, and clinical measures is employed within a simulation/ optimization framework.

Providing more capacity (OR Block time) generates more cost, but the key question is what is the marginal benefit of additional capacity? In constrained optimization, the shadow price is the incremental change per unit of the constraint in the objective value of the optimal solution of an optimization problem obtained by relaxing the constraint. We can expect a non-linear function for the marginal benefit of unit block hours and release block policies to add even more complexity to the model. A release block policy provides essentially free block time over time, so its interaction with the original block size is an interesting issue that warrants investigation.

Another element in our objective function is the cost associated with the waiting time of the patients. The waiting time is defined as the time gap between when a patient calls to make an appointment for surgery and the time the surgery is actually performed (defined as indirect waiting time in literature, Gupta and Denton, 2008). Of course, what counts as waiting time depends on the type of surgery required: we will not penalize every individual for waiting since some procedures are intentionally scheduled multiple weeks in advance because they do not involve emergencies and others, while necessary to preserve a patient's life, do not need to be performed immediately (Semi-urgent surgery).

Patients who expect, but do not receive, immediate service may decide to leave for another surgeon or may be forced to leave because their condition has deteriorated. Table 3 summarizes the cancellation rate of Stafford’s patients through time. Numbers are calculated based on the total counts of canceled surgeries among total number of appointments within a given time period. We expect that as the waiting time increases, the probability of leaving increases, as well. For some specialties such as plastic and Ortho, patients decide to cancel the appointment somewhat early (more than 75% cancel their appointment after a week) while in other specialties (such as podiatry) patients keep their appointment as long as they can.

Table 3: Cancel rate as a function of waiting time across semi-urgent cases

% of semi-urgent cases canceled which have waited after x-days							
Weeks	x-days	Podiatry	Plastic	Otho	GYN	GV	ENT
1	5	15%	83%	76%	37%	50%	44%
2	10	6%	0%	14%	23%	32%	17%
3	15	15%	17%	10%	17%	9%	17%
4	20	12%	0%	3%	13%	9%	11%
5	25	9%	0%	0%	3%	0%	0%
6	30	6%	0%	3%	7%	0%	11%
7	35	3%	0%	0%	0%	0%	0%
8	40	12%	0%	0%	0%	0%	0%
9	45	21%	0%	0%	0%	0%	6%

Based on the above elements we defined our multi-objective function (1) which maximizes the profit through minimizing the block cost incurred by scheduling patients in surgical blocks as well as minimizing waiting time costs[2]. For a period N days into the future and S surgeons using OR blocks, we can represent an objective function as follows.

$$\text{Maximize Profit} = \max \sum_{i=1}^{S} r_i X_i - \sum_{i=1}^{S} c_i T_i - \sum_{i=1}^{S} \sum_{j=0}^{N} r_i \pi(t)_j \tag{1}$$

$\sum_{i=1}^{S} r_i X_i$	Total expected revenue from all cases
$\sum_{i=1}^{S} c_i T_i$	Total OR block time cost
$\sum_{i=1}^{S} \sum_{j=0}^{N} r_i \pi(t)_j$	Total waiting cost
T_i	Total OR block time provided for surgeon i in the period
X_i	Total cases done with surgeon i in the period
r_i	Per case revenue associated with surgeon i case

[2] - we assume there is no overtime allowed

$\pi(t)_j$	Probability of cancellation, given delay time j
c_i	Cost per unit of block time for surgeon i

Model Variable: Cost of OR block time

Operating rooms are significant cost drivers in hospitals and their costs can vary depending upon the type of medical procedure. "How much does one unit (hour) of OR time cost for each specialty/surgeon?" is a common question often asked by operating room (OR) management in order to effectively evaluate and manage the scarce resources. Total OR time provided to a surgeon will consist of his utilized OR block time and OR hours released from underutilized services. So, both allocated OR block hours and release policies will directly affect the total block cost. The best decisions about the allocated block size and release policies should be made considering the difference among surgical block cost. The total block cost of each surgeon can be calculated with the following function (2),

$$\sum_{i=1}^{s} \boldsymbol{c_i T_i} = \sum_{i=1}^{s} \boldsymbol{c_i} * [(allocated\ OR\ hours\ for\ surgeon\ i) - (allocated\ OR\ hours\ released\ to\ other\ services) + (OR\ hours\ released\ from\ underutilized\ services\ used\ by\ suregon\ i)] \qquad (2)$$

Model Variable: Cost of waiting

Delaying surgery may lead to a deterioration of the patient's condition, a poor clinical outcome, an increased risk of death, or an increase in the probability of emergency admission. Recently, policy makers have called for the establishment of target access times for major operations that would minimize the risk of adverse events associated with treatment delays (Sobolev and Kuramoto, 2008). Thus, the risk of adverse events while waiting should be considered explicitly when building a surgery schedule since it will increase the risk of leaving the system (cancellation). In this study, we employ a logistic regression model to estimate the (indirect) waiting cost as the probability of cancellation due to the time gap between request and the appointment. As past research shows, the longer the delay in appointment, the higher the chances that he (she) will cancel the appointment (Gallucci et al., 2005). The waiting cost of each surgeon can be calculated with the following function (3),

$$\boldsymbol{r_i \pi(t)_j} = Waiting\ cost\ of\ patient\ i\ given\ delay\ time\ j\ (expected\ revenue\ loss) \quad (3)$$

$Revenue\ per\ case\ associated\ with\ surgeon\ i\ case\ x\ Probability\ of\ patient\ i\ opting\ out\ of\ the\ procedure$

The waiting cost model that we propose in this study has much in common with existing research on capacity allocation and yield management. We observe, analyze and anticipate patient behavior in response to waiting time in order to maximize yield or profits from a fixed perishable resource (OR time). Although the surgery cost would be fixed over the same procedure, managing the available OR time will affect the overall profit. The daily challenge facing the surgical scheduler is to allocate the available capacity between the priority classes so as to minimize the number of patients whose wait time exceeds a pre-specified, priority-specific target, with greater weight given to any late bookings of higher-priority demand. This requires significant foresight because each day's decision will clearly impact what appointment slots are available for future demand. If lower-priority patients are booked too soon, then there may be insufficient capacity for later-arriving higher-priority demand. Conversely, if lower priority patients are booked too far into the future, there is the potential for idle capacity (Patrick et al., 2008). The arrival times of patients are uncertain, so the scheduler must decide whether or not to reserve the next available OR for the potential next semi-urgent patient and risk the potential for idle capacity. An optimal scheduling strategy will mitigate this risk for semi-urgent patients and the hospital at the same time.

Waiting cost is comprised of the contribution margin of each type of patient multiplied by the probability that the patient will leave, given the indirect waiting time. Yield management was originally used in the airline industry to manage strategic control of seats in order to sell to the right customers at the right times for the right prices. After yield management's success was established in the airline industry in the 1970s, it has grown in many industries and organizations that face the challenge of satisfying customers' uncertain demand with a relatively fixed amount of resources.

Healthcare is an area in which yield (revenue) management has not been intensively used, probably because most segments within this industry are working on a non-profit base and it can be argued that revenue management could raise some ethical issues. But this may not necessarily be true. Hospital cost represents a large and increasing percentage of the national GDP (Gross Domestic Product) and, as any other business that supplies a good or service to its customers, a healthcare unit needs to generate ample revenue for sustainability and future growth. There is no harm in increased revenue when the long term goal is better customer satisfaction through decreasing waiting time while maintaining an acceptable quality of service (Strum et al., 2008).

In the next session, we provide a case study of a hypothetical hospital to illustrate practical aspects of implementation of the multi-objective function in joint optimization of block allocation decisions and block release policies.

Case Study of Joint Optimization of block allocation decisions and block release policies:

We split the study into three subset models with different complexity and assumption levels, as shown in Table 4. This breakdown will give us more insight about the effect of interaction in the final model.

Table 4: Description of models

	Decision 1	**Decision 2**	**Decision 3**	**Decision 4**	**Assumptions**	**Objective**
Model 1	**Xi***	**xi***			-No release allowed (scenario 1)	Best OR allocation with lowest waiting penalty
Model 2			**Ti***	**R***	-Release allowed -Obtain the Xi* & **xi*** of Model1 as input	Best block release policy with lowest waiting penalty
Model 3	**Xi***	**xi***	**Ti***	**R***	-Release allowed (scenario 2)	Best OR allocation and release policies with lowest waiting penalty

<u>Decision variable 1</u>	*Xi*: allocated OR block for each surgeon i*
<u>Decision variable 2</u>	*xi*: Allocated weekly OR block schedule for each surgeon i*
<u>Decision variable 3</u>	*Ti*: Improved release time (Days before surgery to release block i)*
<u>Decision variable 4</u>	*R*: Rules to assign cases to released blocks*

Discussion: For the surgery master schedule, the maximum number of cases on each day or total work hours per surgeon per day is limited based on resources as beds, nurses, equipment and downstream resources. Thus, to control this constraint and exclude its effect in the model we set an upper bound on the number of cases of each type for each day.

In the first model, we assume that no block release policy exists. In other word, surgeons cannot share blocks even if blocks are not fully filled, and each case should be assigned to the blocks of its surgeon. In this model, the best solution of OR block size (Xi*) and the best allocated times for each surgeon (xi*) will be evaluated based on maximizing profit. The optimal output of the first model (the best block size and allocation plan) will be set as an input for the second model, where a block release policy is assumed to exist. Thus, in this stage of our analysis, we are seeking the best release policy given the OR block sizes from the previous level. The third model will yield the results that are of most interest in this study. In it, we expand the scope of the second model and incorporate all decision variables and their interaction without any assumption about block size or release policy. The difference among the results of these three models will provide insight into sensitivity of our objective to policies and decisions.

Simulation software:

The models described in the previous sections were implemented using the ExtendSim simulation environment. ExtendSim is an advanced simulation software that can dynamically model continuous, discrete event, discrete rate, agent-based, linear, non-linear, and mixed-mode systems. The integrated simulation database creates an interface that facilitates dynamic simulation modeling. The models are comprised of blocks that communicate with each other to describe the simulated sequence and the general logic of the model. In addition, for specialized purposes, an ExtendSim custom block can be created that can be programmed in ExtendSim's C-based ModL language. Finally, ExtendSim's Scenario Manager leverages the database to store model factors and responses and run experiments to analyze how a model reacts to different factors.

Scenario analysis and design of experiment

Simulation approaches are well suited for applying a scenario analysis to discover the responses of the simulated system based on different factors. The challenge is to integrate simulation and optimization in order to bring together the capability of the simulation in the scenario analysis ("what-if" analysis) and in describing the dynamics of the system considered and the prescriptive strength of the optimization, i.e., the "what's-best" analysis (Ozcan et al., 2011) . ExtendSim's Scenario Manager provides an easy interface to design different model configurations and run experiments to understand how a model reacts to different factors. When analyzing the model responses, it can be helpful to employ design of experiments (DOE) to reduce the number of model runs required to compare multiple scenarios. This is particularly useful for initial investigations where the modeler needs to determine which factors have the most impact on system performance (Krahl, 2011). The results of DOE can easily be translated to other analysis programs, such as statistical software, for further analysis.

In order to better understand how the best decisions are generated, we have started with an analysis of a simple hypothetical hospital with two ORs (OR1 and OR2) and four types of surgeons/procedures (A, B, C & D) as an abstract version of our real model (Appendix A). We assume that demand for each type of surgery arrives according to a Poisson distribution so the inter-arrival times follow an exponential distribution. Also, we estimate revenue for each surgical case and unit block costs according to each type of surgery provided (Table 5). For simplicity, we assume that the case duration is set at one hour for any type of surgery. The cost of waiting reflects the probability of a patient opting out of the procedure and is represented as a logarithmic function of waiting time. This logarithmic function is shown in Figure 7.

The following formula defines how waiting cost of each patient is calculated in the objective function (as a logarithmic function of waiting time).

Probability of a patient opting out of the procedure =

$$\left[exp(-2.49 + (0.03 * waiting\ time\ of\ \ patient\ i))\Big/(exp(-2.49 + (0.03 * waiting\ time\ of\ \ patient\ i)) + 1)\right]$$

*Waiting cost of patient i (expected $ loss) = Revenue per case associated with surgeon i case * Probability of a patient opting out of the procedure*

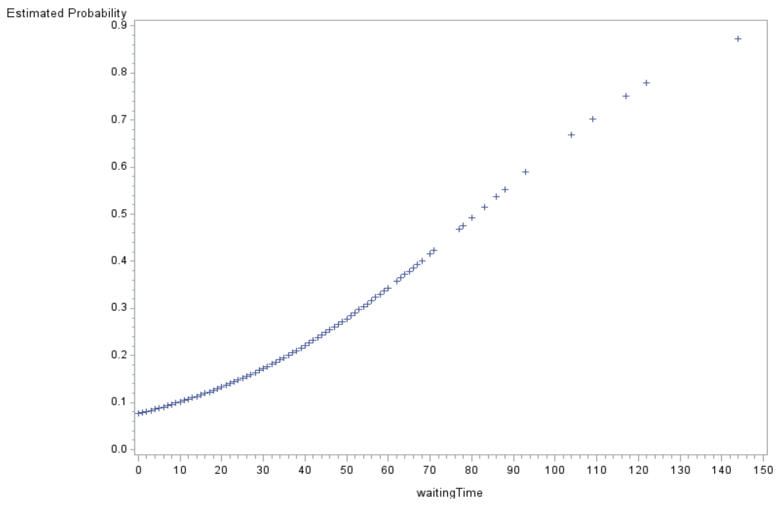


Figure 7: Probability of Cancellation given waiting time

Table 5: Input data for abstract model

	Surgeons/ procedures			
	A	**B**	**C**	**D**
Inter-arrival time distribution	Exp ~ (1.3)	Exp ~ (12)	Exp ~ (6)	Exp ~ (2.8)
Unit revenue ($/case)	5	8	7	6
Unit block cost ($/hr)	3	3	3	3
Contribution margin ($/case)	2	5	4	3

Patrick and Puterman (2008) claim that if average demand exceeds available capacity (regular and overtime[3]), then no optimal schedule can be achieved. So, we define the base capacity such that it meets average demand based on arrival rates and case durations (while wait time may continue to increase owing to the variability in demand). The block size is formed such that it meets weekly average demand. In this model, a cyclic block schedule is used to allocate operating room time to particular surgeons for their elective surgeries (Table 6). It is assumed that ORs are open eight hours a day from 8:00am to 4:00pm.

[3] - We don't have any overtime in this study

Table 6: Cyclic block schedule

	Weekly block schedule								
	Hour	8:00	9:00	10:00	11:00	12:00	13:00	14:00	15:00
Monday	OR1	A	A	A	A	A	A	A	A
	OR2	B	B	B	B	B	■	■	■
Tuesday	OR1	A	A	A	A	A	A	A	A
	OR2	D	D	D	D	D	D	D	D
Wednesday	OR1	A	A	A	A	A	A	A	A
	OR2	D	D	D	D	D	D	D	D
Thursday	OR1	A	A	A	A	A	A	A	A
	OR2	D	D	D	D	D	D	D	D
Friday	OR1	A	A	A	A	A	A	A	A
	OR2	C	C	C	C	C	C	C	C

Working with these hypothetical parameters, we have created a simulation model with a custom "scheduling" module (the simulation interface provided in Appendix B). This scheduling module is programmed to run for two scenarios: 1) block release is prohibited (model 1), and 2) block release is allowed (model 2). For the first scenario, each case is assigned to the next available block in the surgeon's designated block times regardless of the existence of earlier available blocks in other surgeons' block times. In the second scenario, the scheduling algorithm is much more complex. Since blocks are shared after a stipulated release date, a new case may be assigned to any of several available surgeon blocks. Thus, multiple variables must be considered in order to find the best assignment option. Appendix C shows the entire scheduling algorithm which includes scheduling logic and sequential decisions that are programmed in the scheduling block. The program incorporates all assumptions and utilizes all the data tables created in the database. All data for the simulation, included input, current state of the system, objective function and block release policy are stored in the database. This makes it readily useful for simulation and further analysis (Appendix D). The abstract model runs for 187 business days (or six months with 30 days for each month) with the first 7 days excluded as warm-up for the analysis.

The algorithm is based on multiple assumptions:

- Case duration is considered as room duration which includes surgery, cleaning and change over time. There is no time gap between two consecutive cases.

- Operating room cost is assumed to be the same for all procedures and surgeons.
- All patients are considered as urgent patients. In other words, we assume that all request the earliest available block that accommodates their case duration
- The earliest block that a case can be assigned to will be the next day after the request, due to a lead time for preadmission testing and preparation for surgery
- In reality, surgeons cannot schedule elective surgery every day. To limit the number of options and to make the model more realistic, a preference table is provided to represent the days that surgeons are available to conduct surgery.

<u>Model 1: Find the best block size and allocation</u>

In this model, the ExtendSim Scenario Manager can be run to investigate the sensitivity of profit to the block size and allocation plan (Appendix E-G). Three block size scenarios of two, five and eight hours were created for each day (Table 7). Table 8 shows the possible range of block sizes based on the scenarios and the hours needed based on the average demand (arrival rate and case duration). It assumes that all blocks start at 8:00 am every day. Based on the improved solution, end time may vary as either 10:00 am, 1:00pm or 4:00pm. So, the smallest block size is set as two hours.

Table 7: Block size option

Option	**Start Time**	**End Time**	**Block size**
1	8:00	10:00	2 hrs
2	8:00	13:00	5 hrs
3	8:00	16:00	8 hrs

Table 8: Block allocation range

	A	**B**	**C**	**D**
Allocated OR hours range	28-40 hours	2-8 hours	2-8 hours	12-24 hours
Average weekly OR hours demand	30 hours	4 hours	7 hours	14 hours

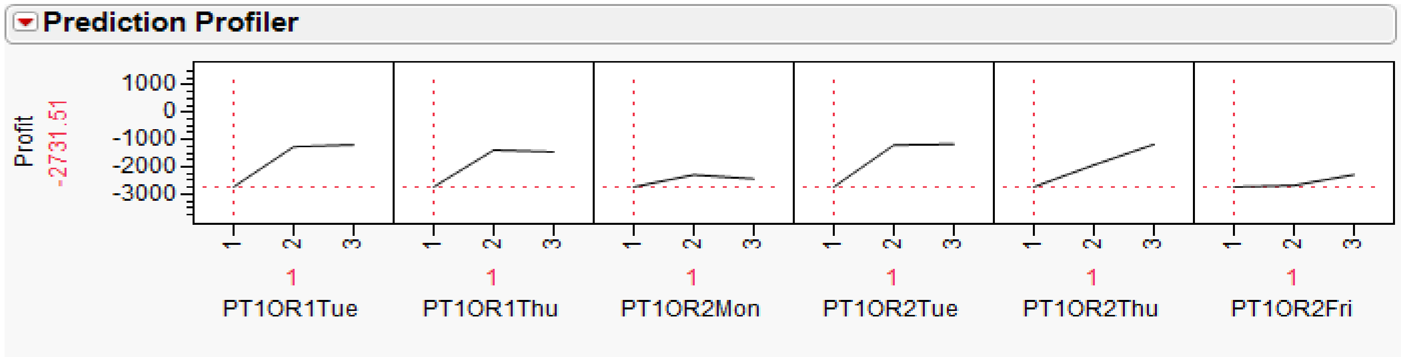
Model 1 Results:

Figure 8 displays the results of the Scenario Manager, which constitute simply the relationship between the six model factors and profit (Table 9). The lines show all the possible what-if scenarios if one input were to change and all other inputs were held constant. It is very easy to see that the block sizes of Surgeon A and D have the biggest impact on profit (have a steeper slope in step 0 of Figure 8).

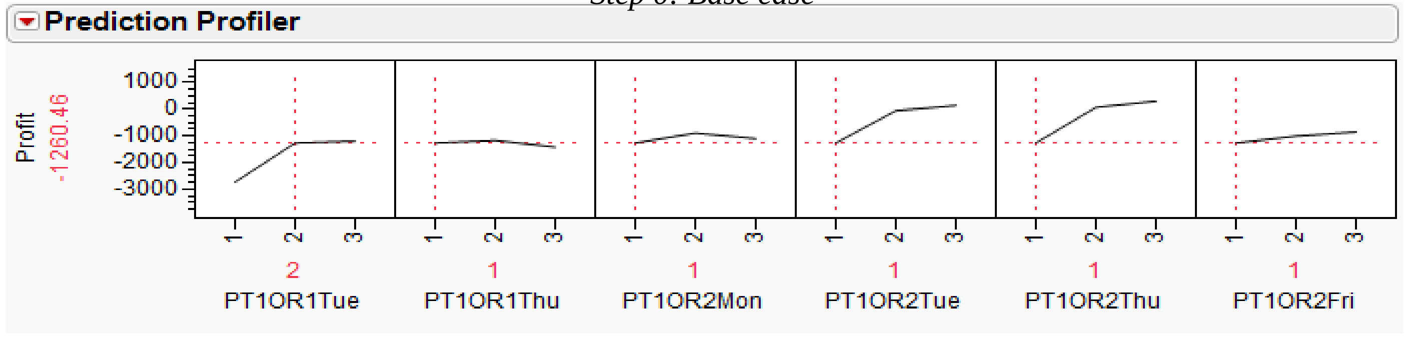
Table 9: Model 1 factors and response

Factors	Response
1. Surgeon A Block size on Tuesday 2. Surgeon A Block size on Thursday 3. Surgeon B Block size on Monday 4. Surgeon C Block size on Friday 5. Surgeon D Block size on Tuesday 6. Surgeon D Block size on Thursday	**Profit**

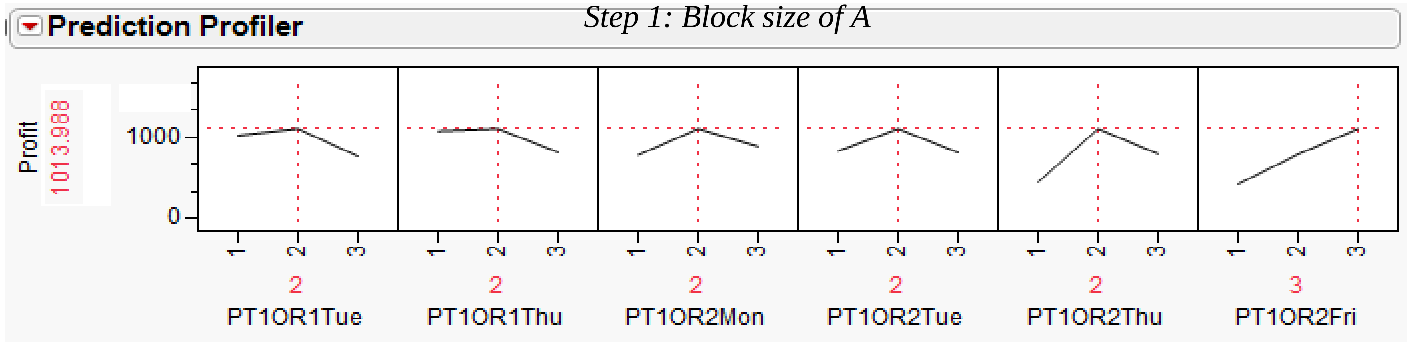
Starting from the base case (step 0), the next step or the next decision is made based on the factor with the most impact on improving the profit. The last figure indicates the best allocation strategy for all surgeons. For Surgeon A there is value in terms of increased profit by increasing block size from 2 hours to 5 hours, but no more value is created by increasing block size from 5 hours to 8 hours. The effect of block size is similar for both Tuesdays and Thursdays (five hours on each day). Although, on average, Surgeon A needs 30 hours a week for his cases, the Scenario Manager shows 34 hours of block time per week would be a better strategy. The flat line for Surgeon B in Step 0 indicates that in comparison with other surgeons' blocks, the block size of Surgeon B will not impact the profit. If Surgeon C decides to extend his or her hours, 8 hours blocks would be better rather than 5 hours. Surgeon D's best block size and allocation is the same as Surgeon A's best result in terms of total hours needed and the amount of service hours on Tuesdays and Thursdays.



Step 0: Base case



Step 1: Block size of A



Last Step: Best allocation plan

Figure 8: Model 1 Results- Scenario manager on output for the prominent block size and allocation

The best block size and allocation plan based on the Scenario Manager result are summarized in Table 10. The best block size results from the scenario manager for Surgeons A, B and D are strictly within the possible block size range while Surgeon C touches his upper bound of 8 hours.

Table 10: The best block size and allocation

	A	B	C	D
Allocated OR hours range	28-40 hours	2-8 hours	2-8 hours	12-24 hours
Average demand	30 hours	4 hours	7 hours	14 hours
Best allocation plan	34 hours	5 hours	8 hours	18 hours
Best weekly schedule	Day – Block Size Mon – 8 hrs Tue – 5 hrs Wed – 8 hrs Thu – 5 hrs Fri – 8 hrs	Day – Block Size Mon – 5 hrs	Day – Block Size Fri – 8 hrs	Day – Block Size Tue – 5 hrs Wed – 8 hrs Thu – 5 hrs

Model 2: Find the best release policy

The next step in this analysis will be to fix the block size and weekly block schedule based on the results of Model 1 and run multiple scenarios on the block release time policy. Two parameters are involved in the block release policy:

(1) The maximum time that a patient can wait for accessing his surgeon's primary block before being considered for scheduling in an off block, and

(2) The minimum number of days before the day of surgery that the block can be released.

Parameter 1 depends on the expectation of the patients about the longest time to wait for surgery and the urgency of the case, but parameter 2 depends on the arrival rate of patients and how quickly blocks are filled. The maximum time that a patient can wait can be estimated for each type of surgery based on survey or statistical methods. Of course, off block surgeries have some amount of inconvenience for the surgeon and team. So, up to some point and based on patient condition, they prefer to keep the case in a primary block until the surgeon decides to search through available off blocks for an earlier time. For this model, we assume it is the same for all types (four days or 96 hours). The minimum number of days before the day of surgery that the block can be released will be analyzed by the Scenario Manager. Since the maximum wait time is set to four days, the minimum scenario range will be considered one to four days for each type of patient.

Model 2 Results:

As shown in Table 11, we examine the effect of four factors (block release time for each type of surgery) on profit. These factors are defined as the number of days before surgery a block can be released such that remaining block hours can be shared among other surgeons who have urgent cases to schedule.

Table 12 is provided to represent the days that surgeons are available to conduct surgery. The surgeons’ preference adds more constraints on the sets of options for the receiving surgeons who want to schedule surgery on released dates. A zero value in the table indicates a day when the surgeon is not available to perform any operation.

Figure 9 shows the first and the last step of Scenario Manager’s outcomes of this model. Although all release policies will affect the profit, a comparison of the release time of surgeon A is shown to have the most effect on final profit. The last figure provides the best release block policy for this model since no more gain in profit is possible after this point.

The concavity of these lines (in the last figure) suggests releasing blocks three days before the surgery date is the best strategy of release policy for this case study. It means that if surgeons release their remaining block hours to others, then the overall hospital profit will be maximized over time. These policies are summarized in Table 13.

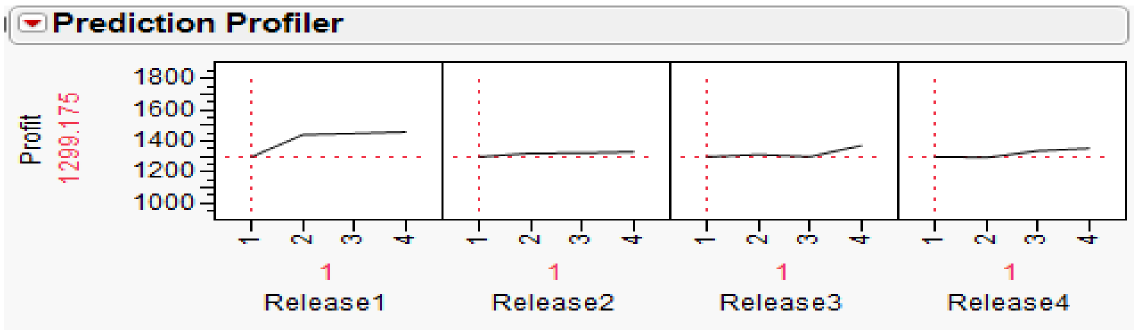
Table 11: Model 2 factors and response

Factors	Response
1. Release day for surgeon A's block 2. Release day for surgeon B's block 3. Release day for surgeon C's block 4. Release day for surgeon D's block	**Profit**

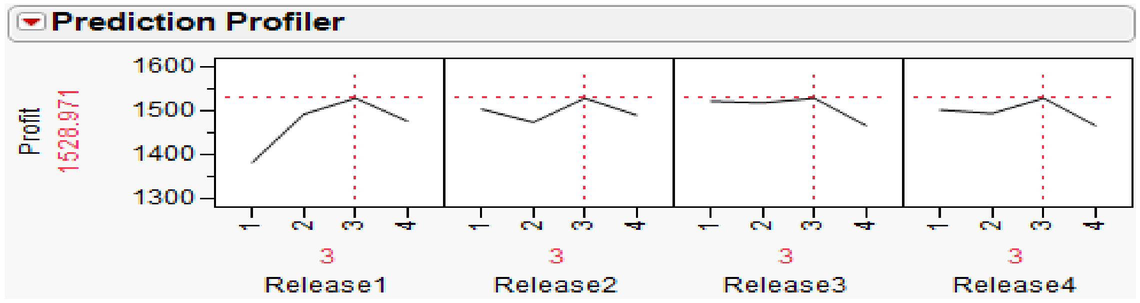
Table 12: Surgeons' preference table

	MON	TUE	WED	THU	FRI
A	1	1	1	1	1
B	1	1	1	1	1
C	1	0	1	0	1
D	0	1	1	1	0

1: does surgery *0: does not surgery*



Step 0: Base case



Last step: Best release policy

Figure 9: Model 2 results- Scenario manager on the superior release policy

Table 13: The best release policy scenario

	A	B	C	D
Release time scenario (# of days before surgery)	1-4 day(s)	1-4 day(s)	1-4 day(s)	1-4 day(s)
Best release policy	3 days	3 days	3 days	3 days

<u>Model 3: Find the best block size, allocation and release policy</u>

This model conducts a comprehensive experiment on the interaction of all block allocation decisions and release policies on profit. This model is a combination of two previous models which contains ten factors consisting of, six factors of block allocation from model 1 and four block release policies from model 2 (Table 14). Since it is a large full factor model with multiple levels of running, two factors are excluded from the scenario analysis model (Release time and block size of block B). For comparability, their value is set as the best output of the two previous models (Five hours block on Monday with a three-day release policy).

Table 14: Model 3 factors and response

Factors	Response
1. Release day for surgeon A's block *2. Release day for surgeon B's block* 3. Release day for surgeon C's block 4. Release day for surgeon D's block 5. Surgeon A Block size on Tuesday 6. Surgeon A Block size on Thursday *7. Surgeon B Block size on Monday* 8. Surgeon C Block size on Friday 9. Surgeon D Block size on Tuesday 10. Surgeon D Block size on Thursday	**Profit**

Model 3 Results:

It is expected that the effect of block size on profit is much more than the effect of release block policy (larger slope for block size factors in step 0 of Figure 10). Although the impact of release policy appears negligible initially, but it becomes a more significant factor with increasing block sizes. In other words, when we are close to the appropriate block size, then release block policies become the best strategy to take advantage of “free” available capacity in improving the profit.

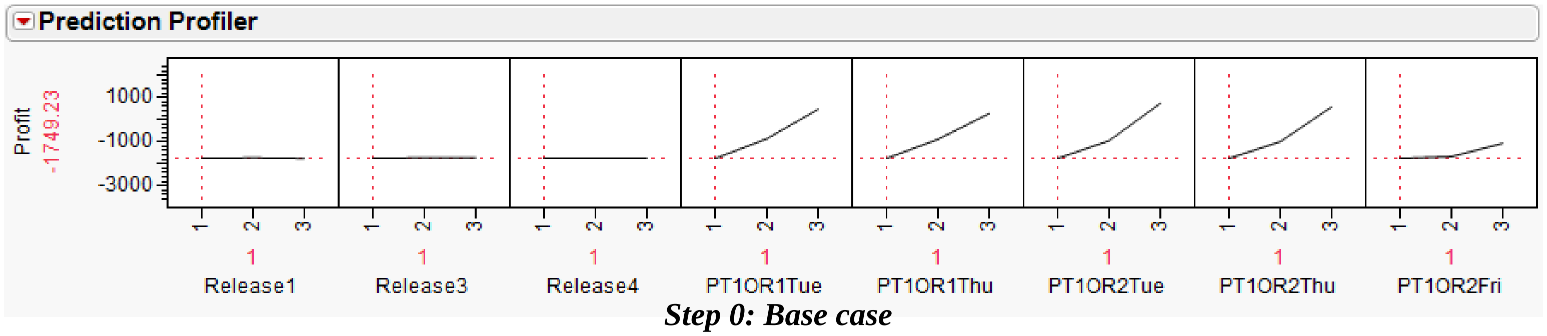
The output of the scenario manager is given in Figure 10. It demonstrates the snapshot of three steps of what-if scenarios and how release policy becomes effective after the block allocation decision process (step 4). The results suggest that utilizing the release policy instead of asking for additional block hours (with more associated cost) would be a viable strategy to improve profit. Although Surgeon C’s block size is reduced to two hours, the overall profit improves in this model. Table 15 summarizes the best decisions developed by Scenario Manager.

Table 15: Model 3 best decision and release policy

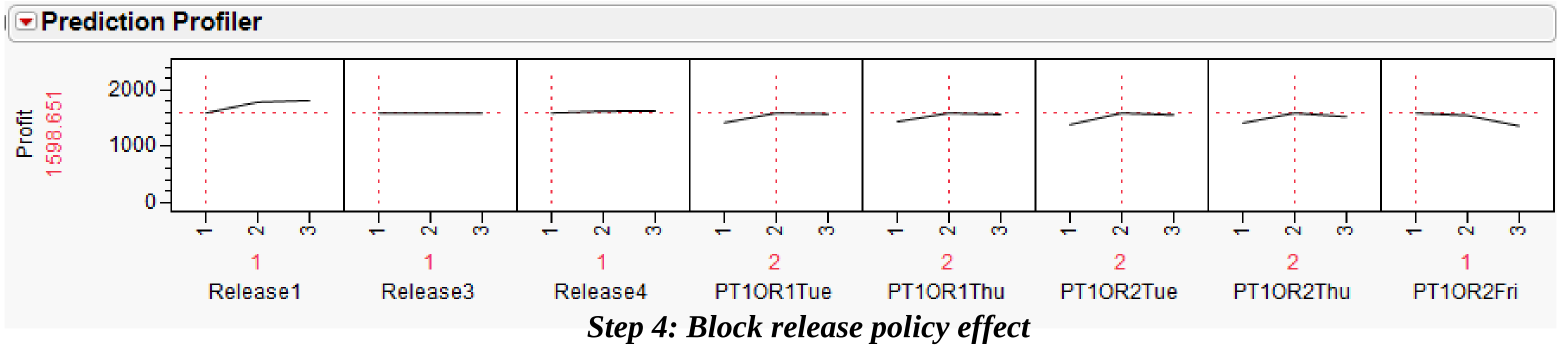
	A		B		C		D	
Best allocation plan	34 hours		5 hours		2 hours		18 hours	
Best weekly schedule	Day	Block Size	Day	Block Size	Day	Block Size	Day	Block Size
	Mon	8 hrs	Mon	5 hrs	Fri	2 hrs	Tue	5 hrs
	Tue	5 hrs					Wed	8 hrs
	Wed	8 hrs					Thu	5 hrs
	Thu	5 hrs						
	Fri	8 hrs						
Best release policy	4 days		3 days		2 days		4 days	

<u>Result(s) comparison:</u>

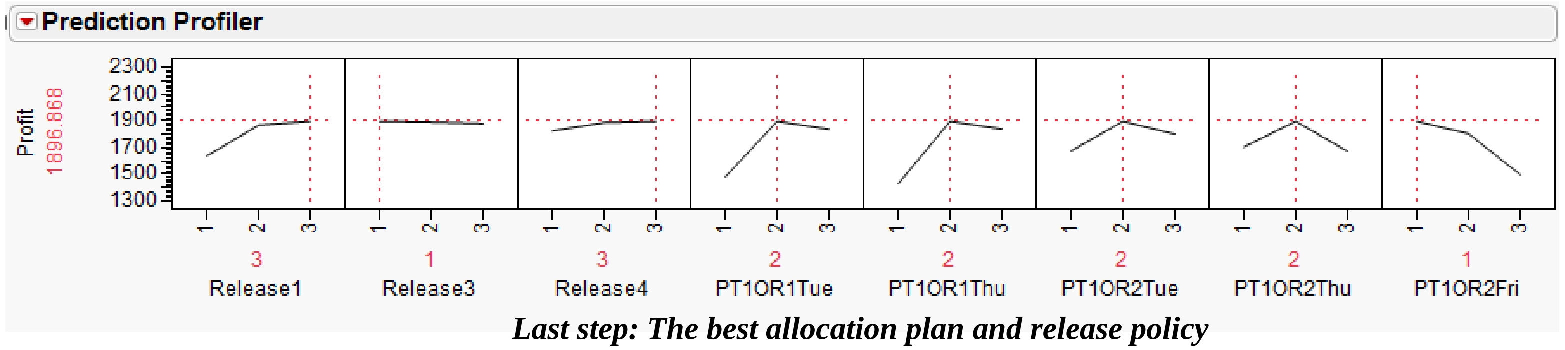
The main goal is to find the best combination of block decisions and release policies that maximize the overall profit. The contribution of these three models is shown in Table 15. These models try to provide the minimum amount of surgical blocks while maximizing the profit (maximize utilization). Models 1 and 2 have the same amount of surgical blocks provided to surgeons, but the difference in profit is due to introducing the release policy in the second model.



Step 0: Base case



Step 4: Block release policy effect



Last step: The best allocation plan and release policy

Figure 10: Model 3 results- Scenario manager on the prominent block allocation and release policy

The second model sets the block size at its improved position which was calculated in the first model, then determines the best release policies. Model 3 is a generalization of the second model whereby the block allocation may not be superior. The difference between model 2 and 3 reveals the interaction effect of scheduling decisions and release policy.The results suggest that ignoring the interaction of decisions will penalize the overall profit. In this abstract model (with given inputs and assumptions) the profit increases more than $1726 a year ($863 per 6-month period) with less total operating block time provided to surgeons (comprising the difference between profit of model 1 and model 3). The median waiting time for the first model is unevenly spread across surgeons since there is no possibility for sharing blocks among surgeons and Surgeon B and C have only one dedicated day a week to perform surgery. The next two models demonstrate how this limitation can be eliminated with block release policies. The median waiting time is more even across surgeons in the next two models because we set a maximum day that a patient can wait and a release date for sharing unfilled blocks. There is no measurable difference between the median waiting times for model 2 and model 3 (Table 16).

Table 16: Comparison of three models

	Model 1		Model 2		Model 3	
Profit	$ 1133		$ 1563		$ 1996	
Total block hours	65 (hrs/week)		65 (hrs/week)		59 (hrs/week)	
Median waiting time for patients (hours)	A	45	A	45	A	47
	B	92	B	48	B	51
	C	94	C	49	C	47
	D	50	D	49	D	48

Although in model 3, Surgeon C has lost six hours of his dedicated OR time per week, it does not have any effect on the median waiting time of his patients. This loss is compensated by earlier block release times for Surgeons A and D and a later release time for Surgeon C (Table 17).

Table 17: Model 2 and 3 best release block policies

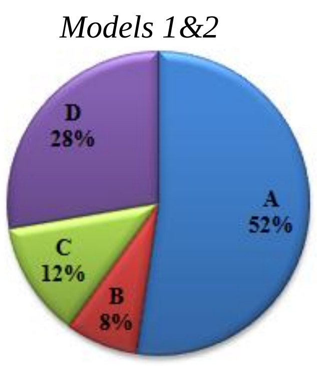
Best release policy	A	B	C	D
Model 2	3 days	3 days	3 days	3 days
Model 3	4 days	3 days	2 days	4 days

Finally, we explore the details of the differences among these three models. The pie charts in Figure 11 demonstrate the block allocation proportions of the four surgeons in Models 1 and 2 (the same as model 1), and of model 3. Comparing the solutions indicates how the third model increases the overall profit with less service hours. It devotes six fewer block hours to Surgeon C and keeps the same amounts for the other three. This reduction did not erode the profit due to new block release policies.

The inter-arrival times of Surgeon C’s patients follow an exponential distribution with mean 6 hours as compared to the inter-arrival time of Surgeons’ A and D arriving requests which are also exponential but with averages of 1.3 and 2.8 hours, respectively. The results establish that it is profitable to reduce service hours of C with less frequent patients and modify the block release policies (later release time for C and earlier release time for A and D) such that these patients can easily fill underutilized hours of surgeons with more frequent patients. This resulted in higher utilizations for the third model (see utilizations in Figure 12).

Next we evaluate the percentage of cases that are done in non-primary blocks (off-block) in models 2 and 3. As can be seen in off-block Table 18, the percentage of primary block surgeries of Surgeons’ A and D remain unchanged in the two models with around 100% of surgeries within their primary blocks. Each number represents the percent of the row surgeries performed in the column surgeon’s block.

The main difference between the results of models 2 and 3 is in Surgeon C’s block, where more than two thirds of the surgeries are done outside of his primary block in model 3. The reason is that in model 3 the block size of C is reduced from 8 hours to 2 hours per week so cases will need to be done in other surgeons’ blocks.



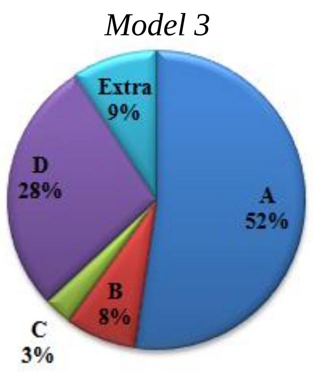


Figure 11: Case mix Block allocation proportions of four surgeons A, B, C & D

Table 18: Percentage of off-block surgeries

	A	B	C	D
A	99.9%	0.0%	0.0%	0.1%
B	15.8%	69.9%	1.5%	12.8%
C	10.1%	17.3%	70.1%	2.5%
D	0.9%	0.0%	0.0%	99.1%

Model 2

	A	B	C	D
A	100.0%	0.0%	0.0%	0.0%
B	14.3%	67.7%	0.0%	18.0%
C	36.2%	21.7%	29.5%	12.6%
D	1.5%	0.0%	0.0%	98.5%

Model 3

In the second model, Surgeon C has eight hours with release time of three days before the day of surgery. In this model, seventy percent of cases are done in C's primary block and around twenty percent in block B. In contrast, when block C reduces to 2 hours per week over seventy percent of the cases are done outside of the primary block, mostly in blocks A and B. Earlier release times increase the opportunity for other surgeons to schedule their cases in alternate non-primary blocks. On the other hand, the primary surgeon will lose his or her access to the dedicated block. This trade-off is defined in the best block release policies for each surgeon.

Finally, the utilization rates of the block are analyzed in Figure 12. In general the utilization rate is improved from model 1 to model 3. While models 1 and 2 have the same amount of surgical block hours; utilization is increased in second model (due to introduction of release policies) except for Surgeon's C block where the utilization declines. The main reason is that 30% of Surgeon's C cases are done out of his block mostly in Surgeons' A and B blocks to keep the waiting time of the patients as low as possible.

In order to avoid patients delays and staff overtime due to OR utilization higher than 85% to 90% for surgeon A and C, extra OR available time can be allocated to them to increase efficiency of an OR without the cost of patient inconvenience. These results outline the advantage of considering the joint impact of allocation decisions and block release policies towards higher profit for hospitals and lower waiting times for patients.

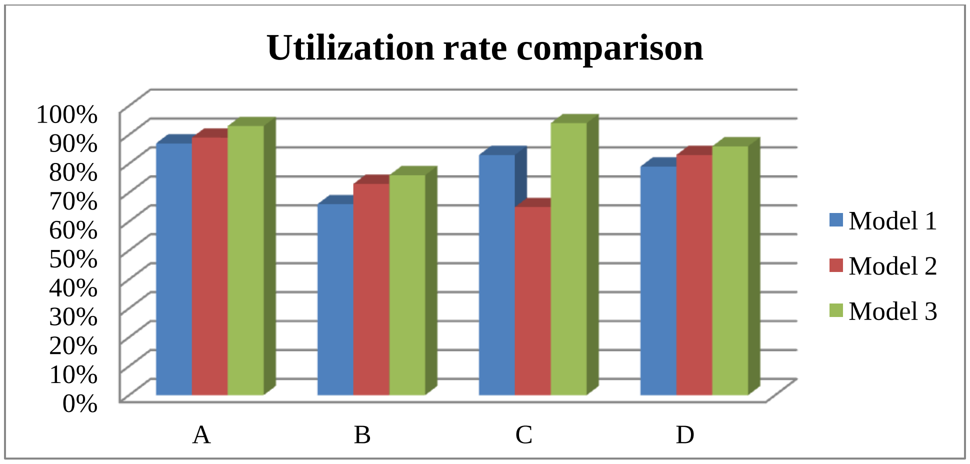


Figure 12: Utilization rate of Surgeons' service hours

Chapter 4 : Research Design and Methodology

Methodology: Development and Evaluation of Case Scheduling Policies

Even though patient scheduling problems have been studied extensively over the last decade, the dynamic allocation of medical capacity in advance of the service date, when future demand for service is still unknown and in the presence of multiple types of patients has received limited attention. In general, three types of scheduling decision problems have been considered: (1) who to serve next, (2) when to schedule the arriving patient and (3) how much capacity to reserve for a particular class of patients. The first question tries to schedule available patients on the day of service (referred as allocation scheduling), while the next two questions focus on scheduling patients in advance of service date (referred as advanced scheduling based on Margerlin and Martin (1978)). Advance patient scheduling decisions usually rely on the expertise of one or two bookings agents and are made without explicitly considering the impact of current decisions on the future performance of the system (Sauré, 2012). Our study addresses the second question where it is important to fix appointment dates soon after multi-priority patients are requesting the surgery. In this dissertation, we acknowledge the importance of developing advanced scheduling concepts and techniques instead of relying on conventional wisdom. The main approaches that have been adopted for surgery scheduling are mathematical programming and simulation modeling. Mathematical programming (especially, integer and dynamic) models have been shown to be useful in capacity planning and resource allocation in many complex systems; while valid simulation models are useful in estimating the actual performance of a planned system in advance. Especially when the system exhibits considerable stochastic behavior or when it is relationally complex, simulation proves to be useful as it possesses an extensive modeling flexibility and allows for a sufficient degree of detail.

Simulation approaches can be classified as static or dynamic, as deterministic or stochastic, and as involving discrete or continuous time (Law and Kelton, 2000). Static simulation models, often called Monte Carlo models, furnish the decision-maker with a range of possible outcomes and the probabilities that will occur for any choice of action and at a particular point in time. In contrast, a dynamic model represents a process as it evolves over time. In deterministic models, a set of input parameters results in a unique output, whereas stochastic models contain at least one probabilistic random variable. As a result, the output of a stochastic simulation model is itself random. In discrete-event models, the state variables change instantaneously at separate points in time, these points in time are the ones at which an event occurs, where an event is defined as an instantaneous occurrence that may change the state of the system, whereas in continuous-time models, the state variables change continuously over time (Sobolev et al., 2011).

A comprehensive review of simulation models of surgical suites published over the past five decades was done by Sobolev et al. (2011). They identified a total of 1,332 publications by searching eight electronic databases. In this section, we will provide a brief review of the more recent and related research. Rising et al. (1973) applied simulation models to evaluate the performance of alternative booking policies in an outpatient clinic considering two-priority patient types. Everett (2002) employed a simulation model to provide decision support for the scheduling of patients waiting for elective surgery in the public hospital system. The model was used to match hospital availability and patient need (urgency level) and also to compare the effectiveness of alternative policies. In a series of papers by Dexter et al. (2003) and Dexter and Macario (2004), a simulation model was applied to study the effect of multiple assigning rules on adding a single elective case to an existing surgical schedule on block release dates ranging from one to five days before the day of surgery.

Denton et al. (2007) used simulation as a tool to evaluate the tradeoff between patient waiting time, OR team waiting time, OR idling and overtime in a multi-room surgical suite. Testi et al (2007) studied the problem of assigning patients to ORs (on a single day) using discrete event simulation to judge the quality of different scheduling policies. Vermeulen et al. (2009) developed a dynamic method for scheduling CT-scan cases within a radiology department. The result of their simulation showed a significant improvement in the number of patients scheduled on time. A simulation study was carried out by Steins et al. (2010) to find new ideas and new planning and scheduling techniques to improve the utilization of overall operating room capacity including pre- and post-operating activities. Persson and Persson (2010) described a discrete-event simulation model to study how resource allocation policies in a department of orthopedics affect the waiting time and utilization of emergency resources, taking into account both patient arrival uncertainty and surgery duration variability. Schutz and Kolisch (2012) adopted a revenue management approach to address the problem of determining whether or not to accept MRI-scan requests for different patient types.

The results and policy insights of our research are based on the two approaches of simulation and Markov decision process (MDP). An MDP models a system in which decisions are made sequentially over time, and future decisions and outcomes depend on current and past decisions (Puterman, 1994). MDPs are useful for studying a wide range of optimization problems solved via dynamic programming. Applying an MDP provides an optimal policy that prescribes how best to manage the system in any contingency. It offers a systematic alternative to the "guess and check" approach that underlies using simulation on its own to determine good policies (Patrick et. al, 2008). However, to determine optimal policies for realistic-sized systems, the MDP model becomes challenging, if not impossible, to apply due to the curse of dimensionality.

Over the past two decades, researchers in operations research, engineering and computer science (Bertsekas and Tsitsiklis, 1996; Sutton and Barto, 1998, Sauré et al., 2012) have developed a new branch of operations research called approximate dynamic programming (ADP) that seeks to overcome such computational challenges. The ADP approach has been employed for the surgery scheduling problem with multi-priority patients addressing the issue of how to balance underutilization cost and the cost for postponing surgeries. ADP methods produce good but not necessarily optimal solutions to the underlying problem. Policies obtained through ADP must be evaluated by testing them in a system simulation model. We use a simulation model to compare the optimal scheduling rules derived from the ADP with a range of alternatives, including current practice.

Patrick and Puterman (2008) worked on scheduling multi-priority patients to available future slots, while simultaneously accounting for uncertain demand over each day. Their objective was to minimize the total penalty cost incurred when patients had to wait longer than a target waiting-time. They modeled this as an infinite-horizon Markov decision process (MDP), and solved it using approximate dynamic programming (ADP). The work of Sauré et al. (2012) represents an extension of the dynamic multi-priority patient scheduling developed with Patrick et al. (2008) in which they consider patients who receive radiation treatment across multiple days and for irregular lengths of time. Erdelyi and Topaloglu (2009) study a problem that involves allocating a fixed amount of daily capacity among entities with different priorities. They consider a finite planning horizon and focus on a set of protection level policies. Liu et al. (2010) develop dynamic policies for a primary care clinic taking into account patients' cancellation and no-show behavior.

Scenario analysis on different strategies on yield management:

Although in some cases long wait time may have little medical impact, in others, excessive wait times can potentially impact health outcomes and result in losing patients. In the Stafford hospital, there are two types of patient studied: semi-urgent who may require immediate treatment, and non-urgent patients whereby it may be medically acceptable to wait up to several weeks.

There is no cost associated with delay in scheduling non-urgent patients (zero waiting cost). In contrast, the hospital will be penalized for postponing the scheduling of semi-urgent patients. Based on cluster analysis, semi-urgent patients are classified into multiple priority classes. In this case, the allocation decision and block release policy factors would affect the overall profit since patients are assigned to the next available slot considering their home block status and their urgency. But what will be the best set of policies for optimal yield across multi-priority patients? Since less-urgent patients are booked further into the future, this raises the question as to how much resource capacity to reserve for later-arriving but higher-priority demand?

The decision of when a patient should be scheduled is made based on the cost for surgery postponement. A numeric solution is formulated to address this problem and multiple strategies are conducted to understand the properties of an optimal scheduling policy. We are looking for sets of superior strategies to better manage health-care resources in order to reduce wait times to acceptable levels without undue additional costs.

Optimal scheduling policy

In order to gain insight into the structure of an optimal scheduling policy with lowest overall waiting cost for patients with different priorities, a series of tests was conducted for a hypothetical hospital with two ORs that are open continuously, seven days per week and twenty-four hours per day. The tests examined two types of patient requests for surgery in one of two surgical operating rooms with Poisson arrival rates of $0 < \lambda_{a,b} < 2$ (with constraint of $\lambda_{a} + \lambda_{a,b} < 2$ for system stability). Each patient type has a no-cancellation option, but the two types have different waiting penalty functions (starting with an assumption of linear penalty functions), and expected revenue. The cost of waiting for Patient A is a linear function $f(t, p_a) = c + p_a * t$, where c is a constant, p_a is the per unit penalty for waiting, and t is the amount of time between request and appointment. Similarly, the cost of waiting for patient B follows $f(t, p_b) = c + p_b * t$. Without loss of generality we assume that $p_a = 1$ and $0.2 < p_b < 5$.

Assuming that a request for surgery of type A has just arrived, the only information that the model requires is the first available space in OR 1 and the first available space in OR 2. Then, given all the inputs above, the request can be directed to either the next available slot in OR1 or the next available slot in OR2 based on the following policies:

1. *Always seek the shortest wait time (also known as FCFS policy)*. This strategy allocates the patient to the next available room regardless of their type and their waiting penalty. If t_1 is the waiting time for room 1 and t_2 is waiting time for room 2, under the above assumptions both requests would follow an exponential distribution. Let t_{min} = min $(t_1;t_2)$ and thus the expected waiting cost of the system can be captured as,

$$E[waiting\ Cost] = Et_{min}[f(t_{min}; p_a)|typeA]P(typeA) + Et_{min}[f(t_{min}; p_b)|typeB]P(typeB)$$

2. *Always schedule patients with lower waiting costs to the room with longer wait times and those with higher waiting costs to the rooms with shorter wait times*. This policy prioritizes patients on the basis of their waiting cost. Let t_{min} = min $(t_1;t_2)$ and t_{max} = max $(t_1;t_2)$, then the expected waiting cost (assuming $p_a < p_b$) will be,

$$E[waiting\ Cost] = Et_{max}[f(t_{max}; p_a)|typeA]P(typeA) + Et_{min}[f(t_{min}; p_b)|typeB]P(typeB)$$

3. *Schedule lower cost cases to the rooms with longer wait times and higher cost cases to rooms with shorter wait times unless the longer-wait-time minus the shorter-wait-time is greater than some constant time T (also known as the "Threshold" policy)*. This policy enhances the flexibility of Policy 2 by adding more constraints to the priority selection and by checking for the optimal T—i.e. the T which incurs minimal cost. The expected cost of waiting incurred under this policy (assuming $p_a < p_b$) is calculated as follow:

Let t_{min} = min (t_1;t_2) and t_{max} = max (t_1;t_2) also $t_{diff} = t_{max} - t_{min}$

$$
\begin{aligned}
E[waiting\ Cost] \\
&= E[f(t_{max}; p_a) | typeA \,\&\, t_{diff}\,]\, P(typeA \,\&\, t_{diff} < T) \\
&+ E[f(t_{min}; p_a) | typeA \,\&\, t_{diff}\,]\, P(typeA \,\&\, t_{diff} > T) \\
&+ E[f(t_{min}; p_b) typeB]\, P(typeB)
\end{aligned}
$$

1. Simulation:

Three discrete-event simulations were generated for 2880 independent time units to estimate overall policy performance (profit and waiting cost) across a range of parameters ($p_a, p_b, \lambda_{a,b}$) with a range of T time values for the last policy. Simulations were run for 10 iterations and 480 time units as a warm-up. The same parameters were applied across all three policies to enable comparability of results. Table 19 summarizes all the assumptions applied in these simulations.

Table 19: Set of assumption for simulation parameters

Parameters	Value
Mean arrival rate type A (per hr)	**Exp ~0 <λ_a<2**
Mean arrival rate type B (per hr)	**Exp ~0 <λ_b<2**
Waiting penalty Coef. A	**p_a =$ 1**
Waiting penalty Coef. B	**p_b= $0.2-5**
T time period gap (under Policy 3)	**T= .2-5 periods**

Figure 13 displays the simulation results for overall profit under each policy. The three independent factors, plus an additional fourth independent factor for the third policy, are mapped against the overall profit.

Assuming that Patient-type B has higher priority than Patient-type A ($^{P_b}/_{P_a} \approx 4$), we expect an exponential reduction in profit under Policy 1 if the arrival rate of Patient-type B exceeds the arrival rate of Patient-type A while, in contrast, profit tends to improve in the same situation under Policy 2. Another finding which differentiates Policies 1 and 2 is the rate of reduction in profits as the arrival rate A and the ratio of waiting costs between the patient types is increased. Although the cost per time-unit of waiting (i.e. the penalty function coefficient) increased linearly in our tests, the overall profit under Policy 2 did not drop at the same pace. Policy 2 treats all patients according to their priority while allowing more attention to be given to urgent patients.

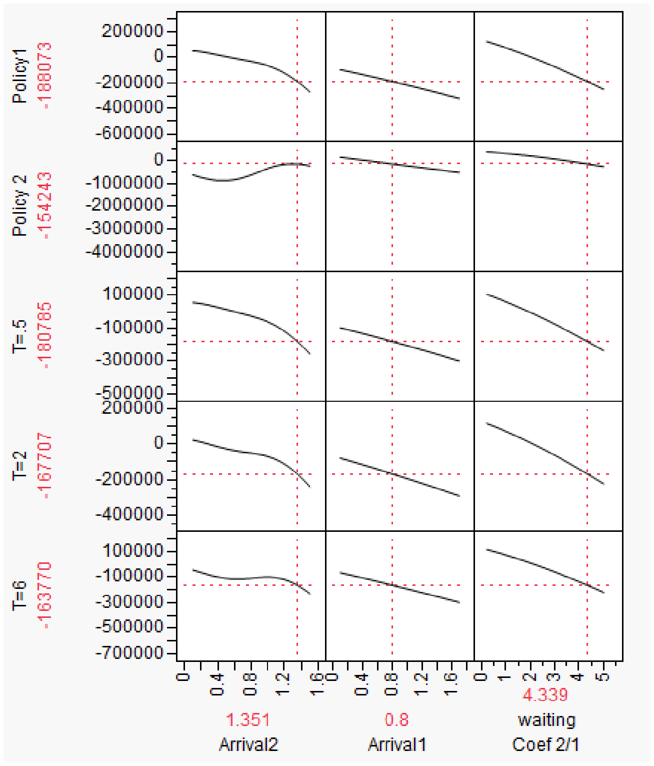
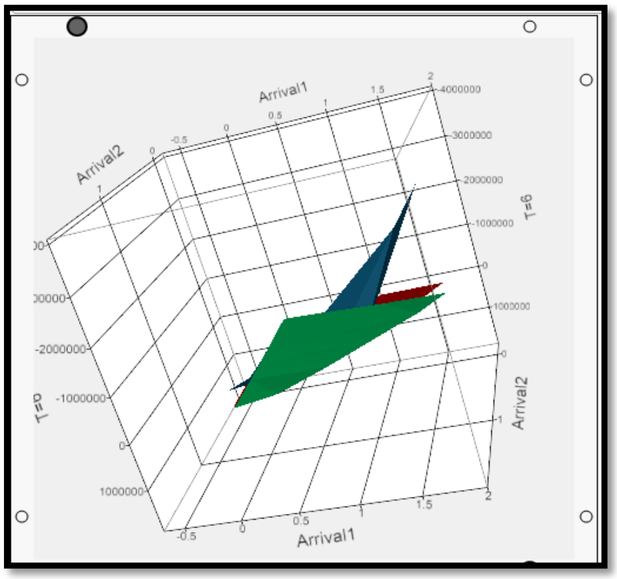
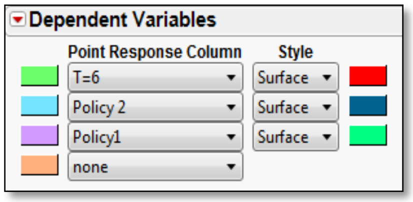


Figure 13: Profit trace of three policies across range of parameters (T=.5, 2, 6 refers to different scenarios of policy 3)

Under this policy, profit gradually increases because additional arrival rates eventually exceed incremental waiting costs, while under Policy 1, the rate of increase in waiting costs is greater than the rate of gain (profit) that can be made by scheduling additional patients (higher utilization). The last three profit traces display overall profit under three different T time threshold values under Policy 3. As T increases from 0.5 to 6 time-units, the profit trace gradually shifts from Policy 1 towards Policy 2. The T value adds more flexibility to the model with regard to the selection and combination of Policies 1 and 2.

Up to this point, we have evaluated the effects of individual policies on overall profit; the next step is to compare the policies across factors at the same time, which allows us to determine the superior policy at each state. In the discussion that follows, we use a surface plot to gain more insight into the performance of policies through a different range of parameters and to discern whether one of these policies is dominant over others in all situations. The surface plot displays three-dimensional views of the above 2-dimensional counter plot. Two separate plots were generated to capture all combinations of arrival rate and ratio of waiting penalties for the two patient types. Figure 14 provides a graphical representation of three profit surfaces displayed for each policy across arrival rates for Patient-types A and B.

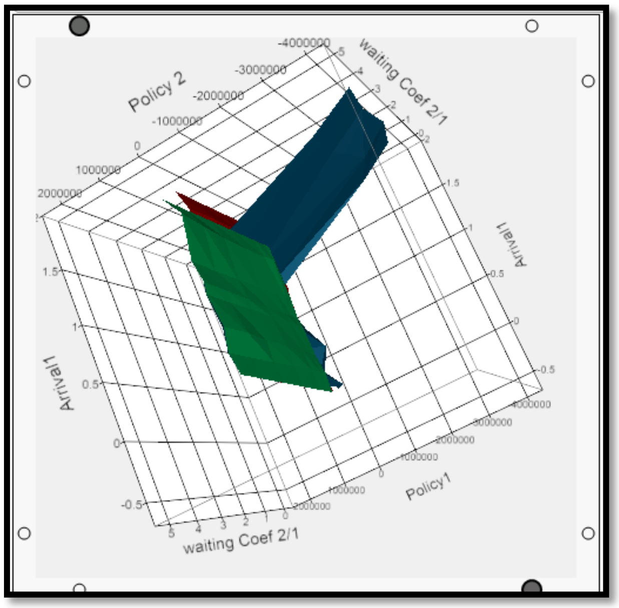


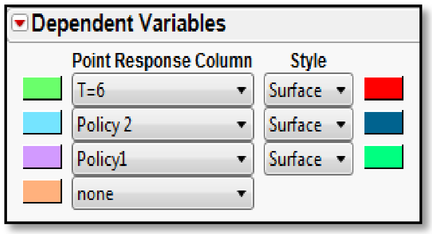

Dependent Variables

	Point Response Column	Style	
🟩	T=6	Surface	🟥
🟦	Policy 2	Surface	🟦
🟪	Policy1	Surface	🟩
🟧	none		

Figure 14: The profit surface under the three policies across arrival rates A and B as independent variables (Policy1="Green", Policy 2="Blue", Policy 3="Red")

The three surfaces show considerable differences. Overall profit under Policy 2 suddenly drops as the arrival rate of Patient-type A changes (refer to the Blue surface) in comparison with the two other policies, which generate slight decreases in profit throughout the range.
Figure 15 displays profit surfaces over all combinations of arrival rate A and the ratio of waiting penalties. Although Policy 1 has a smooth surface over the range, the surface plot of Policy 2 has multiple spikes with a significant rise across arrival rate A.





Dependent Variables

	Point Response Column	Style	
🟩	T=6	Surface	🟥
🟦	Policy 2	Surface	🟦
🟪	Policy1	Surface	🟩
🟧	none		

Figure 15: The profit surface under the three policies across arrival rate A and ratio of waiting penalty as independent variables (Policy1=”Green”, Policy 2=”Blue”, Policy 3=”Red”)

In general, these observations clearly suggest that none of these three policies dominates across all sets of parameters. In both graphs, the surfaces switch their position (optimality) over the range of parameters. But it is the set of parameters that determines what will be the predominant policy.

To identify the predominant policy under different combinations of parameters, scenario analysis was conducted for the selected problem. These results are presented in two separate tables. Table 20 lays out the superior scheduling policy for a scenario in which the two patient types arrive at the same rate. Although the arrival rate is identical in the above mentioned scenarios, the prominent policy varies over the arrival rate range due to differences in system utilization and waiting penalties. In the top two scenarios, where utilization is high, Policy 3 will be the dominant policy if the waiting penalty of one type is at least twice that of the other type; otherwise there is no dominant policy.

Table 20: Superior policy under equal arrival rates for multi-priority patients

Arrival rate $\lambda_{a,b}$	Total arrival $\lambda_a + \lambda_b$	Utilization	Best Policy	Comment
0.91	**1.82**	**91%**	$\begin{cases} Policy\ 3\ ; & \frac{P_b}{P_a} > 3 (T = 2.5) \\ No\ diff\ ; Otherwise \end{cases}$	High system utilization which results in small difference between waiting times of patient A and B
0.80	**1.60**	**80%**	$\begin{cases} Policy\ 3\ ; & \frac{P_b}{P_a} > 2\ (T = 2) \\ No\ diff\ ; Otherwise \end{cases}$	
0.45	**0.91**	**46%**	*Policy 3 (T=2)*	Medium system utilization keeps high priority patient wait time significantly lower than low priority
0.33	**0.67**	**34%**	*Policy 3 (T=1.5)*	
0.10	**0.20**	**10%**	*No difference*	Low system utilization leaves lots of free spaces for all patients

As utilization drops slightly to medium range, Policy 3 becomes dominant across all values of the waiting ratios. The T time threshold varies as utilization of the system changes. On the other hand, if utilization is low, then the system is not sensitive to the choice of scheduling policy. Table 21 shows the superior policy across a range of arrival patterns and waiting cost ratios. The shaded area refers to scenarios in which the system is not stable (e.g. the total arrival rate from both patient types is greater than the available capacity). We can simply divide the results into three sections: dominance of Policy 1 (the bottom left cells), no dominant policy (the right cells), and a combination (the diagonal cells).

In particular, if the arrival rate of low-priority patients is higher than that of higher-priority patients, then Policy 1 becomes dominant for the entire range of waiting cost ratios and arrival rates. Policy 1 does not differentiate between high- and low-priority patients in terms of reserving rooms for the potential arrival of high-priority patients. This policy is simple to apply, but, because all types of patients use the same OR resources when there are urgent patients (with high waiting costs) arriving as frequently as (or faster than) non-urgent patients, the system suffers high waiting costs from urgent patients. However, when utilization is low or low-priority patients arrive at a very low rate (λ=0.1) the choice of scheduling policies makes no difference.

Table 21: The superior policy under sets of parameters (Arrival rate, waiting penalty B/A, utilization)

		Arrival A (λ_a)				
		1.54	**1.11**	**0.80**	**0.45**	**0.10**
Arrival B (λ_b)	**1.54**	▨	▨	▨	$\begin{cases} Policy\ 1\ ; & \frac{P_b}{P_a} < 3 \\ Policy\ 3\ ; & \frac{P_b}{P_a} = 3\ (T = 1.5 \\ Policy\ 2\ ; & \frac{P_b}{P_a} > 3 \end{cases}$	***No difference***
	1.11	▨	▨	$\begin{cases} Policy\ 1\ ; & \frac{P_b}{P_a} < 2 \\ Policy\ 3\ ; & \frac{P_b}{P_a} = 2\ (T = 2) \\ Policy\ 2\ ; & \frac{P_b}{P_a} > 2 \end{cases}$	$\begin{cases} Policy\ 1\ ; & \frac{P_b}{P_a} < 3 \\ Policy\ 3\ ; & \frac{P_b}{P_a} = 3\ (T = 2) \\ Policy\ 2\ ; & \frac{P_b}{P_a} > 3 \end{cases}$	***No difference***
	0.80	▨	***Policy 1***	$\begin{cases} Policy\ 3\ ; & \frac{P_b}{P_a} > 2\ (T = 2 \\ No\ diff & ; Otherwise \end{cases}$	$\begin{cases} Policy\ 1\ ; & \frac{P_b}{P_a} < 2 \\ Policy\ 3\ ; & \frac{P_b}{P_a} = 2\ (T = 1.5 \\ Policy\ 2\ ; & \frac{P_b}{P_a} > 2 \end{cases}$	***No difference***
	0.45	***Policy 1***	***Policy 1***	$\begin{cases} Policy\ 1\ ; & \frac{P_b}{P_a} \leq 2 \\ Policy\ 2\ ; & \frac{P_b}{P_a} > 2 \end{cases}$	*Policy 3 (T=2)*	***No difference***
	0.10	***Policy 1***	***Policy 1***	***Policy 1***	***No difference***	***No difference***

Notes: All policies are based on $^{P_b}/_{P_a} > 1$ *and, in shaded areas, the system is unstable*

Under some scenarios, namely those in which the arrival rate gap between patient types is small, utilization is high and the arrival rate for high-priority patient-types B is higher than for low priority Patient-types A, the third factor; the waiting cost ratio; will define the prominent policy. As shown in Table 21, the optimal policy shifts from the Policy 1 to Policy 3 and finally to Policy 2 as penalty ratio rises. As the penalty ratio goes beyond 2 or 3, Policy 2 becomes dominant, while Policy 3 stays superior at the transition point from Policy 1 to Policy 2. Policy 3 has the same features as Policy 2, with this difference: Policy 3 allocates more attention to non-urgent waiting times as well as urgent cases. It is the T threshold that determines the dominance of Policy 1 or Policy 2 in this case: the T time factor protects non-urgent patients from long waiting times if it exceeds the T time unit threshold. Simulation results show that, if the arrival rate of urgent patients is the same as that of non-urgent patients, it is better to postpone assigning non-urgent patients to longer OR waits, but, if urgent patients come less frequently, it is better to let non-urgent patients be assigned to shorter OR waits.

Our results confirm earlier findings that not one of these policies is superior for all sets of parameters. In conclusion, the decision of which policy is optimal at each state depends on arrival rates A and B (utilization) as well as their waiting penalty. The third policy is a comprehensive policy that triggers whatever combination of Policy 1 and 2 across the range of T values would be optimal for a given condition. But, at the same time, it is the most complex strategy to apply. In general, when we don't have any information about the arrival rates of multi-priority patients or their waiting penalties, it is best first to proceed with Policy 1 since it is simple and less sensitive to parameters than other options. Ultimately, however, having more information about patient types can facilitate the choice of an optimal policy and improve patient waiting times and overall profit.

In our simulation, we applied a static policy for all states and parameters, but our results indicate that a combination of policies may be a better strategy. While the analytic tool of simulation is sufficient for analyzing the performance of each of the three different scenarios separately, it cannot yield the optimal policy for minimizing overall waiting cost in all states. To determine the optimal policy for scheduling multi-priority patients for future dates upon their arrival, given the waiting penalties and current state of the system, it is necessary to turn to mathematical programming.

2. Mathematical programming:

This section formulates the scheduling problem as a discounted infinite-horizon Markov Decision Process (MDP), which can help assign an appointment date to each patient depending on the available appointment schedule at the time of the patient's call. This section is a based on the work of Patrick and Puterman (2008) on scheduling cancer patients for radiation therapy seeking to reduce the potential impact of delays on patients. This model can be applied broadly to healthcare systems that must find optimal ways to utilize limited resources (OR hours) in providing service to multi-priority patients. The discussion below stipulates the decision epochs, state space, action sets, transition probabilities, and state-action costs for this problem.

Decision epochs. The term 'decision epoch' refers to a specific point of time in a day when the scheduler observes the state of the system and takes action. The state is determined by the number of available OR hours on each future day over an N-day booking horizon and by the number of cases in each priority class to be scheduled. The N-day booking horizon is defined as the maximum number of days in advance that a scheduler is allowed to schedule patients. Thus, at the beginning of each decision epoch, there is no appointment booked on the Nth day. For modeling convenience, we assume all appointment requests arrive at the beginning of the day, so the decisions epochs correspond to the beginning of each day.

The State Space. If patients are classified into *i* priority classes, then the state takes the following form:

$$\boldsymbol{S} = (\vec{x}, \vec{y}) = (\textstyle\sum_{i=1}^{I} \sum_{n=1}^{N} x_{in}\,; \sum_{i=1}^{I} y_i\,) \quad ; \quad x_n = \sum_{i=1}^{I} \sum_{n=1}^{N} x_{in}$$

Where x_{in} stands for the number of priority i patients filling appointments on day n and y_i stands for the number of priority i patients requesting to be scheduled. On each day, we assume that a maximum of C_n cases can be performed (C is identical to a fixed-length service period) and that

Q_i maximum number of priority i patients will arrive (Q_i is set as a large number). Therefore, the state space can be represented thus:

$S = \{(\vec{x}, \vec{y}) | x_n \leq C_n, 1 \leq n \leq N; 0 \leq y_i \leq Q_i, 1 \leq i \leq I\}$

The action set. For a given state, the model determines an optimal schedule by evaluating feasible actions at the beginning of each decision epoch. The main task for the scheduler is to assign each arrived demand to the available OR hours in the N-day booking horizon. However, if this is the only action available, then there is a high risk of waiting time going to infinity (an unstable queue) due to the limited resources for realized demand. Therefore, we assume that the scheduler is allowed to divert patients to overtime hours to avoid infinite waiting times, but, to be realistic, we set an upper limit on the number of patients that can be diverted on each day. The action set is defined as $(\vec{a}, \vec{z}) = \{ a_{in}, z_i\}$, where a_{in} is the number of priority i patients scheduled for surgery on day n and z_i is the number of diverted priority i patients. The following boundary conditions for each action ensure that the capacity constraint is not violated, that all waiting patients are booked, and that actions are forced to be positive and integers:

$$x_n + \sum_{i=1}^{I} a_{in} \leq C_n \qquad \forall n \in \{1, \dots, N\}$$

$$\sum_{i=1}^{I} z_i \leq M$$

$$\sum_{n=1}^{N} a_{in} = y_i - z_i \qquad \forall i \in \{1, .., I\}$$

Where $\mathbf{a_{in}}$ and $\mathbf{z_i}$ represents a set of nonnegative integer values and M denotes the maximum number of patients that can be diverted on a day.

Transition Probabilities. The state undergoes a transition whenever new requests for surgery take place. Given that new requests come as $\overrightarrow{y'} = (y'_1, \dots, y'_I)$, then the state will change with the probability $p(\overrightarrow{y'}) = \prod_{i=1}^{I} p(y'_i)$, where $p(y'_i)$ is the probability that y'_i priority i patients request surgery on a given day. We assume that the arrival rate of each priority patient follows Poisson distribution and is independent of others. State transition is expressed as follows:

$$(x_{i1}, x_{i2}, \dots, x_{IN}; y_1, y_2, \dots, y_I) \rightarrow (x_{i2} + \sum_{i=1}^{I} a_{i2}, \dots, x_{IN} + \sum_{i=1}^{I} a_{iN}, 0; {y'}_1, \dots, {y'}_I)$$

The Costs. Two types of costs are associated with the process of scheduling. First, delaying a surgery appointment incurs additional waiting costs per unit time for patients and a high risk of underutilized OR capacity for the system. Second, assigning patients to the first available spot increases the risk of overutilization for high priority patients and the need for surge capacity. We can formulate the cost associated with each state-action set as follows:

$$c(\vec{a}, \vec{z}) = \sum_{i,N} f(n, p_i)\, a_{i,n} + \sum_{i=1}^{I} d(i) z_i + \sum_{=1}^{N} f_n(C_n - x_n - \sum_{i=1}^{I} a_{i,n})$$

Where $f(n, p_i)$ denotes the waiting cost penalty of booking a priority i patient on day n and $d(i)$ is the penalty for diverting a priority i patient and f_n is the unit cost of underutilized capacity. The cost function explicitly balances the cost of postponing a patient surgery against the cost of diverting surgery (revenue loss). The overriding goal is to maintain reasonable wait times while optimizing utilization.

Dynamic programming (The Bellman Equation). The Markov decision process model can be resolved via dynamic programming. Dynamic programming is a method for solving complex optimization problems by breaking multi-period problems down into simpler sub-problems, as Bellman's Principle of Optimality prescribes. In dynamic programming, the value of a decision problem at a certain point in time is expressed in terms of the payoff from some initial choices and the value of the remaining decision problem from those initial choices. Based on the Bellman Equation principles, an optimal policy has the property that, whatever the initial state and value of the decisions are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision. The decision value function $v(\vec{x}, \vec{y})$ breaks a dynamic optimization problem into simpler sub-problems with discounted costs over the infinite horizon for all state-action sets. If γ is the daily discounted factor and D is the set of all possible incoming number of priority i patients, then the discounted Markov decision values function is expressed as follows:

$$v(\vec{x},\vec{y}) = \min_{(\vec{a},\vec{z})\in A_{\vec{x},\vec{y}}} \{c(\vec{a},\vec{z}) + \gamma \sum_{\overrightarrow{y'}\in D} p(\overrightarrow{y'}) v(x_2 + \sum_{i=1}^{I} a_{i2}, \dots, x_N + \sum_{i=1}^{I} a_{iN}, 0; y'_1, \dots, y'_I)\} \qquad (1)$$

This optimality equation describes the lowest possible cost, as a function of state $(\vec{x},\vec{y})$. By calculating the value function, we find the optimal action (or policy) as a function of each state. The main strength of this approach is that fairly general stochastic and nonlinear dynamics can be considered. However, the size of a state space typically grows exponentially in terms of the number of state variables. The above optimality equation suffers from the curse of dimensionality, which makes a direct solution impossible. Suppose that the maximum number of priority i appointment requests that can be possibly received on a single day is $\mathrm{Q_i}$. Then the dimension of the state space would be $\mathrm{C^N} \prod_{\mathrm{i=1}}^{\mathrm{I}} \mathrm{Q_i}$. Note that even with N=14, I=2, Q=6 and C=4 this number equals 9.66×10^9 states, and thus determining the optimal policy is not practically feasible for any realistically sized problem.

Approximate dynamic programming. One approach to dealing with this difficulty is to generate an approximation for the value function within a specific class of functions and then seek to find the optimal value function within this class. This method of solution proceeds as follows:

1. Transform the discounted Markov decision process into a linear program (refers to the relationship between MDP and linear programming.)
2. Approximate the LP value function (ALP) to reduce the dimensionality.
3. Solve the ALP to get the optimal value function.
4. Utilize the result of the optimal value function to determine the optimal state-action policies.

A fundamental result in MDP theory (Puterman, 1994) implies that solving the optimality equation is equivalent to solving the following LP for any strictly positive α which satisfies $\sum_{\vec{x},\vec{y}} \alpha = 1$ (assuming that α has a probability distribution over the initial state of the system),

$$\max_{\vec{v}} \sum_{\vec{x},\vec{y}} \alpha(\vec{x},\vec{y}) v(\vec{x},\vec{y})$$

Subject to

$$c(\vec{a},\vec{z}) + \gamma \sum_{\overrightarrow{d \in D}} [p(\overrightarrow{y'}) v\left(x_2 + \sum_{i=1}^{I} a_{i2}, \dots, x_N + \sum_{i=1}^{I} a_{iN}, 0; y'_1, \dots, y'_I)\right)] \geq v(\vec{x},\vec{y})$$

$$\forall (\vec{a},\vec{z}) \in A_{\vec{x},\vec{y}} \; and \; \vec{x},\vec{y} \in S$$

where $v(\vec{x},\vec{y})$ is a lower bound for the optimal value of the MDP, $v^*(\vec{x},\vec{y})$. Although the equations above transform the original dynamic programming to a set of simpler sub-problems, this model still suffers from dimensionality. A possible solution would be to approximate the value function with an interpretable linear value function (a linear combination of basis functions or states). Thus, we assume the following function as a starting point for approximate value function,

$$v(\vec{x},\vec{y}) = V_0 + \sum_{n=1}^{N} \sum_{i=1}^{I} V_{in} \, x_{in}$$

where V_0 is constant, V_{in} represents the marginal discounted cost of having a patient type i occupied OR hour on day n (V_{in} also depends on arrival rate λ_i).

Substituting the above value function with new linear function results in the following equation:

$$\max_{\vec{V},} \sum_{\vec{x},\vec{y}} \alpha(\vec{x},\vec{y}) (V_0 + \sum_{n=1}^{N} \sum_{i=1}^{I} V_{in} \, x_{in})$$

Subject to

$V_0 + \sum_{n=1}^{N} \sum_{i=1}^{I} V_{in}\ x_{in} - \ \gamma \sum_{\vec{d} \in D} [p(\overrightarrow{y'})(V_0 + \sum_{n=1}^{N} \sum_{i=1}^{I} V_{in}\ (x_{i,n+1} + \sum_{i=1}^{I} a_{i,n+1})] \leq c(\vec{a}, \vec{z})$; $\forall (\vec{a}, \vec{z}) \in \ A_{\vec{x},\vec{y}}\ and\ \vec{x}, \vec{y}\ \in S$

To simplify this, we can utilize the assumption of α as a probability distribution and transform the above formula into the following equation. Thus, X_{in} is random variable with respect to the probability distribution α.

$$\max_{\vec{V}} \left\{ (V_0 + \sum_{n=1}^{N} \sum_{i=1}^{I} E_\alpha[x_{in}] V_{in}) \right\}$$

Subject to

$(1-\gamma)V_0 + \sum_{n=1}^{N} \sum_{i=1}^{I} V_{in}\ (x_{in} - \gamma x_{i,n+1} - \gamma \sum_{i=1}^{I} a_{i,n+1}) \leq c(\vec{a}, \vec{z})$; $\forall (\vec{a}, \vec{z}) \in A_{\vec{x},\vec{y}}\ and\ \vec{x}, \vec{y}\ \in S$

Even when a dynamic program is transformed into a linear program, it still suffers from high dimensionality, which results in a large number of constraints. Alternatively, we can proceed with the dual of the linear programming, which gives us the advantage of a reasonable number of constraints but at the expense of creating an intractable number of variables,

$$\min_{\vec{X}} \sum_{\substack{\vec{x},\vec{y} \in S \\ (\vec{a},\vec{z}) \in A_{\vec{x},\vec{y}}}} X(\vec{x}, \vec{y}, \vec{a}, \vec{z}) c(\vec{a}, \vec{z})$$

Subject to

$$(1-\gamma) \sum_{\substack{\vec{x},\vec{y} \in S \\ (\vec{a},\vec{z}) \in A_{\vec{x},\vec{y}}}} X(\vec{x}, \vec{y}, \vec{a}, \vec{z}) = 1$$

$\sum_{\substack{\vec{x},\vec{y} \in S \\ (\vec{a},\vec{z}) \in A_{\vec{x},\vec{y}}}} X(\vec{x}, \vec{y}, \vec{a}, \vec{z})\ (x_{in} - \gamma x_{i,n+1} - \gamma \sum_{i=1}^{I} a_{i,n+1})\ \geq\ E_\alpha[X_n]$; $\forall n = 1, \dots, N \ \ \forall i = 1, \dots, I$

This problem is still too large to consider all of the variables explicitly. Patrick et al. (2008) have proposed using column generation to solve this problem by leveraging the notion that most of the variables will be non-basic with a value of zero in the optimal solution and generating only a subset of variables which have the potential to improve the objective function. Column generation can be initiated by using a small set S' of feasible state-action pairs of the dual to obtain dual prices as estimates for V_0 and $V_{in,}$and finding one or more constraints in the primal. It then adds the state-action pairs associated with these violated constraints into the set S' before resolving the dual. This process is repeated until no primal constraint is violated.

The challenge in this process is to find an initial feasible set $S'$and also a violated primal constraint. As Patrick et al. proposes, if we consider a state where no available slot exists on the N-booking day, then all incoming arrivals would be diverted as initial feasible state-action pairs. Then finding the most violated primal constraint only involves solving the following integer programming:

$$z(\vec{v}) = \min_{(\vec{x},\vec{y})\in S}[\sum_{i,N} f(n,p_i)\, a_{i,n} + \sum_{i=1}^{I} d(i)z_i + \sum\nolimits_{n=1}^{N} f_n(C - x_n - \sum\nolimits_{i=1}^{I} a_{i,n}) - \sum_{n=1}^{N}\sum_{i=1}^{I} V_{in}\ (x_{in} - \gamma x_{i,n+1} - \gamma \sum_{i=1}^{I} a_{i,n+1}) - (1-\gamma)V_0]$$

St.

$$x_n + \sum_{i=1}^{I} a_{in} \leq C_n \qquad \forall n \in \{1, \dots, N\}$$

$$\sum_{i=1}^{I} z_i \leq M$$

$$\sum_{n=1}^{N} a_{in} + z_i = y_i \qquad \forall i \in \{1, .., I\}$$

Rearranging terms yields the optimal linear value function approximation:

$$z(\vec{v}) = \min_{(\vec{x},\vec{y})\in S}[\sum_{i,N}(f(n,p_i) + \gamma V_{i,n-1} - f_n)a_{i,n} + \sum_{i=1}^{I}\big(d(i)\big)z_i + (\gamma V_{in-1} - V_{in} - f_n)x_{i,n} - (1 - \gamma)V_0]$$

St.

$$x_n + \sum_{i=1}^{I} a_{in} \leq C_n \quad \forall n \in \{1, \dots, N\}$$

$$\sum_{i=1}^{I} z_i \leq M$$

$$\sum_{n=1}^{N} a_{in} + z_i = y_i \quad \forall i \in \{1, .., I\}$$

The coefficients of each action in this equation represent the balance between costs and benefits of taking each action. For each action a_{in} we capture two costs, patients' waiting cost $f(n, p_i)$ due to possible appointment delay as well as the cost of losing available capacity for tomorrow's higher priority patients. These cost are then compared with benefits of not having an underutilized OR, f_n, and reducing additional waiting time for low priority patients, $f(n, p_i)$. Similarly, for each action z_i, there is a cost associated with diverting patient $d(i)$ against the benefits of not postponing a surgery, $f(n, p_i)$ and occupying the space for patient type i. In other words, after the best appointment action is determined, $a_{i,n}$, this cost can be compared with the cost of diverting each patient and the best overall decision can be determined. (Coefficients of x_n, show net loss or gain in each day's value)

Assuming we obtain the optimal value function $V_{in}{}^{*}$, then the next step would be to derive an optimal policy from the above approximate LP solution. We insert the optimal value function into the right-hand side of the optimality equation (1) and solve for the optimal action $(\vec{a}, \vec{z})$ in state $(\vec{x}, \vec{y})$. This involves solving the following integer programming formulation:

$$\min_{(\vec{a},\vec{z})\epsilon A_{\vec{x},\vec{y}}} \left\{ \sum_{i,N} f(n,p_i)\, a_{i,n} + \sum_{i=1}^{I} d(i) z_i + \sum\nolimits_{n=1}^{N} f_n(C - x_n - \sum\nolimits_{i=1}^{I} a_{i,n}) + \gamma \sum_{\vec{d}\in D} [p(\overrightarrow{y'})\,(V_0 + \sum_{n=1}^{N}\sum_{i=1}^{I} V_{in}\,(x_{in} - \gamma x_{i,n+1} - \gamma \sum_{i=1}^{I} a_{i,n+1}) \right\}$$

$$= \min_{(\vec{x},\vec{y})\in S} [\sum_{i,N} (f(n,p_i) + \gamma V_{i,n-1} - f_n) a_{i,n} + \sum_{i=1}^{I} ((d(i)) z_i - (1-\gamma) V_0 + (\gamma^{-1} f_n + \gamma V_{n-1} - V_n) x_n]$$

$$= \min_{(\vec{a},\vec{z})\epsilon A_{\vec{x},\vec{y}}} [\sum_{i,N} (f(n,p_i) + \gamma V_{n-1} - f_n) a_{i,n} + \sum_{i=1}^{I} ((d(i)) z_i] + constant$$

$$= \min_{(\vec{a},\vec{z})\epsilon A_{\vec{x},\vec{y}}} [\sum_{i,N} A_{in}\, a_{i,n} + \sum_{i=1}^{I} Z_i z_i] + constant$$

Thus, the optimal policy is derived from the coefficients A_{in} and Z_i:

$A_{in} = f(n,p_i) + \gamma V_{n-1} - f_n$ $\qquad\qquad$ $Z_i = d(i)$

This model only assigns patients of each priority to those days with $A_{in} \leq 0$ or when the benefit of booking a priority *i* patient on day n exceeds the cost of postponing those patients to a later time. Similarly, it only uses overtime for priority *i* patients for whom the cost of excess waiting and occupied space surpasses the costs of overtime and lost profit[4].

[4] - *The focus of this dissertation is on optimality, to gain insight from mathematical models and to guide us in developing more sophisticated rules and to evaluate rules for OR scheduling. The dimensionality of the mathematical model is very large; solving for optimal value function is beyond scope of this dissertation.*

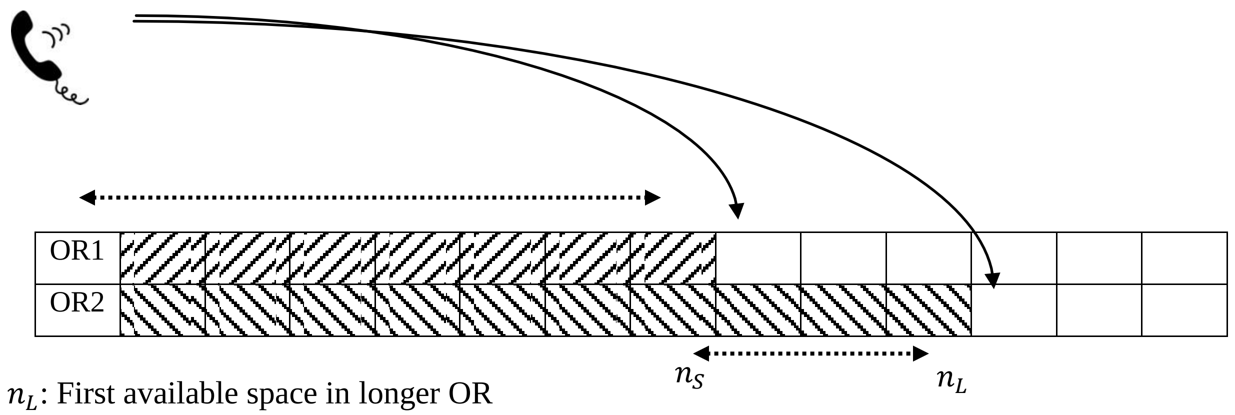
Comparing our original three simple heuristic assignment policies with the MDP optimal value function coefficients guides us to the development of more sophisticated rules (Table 22). It is clear that prior simple heuristic policies have failed to account for all the cost drivers. An optimal policy depends on three factors: waiting penalties, the marginal cost of occupied ORs on each day (which depend on the arrival rate of each patient type) and the cost of underutilization. The prior policies have only partly accounted for these factors in assignment decisions.

Table 22: Comparison of simple heuristic policies based on cost factors

Policy	Waiting penalty	Cost of occupied slot	Cost of underutilization	Arrival rate
Policy 1-FCFS	N	N	Y	N
Policy 2- Priority policy	Y	N	N	N
Policy 3- Threshold priority policy	Y	N	Y	N

Policy 1 focuses purely on maximizing utilization at the expense of high waiting costs for high priority patients while the second policy sacrifices utilization in favor of minimum waiting penalties for high priority patients even if it results in high waiting cost for low priority patients. None of these policies accounts for the cost of occupied slots and arrival rates. Policy 3 seeks a balance between the patient waiting penalty and the risk of underutilization by adding a booking threshold but still fails to address patients' arrival rates. Emphasis on waiting penalties and the cost of underutilization tends to result in the assignment of patients to the first available space while emphasis on the cost of occupied space leads to the reservation of some space for high priority patients and the postponement of surgery for lower priority patients. To include all of these factors and thereby calculate the superior policy under a range of states, we can incorporate a new policy, Policy 4, into the model.

To devise Policy 4, we can begin by finding a superior policy for the highest priority patients. All three factors support assigning high priority patients to the first available space because there is no benefit in reserving any space for future incoming patients when waiting cost increase linearly per unit of time. If we assume that just two priority-levels exist; the only necessary calculation is to examine the marginal costs and benefits of assigning each low priority patient to the next available space under a range of waiting penalties and arrival rates. Suppose Figure 16 represents the current status of a 2-OR hospital with two types of patients. All high priority patients are assigned to the first available space in any room. If a low priority patient requests an appointment, we must decide between assigning him to a shorter-wait OR (OR 1), or postponing his appointment to the next available spot in the room with the longer wait time (OR2)



n_L: First available space in longer OR
n_S: First available space in shorter OR

Figure 16: Current state of 2-OR Hospital

The following equations summarize the cost-benefit of assigning low priority patients to the shortest-wait (first available) OR,

Benefit
Lower waiting penalty ***(for low priority patient)***
$\boldsymbol{P_A} * (\boldsymbol{n_L} - \boldsymbol{n_S})$

$\boldsymbol{P_A}$: Waiting penalty of low priority patient

Cost[5]
Risk of overutilization and postponing high priority patient
$\boldsymbol{P_B * Expected\ number\ of\ spaces\ filled\ with\ high\ penalty\ patient\ in\ next\ n_S\ period}$
$\boldsymbol{P_B * \sum_{n=0}^{\infty} n * \frac{(\lambda_B * n_S)^n * e^{-(\lambda_B * n_S)}}{n!}}$

$\boldsymbol{P_B}$: Waiting penalty of high priority patient

Based on the equation above, we define Policy 4 as follows:

4. Marginal comparison: Schedule higher cost cases to shorter wait-time ORs and lower cost cases to ORs with longer wait times unless the following equation holds or unless benefits outweigh the cost of action (assuming that patient-type B has higher priority):

$$[P_A * (n_L - n_S)] > (P_B * \sum_{n=0}^{\infty} n * \frac{(\lambda_B * n_S)^n * e^{-(\lambda_B * n_S)}}{n!})$$

This policy is comparable to Policy 3 in that it establishes some threshold at which low priority patients will be scheduled in the first available space but what differentiates these two policies is that, unlike Policy 3, which has a fixed threshold, Policy 4 has a threshold which is periodically updated through time. The Threshold T is updated to reflect possible changes in the first available space and the arrival rates of high priority patients while Policy 3 treats the arrival rate as a constant.

In order to determine the effect of arrival rates and waiting penalties on this equality (and threshold), we analyzed results for the given scenarios in Table 23. We assigned low priority patients to the first available space in short wait OR, n_S, if and only if the first available spot in the OR with the longer wait time, n_L , satisfies the inequality in each cell (scenario).

[5] - *Since all surgeries are typically scheduled a few day(s) in advance, cases are not queued for empty OR time and underutilized OR time does not represent lost revenue for the surgical suite so we exclude this cost from our cost/benefit analysis*

Table 23: Variable threshold by arrival rate and waiting penalty

	Waiting penalty Range		
Arrival range; $[\lambda_A, \lambda_B]$; $\lambda_A+\lambda_B < 2$	$P_B = 2P_A$	$P_B = 3P_A$	$P_B = 5P_A$
[λ_A,0.5]	$n_L > 2n_S$	$n_L > 2.5n_S$	$n_L > 3.5n_S$
[λ_A,1]	$n_L > 3n_S$	$n_L > 4n_S$	$n_L > 6n_S$
[λ_A,1.5]	$n_L > 4n_S$	$n_L > 5.5n_S$	$n_L > 7n_S$

Assumptions:

- Waiting penalty coefficient of low priority patient $P_A = \$1$
- Since this equation is checked for every low priority case, we exclude the impact of the low priority patient arrival rate, λ_A

If a low priority patient arrives, and if his waiting penalty is half that of a high priority patient ($P_B = 2P_A$) and if the first availability of a spot in the shorter-wait OR is within the next three time periods ($n_S = 3$), he will be assigned to a shorter-wait OR if and only if the first availability of a spot in the longer-wait OR is longer than six time periods with $\lambda_B = 0.5$, or is longer than nine time periods with $\lambda_B = 1$, or is longer than twelve time periods with $\lambda_B = 1.5$. Under this policy, we tend to risk more underutilization while reserving more space under conditions where high priority patients are arriving more frequently. The same trend holds under different waiting penalties but the same arrival rate, $\lambda_B = 1$. If a low priority patient arrives, and the first availability of a spot in the shorter-wait OR is within the next three days, then Policy 4 assigns him to a shorter-wait OR if and only if the first availability in the longer-wait OR is longer than nine time periods with $P_B = 2$, or longer than twelve time period with $P_B = 3$, or longer than eighteen time periods with $P_B = 5$. Under this policy, we tend to risk more underutilization while reserving more space for high priority patients as the waiting penalty increases, assuming the same arrival rate for high priority patients.

The benefits of Policy 4 can be evaluated against Policies 1, 2 and 3 by simulating its performance across the ranges of parameters. However, without knowing the optimal policy, it is difficult to know how much better one might do with some other heuristic policies and when to stop. In these settings, it would be useful to find an upper bound on the performance of an optimal policy by assuming perfect information from the output of heuristic polices, and to compare the performance of each policy against this upper bound.

To estimate an upper bound on the profit across all patients, we assume perfect information about arrival demand in scheduling period $j \leq m$ and construct an appointment schedule which minimizes the total waiting penalty (or lower bound on waiting cost) across all patients. The proposed approach is inspired by algorithms for machine scheduling which minimize weighted tardiness. For our case, we are given the number of requests of each type and the waiting cost at any given appointment time and we want to identify the lowest possible waiting cost appointment policy. It can be formulated as a linear integer programming, with *i* the number of the request for appointment and *j* the number of the appointment slot. The input data is defined as follows:

i = number of the request for appointment, $i = 1, \ldots, n$
j = number of appointment slots, $j = 1, \ldots, m$
R_i = request time of patient *i*
A_j = start time of appointment slot *j*
P_i = penalty per time unit delay between request and appointment

$$P_i = \begin{cases} P_A\ ;\ patient\ type\ A\ penalty \\ P_B\ ;\ patient\ type\ B\ penalty \end{cases}$$

$$p_{ij} = \begin{cases} P_i(\,A_j - R_i); & R_i \leq A_j \\ \infty; & R_i > A_j \end{cases}$$

$$x_{ij} = \begin{cases} 1\ ; & Patient\ i\ is\ assign\ to\ appointment\ j \\ 0\ ; & Otherwise \end{cases}$$

Letting p_{ij} be the waiting penalty of patient *i* assigned to appointment *j*. Then, the linear integer program can be written as follows:

$$\min \sum_{i=1}^{n} \sum_{j=1}^{m} p_{ij} x_{ij}$$

St.

$$\sum_{j=1}^{m} x_{ij} = 1\,; i = 1, \dots, n$$

$$\sum_{i=1}^{n} x_{ij} \leq 1; j = 1, \dots, m$$

$$x_{ij} = \{0,1\}$$

The first equality requires that each patient is assigned exactly once and the second equality requires that each appointment time is filled by at most one patient or is un-assigned.

The most direct solution approach would be to enumerate all permutations (of patient appointment assignments) and see which one has the lowest cost. The running time for this approach would be factorial in the number of patients, $O(n!)$. Hence, this approach becomes intractable even for a limited number of patients. For our case, under perfect information in the sense that the appointment availability, time of request, type of request, and other important facts, are fully known at the time of constructing a schedule, finding a lower bound on waiting cost can be done in polynomial time. At any point of time we need to schedule the higher penalty patients in the earliest possible time slots, and then schedule the lower penalty patients in the earliest remaining time slots. The proof of optimality is provided in Appendix L using adjacent pairwise interchange. An approach for deriving a lower bound on waiting costs is presented in Algorithm 1 given below.

Visual Basic is used to generate the upper bound profit under perfect information across a range of parameters ($p_a, p_b, \lambda_{\mathrm{a,b}}$) based on the optimal schedule (the lower bound waiting cost) is proved using adjacent pairwise interchange (code is provided in Appendix K). This upper bound on profit provides a useful benchmark for evaluating the performance of other heuristics and bounds. All four discrete-event simulations were generated for 2880 independent time units to estimate overall policy performance (waiting cost) across a range of parameters ($p_a, p_b, \lambda_{\mathrm{a,b}}$) (with range of T time values for the third policy). The same set of parameters was applied across all four policies for comparability of results.

Algorithm 1. A scheme of sequential scheduling decisions algorithm

Input: Let i be the number of the request (patient Id), where $i=1\ldots n$, R_i be request time of patient i, assume i is indexed so that $R_i \leq R_{i+1}$ for $\forall\ i$;

Let j be the number of appointment slots; where $j=1\ldots m$, A_j be start time of appointment slot j, assume j is indexed so that $A_j \leq A_{j+1}$ for $\forall\ j$;

Let $v_i = [A, B]$ be the type of patient i, and $[p_A, p_B]$ be waiting penalty coefficients; where $p_A < p_B$

Output: The upper bound scheduling assignment $[x^*_{ij}]$, where x^*_{ij} is an indicator variable for whether patient i is assigned to appointment j;

```
1. Initialize:
        a. Set initial scheduling policy $[x_{ij}] = \vec{0}$
        b. Get $R_i$, request time of patients
2. Iterate while $i \leq n$, where n is the last request
3. Iterate while $j \leq m$, where m is the last appointment time
        a. $For$ $(\mathrm{j} = 1;\ \mathrm{j} \leq \ \mathrm{m}; \mathrm{j}++)$
            i. Let $C = length[\ v_i], where\ \ v_i = [B]\ and\ \ R_i < j$ and $\sum_{j=1}^{m} x_{ij}\ = 0$
            ii. If $C\ > 0$, set $x^*_{ij} = 1$; $\{i\colon \min(\ R_i)\}$
      End
   End
4. Iterate while $j \leq m$, where m is the last appointment time
      a. $For$ $(\mathrm{j} = 1;\ \mathrm{j} \leq \ \mathrm{m}; \mathrm{j}++)$
            i. If $[x_{ij}] = \vec{0}$
            ii. Let $D = length[\ v_i], where\ \ v_i = [A]\ and\ \ R_i < j$ and $\sum_{j=1}^{m} x_{ij}\ = 0$
            i. If $D\ > 0$, set $x^*_{ij} = 1$; $\{i\colon \min(\ R_i)\}$
      End
   End
   End
   Return $[x^*_{ij}]$
```

Table 24 summarizes the results of these simulations against the derived upper bound (on profit). Simulations were conducted under two different waiting penalty scenarios to evaluate the impact of waiting penalty on policy performance.

Table 24: Comparison of policy performance against upper bound policy under perfect information

$\frac{Waiting\ penalty\ B}{Waiting\ penalty\ A}$	**Arrival rate scenario**	**Simple heuristic policy**			**Marginal comparison Policy**	**Upper bound *(under PI)***
		FCFS	Priority policy	Threshold policy		
$\boldsymbol{P_B/P_A = 3}$	λ_A=1.5, λ_B=0.5	78%	9%	79%	**88%**	100%
	λ_A=1, λ_B=1	63%	70%	69%	**74%**	100%
	λ_A=0.5, λ_B=1.5	52%	55%	54%	**56%**	100%
$\boldsymbol{P_B/P_A = 5}$	λ_A=1.5, λ_B=0.5	73%	4%	73%	**81%**	100%
	λ_A=1, λ_B=1	42%	55%	54%	**60%**	100%
	λ_A=0.5, λ_B=1.5	44%	46%	44%	**46%**	100%

Note: $\lambda_A + \lambda_B < 2,\ \lambda_A and\ \lambda_B$ are rounded up to one decimal

As shown in Figure 17 all policies except Policy 2 perform well (over 75% of upper bound performance) in the case of high priority patients that come less frequent than low priority patients. Policy 2 is obviously the worst choice for this scenario as the waiting time of low priority patients grows exponentially. In general, policy performance decreases as the arrival rate of the high priority patient increases.

The same analysis was conducted for conditions where the waiting penalty gap between high and low priority patient is even higher, $P_B/P_A = 5$ (results are summarized in Figure 18). Comparing the results, we have the same performance trend as for the lower waiting penalty ratio, $P_B/P_A = 3,$ yet with lower performance in all polices for this condition. In summary, as the waiting penalty ratio increases, policy performance drops measurably across all scenarios. Although the marginal comparison policy outperforms all three simple policies across all scenarios, there is no improvement seen under the third scenario where high priority patients arrive three times faster than low priority patients. The main reason is overutilization of space by low priority patients when ORs need to be reserved for upcoming high priority patients.

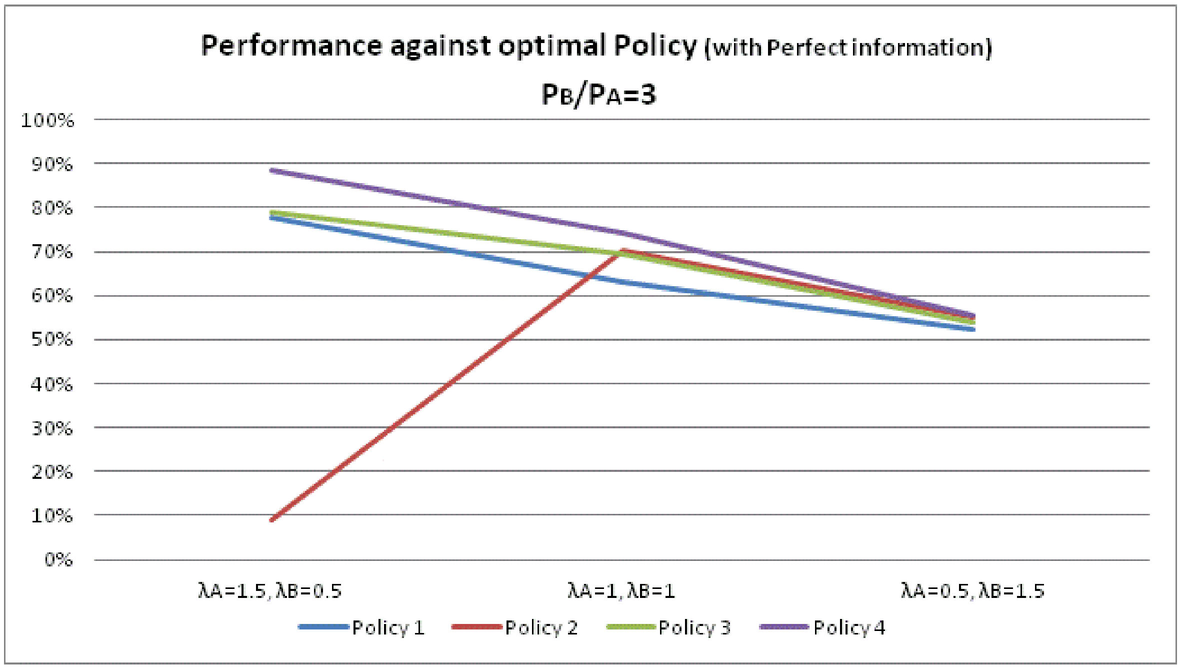


Figure 17: Policy performance against upper bound, where $^{P_B}/_{P_A} = 3$

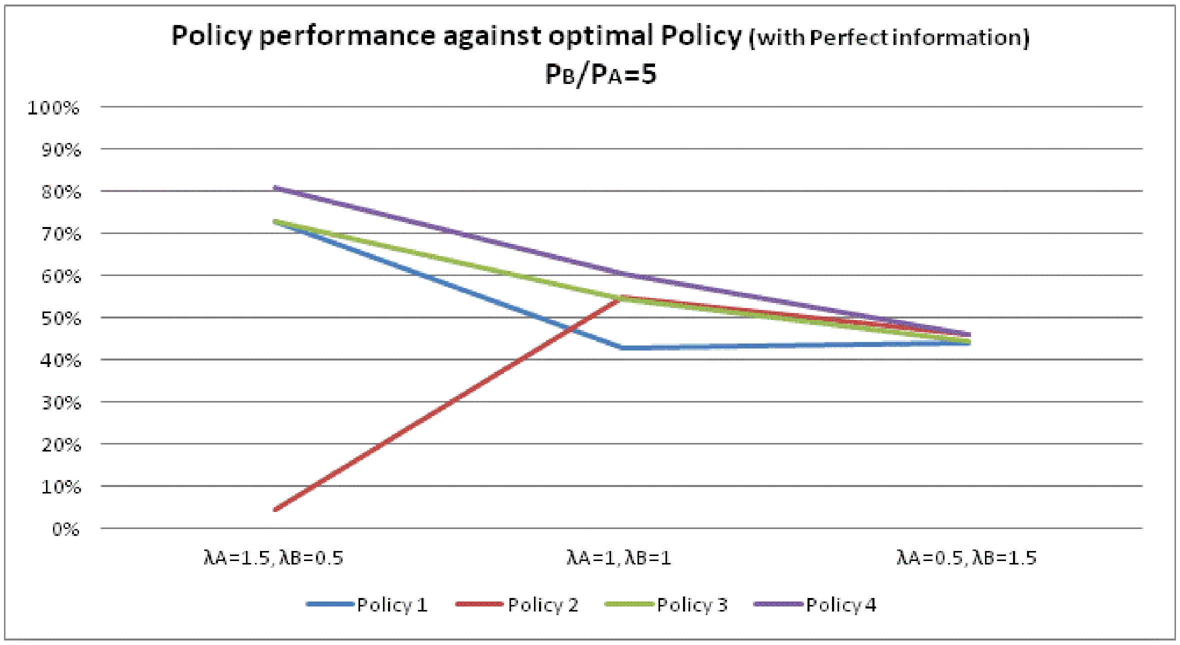


Figure 18: Policy performance in comparison against upper bound, where $^{P_B}/_{P_A} = 5$

Under all of these polices (even Policy 4) we do not postpone any appointment in favor of upcoming high priority patients. This means we can only fill the next space without retaining any space in between for upcoming high priority patients. The goal, then, is to introduce a fifth policy that can account for the arrival rates of both low- and high-priority patients as well as waiting penalties. As presented in Table 25, Policy 4 partially incorporates the cost of occupied slots as it assigns low priority patients to the next longest available space without reserving an interval for upcoming high priority patients. Policy 5 has included the full cost factors in its assigning strategy, as follows:

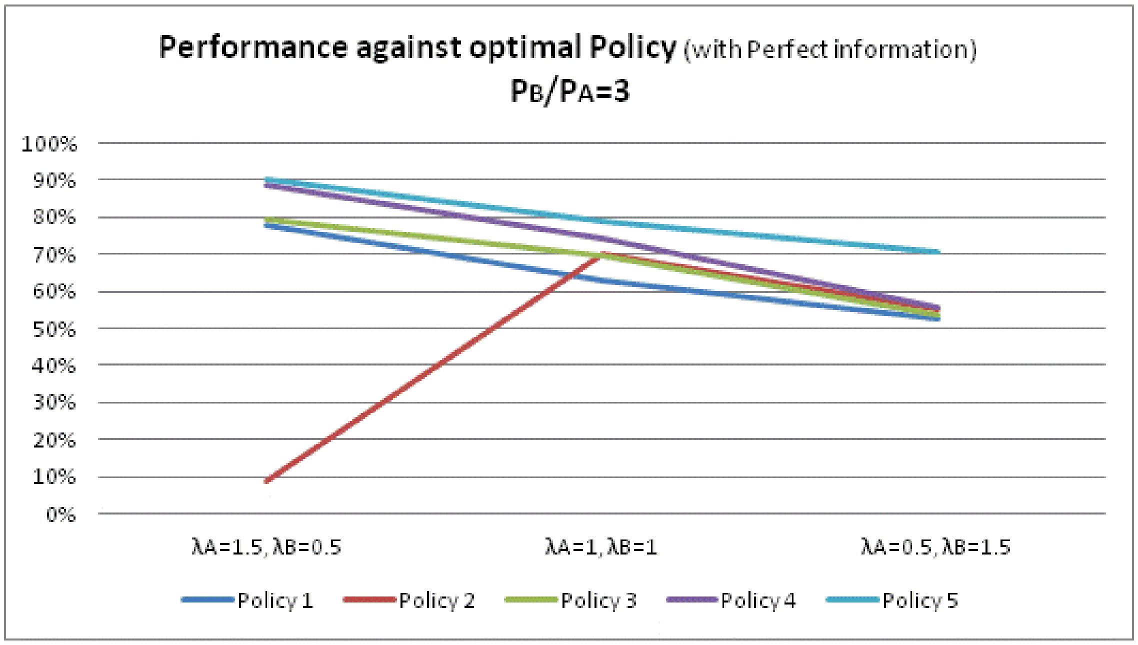
5. Reserve policy: *Schedule higher cost cases to the first available space, n_S and postpone lower cost cases to the next $\frac{\lambda_B}{\lambda_A} \times \frac{P_B}{P_A}$ time period (or further) unless the following equation holds (assuming patient-type B has higher priority),*

$$[P_A * (n_L - n_S)] > (P_B * \sum_{n=0}^{\infty} n * \frac{(\lambda_B * n_S)^n * e^{-(\lambda_B * n_S)}}{n!})$$

Table 25: A Comparison of new reserve policy and heuristic policies in cost factors

Policy	Waiting penalty	Cost of occupied slot	Arrival rate
Policy 1-FCFS	N	N	N
Policy 2- Priority policy	Y	N	N
Policy 3- Threshold priority policy	Y	N	N
Policy 4- Marginal comparison	Y	Y (*focus on low priority*)	Y
Policy 5- Reserve policy	Y	Y	Y

This policy reserves a time frame for upcoming high priority patients considering the arrival rate of both patient types while also maintaining the equation to improve the overall utilization. We conducted the same comparison for Policy 5 and the other heuristic polices, as summarized in Figure 19. We have seen improvement under Policy 5 across all scenarios, with the most improvement in scenarios where high priority patients arrive more frequently. Under this policy we were able to keep policy performance over 70% across all states (Table 26).



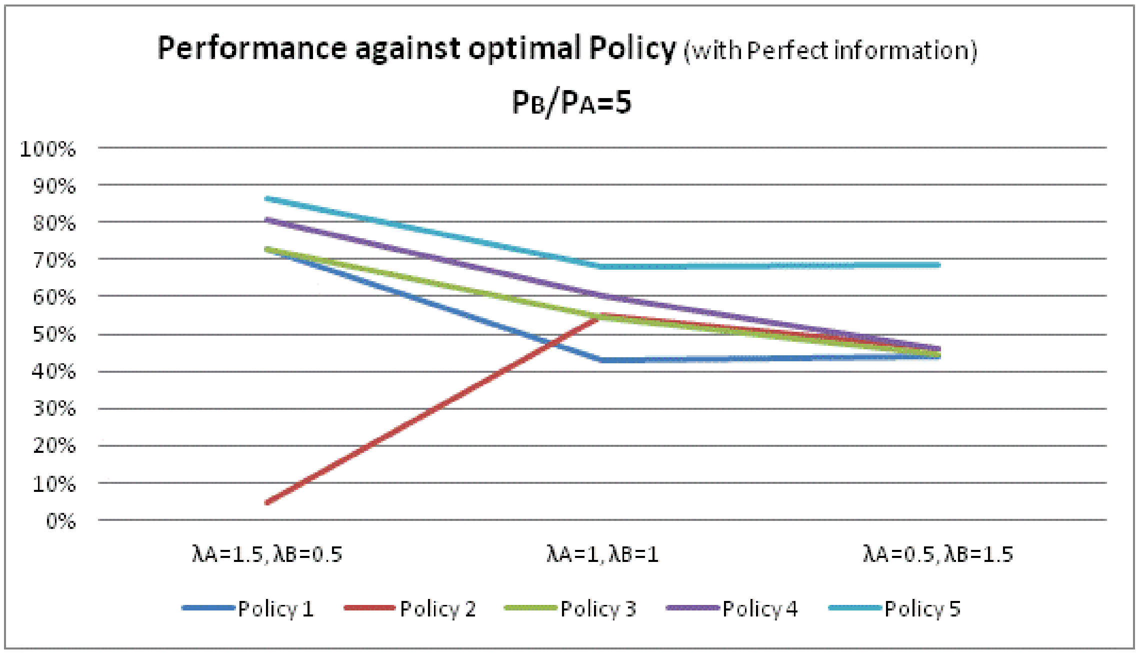


Figure 19: Policy performance in comparison with Upper bound policy, where $^{P_B}/_{P_A} = 3,5$

Table 26: A Comparison of the performance of the reserve policy and heuristic policies against upper bound under perfect information

$\frac{\boldsymbol{Waiting\ penalty}}{\boldsymbol{Waiting\ penalty}}$	**Arrival rate scenario**	**Simple heuristic policy**			**Marginal comparison Policy**	**Reserve Policy**	**Upper bound *(under PI)***
		FCFS	Priority policy	Threshold policy			
$\boldsymbol{P_B/P_A = 3}$	λ_A=1.5, λ_B=0.5	78%	9%	79%	88%	**90%**	100%
	λ_A=1, λ_B=1	63%	70%	69%	74%	**79%**	100%
	λ_A=0.5, λ_B=1.5	52%	55%	54%	56%	**71%**	100%
$\boldsymbol{P_B/P_A = 5}$	λ_A=1.5, λ_B=0.5	73%	4%	73%	81%	**86%**	100%
	λ_A=1, λ_B=1	42%	55%	54%	60%	**69%**	100%
	λ_A=0.5, λ_B=1.5	44%	46%	44%	46%	**70%**	100%

This heuristic policy is comparable to Littlewood's revenue management rule (1972) as both share the same goal to optimally allocate a finite, perishable amount of capacity among two classes of patients who arrive randomly over time. They both attain a protection level that can be used to postpone an arriving request for a lower priority customer in the hope of being able to fulfill the request of a higher priority customer, although the protection level is applied differently in these two rules. In Littlewood's rule, a supplier is looking to improve revenue by setting different prices such that the customers who are willing to pay more are not able to pay less, while the intention of our scheduling heuristic model is to improve revenue by minimizing the overall waiting penalty of patients so that the patients who have higher priority are able to be scheduled first. In addition, Littlewood's rule has multiple assumptions and practices that distinguish it from our model. The first of Littlewood's assumptions which we have relaxed is that demand comes for a particular resource at a particular time, and that, consequently, it is necessary to accept or reject the request at the time of arrival. In contrast, in our appointment model, we assume that demand from low priority patients arriving on a particular day can always be satisfied on another day with a penalty for each time unit of delay. The second assumption of Littlewood's model that we have relaxed is that demand comes in increasing fares, from the lowest to the highest fare. This may be considered natural in such contexts as the airline and hotel industries, since leisure customers usually book early to take advantage of available discounts. However, in our case an arbitrary order of arriving demands is allowed. This is what actually happens in practice, as patients with different priority levels arrive based on needs, concurrently rather than sequentially.

In the next chapter, we demonstrate some practical aspects in the application of the reserve policy in the case of scheduling under multi-priority patient classes.

Chapter 5 : Simulation/optimization for a real world case study

Simulation model of online scheduling in Stafford Hospital

In this study, we present a simulation model for decisions related to surgical scheduling at Stafford Hospital, a small hospital with a capacity of around 2300 cases per year. Approximately 80% of its cases are non-urgent or semi-urgent, and the remaining 20% are urgent cases. Because the current study focuses on the scheduling of elective surgery, the urgent cases have been excluded from analysis (for this reason, part of Stafford's OR hours are eliminated from overall service hours). At Stafford Hospital, non-urgent and semi-urgent patients are treated as elective cases with semi-urgent patients given higher priority than non-urgent cases. The daily challenge facing the scheduler is to allocate the available capacity between these two priority classes so as to minimize indirect waiting time, with greater weight given to any late bookings of semi-urgent patients.

The goal of the current study is to explore multiple scheduling policies that may simultaneously reduce patient waiting time and hospital block costs. These policies involve block hours dedicated to each surgeon or group of surgeons, release time, and priority scheduling. In order to develop the surgery simulation model used here, twelve months of data (January-December 2011) was requested from Stafford Hospital. The data included the dates patients called to make appointments, the dates cases were scheduled, information about the physicians and the surgical procedures information (e.g. the surgeons' specialties and availability), case status (cancel/reschedule), and case duration. Stafford has four operating rooms (OR1-OR4) and six major specialties: orthopedics; plastics; general/vascular; ear, nose and throat; obstetrics/gynecology; and podiatry. At most, three ORs are open Monday through Thursday, and only two are open on Fridays. Twenty surgeons operate actively in these four ORs (we have classified these twenty surgeons to eleven families of surgeons for this study). Stafford uses a modified scheduling strategy, where some blocks are assigned to individual surgeons, groups, or services and the rest are shared among all surgeons. Stafford dedicates around 58% of its service hours (244 hours every two weeks) to individuals or groups, and the remaining 42% are kept open to be shared among all. To increase access to the OR schedule for all users, Stafford employs a predefined block release policy that is calculated so that the required scheduling lead-time accommodates approximately 75% of a service's patients (Table 27).

Table 27: Predefined block release policy across specialties

Specialty	Block Released by
ENT	**7 days prior**
Podiatry	**3 days prior**
General/Vascular	**3 days prior**
OB/GYN	**5 days prior**
Orthopedics	**5 days prior**
Plastic Surgery	**7 days prior**

The hospital administrators believe that this policy maximizes access to the elective schedule for block holders while maintaining sufficient lead-time for other physicians to take advantage of underutilized operating capacity and assign their urgent cases. Under this policy, block holders can fill 75% of their time block prior to the release day and can continue assigning incoming cases to their block after the release date. So, determining the optimal release day is critical in maximizing utilization of ORs and reducing the wait times of urgent patients.

As noted in Chapter 3, each surgeon's patients have a different urgency level, which can be quantified as the maximum number of days they can wait for their surgeon's block. Table 28 is derived from the actual data. This waiting time is consistent with the urgency level perceived by patients, that is, the expectation of the patients about the longest time they are willing to wait. Blocks may be allocated on a weekly or bi-weekly basis. For low-volume surgeons, weekly blocks can be shared among multiple surgeons. Table 29 shows how eight-to-ten hour blocks were shared in a cyclic schedule for even and odd weeks in 2011 among the four ORs at Stafford. The shaded blocks indicate times when a surgical suite is closed.

Table 28: Maximum day that patients will wait to get assigned to their home block

Surgeon	A	B	C	D	E	F	G	H	I	J	K
Days	12	18	43	18	14	29	49	16	27	18	33

Table 29: Weekly Stafford OR schedule

		Week 1									
Hour		**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**
Monday	OR1	A	A	A	A	A	A	A	A	▨	▨
	OR2	B	B	B	B	FCFS	FCFS	FCFS	FCFS	FCFS	FCFS
	OR3	C	C	C	C	FCFS	FCFS	FCFS	FCFS	FCFS	FCFS
	OR4	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
Tuesday	OR1	D	D	D	D	D	D	D	D	▨	▨
	OR2	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
	OR3	E	E	E	E	E	FCFS	FCFS	FCFS	FCFS	FCFS
	OR4	F	F	F	F	F	F	F	F	FCFS	FCFS
Wednesday	OR1	K	K	K	K	K	FCFS	FCFS	FCFS	▨	▨
	OR2	G	G	G	G	G	FCFS	FCFS	FCFS	▨	▨
	OR3	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
	OR4	F	F	F	F	FCFS	FCFS	FCFS	FCFS	FCFS	FCFS
Thursday	OR1	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
	OR2	FCFS	▨	▨	▨	B	B	B	B	FCFS	FCFS
	OR3	E	E	E	E	NORA	NORA	NORA	NORA	▨	▨
	OR4	H	H	H	H	F	F	F	F	FCFS	FCFS
Friday	OR1	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
	OR2	NORA	NORA	NORA	NORA	FCFS	FCFS	FCFS	FCFS	▨	▨
	OR3	E	E	E	E	FCFS	FCFS	FCFS	FCFS	▨	▨
	OR4	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨

		Week 2									
Hour		**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**
Monday	OR1	I	I	I	I	I	I	FCFS	FCFS	▨	▨
	OR2	B	B	B	B	FCFS	FCFS	FCFS	FCFS	FCFS	FCFS
	OR3	C	C	C	C	FCFS	FCFS	FCFS	FCFS	FCFS	FCFS
	OR4	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
Tuesday	OR1	D	D	D	D	D	D	D	D	▨	▨
	OR2	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
	OR3	E	E	E	E	FCFS	FCFS	FCFS	FCFS	FCFS	FCFS
	OR4	F	F	F	F	F	F	F	F	FCFS	FCFS
Wednesday	OR1	C	C	C	C	C	FCFS	FCFS	FCFS	▨	▨
	OR2	FCFS	FCFS	FCFS	FCFS	J	J	J	J	J	J
	OR3	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
	OR4	F	F	F	F	FCFS	FCFS	FCFS	FCFS	FCFS	FCFS
Thursday	OR1	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
	OR2	FCFS	▨	▨	▨	B	B	B	B	FCFS	FCFS
	OR3	E	E	E	E	NORA	NORA	NORA	NORA	▨	▨
	OR4	H	H	H	H	F	F	F	F	FCFS	FCFS

To understand the impact of this cyclic schedule on the development of specific daily schedules, we should first identify the critical components of the scheduling system and the ways in which surgical cases flow through this system. Demand for elective surgery is generated when it is determined by a physician in a clinic or a surgeon making his rounds. Patients then call schedulers to make their appointments. Given the urgency of the patient's case as determined by the surgeon, the current appointment status and the surgeon's availability (specified in Table 31), schedulers assign each incoming request to a specific date in the future. In our model, we assume patients do not have a strong preference for the date they are offered by the scheduler, so they accept the first offer. The scheduler has three options when assigning a case: use the surgeon's own block time, use released time from other surgeons' blocks or use open hours. Although Stafford provides all surgeons equal access to open hours and released hours, it doesn't mean these hours are filled equally by all surgeons. Differences in patient urgency, patients' arrival rates, surgeons' preferences and access to block hours lead to differences in how often surgeons actually use the open hours. Table 30 indicates the percentage of surgeries performed during off-block hours across different specialties.

Table 30: Percentage of off-block hours across surgeons

Specialty	% of operation hours in open or released hours
A	37%
B	20%
C	21%
D	13%
E	10%
F	10%
G	61%
H	15%
I	0%
J	17%
K	11%

Table 31 provides information obtained from 2011 yearly data about surgeons' preferences and available days of the week to do surgery. The number "1" marks the days of the week when each individual or group of surgeons was available to conduct surgery; the "0" indicates restrictions on schedulers' options in picking dates for surgery.

Table 31: Surgeons' availability (preference) status

Surgeon	Mon	Tue	Wed	Thu	Fri
A	1	1	0	0	0
B	1	0	1	1	0
C	1	0	1	1	1
D	0	1	0	0	1
E	0	1	0	1	1
F	0	1	1	1	0
G	1	0	1	0	1
H	0	0	0	1	0
I	1	0	0	0	0
J	0	0	1	0	0
K	0	0	1	0	0

In the period leading up to the day of surgery, patients' requests accumulated but the arrival pattern varied among surgeons: some had many last minute arrivals while others' requests came well in advance. This behavior is shown in Table 32 across all surgeons. Note that many orthopedic patients made appointments less than 3 days before the day of surgery while most podiatry patients made appointments more than 10 days ahead. This behavior is due to the urgency of the cases.

Table 32: Scheduling Lead Time

Day	EN & T	GENERAL /VASCULAR	OBSTETRICS /GYNECOLOGY	ORTHOPEDICS	PLASTICS	PODIATRY
<3	12%	22%	11%	29%	16%	5%
3-5	2%	10%	8%	11%	8%	2%
5-7	6%	11%	3%	9%	8%	6%
7-10	6%	19%	15%	14%	2%	8%
>10	74%	38%	62%	37%	65%	79%

In order to analyze the statistical differences among these surgical service groups, historical data of surgery procedures was compiled and analyzed. Table 33 summarizes this statistical information as follows, starting with the number of non-add-on surgeries (or non-urgent) performed in 2011 across surgeons and specialties:

- The inter-arrival times of patients' requests followed an exponential distribution (as evaluated by JMP software). The coefficient of variation was around 1 for all individuals and groups except for one surgeon whose CoV was 3.8. (For simplicity, we assume it to be 1.)
- Surgery duration was dependent on the type of surgery and the surgeon as well as patients. However, we assume that all patients' surgery durations in the same surgical service group followed the same distribution. To take into account the cleaning time and any possible delay in surgery, surgery time was defined by actual room time for the patient (i.e Patient in/out time). The distribution derived from this data analysis is consistent with empirical studies conducted by May et al. (2000) and Spangler et al. (2004), who found a lognormal distribution for surgery time (evaluated in JMP).
- The percent of cancellations was evaluated based on the total number of cases that were canceled before the day of surgery divided by the total number of cases requested in the same period.
- Actual block hours were defined as the total hours dedicated per 2-week period to each surgeon or group of surgeons. These hours could be released to other surgeons in the case of underutilization after the release time. We have excluded the percentage of hours when surgeons perform urgent cases.
- The percent of semi-urgent patients was calculated based on the number of patients asking for the earliest available spot among the total patients who made appointments for each surgeon.
- Utilization was evaluated for each specialty based on following formula,

$$\frac{Total\ surgical\ hours\ consumed}{Total\ amount\ of\ allocated\ time - Released\ time}$$

Table 33: Descriptive statistics of Stafford data

Specialty	Surgeon/ Group	NumOfCases2011 (Non-Add-Ons)	Mean inter-arrival time (day)	Coefficient of variation (Arrival)	RoomDuration (Lognormal/hrs)	% of cancelation	ActualBlockHRs/ 2Weeks (exclude 20% urgent cases)	% of semi-urgent patients	Utilization
EAR, NOSE AND THROAT	**G**	**115**	**2.50**	**1.02**	**(4.87, .75)**	**16%**	**4**	**73%**	**85%**
GENERAL / VASCULAR	**H**	**70**	**3.20**	**0.94**	**(4.1987,.386)**	**11%**	**6**	**91%**	**48%**
	B	**80**	**3.17**	**1.08**	**(4.25,.38)**	**17%**	**12**	**79%**	**46%**
	E	**295**	**1.06**	**1.30**	**(4,.43)**	**10%**	**20**	**71%**	**71%**
OBSTETRICS/GYNECOLOGY	**F**	**250**	**1.23**	**1.31**	**(4.42,.55)**	**16%**	**28**	**69%**	**43%**
ORTHO	**I**	**35**	**5.32**	**1.85**	**(4.6,.216)**	**12%**	**5**	**94%**	**50%**
	A	**140**	**2.45**	**1.63**	**(4.737,.663)**	**12%**	**7**	**88%**	**61%**
	D	**190**	**1.50**	**3.80**	**(4.439,.531)**	**8%**	**13**	**72%**	**57%**
PLASTICS	**K**	**40**	**4.83**	**1.10**	**(5.13,.54)**	**24%**	**3**	**71%**	**46%**
PODIATRY	**C**	**210**	**1.26**	**1.12**	**(4.115,.46)**	**20%**	**12**	**73%**	**79%**
	J	**45**	**4.80**	**1.10**	**(4.294,.293)**	**9%**	**6**	**96%**	**50%**

Although all the semi-urgent patients expected the earliest available time, the median waiting time for semi-urgent patients varied across surgeons due to differences in the urgency of the cases (as shown in Table 34). Also, this expectation affected the number of times surgeons assigned their patients to released time (to make the waiting shorter for patients)

Table 34: Median Waiting Time (days)

	% of Semi-urgent	Semi-urgent	Non-urgent
A	88%	7	15
B	79%	8	19
C	73%	17	38
D	72%	8	21
E	71%	6	17
F	69%	10	27
G	73%	17	33
H	91%	7	14
I	94%	17	23
J	96%	9	49
K	71%	11	29

Estimation of waiting cost function: Logistic regression

Logistic regression is used to model the relationship between a categorical outcome and one or more explanatory variables. Logistic regression represents both groups of interest as binary variables:

First, for groups that represent characteristics (e.g., gender), the coefficient reflects the impact of independent variables(s) on the likelihood of being in a group (e.g. female).

Second, for groups that represent outcomes or events (e.g., success or failure), the coefficient represents the impact of independent variables(s) on the likelihood of the event happening (e.g. success).

As mentioned earlier, the contribution of this study is to observe, analyze and anticipate patient behavior in response to waiting time in order to maximize yield or profits from a fixed, perishable resource. Patients show distinctive behavior toward waiting time, especially between the groups who need immediate surgery and those who do not.

We employ a logistic regression model to investigate the question "What is the probability of cancellation (leaving) given an expected waiting time for surgery?" and "Does specialty have an effect on this relationship?". The influence of waiting time and urgency level (specialty) on the cancellation rate will be determined through logistic regression.

Real data is used to divide patients into two groups of semi-urgent and non-urgent based on who asks for first available time for surgery and who asks for a convenient date in the near future. Each specialty showed a different level of urgency toward surgery as shown in Table 35.

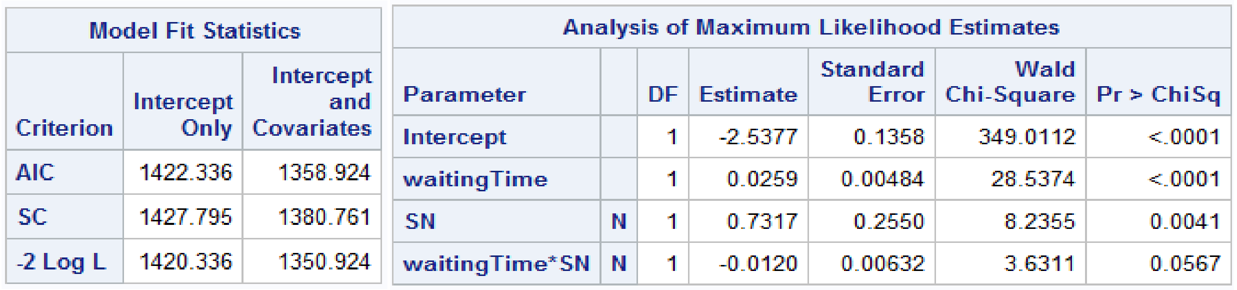
Table 35: Difference among patients behavior of specialties

	Specialties					
	Orthopedics	Plastics	General /vascular	Ear, nose and throat	Obstetrics /gynecology	Podiatry
% of semi-urgent patients	81%	71%	75%	74%	69%	76%
% of non-urgent patients	19%	29%	25%	26%	31%	24%

After running a logistic regression on a year's worth of real data (SAS software is used in logistic regression analysis, Appendix I), with waiting time and urgency of specialties[6] as independent variables and probability of cancellation as an outcome, we obtained the following results:

The Goodness of Fit test confirms that adding the independent variables waiting time and urgency level improves the fit of the model. Also, small p-values imply that the effect of waiting time, urgency level and their interaction are statistically significant in predicting the output (Figure 20). The significant interaction parameter refers to the coefficient difference between two types of patients, semi-urgent and non-urgent.

[6] - patients are divided into two types of urgency, (Urgency=1 as semi-urgent and Urgency=2 as non-urgent patients)



Model Fit Statistics		
Criterion	Intercept Only	Intercept and Covariates
AIC	1422.336	1358.924
SC	1427.795	1380.761
-2 Log L	1420.336	1350.924

Analysis of Maximum Likelihood Estimates						
Parameter		DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept		1	-2.5377	0.1358	349.0112	<.0001
waitingTime		1	0.0259	0.00484	28.5374	<.0001
SN	N	1	0.7317	0.2550	8.2355	0.0041
waitingTime*SN	N	1	-0.0120	0.00632	3.6311	0.0567

Figure 20: Goodness of fit test result

The following logic curves (Figure 21) represent the relationship between the waiting time and urgency level (explanatory variables) and the probability of leaving (dependent variable).

Probability of cancellation for non-urgent patients given their waiting time =

$$\frac{\boldsymbol{e}^{(-(2.5377-0.7317)+(0.0259-0.0120)*\boldsymbol{waitingtime})}}{\boldsymbol{e}^{(-(2.5377-0.7317)+(0.0259-0.0120)*\boldsymbol{waitingtime})}+\mathbf{1}}$$

Probability of cancellation for semi-urgent patients given their waiting time =

$$\frac{\boldsymbol{e}^{(-2.5377+0.0259*\boldsymbol{waitingtime})}}{\boldsymbol{e}^{(-2.5377+0.0259*\boldsymbol{waitingtime})}+\mathbf{1}}$$

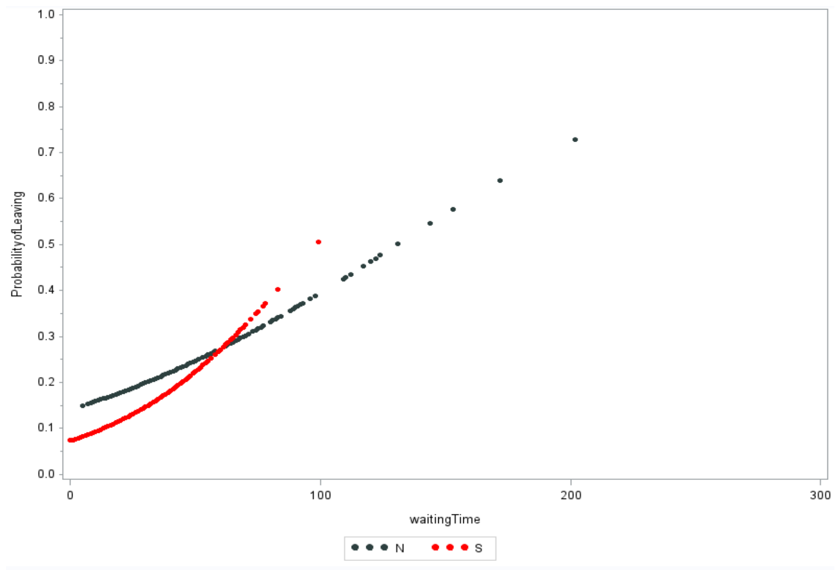


Figure 21: Probability of leaving given waiting time and urgency

As we expected, the probability of cancellation increases (logarithmic) as the waiting time increases, but this relation is not the same for both urgency levels. Semi-urgent patients' tolerance to wait is much less than non-urgent patient. The cost of waiting (sensitivity to probability of cancellation to one more day waiting for surgery) for every extra day is shown in Figure 22. As stated in the figure, semi-urgent patients are highly sensitive to an extra day of waiting (steeper slope function).

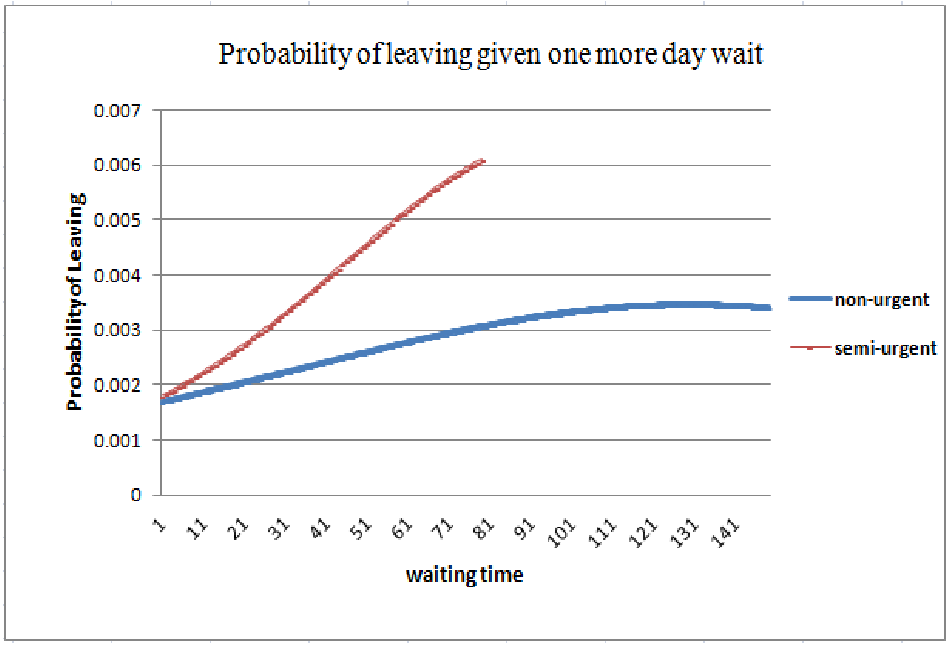


Figure 22: Probability of leaving given one more day of delay

Since non-urgent patients choose to postpone their surgery to a convenient day even though an earlier spot is available at the time of scheduling, we will not penalize these kinds of patients in our objective function and will exclude them from further analysis. We then looked at the behavior of patients among different specialties to see if there is any difference in probability of leaving given waiting time (Figure 23 and 24).

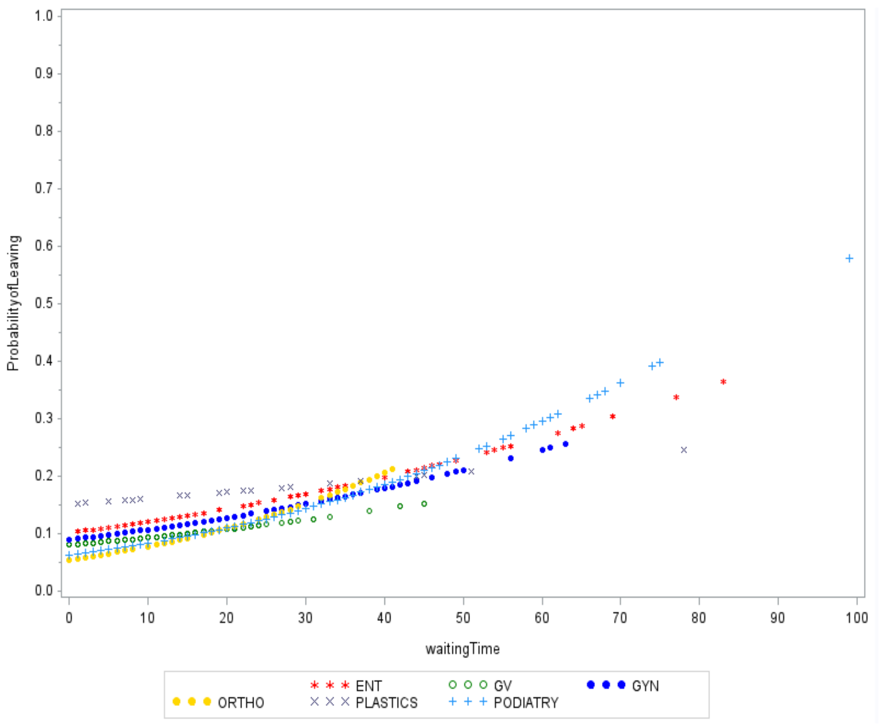
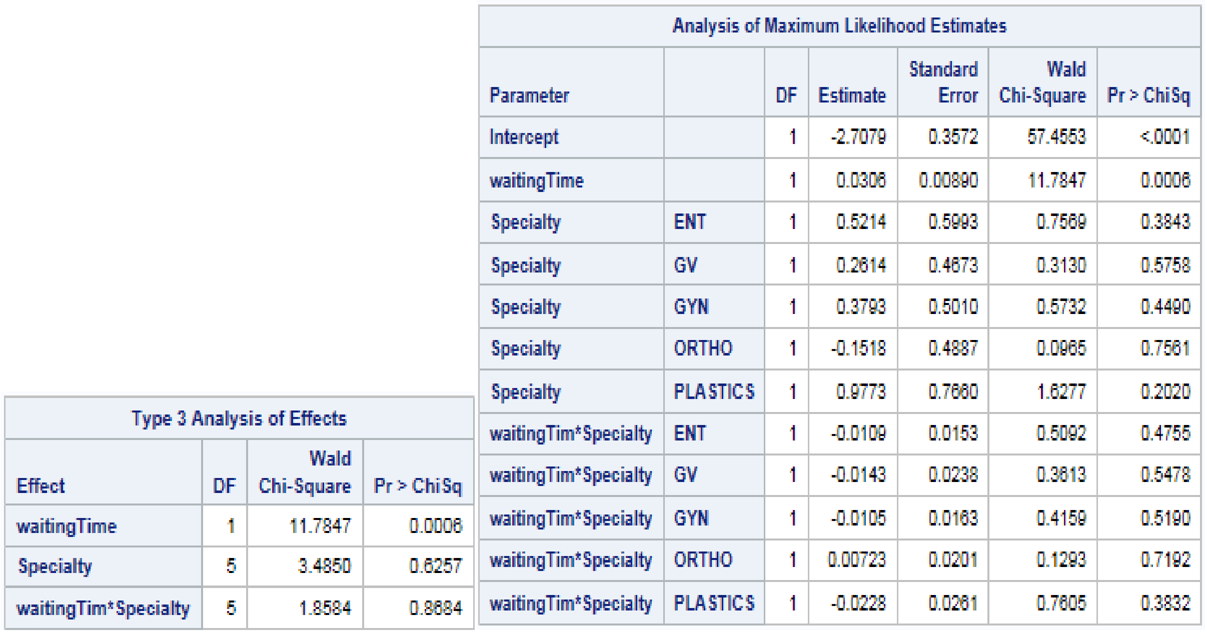


Figure 23: Probability of leaving among specialties



Type 3 Analysis of Effects			
Effect	DF	Wald Chi-Square	Pr > ChiSq
waitingTime	1	11.7847	0.0006
Specialty	5	3.4850	0.6257
waitingTim*Specialty	5	1.8584	0.8684

Analysis of Maximum Likelihood Estimates						
Parameter		DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept		1	-2.7079	0.3572	57.4553	<.0001
waitingTime		1	0.0306	0.00890	11.7847	0.0006
Specialty	ENT	1	0.5214	0.5993	0.7569	0.3843
Specialty	GV	1	0.2614	0.4673	0.3130	0.5758
Specialty	GYN	1	0.3793	0.5010	0.5732	0.4490
Specialty	ORTHO	1	-0.1518	0.4887	0.0965	0.7561
Specialty	PLASTICS	1	0.9773	0.7660	1.6277	0.2020
waitingTim*Specialty	ENT	1	-0.0109	0.0153	0.5092	0.4755
waitingTim*Specialty	GV	1	-0.0143	0.0238	0.3613	0.5478
waitingTim*Specialty	GYN	1	-0.0105	0.0163	0.4159	0.5190
waitingTim*Specialty	ORTHO	1	0.00723	0.0201	0.1293	0.7192
waitingTim*Specialty	PLASTICS	1	-0.0228	0.0261	0.7605	0.3832

Figure 24: Logistic regression Goodness of Fit test result

The result of the logistic regression (Figure 24) did not show any statistically significant difference among cancellation rates of different specialties. Although the median waiting time was distinct among different specialties, the probability of cancellation was not different among them. The same result holds for probability of cancellation among different group(s) of surgeon(s) (multiple groups form a single specialty) (Figure 25 and 26).

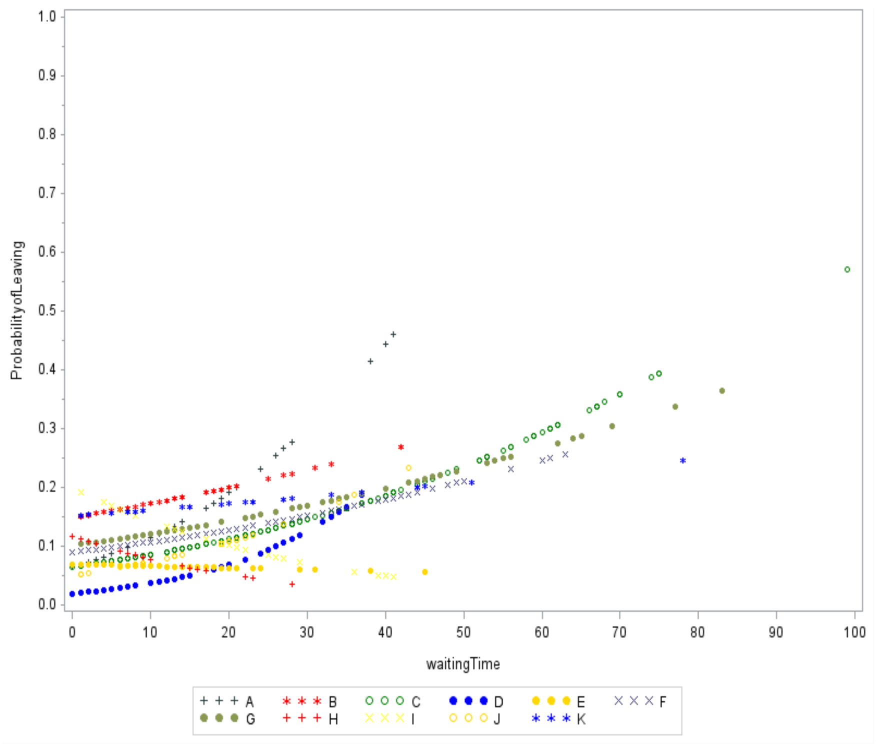
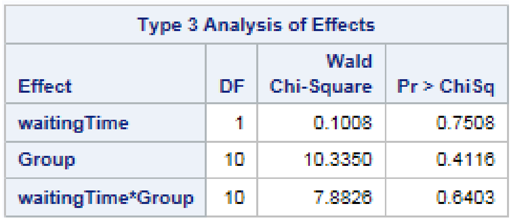
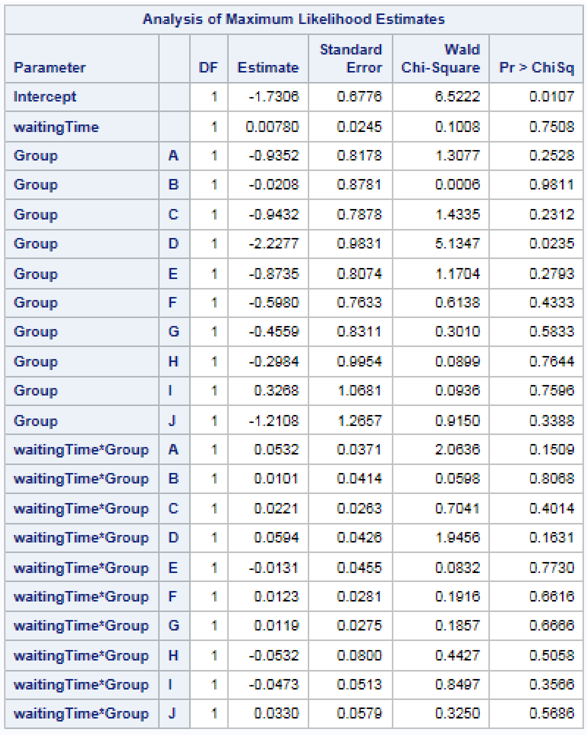


Figure 25: Probability of leaving among different group(s) of surgeon(s)

As the result suggests (shown in Figure 26), the current collection of surgeons had no significant difference in their patients' behavior. We then conducted cluster analysis to place group(s) of surgeon into new groups, or clusters, suggested by the data, so that the difference is significant (SAS code is provided in Appendix J). The result of the analysis is shown in Figure 27 and 28.



Type 3 Analysis of Effects			
Effect	DF	Wald Chi-Square	Pr > ChiSq
waitingTime	1	0.1008	0.7508
Group	10	10.3350	0.4116
waitingTime*Group	10	7.8826	0.6403



Analysis of Maximum Likelihood Estimates						
Parameter		DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept		1	-1.7306	0.6776	6.5222	0.0107
waitingTime		1	0.00780	0.0245	0.1008	0.7508
Group	A	1	-0.9352	0.8178	1.3077	0.2528
Group	B	1	-0.0208	0.8781	0.0006	0.9811
Group	C	1	-0.9432	0.7878	1.4335	0.2312
Group	D	1	-2.2277	0.9831	5.1347	0.0235
Group	E	1	-0.8735	0.8074	1.1704	0.2793
Group	F	1	-0.5980	0.7633	0.6138	0.4333
Group	G	1	-0.4559	0.8311	0.3010	0.5833
Group	H	1	-0.2984	0.9954	0.0899	0.7644
Group	I	1	0.3268	1.0681	0.0936	0.7596
Group	J	1	-1.2108	1.2657	0.9150	0.3388
waitingTime*Group	A	1	0.0532	0.0371	2.0636	0.1509
waitingTime*Group	B	1	0.0101	0.0414	0.0598	0.8068
waitingTime*Group	C	1	0.0221	0.0263	0.7041	0.4014
waitingTime*Group	D	1	0.0594	0.0426	1.9456	0.1631
waitingTime*Group	E	1	-0.0131	0.0455	0.0832	0.7730
waitingTime*Group	F	1	0.0123	0.0281	0.1916	0.6616
waitingTime*Group	G	1	0.0119	0.0275	0.1857	0.6666
waitingTime*Group	H	1	-0.0532	0.0800	0.4427	0.5058
waitingTime*Group	I	1	-0.0473	0.0513	0.8497	0.3566
waitingTime*Group	J	1	0.0330	0.0579	0.3250	0.5686

Figure 26: Logistic regression Goodness of Fit test result

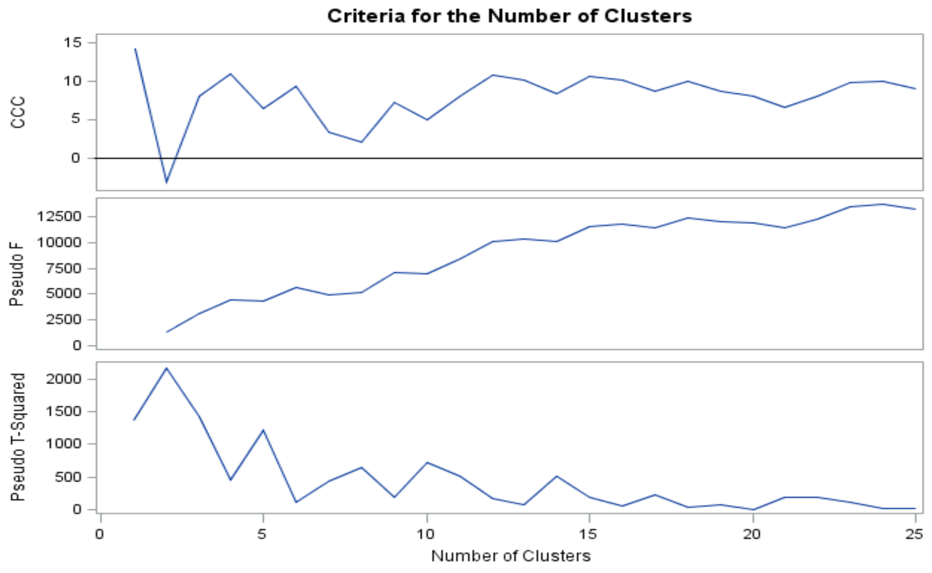
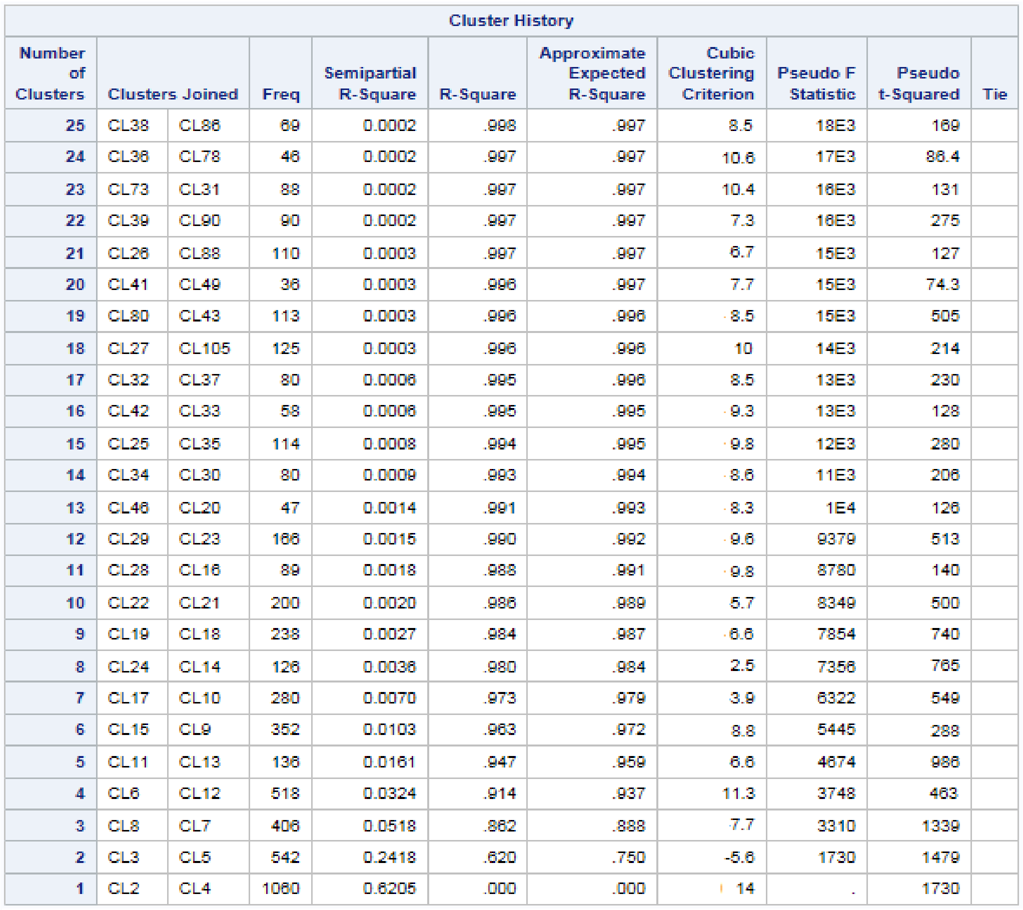


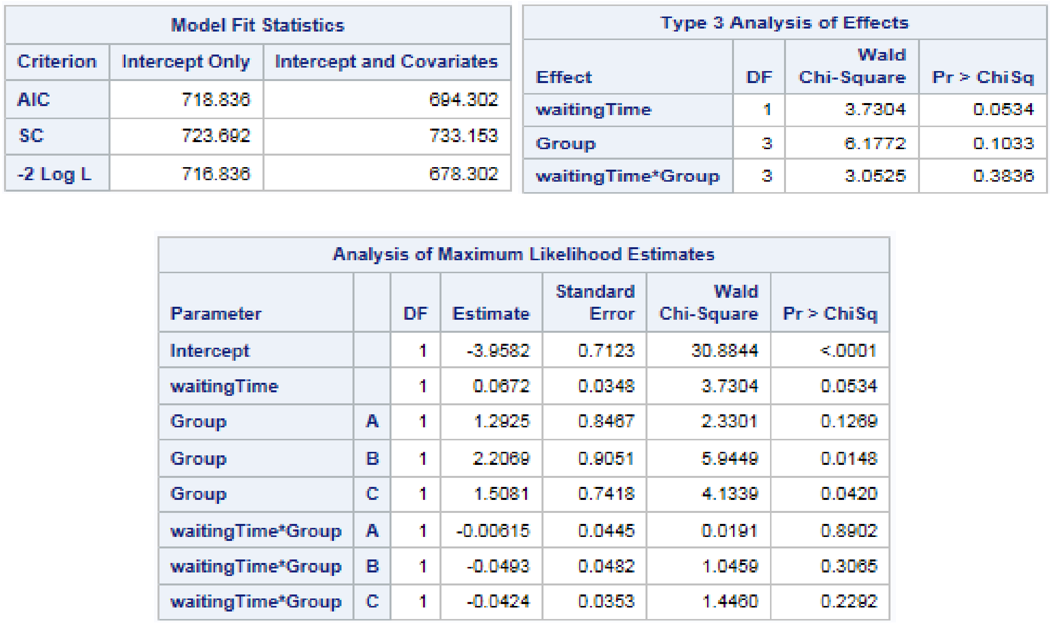
Figure 27: Pseudo-F, cubic clustering criterion (CCC), and Pseudo T-squared statistics for possible cluster solutions



Cluster History										
Number of Clusters	Clusters Joined		Freq	Semipartial R-Square	R-Square	Approximate Expected R-Square	Cubic Clustering Criterion	Pseudo F Statistic	Pseudo t-Squared	Tie
25	CL38	CL86	69	0.0002	.998	.997	8.5	18E3	169	
24	CL36	CL78	46	0.0002	.997	.997	10.6	17E3	86.4	
23	CL73	CL31	88	0.0002	.997	.997	10.4	16E3	131	
22	CL39	CL90	90	0.0002	.997	.997	7.3	16E3	275	
21	CL26	CL88	110	0.0003	.997	.997	6.7	15E3	127	
20	CL41	CL49	36	0.0003	.996	.997	7.7	15E3	74.3	
19	CL80	CL43	113	0.0003	.996	.996	8.5	15E3	505	
18	CL27	CL105	125	0.0003	.996	.996	10	14E3	214	
17	CL32	CL37	80	0.0006	.995	.996	8.5	13E3	230	
16	CL42	CL33	58	0.0006	.995	.995	9.3	13E3	128	
15	CL25	CL35	114	0.0008	.994	.995	9.8	12E3	280	
14	CL34	CL30	80	0.0009	.993	.994	8.6	11E3	206	
13	CL46	CL20	47	0.0014	.991	.993	8.3	1E4	126	
12	CL29	CL23	166	0.0015	.990	.992	9.6	9379	513	
11	CL28	CL16	89	0.0018	.988	.991	9.8	8780	140	
10	CL22	CL21	200	0.0020	.986	.989	5.7	8349	500	
9	CL19	CL18	238	0.0027	.984	.987	6.6	7854	740	
8	CL24	CL14	126	0.0036	.980	.984	2.5	7356	765	
7	CL17	CL10	280	0.0070	.973	.979	3.9	6322	549	
6	CL15	CL9	352	0.0103	.963	.972	8.8	5445	288	
5	CL11	CL13	136	0.0161	.947	.959	6.6	4674	986	
4	CL6	CL12	518	0.0324	.914	.937	11.3	3748	463	
3	CL8	CL7	406	0.0518	.862	.888	7.7	3310	1339	
2	CL3	CL5	542	0.2418	.620	.750	-5.6	1730	1479	
1	CL2	CL4	1060	0.6205	.000	.000	14	.	1730	

Figure 28: Cluster History

The result of hierarchical clustering suggested running the analysis with four or six clusters (refer to the CCC (Cubic Clustering Criteria) peaks in Figure 27). Logistic regression was run for four and six clusters. The result of six clusters is not significant and the AIC (Akaike Information Criterion) is higher in six clusters compared with 4-cluster case, where the preferred model is the one with the minimum AIC value. The result of six clusters is summarized in Appendix H. In addition, the result of logistic regression with four clusters showed significant deference among clusters. The following logic curves (Figure 30) represent the relationship between waiting time, four patient clusters (independent variables) and the probability of leaving (dependent variable).



Model Fit Statistics		
Criterion	Intercept Only	Intercept and Covariates
AIC	718.836	694.302
SC	723.692	733.153
-2 Log L	716.836	678.302

Type 3 Analysis of Effects			
Effect	DF	Wald Chi-Square	Pr > ChiSq
waitingTime	1	3.7304	0.0534
Group	3	6.1772	0.1033
waitingTime*Group	3	3.0526	0.3836

Analysis of Maximum Likelihood Estimates						
Parameter		DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept		1	-3.9582	0.7123	30.8844	<.0001
waitingTime		1	0.0672	0.0348	3.7304	0.0534
Group	A	1	1.2925	0.8467	2.3301	0.1269
Group	B	1	2.2069	0.9051	5.9449	0.0148
Group	C	1	1.5081	0.7418	4.1339	0.0420
waitingTime*Group	A	1	-0.00615	0.0445	0.0191	0.8902
waitingTime*Group	B	1	-0.0493	0.0482	1.0459	0.3065
waitingTime*Group	C	1	-0.0424	0.0353	1.4460	0.2292

Figure 29: Goodness of fit test result

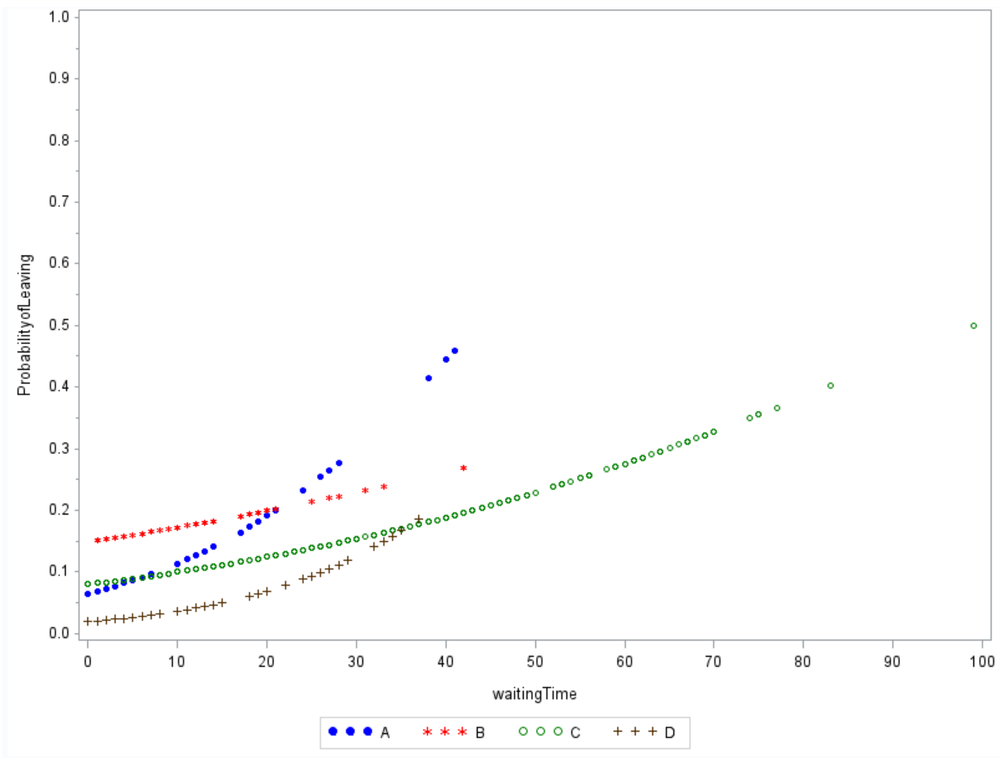


Figure 30: Probability of leaving among four selected clusters

In the following, we have modeled the logistic function for probability of cancellation as a function of waiting times for these four classes of patients. This result will be used in the Stafford simulations.

Probability of cancellation for (semi-urgent) type A patients given their waiting time =

$$\frac{e^{(-(3.958-1.2925)+(0.0672-0.00615)*waitingtime)}}{e^{(-(3.958-1.2925)+(0.0672-0.00615)*waitingtime)}+1} = \frac{e^{(-2.665+(0.061*waitingtime))}}{e^{(-2.665+(0.061*waitingtime))}+1}$$

Probability of cancellation for (semi-urgent) type B patients given their waiting time =

$$\frac{e^{(-(3.958-2.207)+(0.0672-0.0493)*waitingtime)}}{e^{(-(3.958-2.207)+(0.0672-0.0493)*waitingtime)}+1} = \frac{e^{(-1.751+(0.0179*waitingtime))}}{e^{(-1.751+(0.0179*waitingtime))}+1}$$

Probability of cancellation for (semi-urgent) type C patients given their waiting time =

$$\frac{e^{(-(3.958-1.508)+(0.0672-0.0424)*waitingtime)}}{e^{(-(3.958-1.508)+(0.0672-0.0424)*waitingtime)}+1} = \frac{e^{(-2.45+(0.0248*waitingtime))}}{e^{(-2.45+(0.0248*waitingtime))}+1}$$

Probability of cancellation for (semi-urgent) type D patients given their waiting time =

$$\frac{e^{(-(3.958)+(0.0672)*waitingtime)}}{e^{(-(3.958)+(0.0672)*waitingtime)}+1}$$

The model described above was implemented using the ExtendSim simulation environment. Once the simulation was constructed and implemented, it was evaluated to ensure that it adequately represented the actual system.

In our case study in Chapter 3, we concluded that it is advantageous to consider the joint impact of block allocation decisions and block release policies on increasing hospital profits and reducing patient waiting times. Applying the abstract model, with at most 10 factors in the third model, took 24 hours to run the full factorial analysis. The Stafford simulation model consists of many more input factors (32 factors, 21 block allocation factors due to multiple shifts, and 11 block release factors, all in three levels), which makes for a sizable three-factorial model at 3^{32}. Although using a 3^k factorial design allows us to estimate quadratic effects with more design points, it would make the analysis prohibitive or impractical in terms of computational time for full-factorial analysis as k increases. Therefore, factors are usually studied at only two levels, but even at 2 points, full factorial design requires experimentation at all factor combinations. The most common screening method is a fractional factorial design selecting a subset (fraction) of the experimental runs. Furthermore, residual analysis and ANOVA can be used to check the adequacy of the model and to detect the important effects.

Dimension Reduction

Design of experiment in simulation/optimization:

A factorial design is the most common way to study the sensitivity of response to levels of each independent variable and combined with all levels of the other independent variables. Factorial experimental designs investigate the effects of many different factors by varying them simultaneously instead of changing only one factor at a time. Computer simulation models that represent a real-world system generally consist of a large number of input factors and, due to their size and running time, large-scale simulation models can become prohibitively costly and require time-consuming experimental designs to study their behavior.

Multiple methods are introduced to reduce the dimensionality through determining the factors that have significant impact on performance measures (responses) of interest. What makes it a truly daunting task is considering the impact of interaction of model factors in eliminating non-effective factors since an approach of changing one factor at a time is a misleading strategy. The challenge is to determine which factors have the greatest effect on the responses, and to do so with the least amount of simulating. This sensitivity analysis proceeded in two steps:

1. A screening experiment to determine the main drivers

2. A response surface experiment to determine the shape of the effects (linear or curved)

Factor screening experiments are intended to examine all or some of the involved factors to identify those with significant effect on a selected response (output). The identified important factors can then be used in subsequent analyses. Many screening designs have been developed to identify important factors with an economical number of design points and replications.

- Group-screening methods have been widely used for situations with large numbers of factors. The fundamental idea is to identify the important/unimportant factors as a group (Lewis and Dean, 2001). If a group is considered to be important, then subgroups or individual factors within the group should be further screened in a series of steps; otherwise the whole group can be eliminated from further analysis. It is necessary that the factors which are grouped together have the same sign to avoid cancellation in a group (Trocine and Malone, 2001; Dean and Lewis, 2005).
- Factorial and Fractional Factorial (FF) designs are generally considered as the classic factor screening method with different resolutions for different levels of complexity of the response. Fractional factorial designs yield polynomial equations approximating the true response function, with better approximations from higher resolution level designs. These designs can be augmented to incorporate quadratic terms into the metamodel by using Central Composite Designs (CCD) (Yaesoubi, 2006).

The fundamental assumption in fractional factorial design is that certain higher-order interactions are negligible, so information on the main effects and low-order interactions can be obtained by running only a fraction of the complete factorial experiment. The number of required runs in a fractional factorial experiment is much smaller, but the ability to estimate interaction effects is also reduced. Clearly, fractional factorial designs are more efficient than factorial designs, but it is more complicated to appropriately design a fractional factorial.

Investigating factors at many levels may result in a very expensive design. Using a 3^k factorial design lets us estimate quadratic effects, but it requires more design points, especially when k is large, therefore, factors are usually studied at only two levels. A factorial design where all factors are at two levels is called a 2^k factorial design, which is one of the most widely used screening methods in simulation. However, examining each factor at only two levels (the low and high values) does not reveal how the simulation output behaves for factor combinations in the interior of the experimental region. Moreover, it is possible that the choice of low and high level for factors cancels the interaction (Trocine et al., 2000). In practice, a 2^k design can be used to fit a first-order model, and if the model exhibits lack of fit, axial runs are then added to allow the quadratic terms to be incorporated into the model (Montgomery, 2000). Two-level factorial designs assume linearity in the factor effects. Of course, perfect linearity is unnecessary, and the 2^k system will work quite well even when the linearity assumption holds only approximately. However, it is noted that if the interaction terms are added to the main effects or first-order model, then we have a model capable of representing some curvature in the response function (Montgomery, 2000).

Central Composite Designs (CCD) are the most popular class of designs used for fitting a second-order model by using middle levels or center points; however, they also increase the number of required runs. What makes 2^{k-p} fractional factorial designs attractive in factor screening experiments is the efficient number of runs it requires, which is a direct result of effect, i.e. when more effects are confounded, fewer parameters need to be estimated and as a result fewer runs will be needed.

One of the major concerns with fractional factorial designs is that this design may confound a significant interaction effect with other effects; and therefore no information can be gained about the individual interaction effects within this confounded structure. The issue of confounding introduces the concept of resolution of a design. A design's resolution determines the complexity of metamodels that can be fit to the data if the design is used. The following designs are of particular interest in fractional factorial experiments, especially in simulation.

Resolution III designs focus on just finding important main effects; however main effects are confounded with two-factor interactions and two-factor interactions may be confounded with each other. Plackett-Burman designs are well-known in estimating the main effects of k factors in only k+1 runs, when k+1 is divisible by 4.

Resolution IV designs focus on finding main effects and selected 2-way interaction effects. No main effect is confounded with any other main effect or two-factor interaction, but two-factor interactions are confounded with each other.

Finally, Resolution V designs can estimate main effects and all 2-way interaction effects. No main effect or two-factor interaction is confounded with any other main effect or two-factor interaction, but two-factor interactions are confounded with three-factor interactions.

In general, the higher the resolution, the less restrictive the assumptions that are required regarding which interactions are negligible to obtain a unique interpretation of the results. It is more desirable to conduct a Resolution V experiment to be able to estimate separately all the two-way interactions. However, for a large number of factors, it may not be feasible to perform the Resolution V design.

Once a screening experiment has been performed and the important factors determined, the next step is often to perform a response surface experiment to produce a prediction model to determine curvature, detect interactions among the factors, and optimize the process (Telford, 2007). Fitting response surface is a simple and widely applicable approach to in the context of simulation modeling, whereas commonly used methods based on classical statistics (i.e., ANOVA) make unrealistic assumptions such as constant variances and normally distributed residuals. Response surface designs are useful for modeling a curved quadratic surface to continuous factors. A response surface model can pinpoint a minimum or maximum response, if one exists inside the design region. Three distinct values for each factor are necessary to fit a quadratic function, so the standard two-level designs cannot fit curved surfaces. As explained before, central composite designs resolve this issue with combining a two-level fractional factorial and two other kinds of points (Figure 31):

- Center points, for which all the factor values are at the zero (or midrange) value
- Axial points, for which all but one factor are set at zero (midrange) and that one factor is set as outer (axial) values (JMP, 2012)

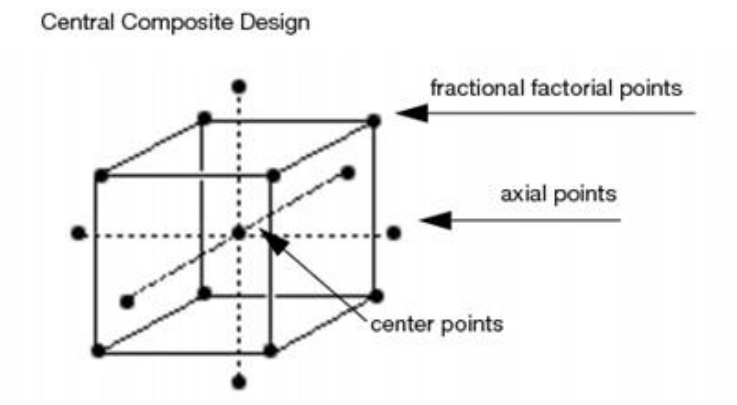


Figure 31: Central composite design

The response prediction profiler can be used to get a close look at the response surface and interactively change variables and look at the effects on the predicted response. The profiler tool explores the prediction equation to answer a number of questions such as, what type of curvature does the response surface have, or what are the predicted values at the corners of the factor space.

JMP software, statistical software developed by SAS Institute Inc., was used to generate a 2^k fractional factorial design of resolution IV. To estimate the main and selected 2-way interaction effects of 32 continuous factors under resolution IV, 64 runs were created. Simulations were run for 4800 time periods under 64 runs. To reduce the variation among the observations, we have generated 10 replications for each run (the inter-arrival time of patient requests for surgery and procedure duration times are both stochastic, following exponential and lognormal distributions, respectively) and threaded the batch mean as one final observation. Prior to performing factorial analysis and model fitting, all factors were internally recoded to -1, 0 and 1 instead of their original units where 0 is in the center of the design, and ±1 are the distance from the center with direction (refer to Figure 32). The relationship between the natural variables, the block schedule and release day variables, and the coded variables is:

$$Type\ 1 = \frac{Block\ schedule\ A - (Block\ schedule\ A_{low} + Block\ schedule\ A_{high})/2}{(Block\ schedule\ A_{high} - Block\ schedule\ A_{low})/2}$$

$$Type\ 2M = \frac{Block\ schedule\ BMonday - (Block\ schedule\ BM_{low} + Block\ schedule\ BM_{high})/2}{(Block\ schedule\ BM_{high} - Block\ schedule\ BM_{low})/2}$$

$$Type\ 2Th = \frac{Block\ schedule\ BThursday - (Block\ schedule\ BTh_{low} + Block\ schedule\ BTh_{high})/2}{(Block\ schedule\ BTh_{high} - Block\ schedule\ BTh_{low})/2}$$

$$\vdots$$

$$Type\ 0Fri = \frac{Block\ schedule\ OpenFriday - (Block\ schedule\ OFri_{low} + Block\ schedule\ OFri_{high})/2}{(Block\ schedule\ OFri_{high} - Block\ schedule\ OFri_{low})/2}$$

And, the same transformation for block release policies is given as,

$$Release\ 1 = \frac{Maximum\ time\ Type\ A\ can\ wait\ for\ Block\ A - (Maximum\ time\ Type\ A_{\ low} + Maximum\ time\ Type\ A_{\ high})/2}{(Maximum\ time\ Type\ A_{\ high} - Maximum\ time\ Type\ A_{\ low})/2}$$

$$\vdots$$

$$Release\ 11 = \frac{Maximum\ time\ Type\ K\ can\ wait\ for\ Block\ K - (Maximum\ time\ Type\ K_{\ low} + Maximum\ time\ Type\ K_{\ high})/2}{(Maximum\ time\ Type\ K_{\ high} - Maximum\ time\ Type\ K_{\ low})/2}$$

That is, they are made dimensionless, measuring the effect of changing each design factor over a one-unit interval regardless of their original metric of factor settings.

Doing this allowed us to test the linear and quadratic components in the relationship between the factors and the dependent variables. Furthermore, coded factors were all estimated with the same precision. The design is orthogonal and the coded variables are also orthogonal. In this study, the values for block release and block hours were recorded on very different scales. Since the metric for these two types of factor is no longer compatible, the magnitudes of the regression coefficients are not compatible either and multicollinearity is unpreventable (Alexander M. T., 1999).

Block Schedule Factors (hr)	Low	High
Block schedule A	13	19
Block schedule BM	6	10
Block schedule Bthu	13	19
Block schedule CM	6	10
Block schedule CWed	6	10
Block schedule D	12	18
Block schedule ETue	6	10
Block schedule EFri	6	10
Block schedule FTue	8	12
Block schedule FThu	13	19
Block schedule G	6	10
Block schedule H	6	10
Block schedule I	6	10
Block schedule J	15	19
Block schedule K	5	9
Block schedule OpenMwk1	14	18
Block schedule OpenMwk2	14	18
Block schedule OpenTue	15	19
Block schedule OpenWed	14	18
Block schedule OpenThu	8	12
Block schedule OpenFri	14	18

Block Release Factors (hr)	Low	High
Maximum Time Type A	32	64
Maximum Time Type B	16	48
Maximum Time Type C	32	64
Maximum Time Type D	32	64
Maximum Time Type E	16	48
Maximum Time Type F	32	64
Maximum Time Type G	48	80
Maximum Time Type H	16	48
Maximum Time Type I	32	64
Maximum Time Type J	16	48
Maximum Time Type K	48	80

Figure 32: Experimental range (-1, +1) across block schedule and release hour factors

The plot of the response distribution (overall profit) is shown in Figure 33, where all the response values are at a reasonable range.

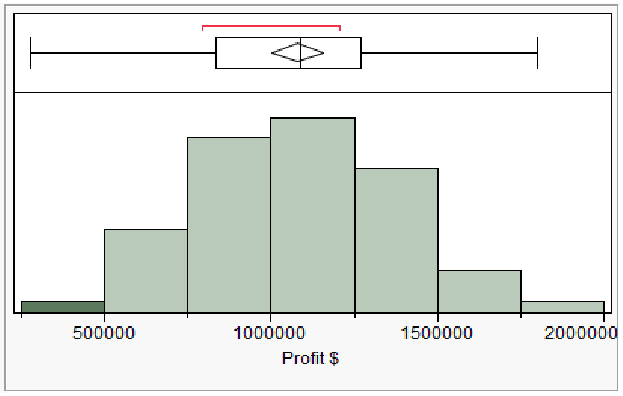
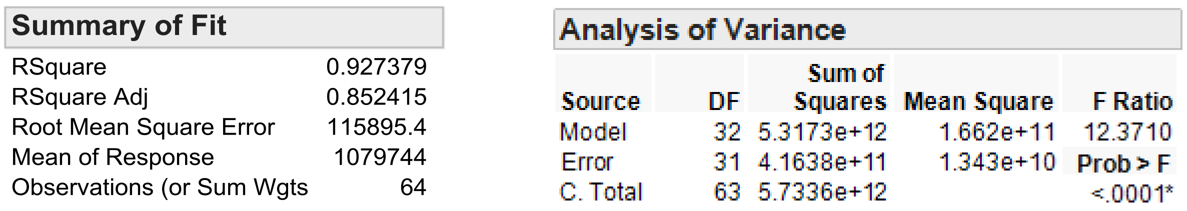


Figure 33: Response distribution of Stafford simulation

Analysis of variance, summary of fit for the 2^k fractional factorial design, and summary of screening design are shown in Figure 34 and 35. The model accounts for 92% of the variation in the data. Reduction in adjusted R^2 clearly shows we have to reduce extra independent factors from further analysis. In the screening design, main and interaction factors with the greatest effect on the response are identified.

The plot of actual versus predicted responses and the normal probability plot for residuals are depicted in Figure 36. Both of these plots indicate that the normal distribution assumption for residuals is reasonable, and there is no significant evidence to suggest the violation of this assumption.



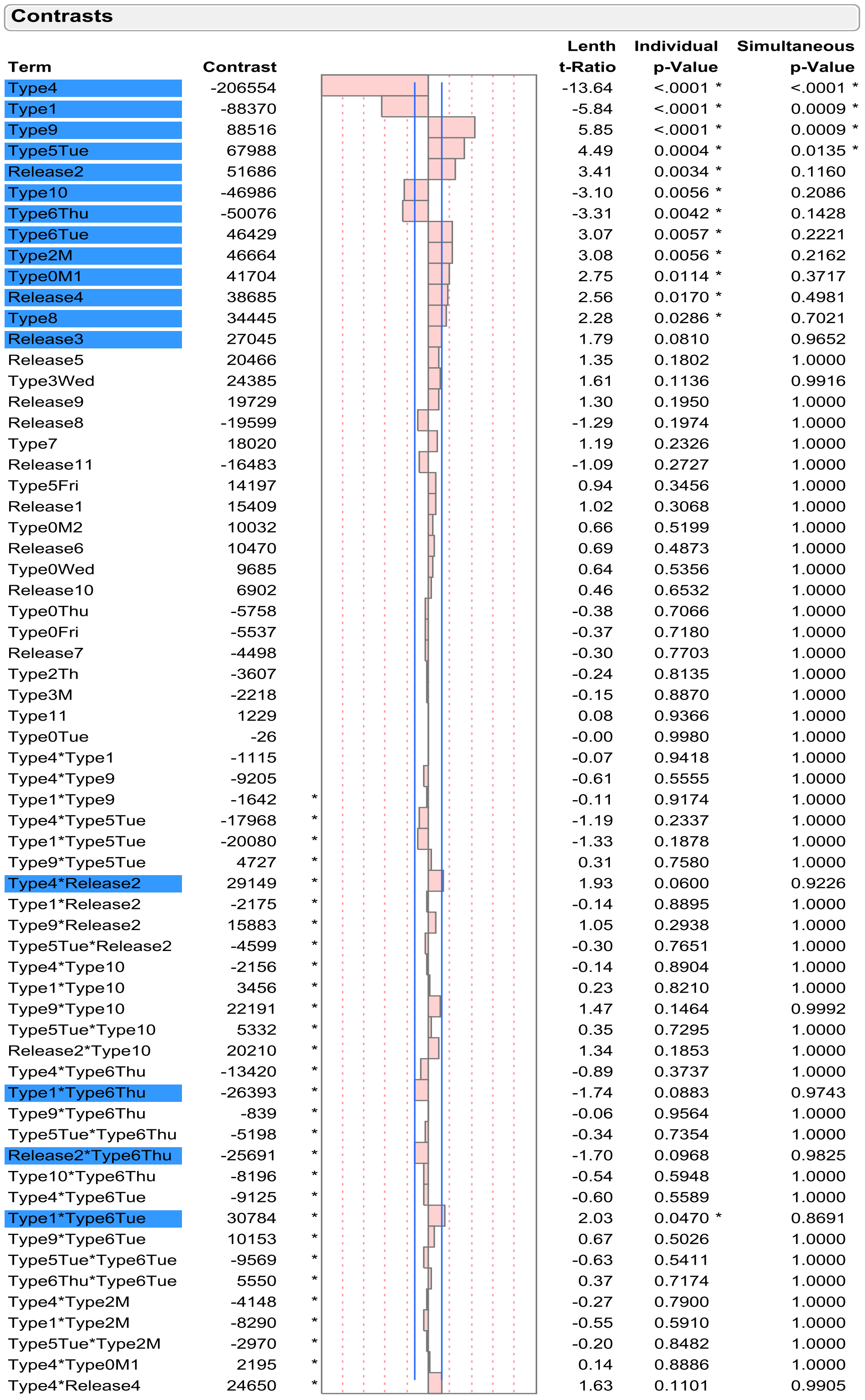
Summary of Fit

RSquare	0.927379
RSquare Adj	0.852415
Root Mean Square Error	115895.4
Mean of Response	1079744
Observations (or Sum Wgts	64

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Ratio
Model	32	5.3173e+12	1.662e+11	12.3710
Error	31	4.1638e+11	1.343e+10	Prob > F
C. Total	63	5.7336e+12		<.0001*

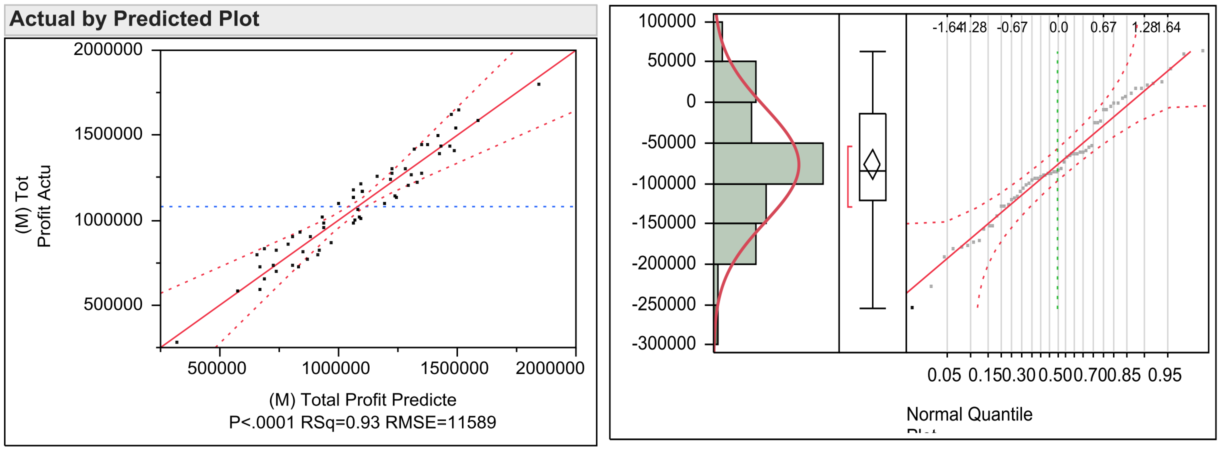
Figure 34: Summary of fit and analysis of variance for 2^k fractional factorial design



Contrasts

Term	Contrast		Lenth t-Ratio	Individual p-Value	Simultaneous p-Value
Type4	-206554		-13.64	<.0001 *	<.0001 *
Type1	-88370		-5.84	<.0001 *	0.0009 *
Type9	88516		5.85	<.0001 *	0.0009 *
Type5Tue	67988		4.49	0.0004 *	0.0135 *
Release2	51686		3.41	0.0034 *	0.1160
Type10	-46986		-3.10	0.0056 *	0.2086
Type6Thu	-50076		-3.31	0.0042 *	0.1428
Type6Tue	46429		3.07	0.0057 *	0.2221
Type2M	46664		3.08	0.0056 *	0.2162
Type0M1	41704		2.75	0.0114 *	0.3717
Release4	38685		2.56	0.0170 *	0.4981
Type8	34445		2.28	0.0286 *	0.7021
Release3	27045		1.79	0.0810	0.9652
Release5	20466		1.35	0.1802	1.0000
Type3Wed	24385		1.61	0.1136	0.9916
Release9	19729		1.30	0.1950	1.0000
Release8	-19599		-1.29	0.1974	1.0000
Type7	18020		1.19	0.2326	1.0000
Release11	-16483		-1.09	0.2727	1.0000
Type5Fri	14197		0.94	0.3456	1.0000
Release1	15409		1.02	0.3068	1.0000
Type0M2	10032		0.66	0.5199	1.0000
Release6	10470		0.69	0.4873	1.0000
Type0Wed	9685		0.64	0.5356	1.0000
Release10	6902		0.46	0.6532	1.0000
Type0Thu	-5758		-0.38	0.7066	1.0000
Type0Fri	-5537		-0.37	0.7180	1.0000
Release7	-4498		-0.30	0.7703	1.0000
Type2Th	-3607		-0.24	0.8135	1.0000
Type3M	-2218		-0.15	0.8870	1.0000
Type11	1229		0.08	0.9366	1.0000
Type0Tue	-26		-0.00	0.9980	1.0000
Type4*Type1	-1115		-0.07	0.9418	1.0000
Type4*Type9	-9205		-0.61	0.5555	1.0000
Type1*Type9	-1642	*	-0.11	0.9174	1.0000
Type4*Type5Tue	-17968	*	-1.19	0.2337	1.0000
Type1*Type5Tue	-20080	*	-1.33	0.1878	1.0000
Type9*Type5Tue	4727	*	0.31	0.7580	1.0000
Type4*Release2	29149	*	1.93	0.0600	0.9226
Type1*Release2	-2175	*	-0.14	0.8895	1.0000
Type9*Release2	15883	*	1.05	0.2938	1.0000
Type5Tue*Release2	-4599	*	-0.30	0.7651	1.0000
Type4*Type10	-2156	*	-0.14	0.8904	1.0000
Type1*Type10	3456	*	0.23	0.8210	1.0000
Type9*Type10	22191	*	1.47	0.1464	0.9992
Type5Tue*Type10	5332	*	0.35	0.7295	1.0000
Release2*Type10	20210	*	1.34	0.1853	1.0000
Type4*Type6Thu	-13420	*	-0.89	0.3737	1.0000
Type1*Type6Thu	-26393	*	-1.74	0.0883	0.9743
Type9*Type6Thu	-839	*	-0.06	0.9564	1.0000
Type5Tue*Type6Thu	-5198	*	-0.34	0.7354	1.0000
Release2*Type6Thu	-25691	*	-1.70	0.0968	0.9825
Type10*Type6Thu	-8196	*	-0.54	0.5948	1.0000
Type4*Type6Tue	-9125	*	-0.60	0.5589	1.0000
Type1*Type6Tue	30784	*	2.03	0.0470 *	0.8691
Type9*Type6Tue	10153	*	0.67	0.5026	1.0000
Type5Tue*Type6Tue	-9569	*	-0.63	0.5411	1.0000
Type6Thu*Type6Tue	5550	*	0.37	0.7174	1.0000
Type4*Type2M	-4148	*	-0.27	0.7900	1.0000
Type1*Type2M	-8290	*	-0.55	0.5910	1.0000
Type5Tue*Type2M	-2970	*	-0.20	0.8482	1.0000
Type4*Type0M1	2195	*	0.14	0.8886	1.0000
Type4*Release4	24650	*	1.63	0.1101	0.9905

Figure 35: Summary of screening design



Goodness-of-Fit Test

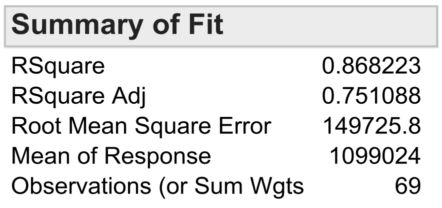
Shapiro-Wilk W Test

W	Prob<W
0.982050	0.4765

Note: Ho = The data is from the Normal distribution. Small p-values reject Ho.

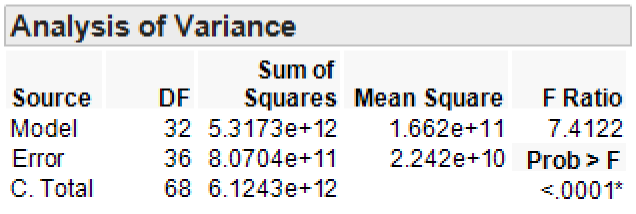
Figure 36: a) actual versus predicted response b) normal probability plot for residuals

Before making judgments about the significance of each factor, center points were added (i.e. all the factors set at their central level) and the analysis was re-run to ensure that the assumption of linearity stands. The center points clearly provide information about the existence of curvature in the system. If curvature is found in the system, then the addition of axial points allows for efficient estimation of the pure quadratic terms (Yaesoubi, 2006). Five center points were added to the model --the replicated points were used to calculate the pure error--which reduced the R^2_{Adj} from 85% to 75% (as well as, reduction in R^2) and the model F-Ratio from 12.37 to 7.4 (Figure 37).



Summary of Fit

RSquare	0.868223
RSquare Adj	0.751088
Root Mean Square Error	149725.8
Mean of Response	1099024
Observations (or Sum Wgts	69



Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Ratio
Model	32	5.3173e+12	1.662e+11	7.4122
Error	36	8.0704e+11	2.242e+10	Prob > F
C. Total	68	6.1243e+12		<.0001*

Figure 37: Summary of fit and analysis of variance for 2^{32-26} fractional factorial design, augmented with center points

In addition, both residual plots show that the residuals for the center points were greater than the residuals for other design points (Figure 38).

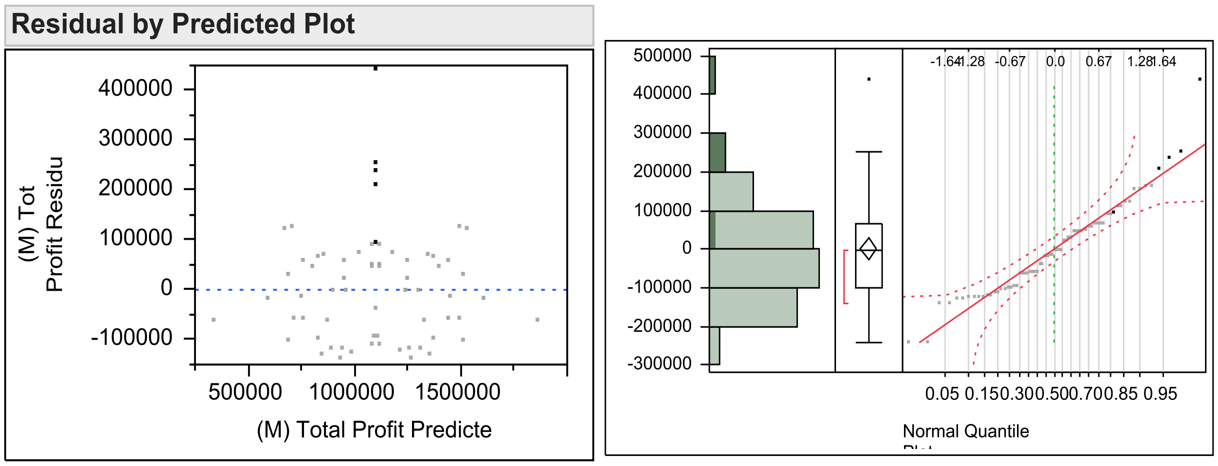
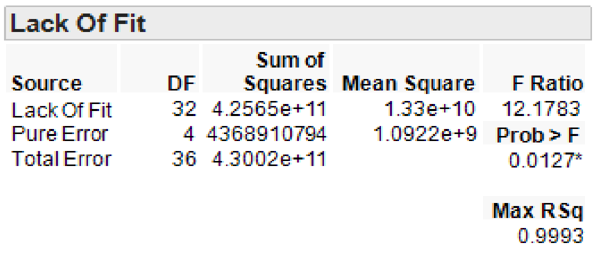


Figure 38: Residual by predicted plot and the residual distribution for 2^k fractional factorial design

To conclude whether a non-linear relationship exists, a lack of fit test was performed on the data. Since the observed statistic, F_0, value was significantly higher than the critical F-value of 5.738, there was sufficient evidence to conclude that, at α-level of 0.05, there is a lack of linear fit (Figure 39).



Lack Of Fit

Source	DF	Sum of Squares	Mean Square	F Ratio
Lack Of Fit	32	4.2565e+11	1.33e+10	12.1783
Pure Error	4	4368910794	1.0922e+9	**Prob > F**
Total Error	36	4.3002e+11		0.0127*
				Max RSq
				0.9993

Figure 39: Lack of fit test result 2^k fractional factorial design

All of this evidence suggests that a first-order polynomial augmented by second-order interactions may not be a good approximation of the response function (Yaesoubi, 2006).

In statistics, a central composite design is an experimental design that is useful in a response surface methodology for building a second order (quadratic) model for the response variable without the need to use a complete three-level factorial experiment. Therefore, a second-order model was devised, using Central Composite Design (CCD) to estimate the quadratic effects in the data. There are many designs available for fitting a second-order model. The most frequently used one is the CCD, introduced by Box and Wilson. It consists of factorial points (from a 2^q design and 2^{q-k} fractional factorial design), central points, and axial points. The center runs contain information about the curvature of the surface: if the curvature is significant, the additional axial points allow the experimenter to obtain an efficient estimation of the quadratic terms. When there is curvature in the response surface, the first-order model is insufficient. A second-order model is useful in approximating a portion of the true response surface with parabolic curvature (Bradley, 2007).

Three main varieties of CCD are available: face-centered, rotatable and inscribed. A face-centered design is obtained by setting the experiment range α at constant distance +1 and - 1 so that it requires only 3 levels of each factor (α=±1) as shown in Figure 40.

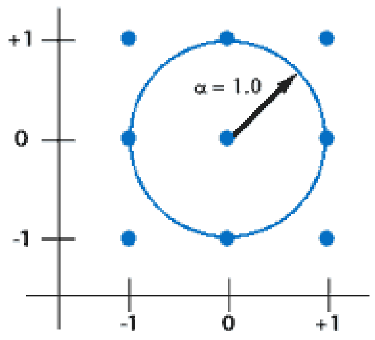


Figure 40: Face-centered design

In rotatable design, the extreme points are at some distance α>1 from the center, based on the properties desired for the design and the number of factors in the design to achieve rotatability. These points establish new extremes for the low and high settings for all factors (Figure 41). This design requires 5 levels for each factor.

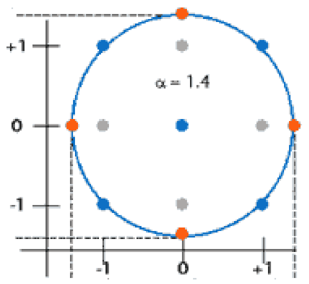


Figure 41: Rotatable design

Situations in which the limits specified for factor settings are truly limits call for inscribed design. This design uses the factor settings as the starting points and creates a factorial or fractional factorial design within those limits (in other words, an inscribed design is a scaled down rotatable design with each factor level of the rotatable design divided by α>1 to generate the inscribed design) (Figure 42). This design also requires 5 levels of each factor (Verseput, 2000).

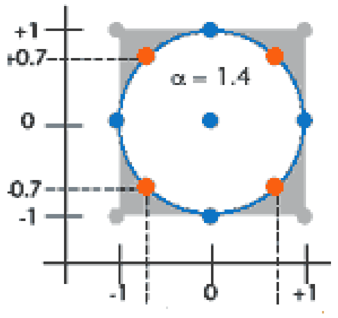
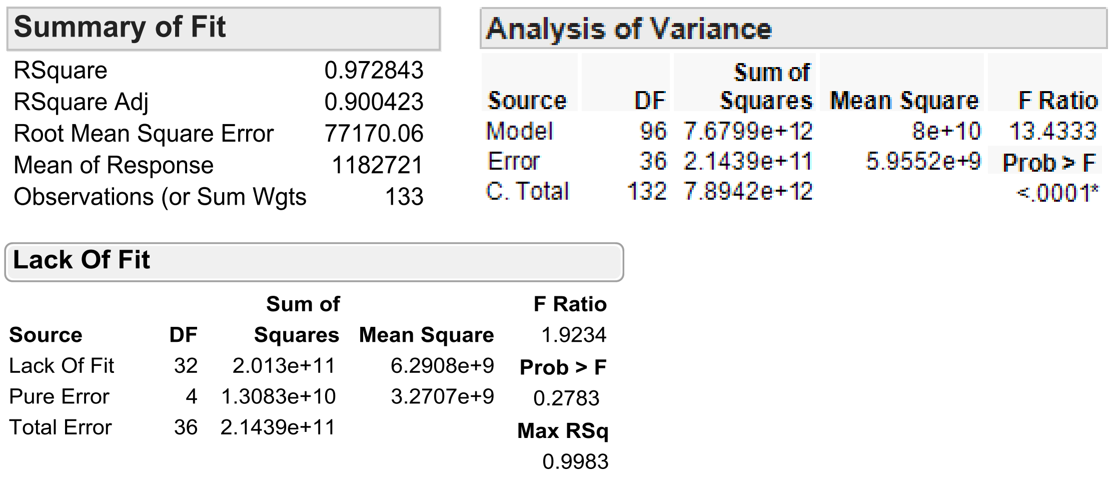


Figure 42: Inscribed design

For this study we chose a face-centered design, because, first, hospitals are not able to operate in OR rooms all the time and, second, it is not practical to set release times lower than some threshold. The inscribed design might have been the best choice if we had known the extreme limits to establish the low and high points for all factors. But, in this case, that information was lacking. Thus, the face-centered CCD was a simpler design to carry out in this situation, which requires operating the process at only three level settings of each variable.

However, applying CCD on all 32 factors was not practical (under resolution V) so, the next step will be to select the important factors under fractional factorial resolution IV.

A fractional factorial of resolution IV augmented with axial and central points was created for this study. It required 64 runs for a fractional factorial of resolution IV, and 2 × 32 runs for axial designs, and 5 runs for central points; thus a total of 133 runs were needed. We used 5 center points for stability of results.[7] The analysis of variance and summary of fit for the design are shown in Figure 43.



Summary of Fit

RSquare	0.972843
RSquare Adj	0.900423
Root Mean Square Error	77170.06
Mean of Response	1182721
Observations (or Sum Wgts	133

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Ratio
Model	96	7.6799e+12	8e+10	13.4333
Error	36	2.1439e+11	5.9552e+9	Prob > F
C. Total	132	7.8942e+12		<.0001*

Lack Of Fit

Source	DF	Sum of Squares	Mean Square	F Ratio
				1.9234
Lack Of Fit	32	2.013e+11	6.2908e+9	Prob > F
Pure Error	4	1.3083e+10	3.2707e+9	0.2783
Total Error	36	2.1439e+11		Max RSq
				0.9983

Figure 43: Summary of fit, analysis of variance, and lack of fit for 2^{32-26} fractional factorial design augmented with axial and center points

Both the R^2 and the model p-value has improved from 92% to 97% under generated second-order model, which implies that 97% of the variation in the dependent variables can be accounted for by the second-order model. Also, the large *p*-value for lack of fit (0.278) indicates the lack of fit is not significant and supports the conclusion that there is little to be gained by introducing additional variables. The normal probability plot for residuals is depicted in Figure 44. The plots show that the normal distribution assumption for residuals seems reasonable, and there is no significant evidence to suggest a violation of this assumption.

[7] - Augmented Design is used to modify an existing 2^k design data table and adds axial points together with center points to transform a screening design to a response surface design.

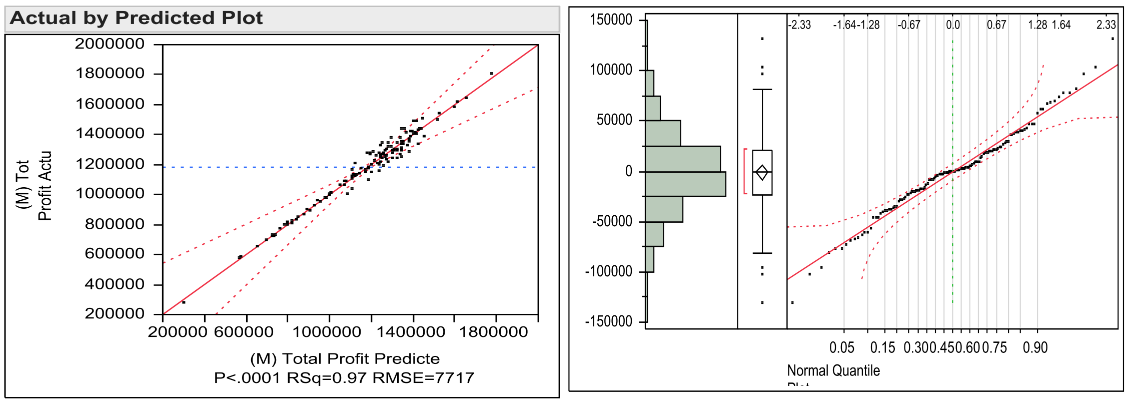
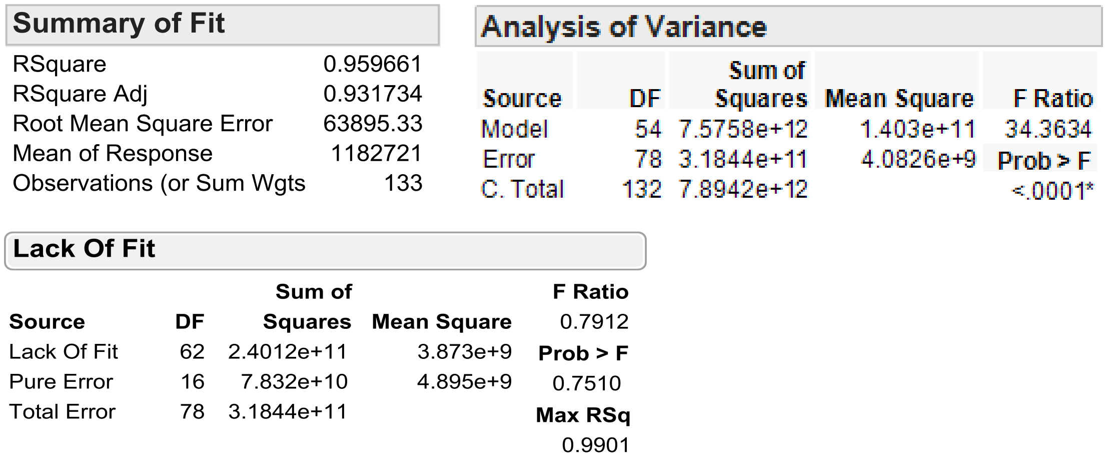


Figure 44: a) normal probability plot for residuals b) actual versus predicted response

As a first step, stepwise regression was run to identify the important factors (Figure 45). This criterion drops any effect with F-Ratio less than 2 from the model, making it a restricted model.



Summary of Fit

RSquare	0.959661
RSquare Adj	0.931734
Root Mean Square Error	63895.33
Mean of Response	1182721
Observations (or Sum Wgts	133

Analysis of Variance

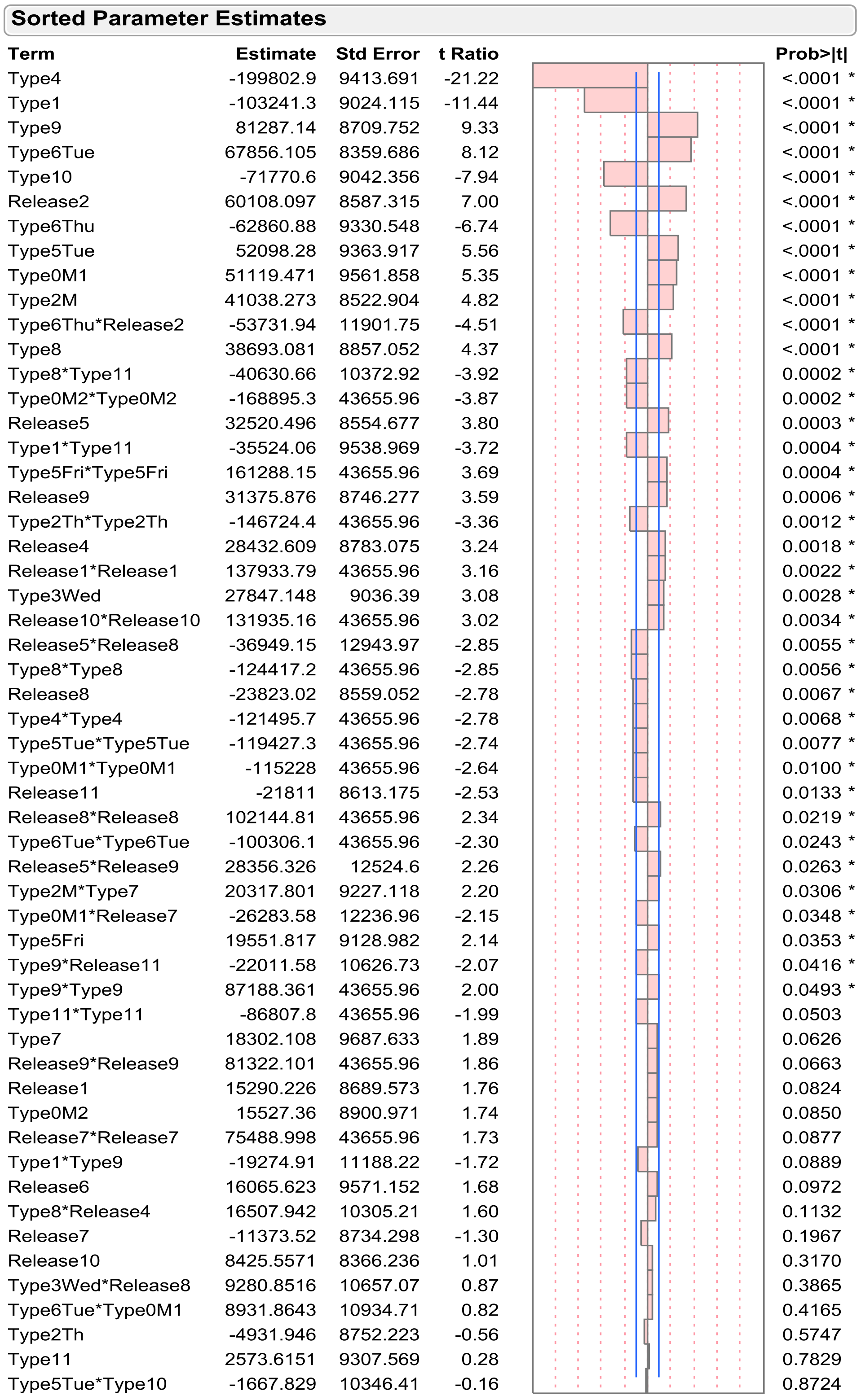
Source	DF	Sum of Squares	Mean Square	F Ratio
Model	54	7.5758e+12	1.403e+11	34.3634
Error	78	3.1844e+11	4.0826e+9	Prob > F
C. Total	132	7.8942e+12		<.0001*

Lack Of Fit

Source	DF	Sum of Squares	Mean Square	F Ratio
				0.7912
Lack Of Fit	62	2.4012e+11	3.873e+9	Prob > F
Pure Error	16	7.832e+10	4.895e+9	0.7510
Total Error	78	3.1844e+11		Max RSq
				0.9901

Figure 45: Summary of fit and analysis of variance for 2^{32-26} fractional factorial design on restricted model

Both the R^2_{Adj} and the model F-Ratio increased with a noticeable jump in the model F-ratio from 12.37 in original first-order model (all factors) to 34.36 in the restricted second-order model. Figure 45 displays the sorted estimated effects of the generated restricted second-level model. As is the case in the parameter estimates, the final model has significant cross product and quadratic factors.



Sorted Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Type4	-199802.9	9413.691	-21.22	<.0001 *
Type1	-103241.3	9024.115	-11.44	<.0001 *
Type9	81287.14	8709.752	9.33	<.0001 *
Type6Tue	67856.105	8359.686	8.12	<.0001 *
Type10	-71770.6	9042.356	-7.94	<.0001 *
Release2	60108.097	8587.315	7.00	<.0001 *
Type6Thu	-62860.88	9330.548	-6.74	<.0001 *
Type5Tue	52098.28	9363.917	5.56	<.0001 *
Type0M1	51119.471	9561.858	5.35	<.0001 *
Type2M	41038.273	8522.904	4.82	<.0001 *
Type6Thu*Release2	-53731.94	11901.75	-4.51	<.0001 *
Type8	38693.081	8857.052	4.37	<.0001 *
Type8*Type11	-40630.66	10372.92	-3.92	0.0002 *
Type0M2*Type0M2	-168895.3	43655.96	-3.87	0.0002 *
Release5	32520.496	8554.677	3.80	0.0003 *
Type1*Type11	-35524.06	9538.969	-3.72	0.0004 *
Type5Fri*Type5Fri	161288.15	43655.96	3.69	0.0004 *
Release9	31375.876	8746.277	3.59	0.0006 *
Type2Th*Type2Th	-146724.4	43655.96	-3.36	0.0012 *
Release4	28432.609	8783.075	3.24	0.0018 *
Release1*Release1	137933.79	43655.96	3.16	0.0022 *
Type3Wed	27847.148	9036.39	3.08	0.0028 *
Release10*Release10	131935.16	43655.96	3.02	0.0034 *
Release5*Release8	-36949.15	12943.97	-2.85	0.0055 *
Type8*Type8	-124417.2	43655.96	-2.85	0.0056 *
Release8	-23823.02	8559.052	-2.78	0.0067 *
Type4*Type4	-121495.7	43655.96	-2.78	0.0068 *
Type5Tue*Type5Tue	-119427.3	43655.96	-2.74	0.0077 *
Type0M1*Type0M1	-115228	43655.96	-2.64	0.0100 *
Release11	-21811	8613.175	-2.53	0.0133 *
Release8*Release8	102144.81	43655.96	2.34	0.0219 *
Type6Tue*Type6Tue	-100306.1	43655.96	-2.30	0.0243 *
Release5*Release9	28356.326	12524.6	2.26	0.0263 *
Type2M*Type7	20317.801	9227.118	2.20	0.0306 *
Type0M1*Release7	-26283.58	12236.96	-2.15	0.0348 *
Type5Fri	19551.817	9128.982	2.14	0.0353 *
Type9*Release11	-22011.58	10626.73	-2.07	0.0416 *
Type9*Type9	87188.361	43655.96	2.00	0.0493 *
Type11*Type11	-86807.8	43655.96	-1.99	0.0503
Type7	18302.108	9687.633	1.89	0.0626
Release9*Release9	81322.101	43655.96	1.86	0.0663
Release1	15290.226	8689.573	1.76	0.0824
Type0M2	15527.36	8900.971	1.74	0.0850
Release7*Release7	75488.998	43655.96	1.73	0.0877
Type1*Type9	-19274.91	11188.22	-1.72	0.0889
Release6	16065.623	9571.152	1.68	0.0972
Type8*Release4	16507.942	10305.21	1.60	0.1132
Release7	-11373.52	8734.298	-1.30	0.1967
Release10	8425.5571	8366.236	1.01	0.3170
Type3Wed*Release8	9280.8516	10657.07	0.87	0.3865
Type6Tue*Type0M1	8931.8643	10934.71	0.82	0.4165
Type2Th	-4931.946	8752.223	-0.56	0.5747
Type11	2573.6151	9307.569	0.28	0.7829
Type5Tue*Type10	-1667.829	10346.41	-0.16	0.8724

Figure 46: Parameter estimates for restricted model

In the next step, important factors were selected among the significant ones (based on Lenth's t-ratio, provided in Table 36) generating over $100K change in the profit (the $100K threshold was set such that it reduced the factors to less than half). Therefore, if a factor had a main effect greater than $50K (or $100K/2) or a quadratic effect greater than $100K, or was involved in a second-order interaction effect greater that $50K, it was declared to be important (Yaesoubi, 2006). Table 36 lists the selected main important factors.

Table 36: List of important factors for Stafford block size

Number	**Factor**	**Estimate**	**Lenth t-Ratio**
1	Type1	-103241	-6.82
2	Type2M	41038	3.42
3	Type2Th (Type2Th* Type2Th)	-146724	2.11
4	Type4	-199803	-16.41
5	Type5Tue	52098	5.14
6	Type5Fri (Type5Fri * Type5Fri)	16093	2.71
7	Type6Tue	67856	3.72
8	Type6Thu	-62861	-3.68
9	Type9	81287	7.16
10	Type10	-71771	-4.11
11	Type0M1	51119	3.55

Considering our results from the earlier case study, we expected the effect of the release block policy on profit (prior to the best block size) to be negligible in comparison with the effect of block size. We can see the same result in Figure 45, the sorted estimated effect size of the release block and block size. Among the top fifteen significant factors, only two factors are release block. Given the small effect of the release block in comparison with block size (refer to Figure 47) and to ensure that the block release policy effect would be considered in the final model, we divided the process of finding important factors into two steps.

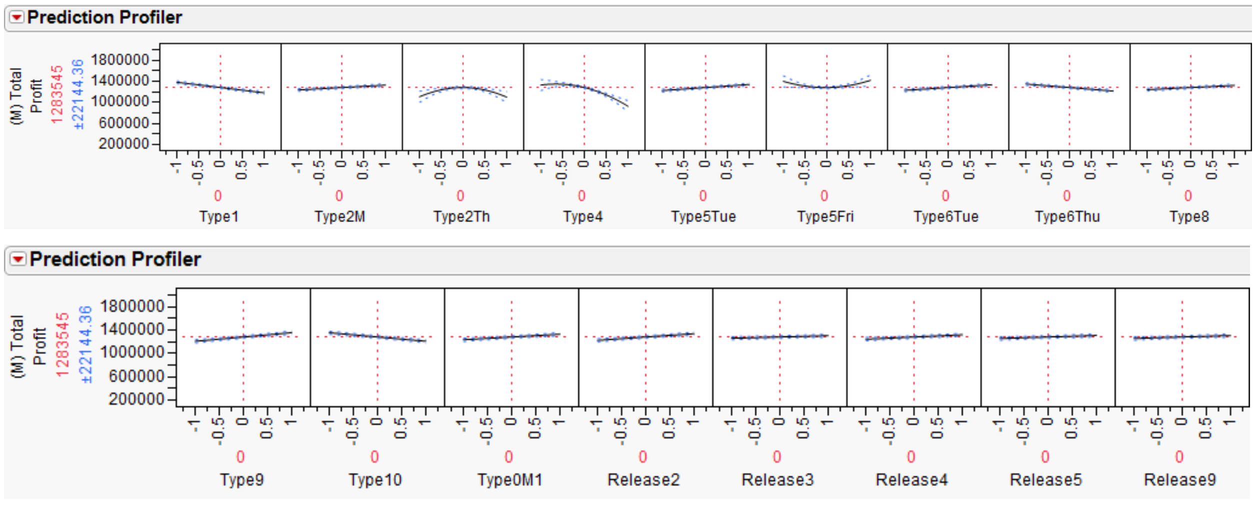
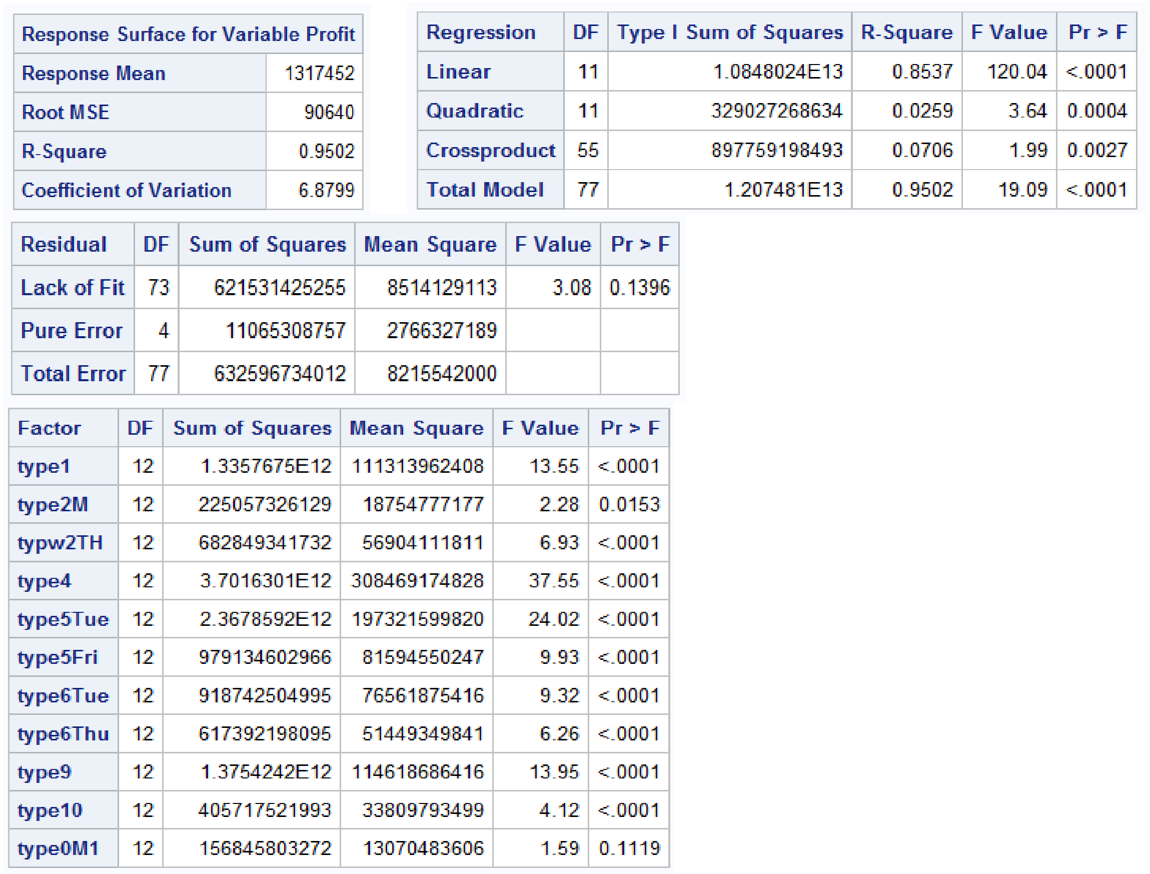


Figure 47: Prediction Profiler

In the first step, the restricted model was run with the top block size factors, and their optimal point was found. As we learned in the case study, the impact of the release policy appears negligible initially, but it becomes more significant in the range of superior block size. So, in the second step, important factors were selected among those important block size factors from the first step and all block release factors within the range of superior main block size factors. These steps are explained in detail below.

Step1. Once a screening experiment has been performed and the important factors determined, the next step is often to perform a response surface experiment to produce a prediction model to determine curvature, detect interactions among the factors, and optimize the process (Telford, 2007). In this study, a second-order model was fitted using face-centered central composite design (CCD) for the eleven factors selected as important. To obtain the second-level model, a central composite design resolution V with center point and axial point was run. This design required 2^{11-4} factorial runs for resolution V design, 22 axial and 5 center points run; thus a total of 155 runs were needed (Matlab is used to create the design). For each run, 10 observations were obtained. The RSREG procedure is used to fit the response surface. The analysis of variance and lack of fit for the model are shown in Figure 47. The factor ANOVA table displays tests for all eleven parameters corresponding to each factor.



Response Surface for Variable Profit	
Response Mean	1317452
Root MSE	90640
R-Square	0.9502
Coefficient of Variation	6.8799

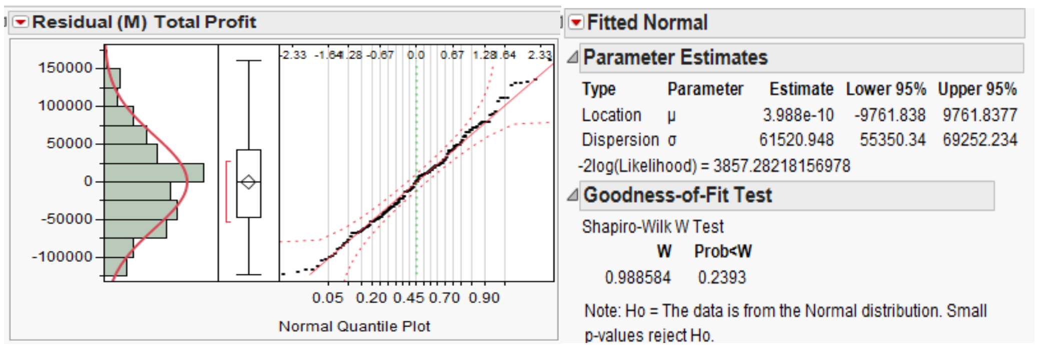
Regression	DF	Type I Sum of Squares	R-Square	F Value	Pr > F
Linear	11	1.0848024E13	0.8537	120.04	<.0001
Quadratic	11	329027268634	0.0259	3.64	0.0004
Crossproduct	55	897759198493	0.0706	1.99	0.0027
Total Model	77	1.207481E13	0.9502	19.09	<.0001

Residual	DF	Sum of Squares	Mean Square	F Value	Pr > F
Lack of Fit	73	621531425255	8514129113	3.08	0.1396
Pure Error	4	11065308757	2766327189		
Total Error	77	632596734012	8215542000		

Factor	DF	Sum of Squares	Mean Square	F Value	Pr > F
type1	12	1.3357675E12	111313962408	13.55	<.0001
type2M	12	225057326129	18754777177	2.28	0.0153
typw2TH	12	682849341732	56904111811	6.93	<.0001
type4	12	3.7016301E12	308469174828	37.55	<.0001
type5Tue	12	2.3678592E12	197321599820	24.02	<.0001
type5Fri	12	979134602966	81594550247	9.93	<.0001
type6Tue	12	918742504995	76561875416	9.32	<.0001
type6Thu	12	617392198095	51449349841	6.26	<.0001
type9	12	1.3754242E12	114618686416	13.95	<.0001
type10	12	405717521993	33809793499	4.12	<.0001
type0M1	12	156845803272	13070483606	1.59	0.1119

Figure 48: Summary of fit, analysis of variance and ANOVA test for CCD

The normal probability plot for residuals is shown in Figure 49. This plot indicates that the normal distribution assumption for residuals appears reasonable; there is no significant evidence to suggest the violation of this assumption. Also, the normality test confirms that the normal distribution assumption holds.



Type	Parameter	Estimate	Lower 95%	Upper 95%
Location	μ	3.988e-10	-9761.838	9761.8377
Dispersion	σ	61520.948	55350.34	69252.234

-2log(Likelihood) = 3857.28218156978

Shapiro-Wilk W Test

W	Prob<W
0.988584	0.2393

Note: Ho = The data is from the Normal distribution. Small p-values reject Ho.

Figure 49: a) normal probability plot for residuals b) Normality test

JMP computes the linear, quadratic, and interaction terms in the model. The estimate of effects for the response variable profit is shown in Figure 50. Analysis of variance indicates that there were significant interactions between the factors. The small *p*-values for linear and quadratic terms also confirm that their contribution is significant to the model, and there is curvature in the response surface.[8]

Next, it was necessary to find the levels of factors that optimized the predicted response, profit. When the response surface is not a plane, it becomes more complicated to determine optimum values. This point, if it exists, will be the set of factors for which the partial derivatives equal to zero. This point is called the stationary point. The stationary point can be either a maximum, a minimum, or a saddle point (Montgomery, 2005).

We may obtain a general mathematical solution for the location of the stationary point as expressing the fitted second-order model in matrix notation, as follows:

$$\hat{y}(X) = b_0 + \sum_{i=1}^{k} b_i\, x_i + \sum_{i=1}^{k} b_{ii} x_i^2 + \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} b_{ij}\, x_i x_j = b_0 + X'\boldsymbol{b} + X'\boldsymbol{B}X \qquad (4.1)$$

Where

$$\text{x}=\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_k \end{bmatrix}, \quad \text{b}=\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_k \end{bmatrix}, \text{ and B}=\begin{bmatrix} b_{11} & b_{12}/2 & \cdots & b_{1k}/2 \\ b_{12}/2 & b_{22} & \dots & b_{2k}/2 \\ \vdots & \vdots & \ddots & \vdots \\ b_{1k}/2 & b_{2k}/2 & \cdots & b_{kk} \end{bmatrix} \qquad (4.2)$$

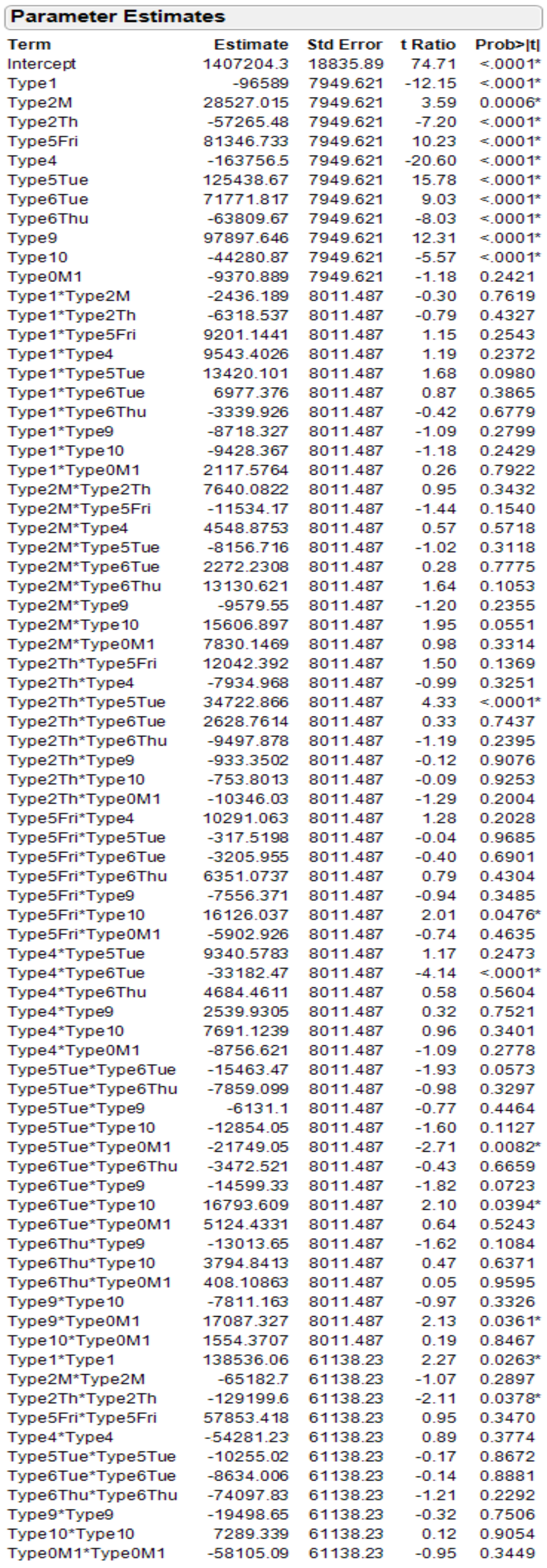
Assuming B is nonsingular, the unique stationary point of the fitted surface occurs at

$$X_s = -\frac{1}{2}\, \boldsymbol{B}^{-1}\boldsymbol{b} \qquad (4.3)$$

Furthermore, by substituting Equation 4.3 into Equation 4.1, we can find the predicted response at the stationary point as:

$$\hat{y}_s = b_0 + \frac{1}{2}\, X_s{}'\boldsymbol{b}$$

[8] -This model follows CCD assumption where each factor x_i is standardized and lies in [-1, +1] range.



Parameter Estimates

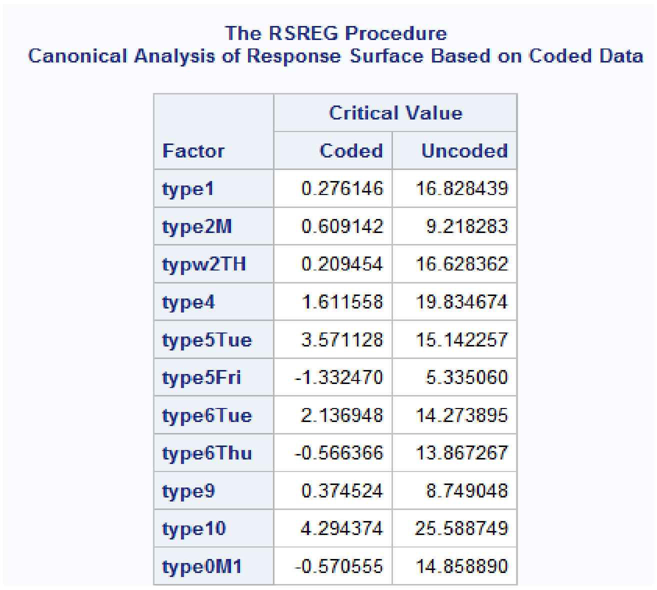
Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	1407204.3	18835.89	74.71	<.0001*
Type1	-96589	7949.621	-12.15	<.0001*
Type2M	28527.015	7949.621	3.59	0.0006*
Type2Th	-57265.48	7949.621	-7.20	<.0001*
Type5Fri	81346.733	7949.621	10.23	<.0001*
Type4	-163756.5	7949.621	-20.60	<.0001*
Type5Tue	125438.67	7949.621	15.78	<.0001*
Type6Tue	71771.817	7949.621	9.03	<.0001*
Type6Thu	-63809.67	7949.621	-8.03	<.0001*
Type9	97897.646	7949.621	12.31	<.0001*
Type10	-44280.87	7949.621	-5.57	<.0001*
Type0M1	-9370.889	7949.621	-1.18	0.2421
Type1*Type2M	-2436.189	8011.487	-0.30	0.7619
Type1*Type2Th	-6318.537	8011.487	-0.79	0.4327
Type1*Type5Fri	9201.1441	8011.487	1.15	0.2543
Type1*Type4	9543.4026	8011.487	1.19	0.2372
Type1*Type5Tue	13420.101	8011.487	1.68	0.0980
Type1*Type6Tue	6977.376	8011.487	0.87	0.3865
Type1*Type6Thu	-3339.926	8011.487	-0.42	0.6779
Type1*Type9	-8718.327	8011.487	-1.09	0.2799
Type1*Type10	-9428.367	8011.487	-1.18	0.2429
Type1*Type0M1	2117.5764	8011.487	0.26	0.7922
Type2M*Type2Th	7640.0822	8011.487	0.95	0.3432
Type2M*Type5Fri	-11534.17	8011.487	-1.44	0.1540
Type2M*Type4	4548.8753	8011.487	0.57	0.5718
Type2M*Type5Tue	-8156.716	8011.487	-1.02	0.3118
Type2M*Type6Tue	2272.2308	8011.487	0.28	0.7775
Type2M*Type6Thu	13130.621	8011.487	1.64	0.1053
Type2M*Type9	-9579.55	8011.487	-1.20	0.2355
Type2M*Type10	15606.897	8011.487	1.95	0.0551
Type2M*Type0M1	7830.1469	8011.487	0.98	0.3314
Type2Th*Type5Fri	12042.392	8011.487	1.50	0.1369
Type2Th*Type4	-7934.968	8011.487	-0.99	0.3251
Type2Th*Type5Tue	34722.866	8011.487	4.33	<.0001*
Type2Th*Type6Tue	2628.7614	8011.487	0.33	0.7437
Type2Th*Type6Thu	-9497.878	8011.487	-1.19	0.2395
Type2Th*Type9	-933.3502	8011.487	-0.12	0.9076
Type2Th*Type10	-753.8013	8011.487	-0.09	0.9253
Type2Th*Type0M1	-10346.03	8011.487	-1.29	0.2004
Type5Fri*Type4	10291.063	8011.487	1.28	0.2028
Type5Fri*Type5Tue	-317.5198	8011.487	-0.04	0.9685
Type5Fri*Type6Tue	-3205.955	8011.487	-0.40	0.6901
Type5Fri*Type6Thu	6351.0737	8011.487	0.79	0.4304
Type5Fri*Type9	-7556.371	8011.487	-0.94	0.3485
Type5Fri*Type10	16126.037	8011.487	2.01	0.0476*
Type5Fri*Type0M1	-5902.926	8011.487	-0.74	0.4635
Type4*Type5Tue	9340.5783	8011.487	1.17	0.2473
Type4*Type6Tue	-33182.47	8011.487	-4.14	<.0001*
Type4*Type6Thu	4684.4611	8011.487	0.58	0.5604
Type4*Type9	2539.9305	8011.487	0.32	0.7521
Type4*Type10	7691.1239	8011.487	0.96	0.3401
Type4*Type0M1	-8756.621	8011.487	-1.09	0.2778
Type5Tue*Type6Tue	-15463.47	8011.487	-1.93	0.0573
Type5Tue*Type6Thu	-7859.099	8011.487	-0.98	0.3297
Type5Tue*Type9	-6131.1	8011.487	-0.77	0.4464
Type5Tue*Type10	-12854.05	8011.487	-1.60	0.1127
Type5Tue*Type0M1	-21749.05	8011.487	-2.71	0.0082*
Type6Tue*Type6Thu	-3472.521	8011.487	-0.43	0.6659
Type6Tue*Type9	-14599.33	8011.487	-1.82	0.0723
Type6Tue*Type10	16793.609	8011.487	2.10	0.0394*
Type6Tue*Type0M1	5124.4331	8011.487	0.64	0.5243
Type6Thu*Type9	-13013.65	8011.487	-1.62	0.1084
Type6Thu*Type10	3794.8413	8011.487	0.47	0.6371
Type6Thu*Type0M1	408.10863	8011.487	0.05	0.9595
Type9*Type10	-7811.163	8011.487	-0.97	0.3326
Type9*Type0M1	17087.327	8011.487	2.13	0.0361*
Type10*Type0M1	1554.3707	8011.487	0.19	0.8467
Type1*Type1	138536.06	61138.23	2.27	0.0263*
Type2M*Type2M	-65182.7	61138.23	-1.07	0.2897
Type2Th*Type2Th	-129199.6	61138.23	-2.11	0.0378*
Type5Fri*Type5Fri	57853.418	61138.23	0.95	0.3470
Type4*Type4	-54281.23	61138.23	0.89	0.3774
Type5Tue*Type5Tue	-10255.02	61138.23	-0.17	0.8672
Type6Tue*Type6Tue	-8634.006	61138.23	-0.14	0.8881
Type6Thu*Type6Thu	-74097.83	61138.23	-1.21	0.2292
Type9*Type9	-19498.65	61138.23	-0.32	0.7506
Type10*Type10	7289.339	61138.23	0.12	0.9054
Type0M1*Type0M1	-58105.09	61138.23	-0.95	0.3449

Figure 50: Parameter estimates for CCD

We can find the location of the stationary point in the experiment region using the general solution in equation 4.1 (Mee, 2009). Note that for our model (values are from Figure 50),

$$b=\begin{bmatrix} -96589 \\ 28527 \\ -57265 \\ -163756 \\ 125438 \\ 81346 \\ 71771 \\ -63809 \\ 97897 \\ -44280 \\ -9370 \end{bmatrix}, \text{ and } B=\begin{bmatrix} 138536 & -710 & -3511 & \cdots & 1801 \\ -710 & -65182 & 4406 & \ldots & 5282 \\ -3511 & 4406 & -129199 & \ldots & -4509 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1801 & 5282 & -4509 & \cdots & -58105 \end{bmatrix}$$

and from equation 4.3, the stationary point is determined at the following location. Both coded and uncoded values are provided based on canonical analysis output,


The RSREG Procedure
Canonical Analysis of Response Surface Based on Coded Data

Factor	Critical Value	
	Coded	Uncoded
type1	0.276146	16.828439
type2M	0.609142	9.218283
typw2TH	0.209454	16.628362
type4	1.611558	19.834674
type5Tue	3.571128	15.142257
type5Fri	-1.332470	5.335060
type6Tue	2.136948	14.273895
type6Thu	-0.566366	13.867267
type9	0.374524	8.749048
type10	4.294374	25.588749
type0M1	-0.570555	14.858890

Figure 51: Canonical Analysis for 11 factors

With predicted responses of 1,455,073, determining some of the larger predicted values (outside the range of [-1, +1]) would require extrapolation.

Once the stationary point is found, it is usually necessary to characterize the response surface in the immediate vicinity of this point; that is, one must determine whether the stationary point is a maximum, a minimum, or a saddle point. We also need to study the relative sensitivity of response to the variables. Although a counter plot is the easiest way when there are just a few variables, performing canonical analysis (an Eigen analysis) of Hessian matrix B is the more appropriate and scientific method (Montgomery, 2005).

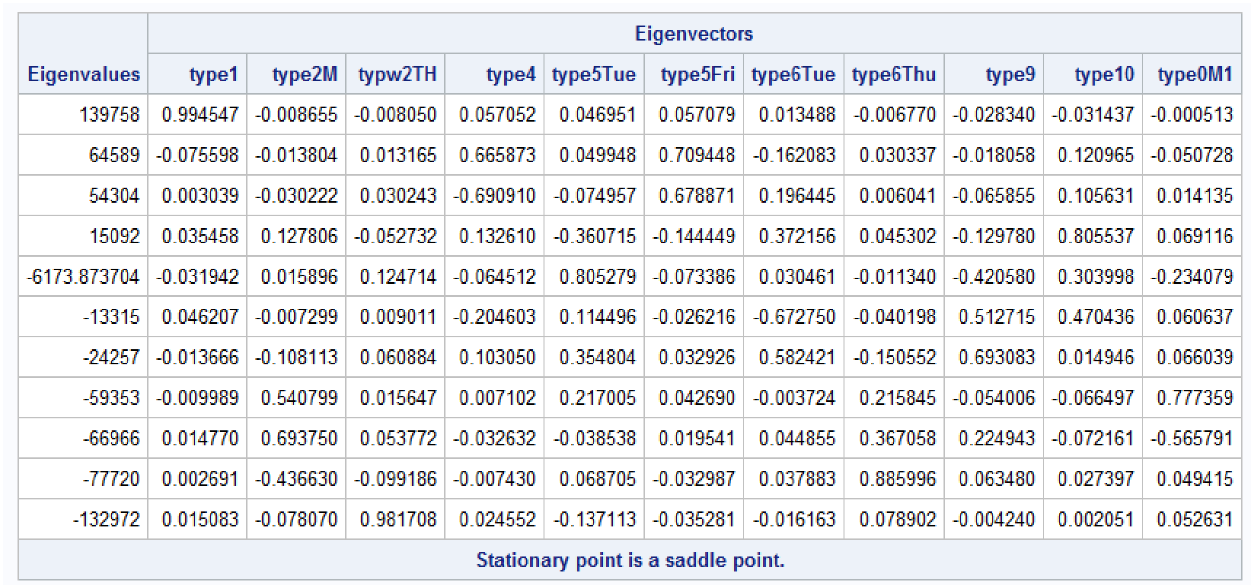
In canonical analysis, a model is transformed into a new coordinate system with the origin at the stationary point and then the axes of this system are rotated until they are parallel to the principal axes of the fitted response surface.

In this study, canonical analysis was used as described above to characterize the stationary point X_s by calculating eigenvalues of Matrix B as the roots of the following determinate equation,

$$[\mathbf{B} - \lambda\,\mathbf{I}] = 0 \qquad (4.4)$$

Accordingly, the roots of the equation become:

The sorted eigenvalue and associated eigenvector of matrix B are presented in Figure 52.



	Eigenvectors										
Eigenvalues	type1	type2M	typw2TH	type4	type5Tue	type5Fri	type6Tue	type6Thu	type9	type10	type0M1
139758	0.994547	-0.008655	-0.008050	0.057052	0.046951	0.057079	0.013488	-0.006770	-0.028340	-0.031437	-0.000513
64589	-0.075598	-0.013804	0.013165	0.665873	0.049948	0.709448	-0.162083	0.030337	-0.018058	0.120965	-0.050728
54304	0.003039	-0.030222	0.030243	-0.690910	-0.074957	0.678871	0.196445	0.006041	-0.065855	0.105631	0.014135
15092	0.035458	0.127806	-0.052732	0.132610	-0.360715	-0.144449	0.372156	0.045302	-0.129780	0.805537	0.069116
-6173.873704	-0.031942	0.015896	0.124714	-0.064512	0.805279	-0.073386	0.030461	-0.011340	-0.420580	0.303998	-0.234079
-13315	0.046207	-0.007299	0.009011	-0.204603	0.114496	-0.026216	-0.672750	-0.040198	0.512715	0.470436	0.060637
-24257	-0.013666	-0.108113	0.060884	0.103050	0.354804	0.032926	0.582421	-0.150552	0.693083	0.014946	0.066039
-59353	-0.009989	0.540799	0.015647	0.007102	0.217005	0.042690	-0.003724	0.215845	-0.054006	-0.066497	0.777359
-66966	0.014770	0.693750	0.053772	-0.032632	-0.038538	0.019541	0.044855	0.367058	0.224943	-0.072161	-0.565791
-77720	0.002691	-0.436630	-0.099186	-0.007430	0.068705	-0.032987	0.037883	0.885996	0.063480	0.027397	0.049415
-132972	0.015083	-0.078070	0.981708	0.024552	-0.137113	-0.035281	-0.016163	0.078902	-0.004240	0.002051	0.052631

Stationary point is a saddle point.

Figure 52: Canonical curvature, Eigenvalues and Eigenvectors

Thus, the canonical form of the fitted model is as follows:

$$y = 1455073 + 139758\mathrm{w}_1^2 + 64589\mathrm{w}_2^2 + 54304\mathrm{w}_3^2 + 15092\mathrm{w}_4^2 - 6173\mathrm{w}_5^2 \pm \cdots - 132972\mathrm{w}_{11}^2$$

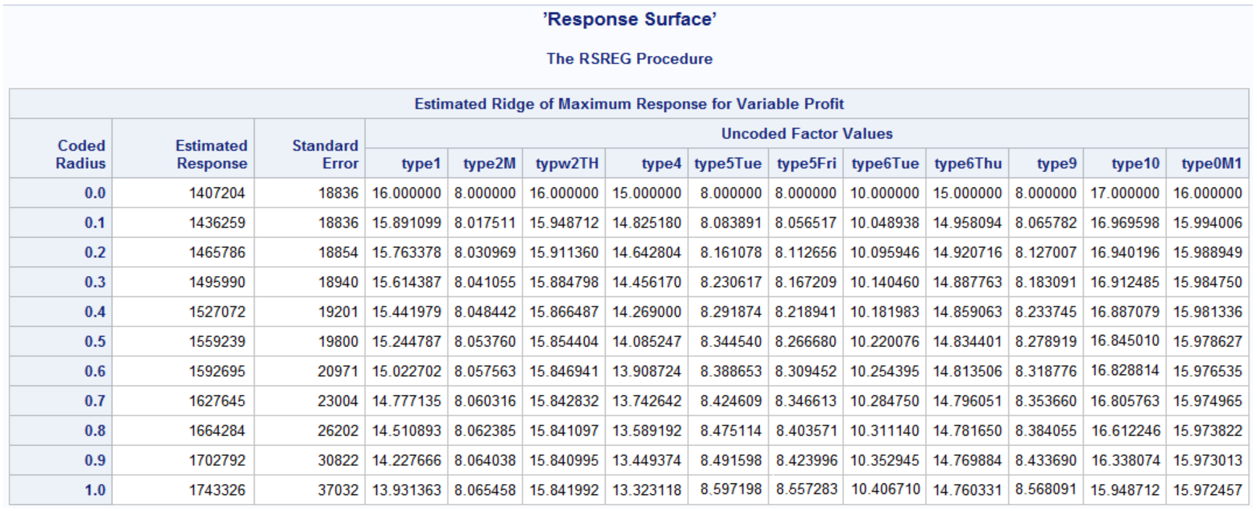
The fitted surface is a saddle surface and unbounded as a consequence of having both positive and negative eigenvalues.

Close examination of the calculated stationary point reveals that not all levels fall within the region of experiment. The optimum values of Type4, Type5Tue, Type5Fri, Type6Tue, and Type10 extrapolate the limits of the experimental design [-1, +1]. In many first-order cases, as well as second-order cases where a saddle point or the stationary point is found to be distant, the most useful further action is to decide in which direction to explore further. Also, comparing the resulting profit at saddle point (1,455,073) with the profit at the current production level (the center point for all main factors) (1,407,204), it is evident that any further improvement in the profit will require us to move towards the curve up direction instead of extrapolating around the extended experiment region to find the optimum.

Ridge analysis of the response surface was performed[9] to locate the optimal response value (and its associated variable levels) within the boundaries of the region (Figure 53). The ridge starts at the midway point (between the highest and the lowest values of the factors), and the point on the ridge at radius 1.0 from the midway point is the collection of factor settings that optimizes the predicted response at this radius. Thus, the ridge analysis can be used as a tool to help interpret an existing response surface or to indicate the direction in which further experimentation should be performed within the boundaries of the region.

The ridge analysis output indicates that maximum profit results from relatively lower block hours for almost all surgeons' blocks, with the exception of type 2 and type 0. The desirability prediction, shown in Figure 54, confirms this conclusion.

[9] – SAS `proc rsreg` is used for this analysis



'Response Surface'

The RSREG Procedure

Estimated Ridge of Maximum Response for Variable Profit													
			Uncoded Factor Values										
Coded Radius	Estimated Response	Standard Error	type1	type2M	typw2TH	type4	type5Tue	type5Fri	type6Tue	type6Thu	type9	type10	type0M1
0.0	1407204	18836	16.000000	8.000000	16.000000	15.000000	8.000000	8.000000	10.000000	15.000000	8.000000	17.000000	16.000000
0.1	1436259	18836	15.891099	8.017511	15.948712	14.825180	8.083891	8.056517	10.048938	14.958094	8.065782	16.969598	15.994006
0.2	1465786	18854	15.763378	8.030969	15.911360	14.642804	8.161078	8.112656	10.095946	14.920716	8.127007	16.940196	15.988949
0.3	1495990	18940	15.614387	8.041055	15.884798	14.456170	8.230617	8.167209	10.140460	14.887763	8.183091	16.912485	15.984750
0.4	1527072	19201	15.441979	8.048442	15.866487	14.269000	8.291874	8.218941	10.181983	14.859063	8.233745	16.887079	15.981336
0.5	1559239	19800	15.244787	8.053760	15.854404	14.085247	8.344540	8.266680	10.220076	14.834401	8.278919	16.845010	15.978627
0.6	1592695	20971	15.022702	8.057563	15.846941	13.908724	8.388653	8.309452	10.254395	14.813506	8.318776	16.828814	15.976535
0.7	1627645	23004	14.777135	8.060316	15.842832	13.742642	8.424609	8.346613	10.284750	14.796051	8.353660	16.805763	15.974965
0.8	1664284	26202	14.510893	8.062385	15.841097	13.589192	8.475114	8.403571	10.311140	14.781650	8.384055	16.612246	15.973822
0.9	1702792	30822	14.227666	8.064038	15.840995	13.449374	8.491598	8.423996	10.352945	14.769884	8.433690	16.338074	15.973013
1.0	1743326	37032	13.931363	8.065458	15.841992	13.323118	8.597198	8.557283	10.406710	14.760331	8.568091	15.948712	15.972457

Figure 53: Ridge Analysis

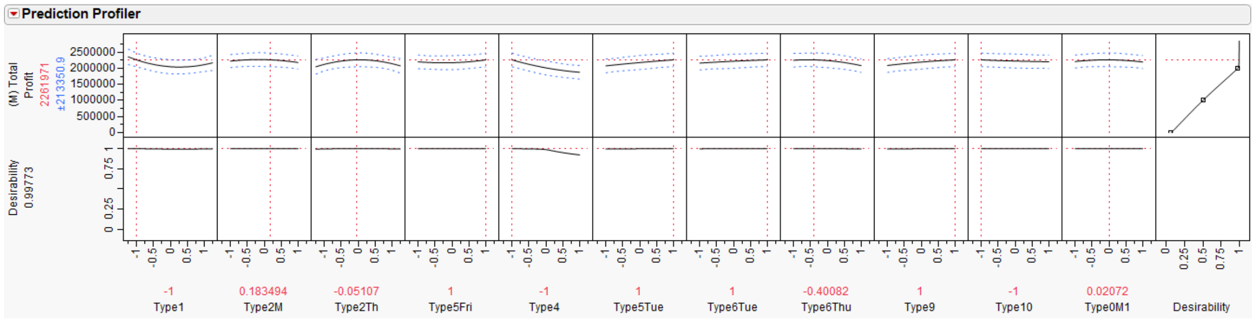


Figure 54: The Desirability Profiler

To summarize, at the predicted response, profit reached \$2.26M, 60% higher than \$1.4M estimated profit at the current operational level (experiment region at center point). The modified stationary point is thus:

$$X_{s'} = \begin{bmatrix} -1 \\ 0.02 \\ -0.05 \\ -1 \\ 1 \\ 1 \\ 1 \\ -0.42 \\ 1 \\ -1 \\ 0.02 \end{bmatrix}$$

This result suggests that the superior policy would be to reduce the block size in eight out of eleven factors and keep the block size at the current level for the rest. The willingness to reduce overall block size across all specialties and increase utilization may not be realistic or practical. As the *OR Manager* has published, in 2012 and 2013, the median utilization rate for operating rooms was around 75%, with top utilizations of 85% to 90% and vastly different rates among specialties because some specialties need further support from primary services due to complexity of their cases. To summarize, actual utilization rates are affected by the risk of overtime combined with complex patient mixes. High utilization rates require extremely good supporting systems, particularly with respect to bed availability, pre-admissions testing and PACU access. Otherwise, the benefits of high utilization will be outweighed by the costs of excessive overtime and staffing.

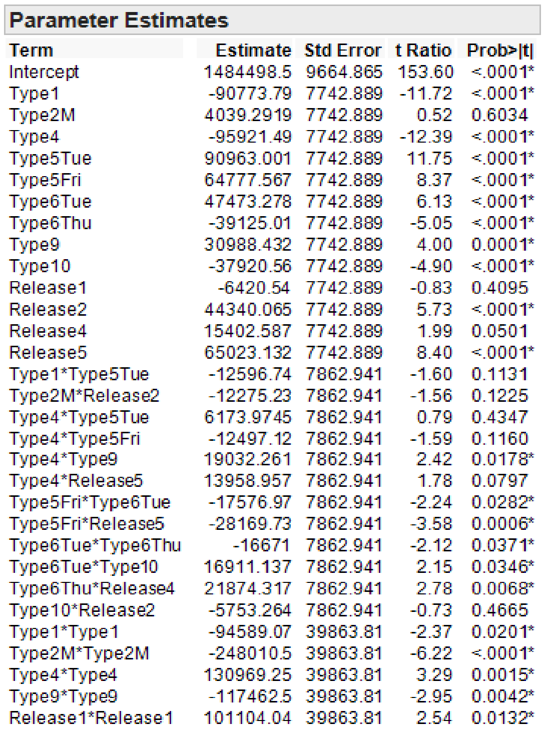
So it is necessary to consider realistic utilization rates for specialties, benchmarks that take into account the specialties' patient mix characteristics and the hospital's willingness to accept the risk of overtime (Van Houdenhoven et al., 2007). To check the robustness of our model, we ran the simulation for three scenarios of target utilization rate: 60%, 75% and 90%. For simplicity, we assume the same utilization rate for all specialties. Table 37 summarizes the simulation results for optimized block size factors under each utilization rate.

Table 37: The best block size for a range of utilization rate

	Utilization							
Factors	**100%**		**90%**		**75%**		**60%**	
Type1	-1	*reduce*	-1	*reduce*	-0.6	*reduce*	-0.3	*reduce*
Type2M	0.02	*no change*	0.02	*no change*	-0.1	*increase*	-0.6	*increase*
Type2Th	-0.05	*no change*	-0.03	*no change*	0.03	*no change*	0.15	*increase*
Tyep4	-1	*reduce*	-1	*reduce*	-0.8	*reduce*	-0.4	*reduce*
Type5Tue	1	*reduce*	0.5	*reduce*	-1	*increase*	-1	*increase*
Type5Fri	1	*reduce*	0.2	*reduce*	0.2	*reduce*	-1	*increase*
Type6Tue	1	*reduce*	-0.1	*increase*	-0.2	*increase*	-0.9	*increase*
Type6Thu	-0.4	*reduce*	-0.3	*reduce*	0.2	*increase*	0.9	*increase*
Type9	1	*reduce*	1	*reduce*	0.8	*reduce*	-0.7	*increase*
Type10	-1	*reduce*	-0.1	*no change*	0.02	*no change*	-0.02	*no change*
Type0M1	0.02	*no change*	1	*increase*	1	*increase*	1	*increase*

As shown in the table, as utilization rates are reduced, the best value tends to increase the block size. Based on this result we have decided to proceed with utilization rate of 75% as most of the optimal values are within the experiment region (no extrapolation is required) and also it is compatible with the industry benchmark as published in *OR manager*.

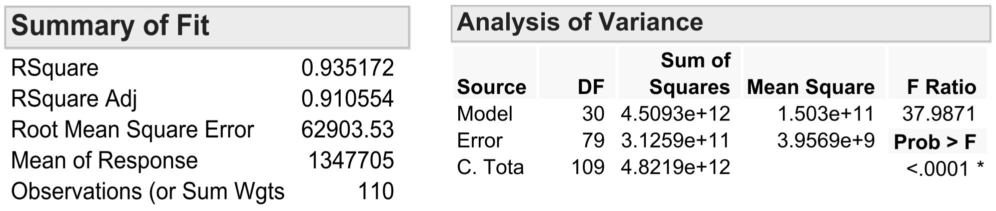
Step 2. In this step, a factor screening experiment was conducted on these remaining eleven important block size factors as well as all the release block factors to identify those which do not have a significant effect on profit. A total of 1073 runs were needed (referring to discrete-valued Walsh functions) in order to estimate all 276 coefficients $(n+2)(n+1)/2$ in a full quadratic model with 22 factors, a design with at least 2^{22-12} factorial, 44 axial and 5 center points runs. Before running this analysis, important block size factors were set at their optimal level, as estimated in Step 1 (which established a new center level for the experiment region) while less-important block size factors were maintained at their current operational level. The result of stepwise regression is shown in Figure 55 and 56. Any effects with F-Ratio less than 2 are eliminated from the model, making it a restricted model.



Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	1484498.5	9664.865	153.60	<.0001*
Type1	-90773.79	7742.889	-11.72	<.0001*
Type2M	4039.2919	7742.889	0.52	0.6034
Type4	-95921.49	7742.889	-12.39	<.0001*
Type5Tue	90963.001	7742.889	11.75	<.0001*
Type5Fri	64777.567	7742.889	8.37	<.0001*
Type6Tue	47473.278	7742.889	6.13	<.0001*
Type6Thu	-39125.01	7742.889	-5.05	<.0001*
Type9	30988.432	7742.889	4.00	0.0001*
Type10	-37920.56	7742.889	-4.90	<.0001*
Release1	-6420.54	7742.889	-0.83	0.4095
Release2	44340.065	7742.889	5.73	<.0001*
Release4	15402.587	7742.889	1.99	0.0501
Release5	65023.132	7742.889	8.40	<.0001*
Type1*Type5Tue	-12596.74	7862.941	-1.60	0.1131
Type2M*Release2	-12275.23	7862.941	-1.56	0.1225
Type4*Type5Tue	6173.9745	7862.941	0.79	0.4347
Type4*Type5Fri	-12497.12	7862.941	-1.59	0.1160
Type4*Type9	19032.261	7862.941	2.42	0.0178*
Type4*Release5	13958.957	7862.941	1.78	0.0797
Type5Fri*Type6Tue	-17576.97	7862.941	-2.24	0.0282*
Type5Fri*Release5	-28169.73	7862.941	-3.58	0.0006*
Type6Tue*Type6Thu	-16671	7862.941	-2.12	0.0371*
Type6Tue*Type10	16911.137	7862.941	2.15	0.0346*
Type6Thu*Release4	21874.317	7862.941	2.78	0.0068*
Type10*Release2	-5753.264	7862.941	-0.73	0.4665
Type1*Type1	-94589.07	39863.81	-2.37	0.0201*
Type2M*Type2M	-248010.5	39863.81	-6.22	<.0001*
Type4*Type4	130969.25	39863.81	3.29	0.0015*
Type9*Type9	-117462.5	39863.81	-2.95	0.0042*
Release1*Release1	101104.04	39863.81	2.54	0.0132*

Figure 55: Parameter estimates for 22 factors



Summary of Fit

RSquare	0.935172
RSquare Adj	0.910554
Root Mean Square Error	62903.53
Mean of Response	1347705
Observations (or Sum Wgts	110

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Ratio
Model	30	4.5093e+12	1.503e+11	37.9871
Error	79	3.1259e+11	3.9569e+9	**Prob > F**
C. Tota	109	4.8219e+12		<.0001 *

Figure 56: Summary of fit and analysis of variance for $^{222-12}$ fractional factorial design

The top five block release factors from our initial experiment (with 32 factors) are shown in Table 38.

Table 38: List of important factors for Stafford block release time

Number	Factor	Main effect	F-ratio
1	Release1	15290	3.1
2	Release2	60108	49.0
3	Release4	28433	10.5
4	Release5	32520	14.5
5	Release9	31376	12.9

The result of screening analysis confirmed that four of these five release factors are among the final thirteen important factors.

From analysis, it was possible to proceed to a response surface experiment to determine the best value for the 13 selected important factors, which include 9 block size and 4 block release time factors. To estimate a full quadratic model with 13 factors, a CCD design was created with 2^{13-5} factorial, 26 axial and 5 center point runs, a total of 287 runs. The analysis of variance and lack of fit for the model are shown in Figure 57.

The estimate of effects for the profit is shown in Figure 58. Analysis of variance indicated significant interactions among the factors. The small *p*-values for the linear and quadratic terms also confirmed that the factors' contribution is significant to the model, and there is curvature in the response surface.

Response Surface for Variable Profit	
Response Mean	1342901
Root MSE	72632
R-Square	0.9311
Coefficient of Variation	5.4086

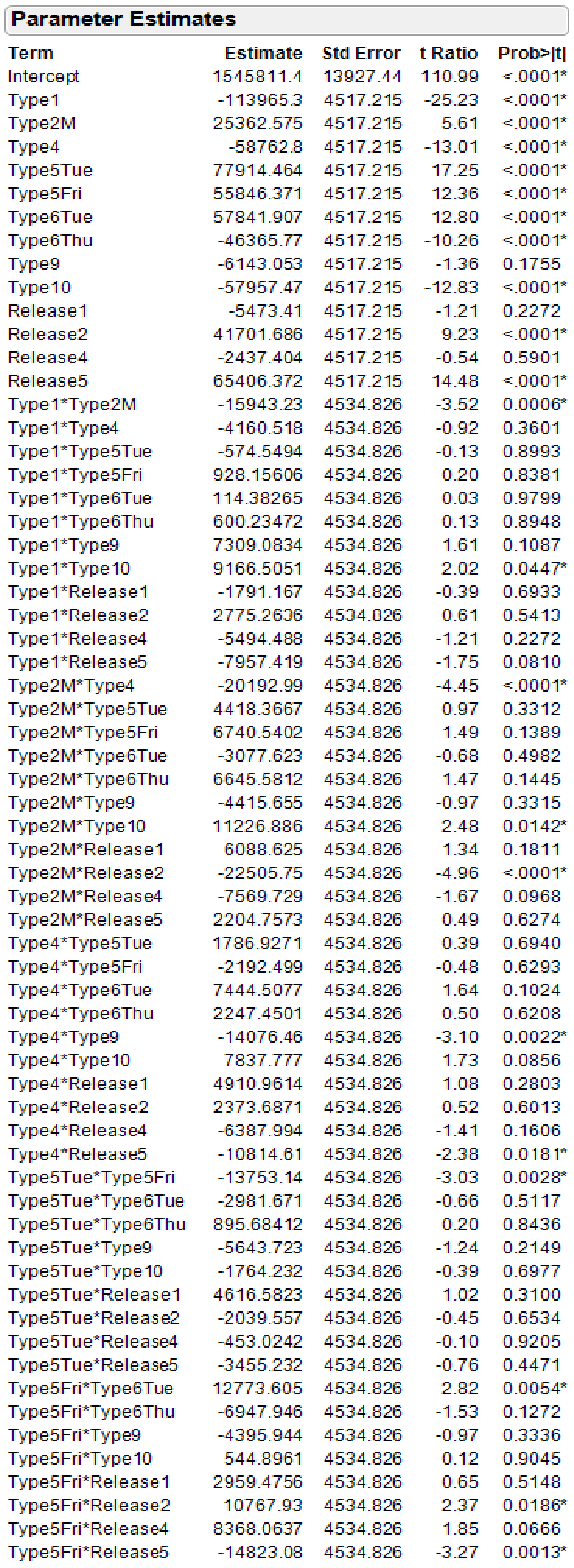
Regression	DF	Type I Sum of Squares	R-Square	F Value	Pr > F
Linear	13	1.0632548E13	0.7625	155.04	<.0001
Quadratic	13	1.3258119E12	0.0951	19.33	<.0001
Crossproduct	78	1.0250125E12	0.0735	2.49	<.0001
Total Model	104	1.2983373E13	0.9311	23.66	<.0001

Residual	DF	Sum of Squares	Mean Square	F Value	Pr > F
Lack of Fit	178	944768183110	5307686422	1.38	0.4231
Pure Error	4	15360366830	3840091707		
Total Error	182	960128549940	5275431593		

Factor	DF	Sum of Squares	Mean Square	F Value	Pr > F
type1	14	3.4979672E12	249854801073	47.46	<.0001
type2M	14	558529034267	39894931019	7.58	<.0001
type4	14	1.131737E12	80838360507	15.36	<.0001
type5Tue	14	1.6480861E12	117720437326	22.36	<.0001
type5Fri	14	1.0316662E12	73690439359	14.00	<.0001
type6Tue	14	998185739137	71298981367	13.54	<.0001
type6Thu	14	596801085848	42628648989	8.10	<.0001
type9	14	122184391735	8727456552	1.66	0.0680
type10	14	999084787563	71363199112	13.56	<.0001
Release1	14	132531103711	9466507408	1.80	0.0416
Release2	14	686858138718	49061295623	9.32	<.0001
Release4	14	118599367901	8471383421	1.61	0.0801
Release5	14	1.2700553E12	90718236611	17.23	<.0001

Figure 57: Summary of fit and analysis of variance for 2^k fractional factorial design

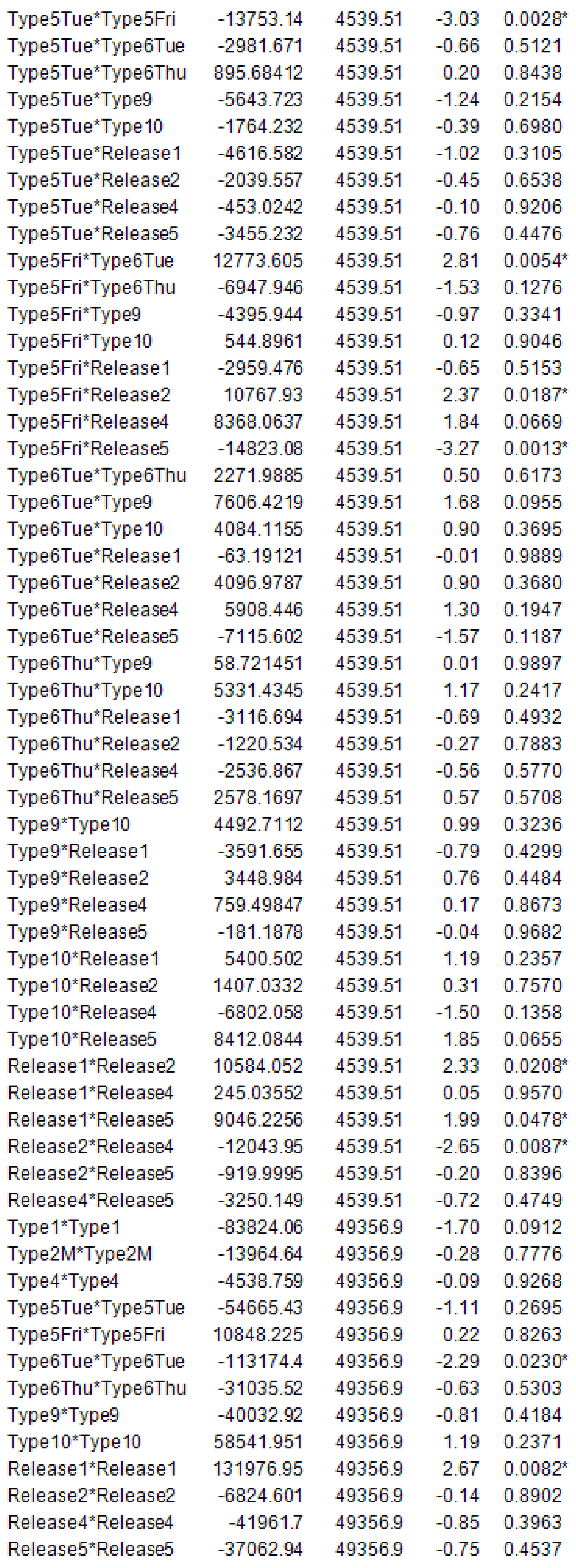
Figure 58: Parameter estimates for 13 factors



Parameter Estimates

Term	Estimate	Std Error	t Ratio	Prob>|t|
Intercept	1545811.4	13927.44	110.99	<.0001*
Type1	-113965.3	4517.215	-25.23	<.0001*
Type2M	25362.575	4517.215	5.61	<.0001*
Type4	-58762.8	4517.215	-13.01	<.0001*
Type5Tue	77914.464	4517.215	17.25	<.0001*
Type5Fri	55846.371	4517.215	12.36	<.0001*
Type6Tue	57841.907	4517.215	12.80	<.0001*
Type6Thu	-46365.77	4517.215	-10.26	<.0001*
Type9	-6143.053	4517.215	-1.36	0.1755
Type10	-57957.47	4517.215	-12.83	<.0001*
Release1	-5473.41	4517.215	-1.21	0.2272
Release2	41701.686	4517.215	9.23	<.0001*
Release4	-2437.404	4517.215	-0.54	0.5901
Release5	65406.372	4517.215	14.48	<.0001*
Type1*Type2M	-15943.23	4534.826	-3.52	0.0006*
Type1*Type4	-4160.518	4534.826	-0.92	0.3601
Type1*Type5Tue	-574.5494	4534.826	-0.13	0.8993
Type1*Type5Fri	928.15606	4534.826	0.20	0.8381
Type1*Type6Tue	114.38265	4534.826	0.03	0.9799
Type1*Type6Thu	600.23472	4534.826	0.13	0.8948
Type1*Type9	7309.0834	4534.826	1.61	0.1087
Type1*Type10	9166.5051	4534.826	2.02	0.0447*
Type1*Release1	-1791.167	4534.826	-0.39	0.6933
Type1*Release2	2775.2636	4534.826	0.61	0.5413
Type1*Release4	-5494.488	4534.826	-1.21	0.2272
Type1*Release5	-7957.419	4534.826	-1.75	0.0810
Type2M*Type4	-20192.99	4534.826	-4.45	<.0001*
Type2M*Type5Tue	4418.3667	4534.826	0.97	0.3312
Type2M*Type5Fri	6740.5402	4534.826	1.49	0.1389
Type2M*Type6Tue	-3077.623	4534.826	-0.68	0.4982
Type2M*Type6Thu	6645.5812	4534.826	1.47	0.1445
Type2M*Type9	-4415.655	4534.826	-0.97	0.3315
Type2M*Type10	11226.886	4534.826	2.48	0.0142*
Type2M*Release1	6088.625	4534.826	1.34	0.1811
Type2M*Release2	-22505.75	4534.826	-4.96	<.0001*
Type2M*Release4	-7569.729	4534.826	-1.67	0.0968
Type2M*Release5	2204.7573	4534.826	0.49	0.6274
Type4*Type5Tue	1786.9271	4534.826	0.39	0.6940
Type4*Type5Fri	-2192.499	4534.826	-0.48	0.6293
Type4*Type6Tue	7444.5077	4534.826	1.64	0.1024
Type4*Type6Thu	2247.4501	4534.826	0.50	0.6208
Type4*Type9	-14076.46	4534.826	-3.10	0.0022*
Type4*Type10	7837.777	4534.826	1.73	0.0856
Type4*Release1	4910.9614	4534.826	1.08	0.2803
Type4*Release2	2373.6871	4534.826	0.52	0.6013
Type4*Release4	-6387.994	4534.826	-1.41	0.1606
Type4*Release5	-10814.61	4534.826	-2.38	0.0181*
Type5Tue*Type5Fri	-13753.14	4534.826	-3.03	0.0028*
Type5Tue*Type6Tue	-2981.671	4534.826	-0.66	0.5117
Type5Tue*Type6Thu	895.68412	4534.826	0.20	0.8436
Type5Tue*Type9	-5643.723	4534.826	-1.24	0.2149
Type5Tue*Type10	-1764.232	4534.826	-0.39	0.6977
Type5Tue*Release1	4616.5823	4534.826	1.02	0.3100
Type5Tue*Release2	-2039.557	4534.826	-0.45	0.6534
Type5Tue*Release4	-453.0242	4534.826	-0.10	0.9205
Type5Tue*Release5	-3455.232	4534.826	-0.76	0.4471
Type5Fri*Type6Tue	12773.605	4534.826	2.82	0.0054*
Type5Fri*Type6Thu	-6947.946	4534.826	-1.53	0.1272
Type5Fri*Type9	-4395.944	4534.826	-0.97	0.3336
Type5Fri*Type10	544.8961	4534.826	0.12	0.9045
Type5Fri*Release1	2959.4756	4534.826	0.65	0.5148
Type5Fri*Release2	10767.93	4534.826	2.37	0.0186*
Type5Fri*Release4	8368.0637	4534.826	1.85	0.0666
Type5Fri*Release5	-14823.08	4534.826	-3.27	0.0013*

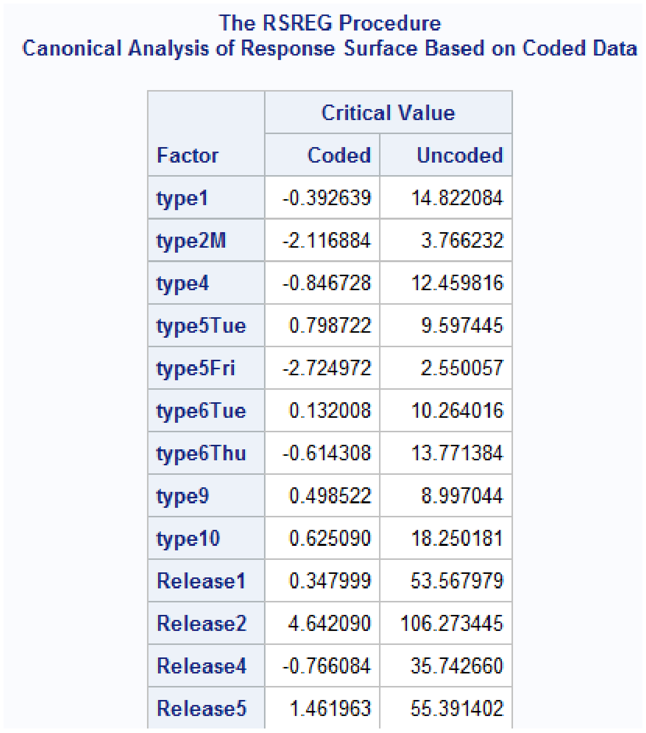
Figure 58 Continued: Parameter estimates for 13 factors



Type5Tue*Type5Fri	-13753.14	4539.51	-3.03	0.0028*
Type5Tue*Type6Tue	-2981.671	4539.51	-0.66	0.5121
Type5Tue*Type6Thu	895.68412	4539.51	0.20	0.8438
Type5Tue*Type9	-5643.723	4539.51	-1.24	0.2154
Type5Tue*Type10	-1764.232	4539.51	-0.39	0.6980
Type5Tue*Release1	-4616.582	4539.51	-1.02	0.3105
Type5Tue*Release2	-2039.557	4539.51	-0.45	0.6538
Type5Tue*Release4	-453.0242	4539.51	-0.10	0.9206
Type5Tue*Release5	-3455.232	4539.51	-0.76	0.4476
Type5Fri*Type6Tue	12773.605	4539.51	2.81	0.0054*
Type5Fri*Type6Thu	-6947.946	4539.51	-1.53	0.1276
Type5Fri*Type9	-4395.944	4539.51	-0.97	0.3341
Type5Fri*Type10	544.8961	4539.51	0.12	0.9046
Type5Fri*Release1	-2959.476	4539.51	-0.65	0.5153
Type5Fri*Release2	10767.93	4539.51	2.37	0.0187*
Type5Fri*Release4	8368.0637	4539.51	1.84	0.0669
Type5Fri*Release5	-14823.08	4539.51	-3.27	0.0013*
Type6Tue*Type6Thu	2271.9885	4539.51	0.50	0.6173
Type6Tue*Type9	7606.4219	4539.51	1.68	0.0955
Type6Tue*Type10	4084.1155	4539.51	0.90	0.3695
Type6Tue*Release1	-63.19121	4539.51	-0.01	0.9889
Type6Tue*Release2	4096.9787	4539.51	0.90	0.3680
Type6Tue*Release4	5908.446	4539.51	1.30	0.1947
Type6Tue*Release5	-7115.602	4539.51	-1.57	0.1187
Type6Thu*Type9	58.721451	4539.51	0.01	0.9897
Type6Thu*Type10	5331.4345	4539.51	1.17	0.2417
Type6Thu*Release1	-3116.694	4539.51	-0.69	0.4932
Type6Thu*Release2	-1220.534	4539.51	-0.27	0.7883
Type6Thu*Release4	-2536.867	4539.51	-0.56	0.5770
Type6Thu*Release5	2578.1697	4539.51	0.57	0.5708
Type9*Type10	4492.7112	4539.51	0.99	0.3236
Type9*Release1	-3591.655	4539.51	-0.79	0.4299
Type9*Release2	3448.984	4539.51	0.76	0.4484
Type9*Release4	759.49847	4539.51	0.17	0.8673
Type9*Release5	-181.1878	4539.51	-0.04	0.9682
Type10*Release1	5400.502	4539.51	1.19	0.2357
Type10*Release2	1407.0332	4539.51	0.31	0.7570
Type10*Release4	-6802.058	4539.51	-1.50	0.1358
Type10*Release5	8412.0844	4539.51	1.85	0.0655
Release1*Release2	10584.052	4539.51	2.33	0.0208*
Release1*Release4	245.03552	4539.51	0.05	0.9570
Release1*Release5	9046.2256	4539.51	1.99	0.0478*
Release2*Release4	-12043.95	4539.51	-2.65	0.0087*
Release2*Release5	-919.9995	4539.51	-0.20	0.8396
Release4*Release5	-3250.149	4539.51	-0.72	0.4749
Type1*Type1	-83824.06	49356.9	-1.70	0.0912
Type2M*Type2M	-13964.64	49356.9	-0.28	0.7776
Type4*Type4	-4538.759	49356.9	-0.09	0.9268
Type5Tue*Type5Tue	-54665.43	49356.9	-1.11	0.2695
Type5Fri*Type5Fri	10848.225	49356.9	0.22	0.8263
Type6Tue*Type6Tue	-113174.4	49356.9	-2.29	0.0230*
Type6Thu*Type6Thu	-31035.52	49356.9	-0.63	0.5303
Type9*Type9	-40032.92	49356.9	-0.81	0.4184
Type10*Type10	58541.951	49356.9	1.19	0.2371
Release1*Release1	131976.95	49356.9	2.67	0.0082*
Release2*Release2	-6824.601	49356.9	-0.14	0.8902
Release4*Release4	-41961.7	49356.9	-0.85	0.3963
Release5*Release5	-37062.94	49356.9	-0.75	0.4537

Figure 58 Continued: Parameter estimates for 13 factors

Using the general solution in equation 4.3, it was possible to locate the stationary point, as shown below:



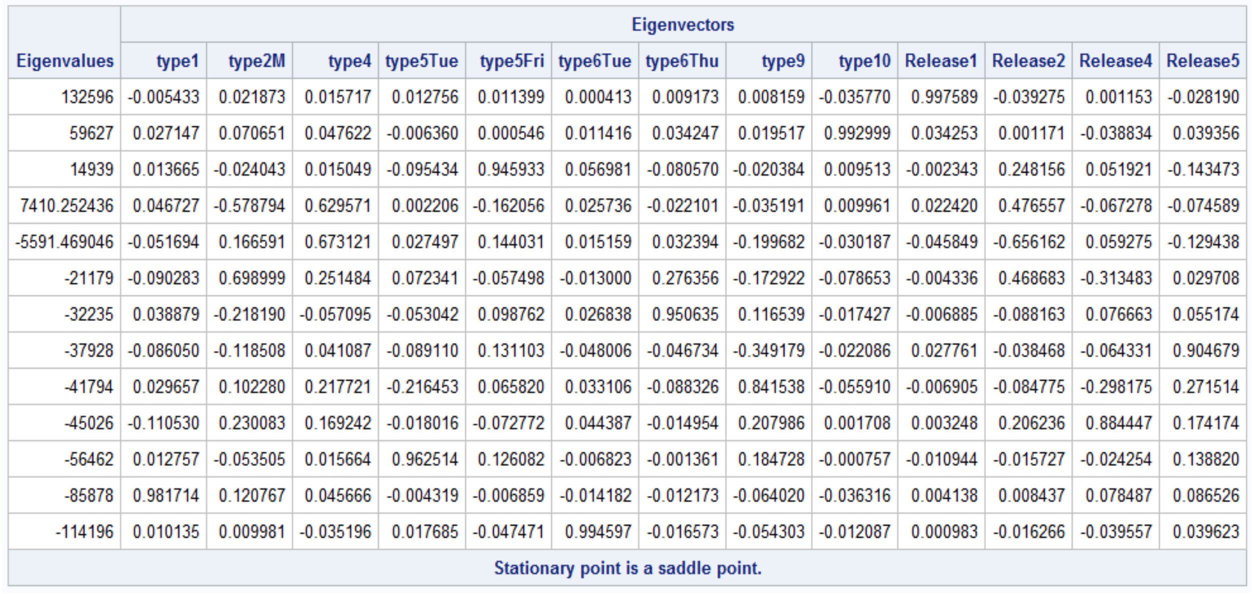
The RSREG Procedure
Canonical Analysis of Response Surface Based on Coded Data

Factor	Critical Value	
	Coded	Uncoded
type1	-0.392639	14.822084
type2M	-2.116884	3.766232
type4	-0.846728	12.459816
type5Tue	0.798722	9.597445
type5Fri	-2.724972	2.550057
type6Tue	0.132008	10.264016
type6Thu	-0.614308	13.771384
type9	0.498522	8.997044
type10	0.625090	18.250181
Release1	0.347999	53.567979
Release2	4.642090	106.273445
Release4	-0.766084	35.742660
Release5	1.461963	55.391402

Figure 59: Canonical Analysis for 13 factors

The predicted response at the stationary point was $1,664,242, with some extrapolation required to determine some of the larger predicted values, 4 out of 13 predicted values are located outside the experiment region(outside the range of [-1, +1]). Canonical analysis was conducted on the Hessian matrix B to classify stationary point into maximum, minimum or saddle point. Our finding of both positive and negative eigenvalues, as shown below, indicated that the stationary point is a saddle point.

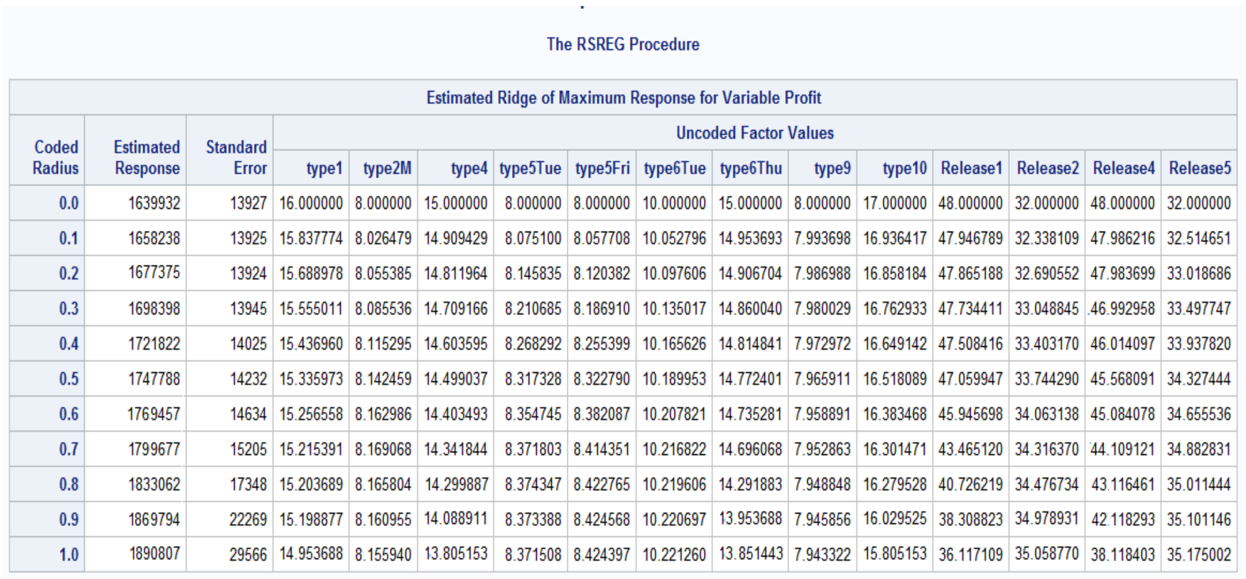
To determine the best path toward improving the profit (response) away from the saddle point, a ridge analysis was conducted. The path was started from midpoint location in the experiment region. The result is summarized in Figure 60.



	Eigenvectors												
Eigenvalues	type1	type2M	type4	type5Tue	type5Fri	type6Tue	type6Thu	type9	type10	Release1	Release2	Release4	Release5
132596	-0.005433	0.021873	0.015717	0.012756	0.011399	0.000413	0.009173	0.008159	-0.035770	0.997589	-0.039275	0.001153	-0.028190
59627	0.027147	0.070651	0.047622	-0.006360	0.000546	0.011416	0.034247	0.019517	0.992999	0.034253	0.001171	-0.038834	0.039356
14939	0.013665	-0.024043	0.015049	-0.095434	0.945933	0.056981	-0.080570	-0.020384	0.009513	-0.002343	0.248156	0.051921	-0.143473
7410.252436	0.046727	-0.578794	0.629571	0.002206	-0.162056	0.025736	-0.022101	-0.035191	0.009961	0.022420	0.476557	-0.067278	-0.074589
-5591.469046	-0.051694	0.166591	0.673121	0.027497	0.144031	0.015159	0.032394	-0.199682	-0.030187	-0.045849	-0.656162	0.059275	-0.129438
-21179	-0.090283	0.698999	0.251484	0.072341	-0.057498	-0.013000	0.276356	-0.172922	-0.078653	-0.004336	0.468683	-0.313483	0.029708
-32235	0.038879	-0.218190	-0.057095	-0.053042	0.098762	0.026838	0.950635	0.116539	-0.017427	-0.006885	-0.088163	0.076663	0.055174
-37928	-0.086050	-0.118508	0.041087	-0.089110	0.131103	-0.048006	-0.046734	-0.349179	-0.022086	0.027761	-0.038468	-0.064331	0.904679
-41794	0.029657	0.102280	0.217721	-0.216453	0.065820	0.033106	-0.088326	0.841538	-0.055910	-0.006905	-0.084775	-0.298175	0.271514
-45026	-0.110530	0.230083	0.169242	-0.018016	-0.072772	0.044387	-0.014954	0.207986	0.001708	0.003248	0.206236	0.884447	0.174174
-56462	0.012757	-0.053505	0.015664	0.962514	0.126082	-0.006823	-0.001361	0.184728	-0.000757	-0.010944	-0.015727	-0.024254	0.138820
-85878	0.981714	0.120767	0.045666	-0.004319	-0.006859	-0.014182	-0.012173	-0.064020	-0.036316	0.004138	0.008437	0.078487	0.086526
-114196	0.010135	0.009981	-0.035196	0.017685	-0.047471	0.994597	-0.016573	-0.054303	-0.012087	0.000983	-0.016266	-0.039557	0.039623

Stationary point is a saddle point.

Figure 60: Canonical curvature, Eigenvalues and Eigenvectors for 13 factors



The RSREG Procedure

Estimated Ridge of Maximum Response for Variable Profit

Coded Radius	Estimated Response	Standard Error	Uncoded Factor Values												
			type1	type2M	type4	type5Tue	type5Fri	type6Tue	type6Thu	type9	type10	Release1	Release2	Release4	Release5
0.0	1639932	13927	16.000000	8.000000	15.000000	8.000000	8.000000	10.000000	15.000000	8.000000	17.000000	48.000000	32.000000	48.000000	32.000000
0.1	1658238	13925	15.837774	8.026479	14.909429	8.075100	8.057708	10.052796	14.953693	7.993698	16.936417	47.946789	32.338109	47.986216	32.514651
0.2	1677375	13924	15.688978	8.055385	14.811964	8.145835	8.120382	10.097606	14.906704	7.986988	16.858184	47.865188	32.690552	47.983699	33.018686
0.3	1698398	13945	15.555011	8.085536	14.709166	8.210685	8.186910	10.135017	14.860040	7.980029	16.762933	47.734411	33.048845	46.992958	33.497747
0.4	1721822	14025	15.436960	8.115295	14.603595	8.268292	8.255399	10.165626	14.814841	7.972972	16.649142	47.508416	33.403170	46.014097	33.937820
0.5	1747788	14232	15.335973	8.142459	14.499037	8.317328	8.322790	10.189953	14.772401	7.965911	16.518089	47.059947	33.744290	45.568091	34.327444
0.6	1769457	14634	15.256558	8.162986	14.403493	8.354745	8.382087	10.207821	14.735281	7.958891	16.383468	45.945698	34.063138	45.084078	34.655536
0.7	1799677	15205	15.215391	8.169068	14.341844	8.371803	8.414351	10.216822	14.696068	7.952863	16.301471	43.465120	34.316370	44.109121	34.882831
0.8	1833062	17348	15.203689	8.165804	14.299887	8.374347	8.422765	10.219606	14.291883	7.948848	16.279528	40.726219	34.476734	43.116461	35.011444
0.9	1869794	22269	15.198877	8.160955	14.088911	8.373388	8.424568	10.220697	13.953688	7.945856	16.029525	38.308823	34.978931	42.118293	35.101146
1.0	1890807	29566	14.953688	8.155940	13.805153	8.371508	8.424397	10.221260	13.851443	7.943322	15.805153	36.117109	35.058770	38.118403	35.175002

Figure 61: Ridge analysis for 13 factors

The predicted response reached \$2M, 44% higher than the estimated profit of \$1.4M at the current operational level (the experiment region at the center point). The modified stationary point is thus located at:

$$X_{s'} = \begin{bmatrix} -0.77 \\ 0 \\ -0.95 \\ 0.1 \\ 0.2 \\ -0.08 \\ -1 \\ 0 \\ -0.88 \\ -1 \\ 1 \\ -1 \\ 1 \end{bmatrix}$$

This result suggests that in order to achieve a superior OR allocation plan, four out of the nine block factors--Type 1, Type 4, Type 6 and Type 10 --would have to be reduced further. However not all of the reductions in block size are due to block release optimization. For Type 1 and Type 4, introducing the block release factor into the model results in a further reduction of block sizes, but, when release effects is excluded from the model, even further reductions are required in order to achieve optimization under a lower utilization rate (referring to Table 37, superior block schedule decision under 60% utilization rate).

Thus, we have excluded Types 1 and 4 block size reduction from our analysis of the marginal benefits of the joint optimization of allocation and release policy. In addition, the effect of introducing block release factors into the model suggests that setting earlier release times for Block Types 2 and 5 and postponing release times for Block Type 1 and Type 4 would be a viable way to improve profits or to reduce overall waiting cost. More explanation, including the reasoning for and potential financial gains associated with these results, is provided in the next chapter.

Step 3. The third and final step will be to optimize the scheduling policy so that overall waiting costs are minimized.

The application of reserve policy in case scheduling for the multi-priority patient

As noted in Chapter 2, the third and the last step in the process of optimizing individual patient scheduling must center on daily decisions about patient scheduling policies where the waiting costs vary—that is, in cases of multiple-priority patients with semi-urgent needs. This step in the optimization process applies only to those patients who are scheduled in open hours or in released hours since for the patients who are scheduled in home blocks, their surgery times are assigned on a first-come-first-served basis. We assume the best release policy remain unchanged from the result of the joint-optimization of block hour and release policy.

Around 25% of Stafford’s surgeries are performed either in open hours or in released hours. This means that an improved appointment policy can generate an improved overall yield (profit) from these patients by reducing their waiting times. The previous chapter reported on the benefit of our reserve policy for the scheduling of two-priority patients. These findings can be extended to multi-priority patient scheduling by introducing booking limits (protection levels) for each priority level. The protection level is calculated for every combination of classes while comparing each class with all other priority classes. The results of the simulation model presented here incorporate the idea of accepting/postponing requests for surgeries from several competing classes of patients who present fluctuating demands and service hours. Traditionally, the primary concern in the healthcare operations literature has been how to reduce operating costs and increase OR utilization. Since the survival and prosperity of the surgical suite in the long run also depends on the revenue it generates, it is also crucial to investigate how to better manage the mix of patients that request elective surgeries, with the goal of increasing the expected revenue generated by the surgical department and reducing the risk of delays and cancellations of surgical cases (Stanciu et al., 2010).

The problem of allocating service capacity among several competing customer classes, who arrive randomly over a period of time, has been studied in diverse applications including airlines, hotels and car rentals. In particular, airline Revenue Management (RM) has been studied thoroughly; see McGill and Van Ryzin (1999) and Talluri and Van Ryzin (2004) for detailed reviews. Whereas capacity reservation is also an important aspect of health care access management, there are important differences that make it difficult to simply “tweak” existing models to fit the needs of the health care industry. For example, of the various models suggested for airline RM, comparisons with the Expected Marginal Seat Revenue (EMSR) model (see Belobaba, 1989) help to highlight the complexity of healthcare operations (Gupta and Denton, 2008).

To outline the main differences between Stafford's multi-priority policy and the earlier two-priority case, we briefly review the following:

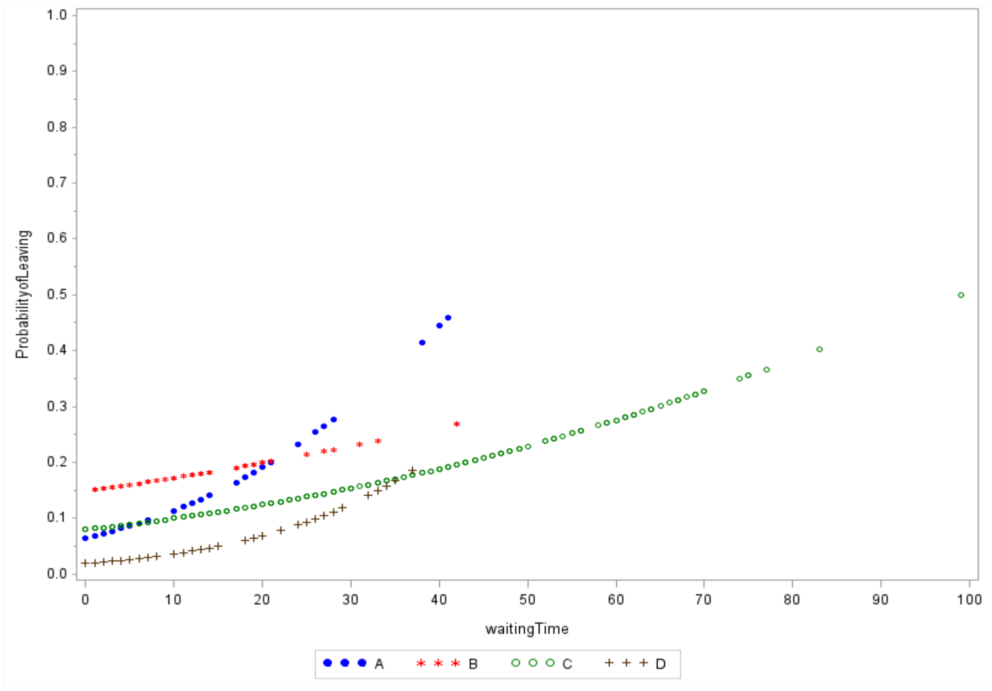
Urgency level. In the 2-priority case, the waiting cost function was linear (e.g., $f(t, p_b) = c + p_b * t, with\ the\ same\ intercept\ c\ for\ type\ a\ and\ type\ b$), which means one type of patient is always dominant (or has higher priority). In contrast, at Stafford, the waiting cost function follows a log function, where the priority of patients changes as a function of waiting time. The probability of cancellation and the expected revenue per case defines the overall cost of waiting for each patient type. Expected revenue per case is calculated based on the average revenue per hour of operation (including pre-op, surgery and post-op). Due to the confidentiality of financial data, proportional revenue per hour is used for comparing patient types. Waiting costs are quantified as the potential loss of profit due to a patient leaving the system without getting the surgery. It can be represented as follows.

$$\sum_{i=1}^{s} \sum_{j=0}^{N} r_i \pi_j \ ;$$

r_i,	*Expected revenue per case, associated with surgeon i*
π_j,	*Probability of cancellation, given j waiting time unit*

Stafford does not have any type of patient who is the highest priority at all times. Waiting penalties ($ profit loss) given waiting time are presented in Figure 62. Type B patients have the highest priority until time 45, after which Type A becomes the highest priority. This function makes the assigning decision more challenging because the decision must depend on the state of the system.

Secondary arrival rate. In order to calculate the protection level for multi-class patients, the arrival rate for all types of patients must first be calculated. The arrival rate is not the original arrival rate of each patient type but rather the arrival demand for a released block or open hours



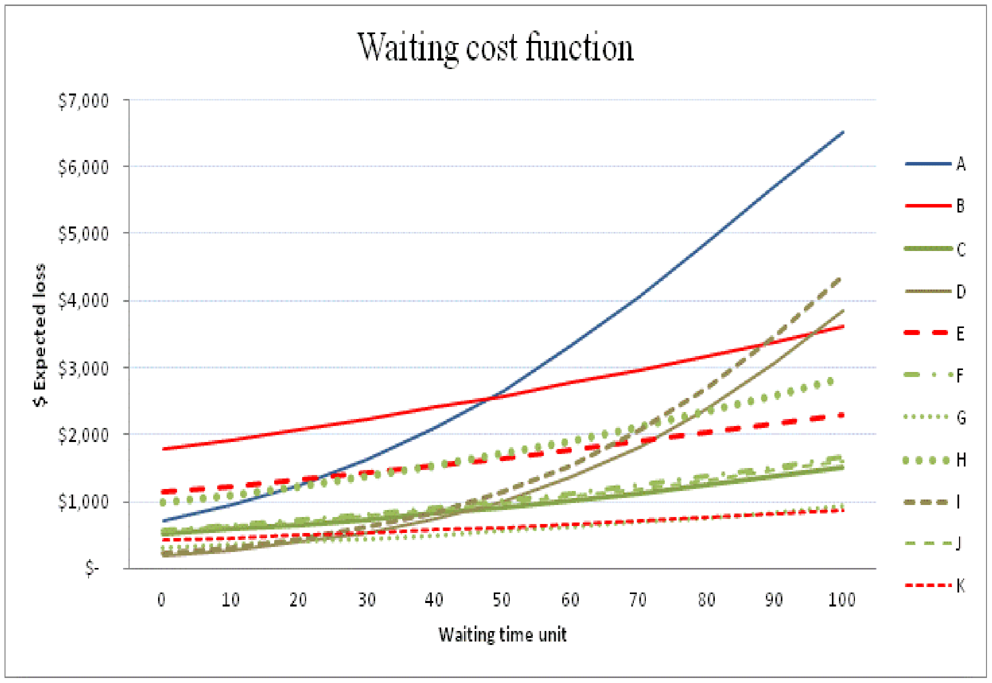


Figure 62: a) Probability of cancellation given waiting time based on historical data b) Waiting penalty ($ profit loss) given waiting time calculated based on probability of cancellation and revenue

(we refer to this as “secondary arrival demand”), which may be completely different from the initial arrival rate due to differences in original block hours, urgency levels and original arrival rates of patients.

To estimate the secondary arrival rate (and overall demand) for released time and open ORs, the Stafford simulation was run under the superior scheduling block time and release policy (Step 2 optimization). This result has been used to estimate the optimal secondary demand rate for off-block hours (Table 39).

Table 39: Secondary arrival rate driven from simulation under superior block size/release policy

Specialty	Surgeon/Group	Initial Mean inter-arrival time (day)	Secondary Mean inter-arrival time (day)
EAR, NOSE AND THROAT	G	2.53	4.46
GENERAL / VASCULAR	H	3.20	17.67
	B	3.17	13.78
	E	1.06	7.05
OBSTETRICS/GYNECOLOGY	F	1.23	6.80
ORTHO	I	5.32	19.40
	A	2.45	4.22
	D	1.50	5.44
PLASTICS	K	5.93	47.88
PODIATRY	C	1.26	5.88
	J	4.80	20.34

Surgeons’ preferences and available days of the week to do surgery. In the 2-priority patient case, surgeons were available and willing to perform surgery on any day. In contrast, at Stafford, surgeons are only available to conduct surgery on a limited number of days. This additional restriction make the scheduling decision even more challenging since the optimal scheduling decisions will be a function of waiting time as well as the day of week for the available released hours. Due to the Markov property (memoryless), it does not matter if we have released hours for higher priority patient before this available time, so each available released block is evaluated separately but we will give a higher penalty weight to those patients whose surgeons have less opportunity to assign their case to open and/or released hours.

Open blocks versus released block. There is a slight difference between available capacity in open and released OR hours. As hospitals already have committed staff for released hours, schedulers tend to fill released hours first and then look for possible open ORs. This difference results in revenue lost for unfilled released hours, while there is no penalty for unfilled open ORs. To incorporate this difference in our study, we have assumed two scenarios for cost per hour of open hours of 1.5 and 2.0 times the cost per hour of released block. This cost difference makes it possible to compare the trade-off between the cost of leaving released hours unfilled and the cost of unnecessary open hours.

These features and the need to accommodate urgent demand, make it more difficult to apply popular heuristic revenue management methods such as the Expected Marginal Seat Revenue (EMSR) model, for the surgery scheduling (access) decisions. Currently, at Stafford, surgeries are assigned to off-block hours based on first-come-first-serve strategy. The proposed reserve policy at Stafford exploits the multi-priority patients in assigning patients as follow,

---

Algorithm 2. A scheme of Stafford proposed reserve policy algorithm

---

Given a patient C has exceeded the maximum release time threshold, T_{maxI} and is eligible to assign to release/open time.

1. Obtain
   a. Patient type, $I = \{A, B, \dots, K\}$
   b. Arrival rates (secondary arrival rate), λ_{sI}
   c. Maximum release time of each patient type, T_{maxI}
   d. Surgeons' preference matrix , M
2. Find the first available spot, R_{t1} for patient C in released hours, which
   a. matches with the surgeon's preference matrix (1.d) and
   b. is within the maximum release time of the patient (1.c, refer to block release policy)

3. Look up the surgeons' preference table for this day (the weekday associated with R_{t1}) and identify all patient types whom their surgeon(s) can perform surgery on this day (e.g., Monday)
4. Calculate the priority (expected waiting loss) of selected patients, $P_{c,comp}$, based on secondary arrival rates, λ_{sI} , and the waiting cost at time unit T_1, where T_1 is calculated based on $T_1 = (R_{t1} - current\ time) > 0$, and sort patients in ascending priority order

$$\begin{cases} P_c = \pi_{cs}(T_1) * r_c & ; case\ c \\ P_{comp} = {}^{1}/_{T_1} * \left(1 - e^{-\lambda_k T_1)}\right) * \gamma \sum_{t=0}^{T_1} \pi_{comp}(t) * r_{comp} & ; competitor\ cases \end{cases}$$

- r_c, r_{comp}: *\$ Revenue per case*
- ${}^{1}/_{T_1} * \left(1 - e^{-\lambda_k T_1)}\right)$: *Probability of arrival for competitor cases in the next* T_1*time unit*
- $\gamma \sum_{t=0}^{T_1} \pi_{comp}(t)$: *Probability of leaving for each competitor case, given waiting time t;* γ: $daily\ discounted\ factor$
- $P_{comp} = {}^{1}/_{T_1} * \left(1 - e^{-\lambda_k T_1}\right) * \gamma \sum_{t=0}^{T_1} \pi_{comp}(t) * r_{comp}$:
  E$xpected\ waiting\ cost\ penalty\ of\ competitor$

5. Iterate until simulation time ends,
   a. If the requesting patient type *C*, has the highest priority rank, assign it to this available spot R_{t1},
   b. Else if, in spite of allocation of higher priority type(s) to this released block, and there is still enough space remaining, assign the case *C* to R_{t1}
6. Else, find the next open/release hour, R_{t2} for all higher priority patients and this patient *C*, which
   a. matches with surgeon's preference matrix and
   b. $R_{t2} \leq$ maximum time of the patient (refer to block release policy)
7. Calculate cost of postponing each higher priority patients to the next available spot, R_{t2} as follow, $T_2 = (R_{t2} - current\ time) > 0$

$$ {}^{1}/_{T_2} * \left(1 - e^{\lambda_k T_2}\right) * \gamma \sum_{t=0}^{T_2} \pi_{comp}(t) * r_{comp} * P_{openB}$$

- $P_{openB} \geq 1$: Penalty of using open hour versus utilizing current release hour

$$P_{openB} = \begin{cases} \dfrac{Cost\ per\ hour\ of\ open\ block}{cost\ per\ hour\ of\ release\ block}\ ; & R_{t2}\ belongs\ to\ Open\ hours \\ 1\ ; & R_{t2}\ belongs\ to\ release\ hours \end{cases}$$

a. If the cost of postponing case C to the next available spot is higher than postponing higher priority case(s), or if following equation holds, schedule case C to the first available space, R_{t1},

$$P_c = \pi_{cs}(T_2) * r_c * P_{openB} > \max_{k \in comp} \left\{ {}^{1}/_{T_2} * \left(1 - e^{-\lambda_k T_2}\right) * \gamma \sum_{t=0}^{T_2} \pi_k(t) * r_k * P_{openB} \right\}$$

b. Else, return to 6.

An experiment is conducted for the proposed reserve policy on the Stafford simulation from the step 2 final result, Stafford under superior block size and block release policy. The same duration of 4800 time units is selected in order to ensure the comparability of results. We have a fixed amount of daily release/open capacity. Patients of different priorities who have exceeded their maximum wait time arrive randomly over time and it must be decided whether to assign a case to the first available release block or postpone it to the later time (either open or own block) in order to reserve earlier hours to higher priority patients. There is always a risk of underutilization in the event that release hours are not fully used. On the other hand, proceeding with filling released hours with lower priority cases will incur additional waiting cost for higher priority patients. In order to capture this tradeoff in the simulation, we ran the simulation under two scenarios; cost of open hours being 1.5, and 2.0 times cost of released block. The higher the cost, the greater the tendency to assign cases to released hours rather than open hours (thereby penalize under-utilization more). We will evaluate the overall cost (including underutilization and waiting cost) under these scenarios. We are interested in minimizing the total expected cost (or maximizing expected profit) over a finite planning horizon considering patient priority. More explanation, including the reasoning for and potential financial gains/loss associated with these results, is provided in the next chapter.

Chapter 6 : Summary and Conclusion

Figure 62, shows the marginal profit gain at each step of our optimization process. To summarize, we took two steps to estimate improved values for block schedule and release policy factors. The purpose of Step 1 of the process was to find superior block schedule allocations for multi-priority patients in the experiment region. The results of Step 1 were used in Step 2 to establish a joint optimization of allocation and block release policy. Overall, we found that profits could be increased 44% above the current Stafford operational level, with 21% of the improvement coming in Step 1 through the combination of better alignment of the OR time allocations with the requirements of each surgical specialty and relaxed utilization rate assumptions.

An additional 10% improvement was gained merely by optimizing the release block policy with reference to the patient waiting penalty function or priority-level and the block utilization rate. The remaining 9% was due to the effect of further OR block size reductions of Types 1 and 4 surgeons, who had been excluded from our analysis of the marginal benefits of the joint optimization of allocation. The first step of optimization involved reducing OR block sizes when there was minimum impact on patients' waiting time or increasing OR block sizes when the additional gain in terms of waiting-cost reductions outweighed the costs of incremental block hours.

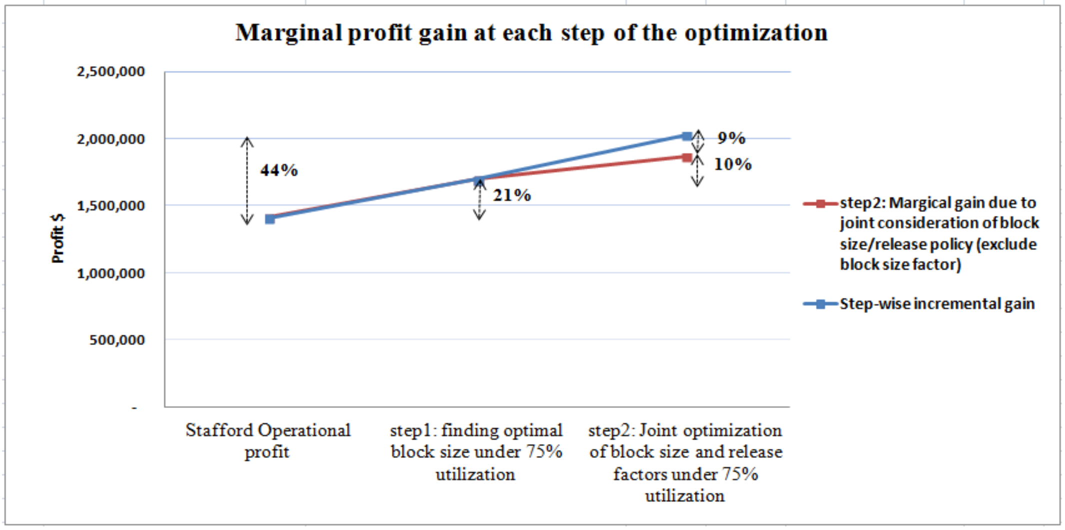


Figure 63: Marginal profit gain at each step of optimization

In contrast, the second step emphasized improving utilization by means of an improved release policy, based on the principle that the value of postponing release time must be calculated in terms of the valued gained by dedicating space to higher priority patients or releasing blocks earlier to provide more access to higher priority patients and thereby reducing overall waiting time.

Table 40 summarizes the step-wise optimization results and shows how these results compare with the current operational level at Stafford. As shown below, both OR block costs and waiting costs can be reduced through these two steps. In Step 1 alone, waiting costs can be reduced by 5% by means of 1) improved access to earlier off-block OR times for high priority patients via increasing open OR hours, 2) the addition of OR block times for medium priority surgeons with more frequent schedules, and the provision of better opportunities for high priority patients to use unfilled blocks.

Table 40: Summary of step-wise optimization results

	Stafford (under 64% utilization)	Optimized block (under 75% utilization)	% change	Optimized block and Release (under 75% utilization)	% change
Total # of performed surgeries	792	803	1%	809	1%
Revenue	$9,633,062	$9,734,990	1%	$9,849,774	1%
Total block cost	$6,635,124	$6,519,367	-2%	$6,345,443	-3%
Total waiting cost	$1,590,791	$1,519,078	-5%	$1,478,226	-3%
Profit	$1,407,147	$1,696,545	21%	$2,026,105	19%
Total occupied block hours	1601	1603	0%	1595	-1%
Total occupied open block hrs	306	405	32%	414	2%
Total occupied release block hrs	73	83	14%	103	24%
Initial block hours available	3155	3047	-3%	2984	-3%
Utilization	64%	70%	6%	73%	3%

As explained above, in the first step, the total number of occupied open blocks was increased by 32%, thereby reducing the waiting time of those patients who couldn’t find available OR times in another surgeons’ released block or whose first available space in surgeon’s block was extremely delayed. Step 2, in turn, generated a significant change in occupied release hours due to a policy for earlier release time and the channeling of high priority patients into unfilled spaces, thereby improving utilization and waiting time. Although all the simulations were run for the same duration (of 4800 time units) in order to ensure the compatibility of results, the model shows that revenue as well as the number of performed surgeries can be increased through step-wise optimization. Revenue is generated based on the number of performed surgeries (exclude waiting and canceled cases) by the end of the simulation time. As utilization increases and waiting times decrease through the optimization steps, Stafford can perform more surgeries over the same period of time. Although the rate of utilization improved from 64% to 70% in the first step of the model's implementation, it did not reach the target rate of 75% due to variations in demand such as variation in case durations, arrival rates (coefficient of variation), the probability of cancellations (multi-priority patients) and restrictions on providing services such as surgeons' preferences or limitations in the availability of equipment. Although we cannot control such variables, we can reduce their effects by offering more flexibility to surgeons to assign their cases based on a first-come-first-served rule. An optimal release policy, which considers both differences in waiting costs and multi-class patients, can effectively control variation and improve utilization across surgeons. As can be seen in Figure 63, the utilization rate improved to 73% in the second step of the model's implementation thanks to modifications in release times. Figure 64 displays the resulting utilization rates across patient types.

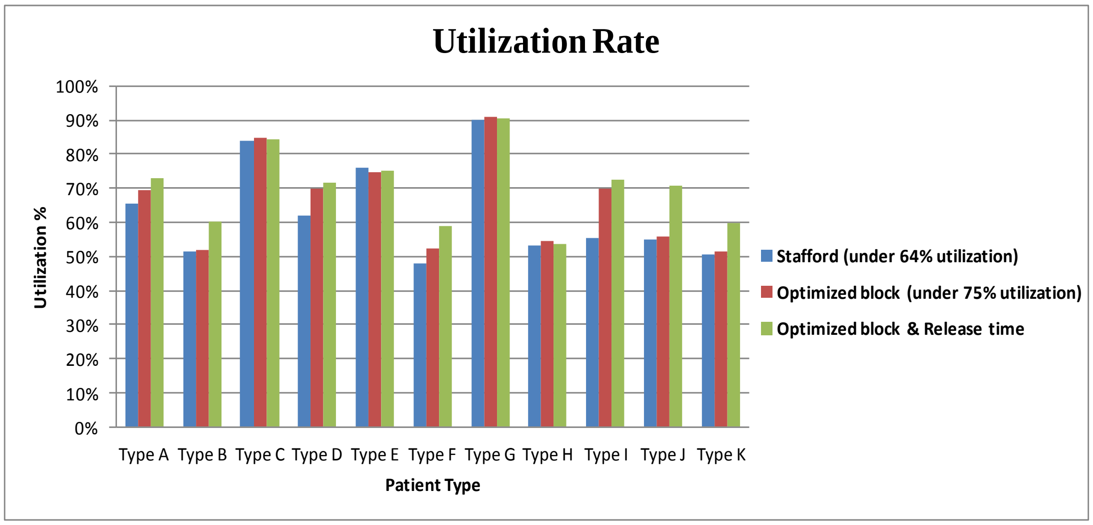


Figure 64: Utilization rate across surgeon types and through each optimization step

Utilization has improved for almost all groups, partially due to the modification of the utilization rate from the current rate of 64% to 75% and partially due to the optimization of the release policy considering the waiting-penalty function of multi-class patients. Although the same percentage of block size reduction is achieved in both steps (a 3% reduction in initial block hours available), the second optimization step makes a greater contribution to utilization improvement by providing fair access to all patients via a better release policy.

Table 41 lays out the result of the improved block schedule allocations across specialties. In the model, the overall block size is reduced by 6% with a combination of increases and reductions in block size. The highest percentage of block size reductions occur for Types A, D and I, all ortho surgeons. These types of surgery have the highest priority with the steepest waiting cost function and the highest coefficient of variation in arrival rates. These surgeons also have the shortest scheduling lead time, so keeping the block size unchanged and releasing blocks earlier would not be a viable way to reduce waiting times or block costs. The results indicate that, for these high priority patients, it is more valuable (lower waiting penalties) to have fewer allocated blocks aggregated on one day while allocating more surgeries to off-block hours with higher flexibility or spreading surgeons' block hour across multiple days instead of one day every other week.

Table 41: Summary of change in block schedule allocations in stage-wise optimization

Surgeon /Group	**Stafford (under 64% utilization)**	**Optimized block (under 75% utilization)**	**Optimized block and Release (under 75% utilization)**	**Optimal Stafford (under 75% utilization)**	**% change**
Type A	192	-27%	-7%	130	**-32%**
Type B	239	0%	0%	240	**0%**
Type C	288	0%	0%	288	**0%**
Type D	312	-22%	-14%	202	**-35%**
Type E	480	5%	0%	502	**5%**
Type F	648	0%	-6%	608	**-6%**
Type G	96	0%	0%	96	**0%**
Type H	144	0%	0%	144	**0%**
Type I	120	-40%	0%	72	**-40%**
Type J	120	0%	-20%	96	**-20%**
Type K	96	0%	0%	96	**0%**
Open block hrs	416	23%	0%	512	**23%**
Total	**3151**	**-3%**	**-3%**	**2984**	**-5%**

It is also clear from the improved block time allocation decision result in Table 41 that an increase is needed in block sizes--both in dedicated block times and open block hours-- to provide more access to higher priority patients notwithstanding the high cost of block hours. Figure 65 shows how block size reallocation and overall block size reduction affected waiting times across different types of surgery, as well as how each of the optimization steps contributed to the shift in the distribution of OR services, and thus to the rebalancing of resources across competing classes of demand. It is obvious that waiting time has not been reduced for all types of patients nor is the reduction rate the same among groups. This difference in outcome is due to differences in the sensitivity of patients to waiting times.

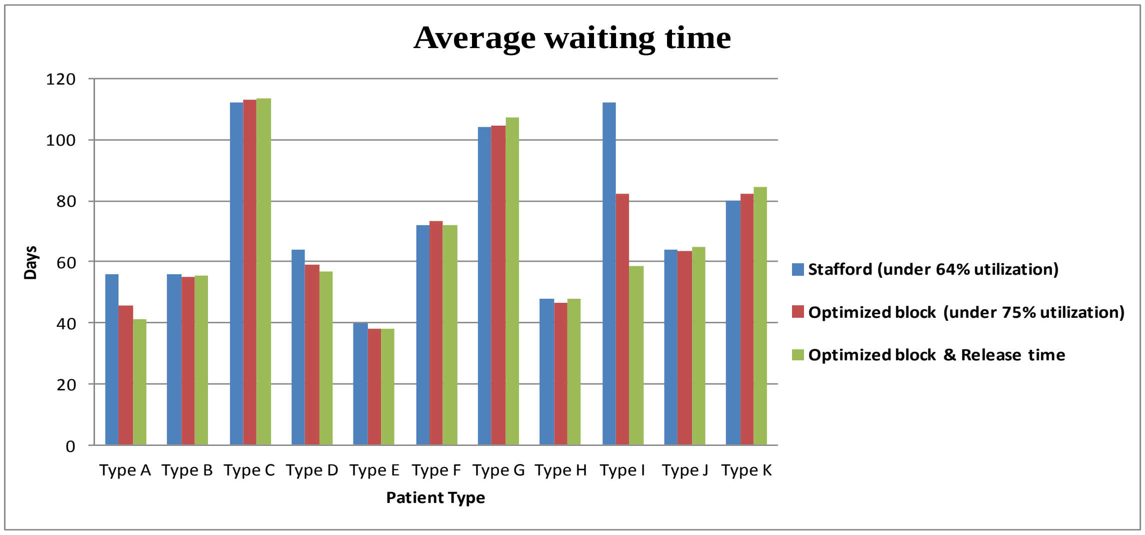


Figure 65 : Average waiting time across patient types and through each optimization step

As illustrated in Figure 66, each surgeon/group shows different sensitivity to waiting times. Some start with a very low waiting penalty but see a rapid increase in the waiting penalty over time; others show less sensitivity to the waiting time with nearly constant waiting penalties over time. Table 42 provides the details on how this behavior causes priority ranking changes over time among these groups. For example, Type A, with steepest slope, stands at the fourth rank at the start but, after a short waiting time, reaches the first rank while Type G's ranking just fluctuates around ranks nine to eleven over time. This behavior adds more complexity to the optimal block allocation and release policy decision at Stafford.

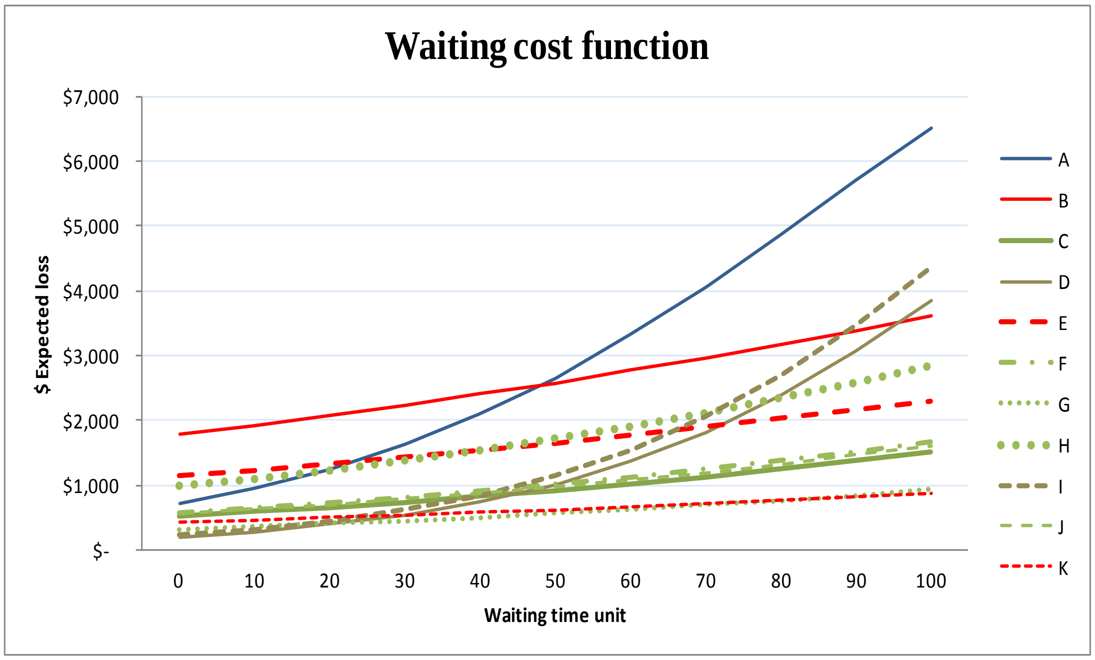


Figure 66: Waiting cost functions across surgeon/group; indicates $ profit loss per waiting time unit

Table 42: Time-based priority of surgeon/group

Priority-Ranking (t)		**Time**										
Specialty	**Surgeon /Group**	**0**	**10**	**20**	**30**	**40**	**50**	**60**	**70**	**80**	**90**	**100**
A-Ortho	**Type A**	4	4	3	2	2	1	1	1	1	1	1
B-GEN/ VASC	**Type B**	1	1	1	1	1	2	2	2	2	3	4
C-PODIATRY	**Type C**	7	7	7	7	8	9	9	9	9	9	9
D-Ortho	**Type D**	11	11	11	9	9	7	6	6	4	4	3
E- GEN/ VASC	**Type E**	2	2	2	3	3	4	4	5	6	6	6
F- GYN	**Type F**	5	5	5	5	5	6	7	7	7	7	7
G-ENT	**Type G**	9	9	10	11	11	11	11	11	10	10	10
H- GEN/ VASC	**Type H**	3	3	4	4	4	3	3	3	5	5	5
I-Ortho	**Type I**	10	10	9	8	7	5	5	4	3	2	2
J- PODIATRY	**Type J**	6	6	6	6	6	8	8	8	8	8	8
K-Plastic	**Type K**	8	8	8	10	10	10	10	10	11	11	11

As explained before, Stafford has a modified block approach in which 75% of OR hours are dedicated to surgeons/group and 25% are kept open to share with other groups. Stafford employs a predefined block release policy that is calculated so that the required scheduling lead-time accommodates 75% of a service's patients. This time must be managed carefully to be fair to both block owners and other groups. Our objective was to derive an improved release policy based on historical data so that waiting times and overall block costs could be minimized across multi-priority patients. Figure 67 displays the percentage of cases that are handled in non-primary blocks (off-block %) across groups through each optimization step. The percentage of off-block surgeries for Surgeons A, D and I has increased more than 15% due to high waiting times and the surgeons' inflexible schedules. On the other hand, the percentage of surgeries that are performed off-block have been reduced for Surgeons G and K, who have the lowest priority patients but have also seen higher waiting costs. Our decision of block size reduction in the first step along with postponing release time for high priority surgeons resulted in less possible off-block opportunity for these two surgeons to assign their surgeries to off-block hours. This decision resulted in forcing low priority patients into a home block for Surgeon G and K, essentially freeing space and giving higher priority patients a better chance at being scheduled for off-block surgeries.

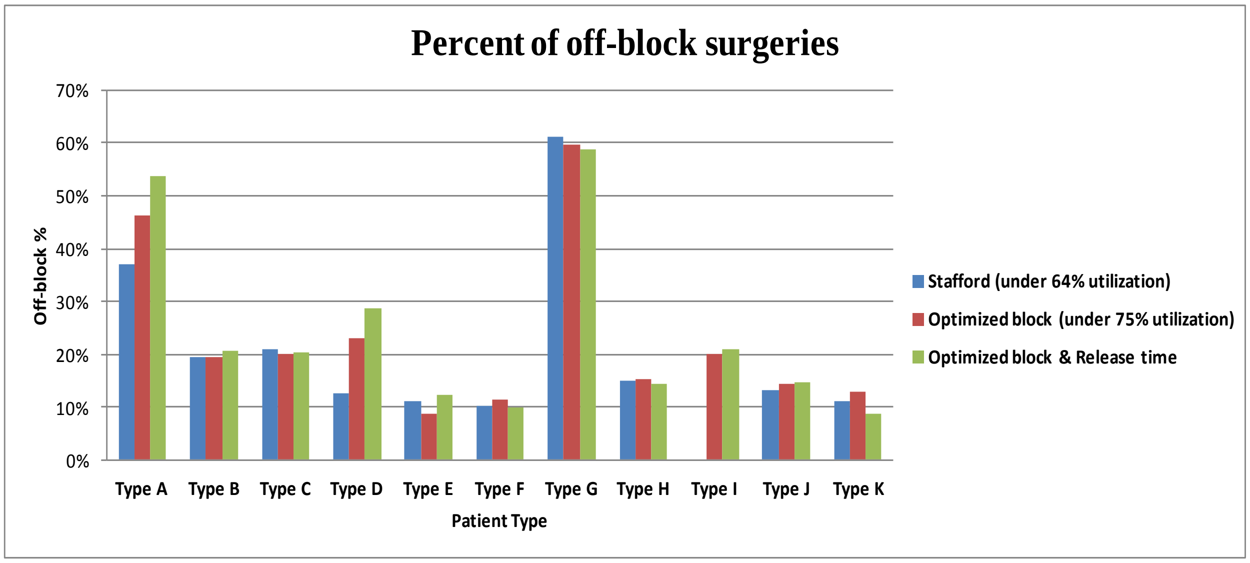


Figure 67: Percent of off-block surgeries across patient types and through each optimization step

Figure 68 presents summary statistics regarding the percentage of off-block surgeries at Stafford under an improved block and release policy. Each number represents the percent of the surgeries of each type performed in blocks allocated to each type, under existing conditions and under improved conditions. Comparing these two tables indicates that the improved release policy results in lower waiting times and higher utilization, as summarized below:

- Postponing the release time of Type A and Type D surgeons has resulted in more reliable block times for these high priority patients. As a result, there is less sharing with other surgeons. For example, referring to the current schedule at Stafford (Figure 68a), Type G has performed 2% of his surgeries in Surgeon A's block, with the highest waiting penalty and shortest scheduling lead time, while under the superior policy (Figure 68b) no surgery has been performed in Surgeon A's block hours.
- The utilization rate has improved by 10% for Surgeon B (refer to Figure 63) due to an increase in the percentage of off-block cases performed in Surgeon's B block hours. The improved release policy results in earlier release time with more available time for use by other groups. This decision resulted in a slight increase in off-block percentage for surgeon B for those cases that come in at the last minute but the utilization improvement appears to have offset the increase in open block hours.
- Although we have increased Surgeon's E block size in the first step, the utilization rate stayed the same due to a higher percentage of off-block cases performed in Surgeon's E block hours as a result of earlier release time with more available time for use by other groups.

Table 43 provides the complete results of the optimization including a short description of the surgeon/group properties and the proposed action for each group in terms of block hour reductions (or increases) and release time adjustments along with the rationale for the changes. The results confirm the earlier statement that "the main goal of optimization is to improve profit with block cost reduction where there is a minimum effect on waiting cost and reducing waiting cost considering waiting penalty function for multi-class patients."

	Type A	Type B	Type C	Type D	Type E	Type F	Type G	Type H	Type I	Type J	Type K	Open Block
Type A	63%	8%	1%	3%	7%	5%	1%			1%		11%
Type B		80%										20%
Type C		2%	79%		1%		1%					17%
Type D				87%								13%
Type E					89%							11%
Type F		0.7%			1%	90%						9%
Type G	2%				5%	2%	39%			1%		51%
Type H		7.5%				7.5%		85%				0%
Type I									100%			0%
Type J										87%		13%
Type K					3%	1%	5%			2%	89%	0%

	Type A	Type B	Type C	Type D	Type E	Type F	Type G	Type H	Type I	Type J	Type K	Open Block
Type A	46%	10%		1%	11%	9%	1%					22%
Type B		78%										22%
Type C		1%	80%									19%
Type D				71%	1%	4%						24%
Type E					88%							12%
Type F		2%			1%	90%						7%
Type G					5%		41%			1%		53%
Type H		6%			4%	4%		85%				3%
Type I		10%							79%			11%
Type J		2%								85%		13%
Type K					2%		3%			1%	91%	3%

Figure 68: a) Percentage of off-block surgeries in Stafford b) Percentage of off-block surgeries under superior block schedule and release policy

The third and final step focused on evaluating the case scheduling or reserve policies for multi-priority patients. Currently, at Stafford, released or open hours are occupied according to first-come first-serve policy. We generated two scenarios of reserve policy under different penalizing functions for using open hours versus release block hours and we estimated total expected cost by simulating Algorithm 2 (introduced in Chapter 5) for the population of patients who are eligible to be assigned to either released or open hours.

Table 43: Detailed description of surgeon/group along with the result of optimization

Surgeon /Group	Description	Result
Type A	The highest priority patient type with short scheduling lead time (urgent), utilization of 66%, and the steepest waiting cost function.	In spite of high priority level, a reduction in block hours is proposed along with postponement of release time to create access to more reliable space and re-direct patients to use even more off-block hours (current off-block is high). Performing surgery only one day every other week results in higher waiting costs for these high priority patients. By assigning more than half of these patients to off-block hours, the average waiting time is reduced by 26%
Type B	A high priority patient type with a low utilization rate of 51%. (Surgeon B performs three days a week every week), and very high and flat waiting cost function.	In spite of low utilization, no reduction in block size is proposed due to high waiting penalty and short scheduling lead time for these patients. However, a shift of release time to earlier times is proposed so that underutilized hours can be shared with other surgeons, improving overall utilization.
Type C	A low priority patient type with long scheduling lead time, a high cancellation rate, high utilization, and a flat waiting cost function.	Because it has no significant effect on profit , Type C was not selected as an important factor in factor screening so it remains at its current operational level
Type D	A medium-to-high priority patient type with low utilization and a steep waiting cost function	A reduction of block size is proposed but with a postponement of the release time and a redirection of more patients to off-block spaces. Allowing surgeons to assign patients on a more flexible schedule helps to reduce waiting times for these high-priority patients by 11%
Type E	A medium-to-high priority patient type at the target utilization, with medium waiting costs that show a flat slope function	A slight (5%) increase in block hours is proposed, while releasing the unfilled spaces earlier to share with others. Given the short lead time and the surgeon's need for a specialized room, an increase in block hours is an appropriate decision for these medium-to-high priority patients. Increasing the block hours along with earlier release time will reduce the waiting time while letting higher priority patient utilize unfilled spaces on any of three days each week.

Table 43 Continued: Detailed description of surgeon/group along with the result of optimization

Surgeon /Group	Description	Result
Type F	A low-to-medium priority patient type with low utilization, and a flat waiting function	This type of surgery requires a specialized room so, even with the low utilization, only a 6% reduction in block size is proposed. There is no need to retain the entire block for this low-priority patient type with its high coefficient of variation in arrival rates. Changes to an earlier release time for more surgeons will redirect cases out of this block hours to released hours of other blocks.
Type G	The lowest priority patient type, with very high waiting time, a high utilization rate, high off-block use, and a low, flat waiting cost function	Type G did not emerge as an important type during factor screening, so it remains at its current operational level.
Type H	A medium-to-high priority patient type with low utilization (Surgeon H performs one day per week with high % of semi-urgent patients) and a medium waiting cost function with mild slope	Type H did not emerge as an important type during factor screening, so it remains at its current operational level.
Type I	A medium priority patient type with low utilization, a very low waiting penalty at low waiting times, but a steeply rising slope over time	Because this surgeon has a low number of patients and only performs surgery one day every other week, a reduction in the block size is proposed while re-directing patients to off-block hours, thereby facilitating more flexible scheduling.
Type J	A low priority patient type, with low utilization (one day of surgery every other week), a very high percentage of semi-urgent patients, and a medium-to-flat waiting cost function	To facilitate release-time optimization, a reduction in block size is proposed for this low-priority patient by re-directing a higher portion of cases to off-block hours while re-directing other surgeons' cases who use this surgeon's hours out of this block. This decision has minimum effect on waiting cost of patient type J.
Type K	A low priority patient type with low utilization, a low volume of surgery, low initial block hours, a high cancellation rate, and low penalty with a flat waiting cost function	Although this type of surgery occurs only one day every other week, it was not identified as an important factor. The block release is already set at the earliest possible time, taking advantage of the high cancellation rate and allowing high priority patients to occupy unfilled space.

In Figure 69, we plot the marginal profit gain/loss utilizing a reserve policy under two scenarios of open block cost, 1.5 times and 2.0 times the cost of occupying release hours. Although one expects the performance to improve under the reserve policy when we have multi-priority patients, comparing the performances of two scenarios indicates that applying the reserve policy does not necessarily result in superior performance under all conditions. Rather, it depends on the optimal balance between underutilization cost and waiting cost. A penalty of using open hours versus released hours performs quite well in balancing these two cost types.

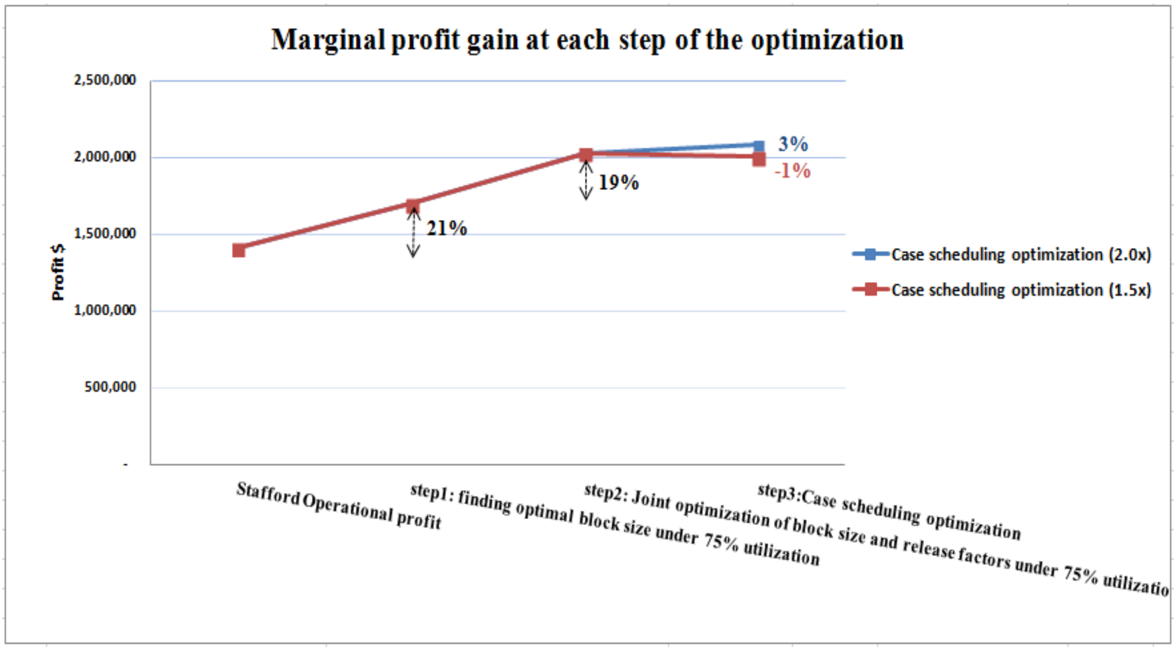


Figure 69: Marginal $ profit gain/loss at case scheduling optimization

In the case of a higher penalty in occupying open hours (2.0x) than release hours, we gained some improvement in profit due to a reduction in overall waiting time with a minimal reduction in utilization rate. Under this scenario, we tend to fill release block first rather than assigning them to open blocks in exchange for the risk of higher waiting cost for future higher priority patients. However, the system performs substantially worse under 1.5 times penalty (and even worse than the first-come-first serve policy), showing the importance of the under-utilization rate while protecting capacity for the future high priority cases.

A comparison of utilization rates among surgeons (in Figure 70) indicates that, under the first scenario, more cases are assigned to open hours thus the utilization rate is reduced in order to keep more space for possible high priority patients. While under the second scenario, we are inclined to assign cases to the last spot in the release time block range. In this case, utilization will improve with minimal impact on the waiting time of high priority patients. We noticed that under the higher penalty for open block, fewer cases are assigned to open blocks, which indicates there is higher chance to force high priority cases to wait for longer times in the system.

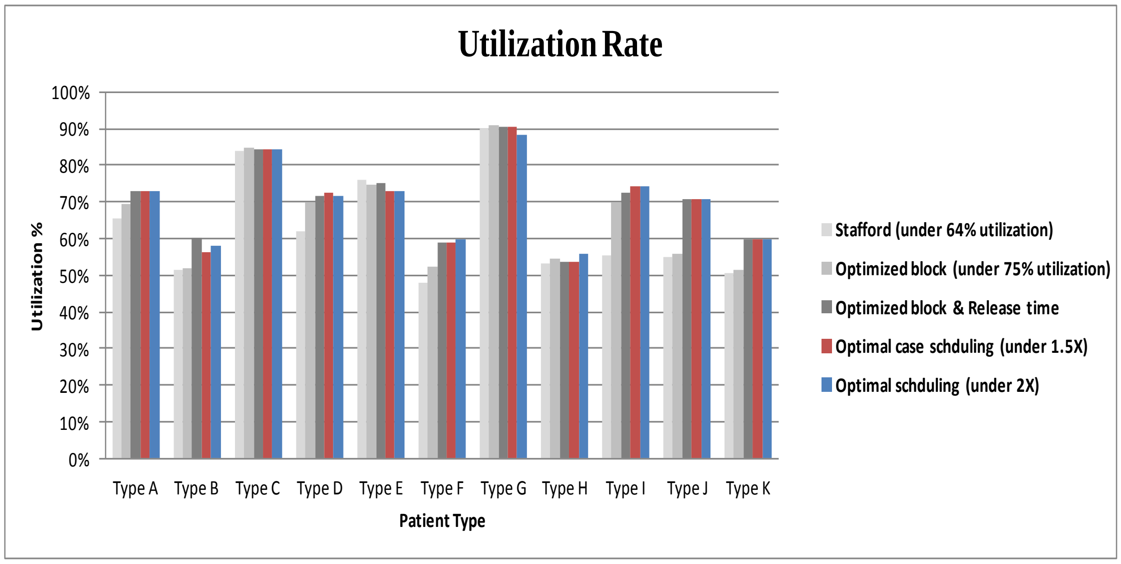


Figure 70: Utilization rate across patient types under improved case scheduling

However, assigning a smaller number of jobs does not immediately translate to a higher waiting cost for higher priority patients. It is due to having open hours in the near future if the released block has already been filled. Figure 71 shows how the improved case scheduling policy affected waiting times across different types of surgery, as well as how open block penalty cost contributed to waiting cost, and thus to the rebalancing of resources across competing classes of demand.

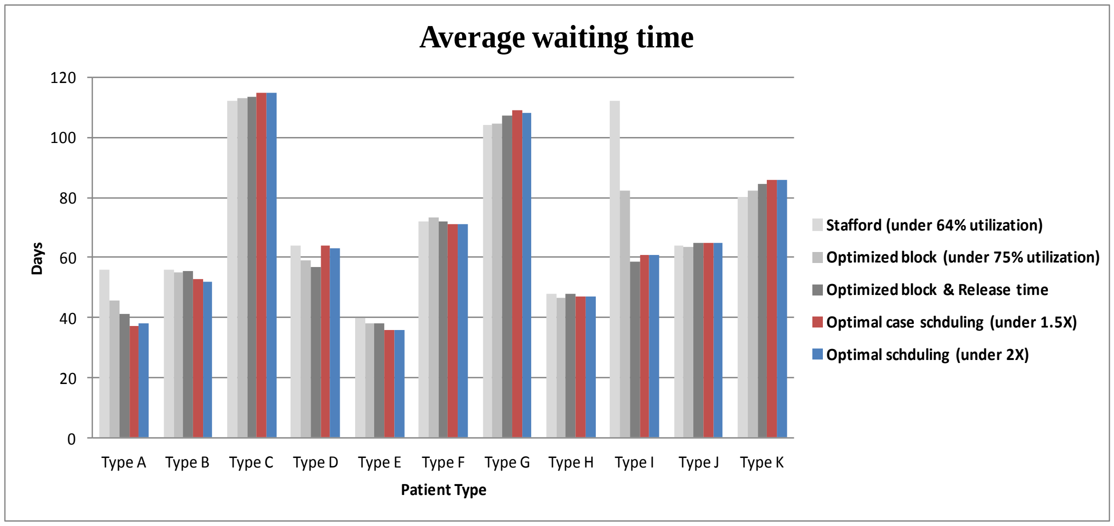
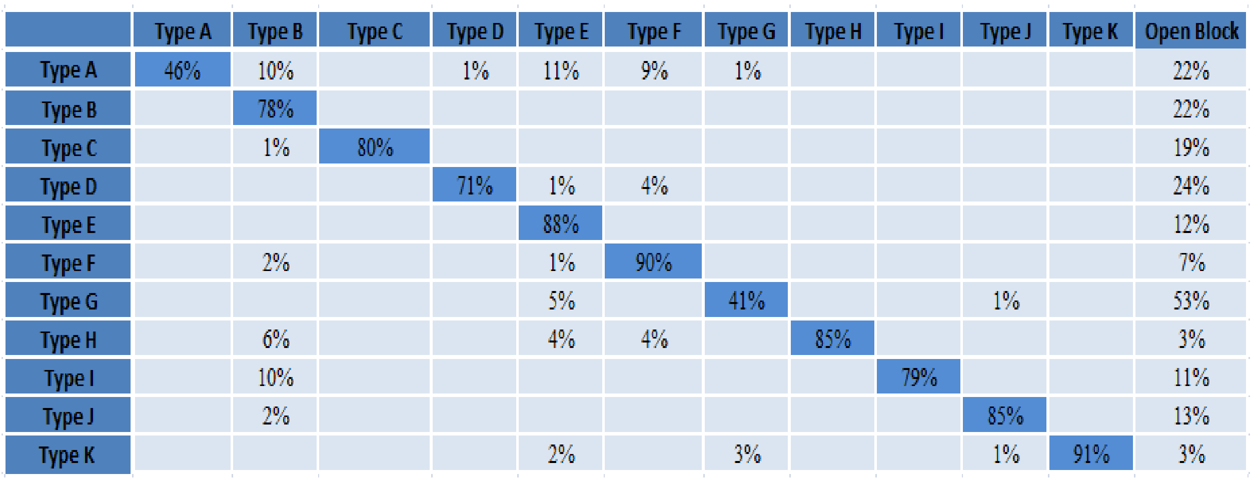


Figure 71: Average waiting time across patient types under improved case scheduling

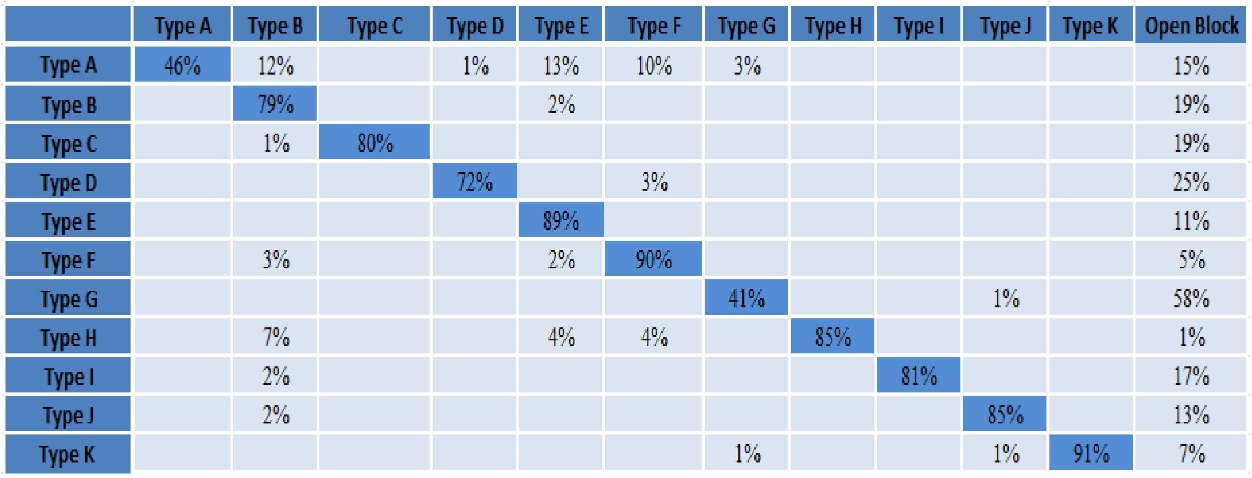
Figure 72 displays the percentage of cases that are handled in primary or non-primary blocks (off-block %) across groups under FCFS and reserve scheduling policy. The reserve policy tends to postpone low priority cases with high arrival rate in favor of high priority cases (referring to patient type G and D in Figure 72 (b)). We see the same behavior under both scenarios of reserve policy with the difference that a greater percentage of cases are still assigned to release block as we penalize open blocks more.

Concluding Remarks:

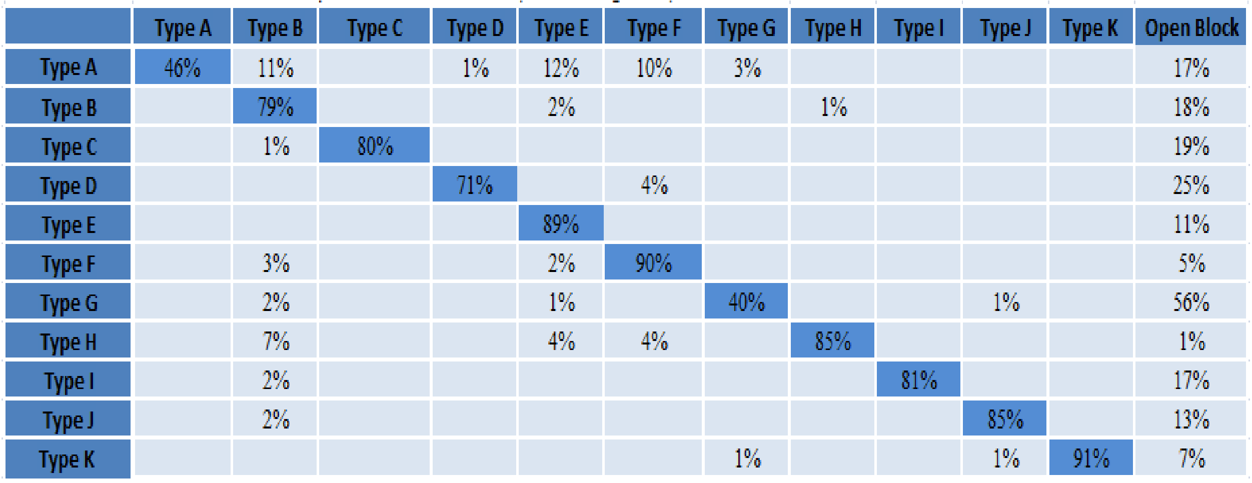
In this dissertation, we focused on developing realistic models for elective surgery scheduling for multi-priority patients in order to solve the joint optimization of allocation and release policy decisions and providing practical insights for practitioners. The goal was not to instruct hospitals as to how they should schedule their cases and block times, but rather to provide options for practitioners to explore. Parameters, conditions, and goals may vary from hospital to hospital, so having some rules of thumb helps to tailor these differences. The majority of earlier studies looked at either a single allocation scheduling stage (case mix planning, surgical master schedule or elective case scheduling) or a reduction in surgery waiting lists (in contrast with online scheduling).



	Type A	Type B	Type C	Type D	Type E	Type F	Type G	Type H	Type I	Type J	Type K	Open Block
Type A	46%	10%		1%	11%	9%	1%					22%
Type B		78%										22%
Type C		1%	80%									19%
Type D				71%	1%	4%						24%
Type E					88%							12%
Type F		2%			1%	90%						7%
Type G					5%		41%			1%		53%
Type H		6%			4%	4%		85%				3%
Type I		10%							79%			11%
Type J		2%								85%		13%
Type K					2%		3%			1%	91%	3%



	Type A	Type B	Type C	Type D	Type E	Type F	Type G	Type H	Type I	Type J	Type K	Open Block
Type A	46%	12%		1%	13%	10%	3%					15%
Type B		79%			2%							19%
Type C		1%	80%									19%
Type D				72%		3%						25%
Type E					89%							11%
Type F		3%			2%	90%						5%
Type G							41%			1%		58%
Type H		7%			4%	4%		85%				1%
Type I		2%							81%			17%
Type J		2%								85%		13%
Type K							1%			1%	91%	7%



	Type A	Type B	Type C	Type D	Type E	Type F	Type G	Type H	Type I	Type J	Type K	Open Block
Type A	46%	11%		1%	12%	10%	3%					17%
Type B		79%			2%			1%				18%
Type C		1%	80%									19%
Type D				71%		4%						25%
Type E					89%							11%
Type F		3%			2%	90%						5%
Type G		2%			1%		40%			1%		56%
Type H		7%			4%	4%		85%				1%
Type I		2%							81%			17%
Type J		2%								85%		13%
Type K							1%			1%	91%	7%

Figure 72: Percentage of off-block surgeries under (a) superior block schedule and release policy (b) reserved policy and open block penalty 1.5X (c) reserved policy and open block penalty 2.0X

However, this study (Chapter 2) addressed the importance of joint optimization specifically under online scheduling. Recognizing this fact, we characterized a mathematical programming model and conducted simulation modeling (Chapter 3) to examine a class of policies that are characterized by reserve levels. We drew valuable scheduling insights from our extensive computational study and from suggested solutions from the case studies. Numerical results from Stafford hospital (Chapter 4) showed that consideration of patient priority resulted in better performance as compared to a schedule that ignores the patient priority (using first-come-first-serve policy), despite the fact that the reserve policy only applies to 25% of Stafford's patient.

Hospitals should be in a continuous search for more efficient and timely utilization of their resources (time, ORs, personnel) in order to better respond to patients' requests for service. What we offer in this study would help hospitals make better decisions at the strategic and operational level. In addition to answering the questions on how many ORs to allocate per surgeons/specialty, this study offers insights about how much time to optimally reserve for higher priority patients, and what would be an improved release time. The problem of allocating multi-priority patients is complex and requires incorporating some revenue management techniques that, despite their proven results in airline and hotel management, are still not very widespread in healthcare. We presented the specifics of these techniques when applied to healthcare and we believe that continued research in this direction will provide hospital managers, practitioners and schedulers with adequate decision making tools.

Insights for the Practitioner

In this section, we convey the key insights of this study in a way that is clear, relevant and actionable, so that practitioners can use our findings to make the best possible business decisions in their own contexts. More importantly, this section will allow practitioners to develop a real-world understanding of the complexities of this field. The section addresses compromises practitioners must make when deciding how to allocate and release blocks, as well as how they can take into account different levels of tolerance for waiting across different classes of patients.

Many hospitals have policies about allocating block hours among specialties based on the demand for surgeries regardless of differences across specialties or levels of urgency for different classes of patients. Although calculations of average demand and utilization are a good starting point for estimating the number of hours a surgical group needs to schedule, these calculations cannot provide answers to questions such as '*When do we need to allocate more block time than average demand would suggest?'* or '*Which mix of open/block scheduling strategy is best for each combination of patients?'* The concepts of waiting penalties, arrival rates, and scheduling lead time can serve as useful indicators for allocating resources among surgeons. For example,

- When a surgical group has high priority patients with short lead times who require specialized rooms, then an increase in block hours along with earlier release time is recommended, so that underutilized hours can be shared with other surgeons, improving overall utilization.
- When a surgical group has patients with very steep waiting penalties, short lead times and a high coefficient of variation in demand, then a reduction in block hours and higher use of open/released hours along with postponement of release time is recommended, so as to create more reliable access to space.

Although our results confirm that no single case scheduling policy is superior for all sets of parameters, our findings do suggest some useful *rules of thumb for scheduling multi-priority patients*:

- In general, when there is little or no information about the arrival rates of multi-priority patients or their waiting penalties, it is best to proceed with a first-come, first-serve (FCFS) policy, since this policy is simple and less sensitive to parameters than other policies, and also guarantees the highest rate of utilization. However, having more information about patient types can facilitate the choice of an optimal policy and improve patient waiting times and overall profit.
- If lower priority patients are arriving at a faster rate than higher priority patients, then a FCFS will be the simplest, but not the best, policy for the entire range of waiting cost-ratios. At the same time, a prioritization policy, i.e. a policy of

postponing all lower priority patients until a future time will likely be the worst choice because it results in unnecessary reserve space for higher priority and an exponential increase in lower priority waiting time.

- If higher-priority patients are arriving at a faster rate than lower-priority patients, then it is best to book less-urgent patients further into the future and to reserve more space in the immediate future for high priority patients. This raises the question of *how much capacity should be reserved for later-arriving but higher-priority demand*, which we refer to as the *threshold-based reserve policy.*

  The answer to this question will depend on your patients' arrival rates and ratios of waiting penalties: as either or both of these numbers increases, the threshold will also increase. This simple policy can help you estimate the necessary protection limit for every class of patient.

- If the arrival rate of urgent patients is about the same as that of non-urgent patients, the optimal policy varies due to differences in system utilization and waiting penalties:
    - If the utilization rate and ratios of waiting penalties are both high, it makes sense to apply the threshold reserve policy, postpone lower penalty patients to the future. Otherwise, if ratios of waiting penalties are close to one, there is no significant difference in the choice of policy.
    - If the utilization rate is medium (50%), it makes sense to apply the threshold reserve policy across all waiting penalty ratios, and
    - If utilization is low, the choice of scheduling policies makes little or no difference.

Another set of questions that operating room managers frequently ask concerns released blocks: *what is the optimal time to release allocated blocks and who should access released hours?* A common approach is to allocate released blocks a fixed number of days before surgery, without regard for differences across specialties and different levels of patient-urgency. Because of variability in demand across specialties, however, this approach often results in uneven utilization and waiting costs.

As a best practice, when waiting penalties are steep, lead time is short, and the coefficient of variation is high, it makes sense to postpone the release of blocks. This practice can insure that there is space available for late-coming, high priority patients. In contrast, if waiting penalties are flat, the coefficient of variation is low (close to 1) and lead time is long, then it makes sense to release blocks earlier. This practice can improve utilization with minimal impact on waiting costs.

The optimal way to determine the recipients for released blocks is to base the decision on the ratio of waiting penalties to underutilization costs, rather than on fixed cost numbers. This approach provides superior *access rules for release hours,* because it facilitates flexibility, and it provides fair opportunities for all types of patients to access the released blocks, so long as the cost of waiting penalties is greater than the cost of underutilization.

This access rule also applies to recipients for open or shared blocks with one caveat: filling an open hour costs more than utilizing current released blocks since current released blocks are already fully staffed. To account for this difference and to encourage the use of release hours, a penalty needs to be applied for using open hours; this penalty provides flexibility for practitioners in determining whether to open extra ORs or to utilize current release hours. Hospitals with surgical groups that are not accustomed to releasing unfilled blocks may want to start off with a very small penalty value so as to encourage only a small amount of sharing of released blocks. As surgical groups get more used to the idea, the hospital can encourage more release time with higher penalties for open hours, and potentially require fewer ORs.

References

- Ahn H., Kreder H., Mahomed N., Beaton D. and Wright J. G. (2011). Empirically derived maximal acceptable wait time for surgery to treat adolescent idiopathic scoliosis *CMAJ*, 183 (9), E565-70.
- Aringhieri R., Landa P., Soriano P., Tànfani E., and Testi A. (2015). A two level metaheuristic for the operating room scheduling and assignment problem. *Computers & Operations Research* 54, 21–34.
- Belobaba P. P. (1987). Airline yield management: An overview of seat inventory control. *Transp. Sci*. 21, 63–73.
- Belobaba P. P. (1989). Application of a probabilistic decision model to Airline seat inventory control. *Opns. Res* 37, 183–197.
- Beliën J., and Demeulemeester E. (2007). Building cyclic master surgery schedules with leveled resulting bed occupancy. *European Journal of Operational Research* 176, 1185-1204.
- Bertsekas, D., and Tsitsiklis J. (1996). *Neuro-Dynamic Programming*. Athena Scientific, Belmont, MA.
- Bishai D. M., and Lang H. C. (2000). The willingness to pay for wait reduction: the disutility of queues for cataract surgery in Canada, Denmark, and Spain. *Journal of Health Economics* 19(2), 219–30.
- Blake J. T., Dexter F., and Donald J. (2002). Operating room manager's use of integer programming for assigning block time to surgical groups: A case study. *Anesthesia and Analgesia* 94, 143–148.
- Blake J., and Donald J. (2002). Mount Sinai Hospital uses integer programming to allocate operating room time. *Interfaces* 32(2), 63–73.
- Blake J. T. and Carter M. W. (2002). A goal programming approach to strategic resource allocation in acute care hospitals. *Eur J Oper Res* 140, 541–561.
- Bradley N. (2007). *The response surface methodology*. Thesis. Indiana University, South Bend.
- Cardoen B., Demeulemeester E., and Beliën J. (2009). Optimizing a multiple objective surgical case sequencing problem. *Int. J. Prod. Econ*. 119, 354-366.

- Cardoen B., Demeulemeester E., and Beliën J. (2010). Operating room planning and scheduling: a literature review. *Eur J Oper Res* 201(3), 921–932.
- Chow V.S., Puterman, M. L., Salehirad N., Huang W., and Atkins D. (2008). Reducing surgical ward congestion at the Vancouver Island Health Authority through improved surgical scheduling. Technical Report, Centre for Operations Excellence, the University of British Columbia.
- Dean A. M. and Lewis S. M. (2005). *Screening*, New York: Springer-Verlag.
- Denton B., Viapiano J., and Vogl A. (2007). Optimization of surgery sequencing and scheduling decisions under uncertainty. *Health Care Management Science* 10, 13-24.
- Denton B. T., Miller A. J., Balasubramanian H. J., and Huschka T. R. (2010). Optimal allocation of surgery blocks to operating rooms under uncertainty. *Operations Research, 58(4)*, 802-816.
- Dexter F., Ledolter J., and Wachtel R. E. (2005). Tactical decision making for selective expansion of operating room resources incorporating financial criteria and uncertainty in sub-specialties’ future workloads. *Anesth Analg* 100, 1425–1432.
- Dexter F., and Traub R. D. (2002). How to schedule elective surgical cases into specific operating rooms to maximize the efficiency of use of operating room time. *Anesthesia Analgesia* 94, 933–942.
- Dexter F., Traub R. D., and Macario A. (2003). How to release allocated operating room time to increase efficiency - Predicting which surgical service will have the most under-utilized operating room time. *Anesthesia & Analgesia* 96, 507-512.
- Dexter F., and Macario A. (2004). When to release allocated operating room time to increase operating room efficiency. *Anesthesia & Analgesia* 98, 758-762.
- Dexter F., Macario A., Qian F., and Traub R. D. (1999). Forecasting surgical groups’ total hours of elective cases for allocation of block time - Application of time series analysis to operating room management. *Anesthesiology* 91, 1501-1508.
- Dexter F., Macario A., and O’Neill L. (2000). Scheduling surgical cases into over flow block time-computer simulation of the effects of scheduling strategies on operating room labor costs. *Anesth Analg* 90, 980–988.

- El-Darzi E., Vasilakis C., Chaussalet T., and Millard P. H. (1998). A simulation modelling approach to evaluating length of stay, occupancy, emptiness and bed blocking in a hospital geriatric department. *Health Care Manage Sci* 1, 143–149.
- Erdelyi A., and Topaloglu H. (2009). Computing protection level policies for dynamic capacity allocation problems by using stochastic approximation methods. *IIE Transactions*, 41, 498-510.
- Everett J. E. (2002). A Decision Support Simulation Model for the Management of an Elective Surgery Waiting System. *Health Care Management Science,* 5 (2), 89-95.
- Fei H., Meskens N., and Chu C. (2009). A planning and scheduling problem for an operating theatre using an open scheduling strategy. *Computers & Industrial Engineering*, 221-230.
- Gallucci G., Swartz W., and Hackerman F. (2005). Brief reports: Impact of the wait for an initial appointment on the rate of kept appointments at a mental health center. *Psychiatr. Serv*. 56(3), 344-346.
- Gerchak Y., Gupta D., and Henig M. (1996). Reservation planning for elective surgery under uncertain demand for emergency surgery. *Management Science* 42(3), 321-334.
- Guinet A., and Chaabane S. (2003). Operating theatre planning. *Int J Prod Econ* 85, 69–81.
- Gupta D. (2007). Surgical suites‟ operations management”. *Production and Operations Management*, *16(6)*, 689-700.
- Gupta D., and Denton B. (2008). Appointment scheduling in health care: Challenges and opportunities. *IIE Transactions* 40, 800–819.
- Hamilton D. M., and Breslawski S. (1994). Operating room scheduling: Factors to consider. *Assoc. Operat. Room Nurses J*. 59, 665–680.
- Hans E. W., Wullink G., van Houdenhoven M., and Kazemier G. (2005). Robust surgery loading. Technical Report Beta-wp141, Department of Operational Methods for Production and Logistics, University of Twente.
- Hans E., Wullink G., van Houdenhoven M., and Kazemier G. (2008). Robust surgery loading. *European Journal of Operational Research*, *185*, 1038-1050.

- Hering W., and Herrmann J. (2011). *The Single-day Surgery Scheduling Problem: Sequential Decision-making and Threshold-based Heuristics*. Springer.
- Herring W. L., and Herrmann J. W. (2011). A stochastic dynamic program for the single-day surgery scheduling problem. *IIE Transactions on Healthcare Systems Engineering* 1, 213–225.
- Herring W. L. (2011). *Prioritizing patient: Stochastic dynamic programming surgery scheduling and mass casualty incident triage*, Ph.D., University of Maryland, College Park, 188 pages.
- Hughes W. L., and Soliman S. Y. (1985). Short-term case mix management with linear programming. *Hospital Health Serv*. Admin. 30, 52–60.
- Jebali A, Alouane, A., and Ladet P. (2006). Operating rooms scheduling. *Int J Prod Econ*; 99, 52–62.
- JMP User Guide, Release 10, 2012. SAS Institute Inc.
- Krahl D. (2011). ExtendSim Technology: Scenario Management. *Simulation Conference (WSC), Proceedings of the 2011 Winter*, 12-19
- Law M. A., and Kelton W. D. (2000). *Simulation modeling and analysis*, 3rd ed. McGraw Hill, Singapore.
- Lewis S. M. and Dean A. M. (2001). Detection of interactions in experiments on large numbers of factors (with discussion). *J. Roy. Statist. Soc*. Ser. B 63, 633-672
- Littlewood K. (1972). Forecasting and control of passengers, in `Proceedings of the 12th AGIFORS', New York, 95-117.
- Liu N., Ziya S., and Kulkarni V. G. (2010). 'Dynamic scheduling of outpatient appointments under patient no-shows and cancellations', *Manufacturing & Service Operations Management* 12(2), 347–364.
- Ma G., and Demeulemeester E. (2012). A multilevel integrative approach to hospital case mix and capacity planning. *Computers & Operations Research*.
- Mannino C., Nilssen E. J., and Nordlander T. E. (2012). A pattern based, robust approach to cyclic master surgery scheduling. *J Sched* 15, 553–563.
- Magerlin J. M, Martin J. B. (1978). Surgical demand scheduling: a review. *Health Serv Res* 13(4), 418-433.

- Marques I., Eugénia Captivo M., and Vaz Pato M. (2012) An integer programming approach to elective surgery scheduling analysis and comparison based on a real case. *OR Spectrum* 34, 407–427.
- May J. H., Strum D. P., and Vargas L. G. (2000). Fitting the lognormal distribution to surgical procedure times, *Decision Sciences* 31, 129-148.
- Mee R. (2009). *A Comprehensive Guide to Factorial Two-Level Experimentation*. New York: Springer
- Min D., and Yih Y. (2010). An elective surgery scheduling problem considering patient priority. *Computers & Operations Research* 37, 1091–1099.
- McGill J., and van Ryzin G. (1999). Revenue management: Research overview and prospects. *Transportation Science* 33, 233-256.
- McLane-Kinzie D. (2005). *Scheduling Strategies for Ambulatory Surgery Centers*. HCPro, Inc.
- McManus M., Long M., Cooper A., Mandell J., Berwick D., Pagano M., and Litvak E. (2003). Variability in surgical caseload and access to intensive care services. *Anesthesiology* 98(6), 1491-1496.
- Montgomery D. C. (2000). *Design and Analysis of Experiments*. 5 ed.: John Wiley & Sons.
- Montgomery D. C. (2005). *Design and Analysis of Experiments: Response surface method and designs*. New Jersey: John Wiley and Sons, Inc.
- Ogulata S. N., and Erol R. (2003). A hierarchical multiple criteria mathematical programming approach for scheduling general surgery operations in large hospitals. *J Med Syst.* 27(3), 259-70.
- Ozcan Y. A., Tanfani E., and Testi A. (2011). A simulation-based modeling framework to deal with clinical pathways. *Simulation Conference (WSC), Proceedings of the 2011 winter*, 1190 – 1201.
- Ozkarahan I. (2000). Allocation of surgeries to operating rooms using goal programming. *J Med Syst* 24(6), 339–378.

- Pandit J. J., Pandit M., and Reynard J. M. (2010). Understanding waiting lists as the matching of surgical capacity to demand: are we wasting enough surgical time? *Anesthesia*, 65, 625–640.
- Patrick J. and Puterman M. L. (2008) Reducing Wait Times through Operations Research: Optimizing the Use of Surge Capacity. *Healthcare Policy* Vol.3 No.3.
- Patrick J., Puterman M. L., and Queyranne M. (2008). Dynamic multipriority patient scheduling for a diagnostic resource. *Operations Research* .56 (6), 1507-1525.
- Patterson, P. (1996). What makes a well-oiled scheduling system? *OR Manager*, 12(9), 19-23
- Persson, M., and Persson J. A. (2010). Analyzing management policies for operating room planning using simulation, *Health Care Management Science*, 13 (2), 182-191.
- Price C., Golden B., Harrington M., Konewko R., Wasil E., and Herring W. (2011). Reducing boarding in a post-anesthesia care unit. *Production and Operations Management*, 20 (3), 431-441.
- Puterman M. (1994). Markov Decision Processes. New York: John Wiley and Sons.
- Rising E., Baron R., Averill B. (1973). A systems analysis of a university-health service outpatient clinic. *Operations Research* 21 (5), 1030–1047.
- Robbins W. A., and Tuntiwongbiboon N. (1989). Linear programming is a useful tool in case-mix management. *Healthcare Finan. Manage*. 43, 114–116.
- Samanloiglu F., Ayag Z., Batili B., Evcimen E., Yilmaz G., and Atalay O. (2010). Determining master schedule of surgical operations by integer programming: a case study. *Proceeding of the 2010 Industrial Engineering Research Conference*, June 5-9, Cancun, Mexico
- Santibañez P., Begen M., and Atkins D. (2007). Surgical block scheduling in a system of hospitals: an application to resource and wait list management in a British Columbia health authority. *Health Care Manag Sci* 10, 269–282.
- Santibañez P., Begen M., and Atkins D. (1997). Managing surgical waitlists for a British Columbia health authority. *Research report, Centre for Operations Excellence, Sauder.*
- Sauré A. (2012). *Approximate Dynamic Programming Methods for Advance Patient Scheduling.* PhD dissertation, The University of British Columbia.

- 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, 573–584
- Schütz H., and Kolisch R. (2012). Approximate dynamic programming for capacity allocation in the service industry. *European Journal of Operational Research* 218 (1), 239–250.
- Sciomachen A., Tanfani E., and Testi A. (2005). Simulation models for optimal schedules of operating theatres. *Int J Simul* 6 (12/13), 26–34.
- Sier D., Tobin P., and McGurk C. (1997). Scheduling surgical procedures. *Journal of the Operational Research Society*, 48, 884-891.
- Sobolev B., and Kuramoto L. (2008). *Analysis of Waiting-Time Data in Health Services Research*. Springer.
- Sobolev B., Fradet G., Hayden R., Kuramoto L., Levy A., and FitzGerald M. (2008). Delay in admission for elective coronary-artery bypass grafting is associated with increased in-hospital mortality. *BMC Health Services Research,* 81-85.
- Sobolev B., Sanchez V., and Vasilakis C. (2011). Systematic review of the use of computer simulation modeling of patient flow in surgical care. *J Med Syst* 35, 1–16.
- Spangler W. E., Strum D. P., Vargas L. G., and May J. H. (2004). Estimating procedure times for surgeries by determining location parameters for the lognormal model. *Health Care Management Science*, 7, 97-104.
- Stanciu A., Vargas L., and May J. (2010). “A revenue management approach for managing operating room capacity,” *Proceedings of the 2010 Winter Simulation Conference*.
- Steins K., Persson F., and Holmer M. (2010). Increasing utilization in a hospital operating department using simulation modeling. *Simulation-Transactions of the Society for Modeling and Simulation International* 86, 463-480.
- Stenevi U., Lundstrom M., and Thorburn W. (2000). The cost of cataract patients awaiting surgery. *Acta Ophthalmologica Scandinavica*; 78(6), 703–5.
- Strum D. P. Vargas L. G., and May J. H. (1999). Surgical subspecialty block utilization and capacity planning: a minimal cost analysis model. *Anesthesiology, 90(4)*, 1176-1185.

- Strum D., Vargas L. G., and May J. H., Spangler W., and Stanicu A. (2008). "Operating room scheduling: capacity planning, and revenue management." *Anesthesia Informatics*, Stonemetz J., Ruskin K. (eds.). Springer Verlag, 359-390.
- Talluri K., and Van Ryzin G. (2004). *The Theory and Practice of Revenue Management.* Springer.
- Tanfani E., and Testi A. (2010). A pre-assignment heuristic algorithm for the master surgical schedule problem (MSSP). *Annals of Operations Research*, 178, 105–119.
- Telford J. K. (2007). A Brief Introduction to Design of Experiments. *Johns Hopkins APL Technical Digest*, Volume 27, Number 3.
- Testi A., Tanfani E., and Torre G. (2007). A three-phase approach for operating theatre schedules. *Health Care Management Science*, 10, 163-172.
- Trocine L. and Malone L. (2000). Finding important independent variables through screening designs: A comparison of methods. *In Proceeding of the 2000 Winter Simulation Conference*, 749–754.
- Trocine L. and Malone L. (2001). An overview of newer advanced screening methods for the initial phase in an experimental design. *In Proceedings of the 2001 Winter Simulation Conference.*
- Wachtel R. E., and Dexter F. (2008). Tactical increases in operating room block time for capacity planning should not be based on utilization. *Anesth Analg*. 106(1), 215-26.
- Van Houdenhoven M., Van Oostrum J. M., Hans E. W., Wullink G. and Kazemier G. (2007). Improving operating room efficiency by applying bin-packing and portfolio techniques to surgical case scheduling. *Anesthesia & Analgesia*, 105(3), 707–714.
- Van Oostrum J. M., van Houdenhoven M., Hurink J. L., Hans E.W., Wullink G., and Kazemier G., (2008). A master surgical scheduling approach for cyclic scheduling in operating room departments. *OR Spectrum*, 30(2), 355-374.
- Vansteenkiste N., Lamote C., Vandersmissen J., Luysmans P., Monnens P., De Voldere G., Kips J., and Rademakers F. E. (2012). Reallocation of operating room capacity using the due-time model. *Medical Care* 50, 779-784.

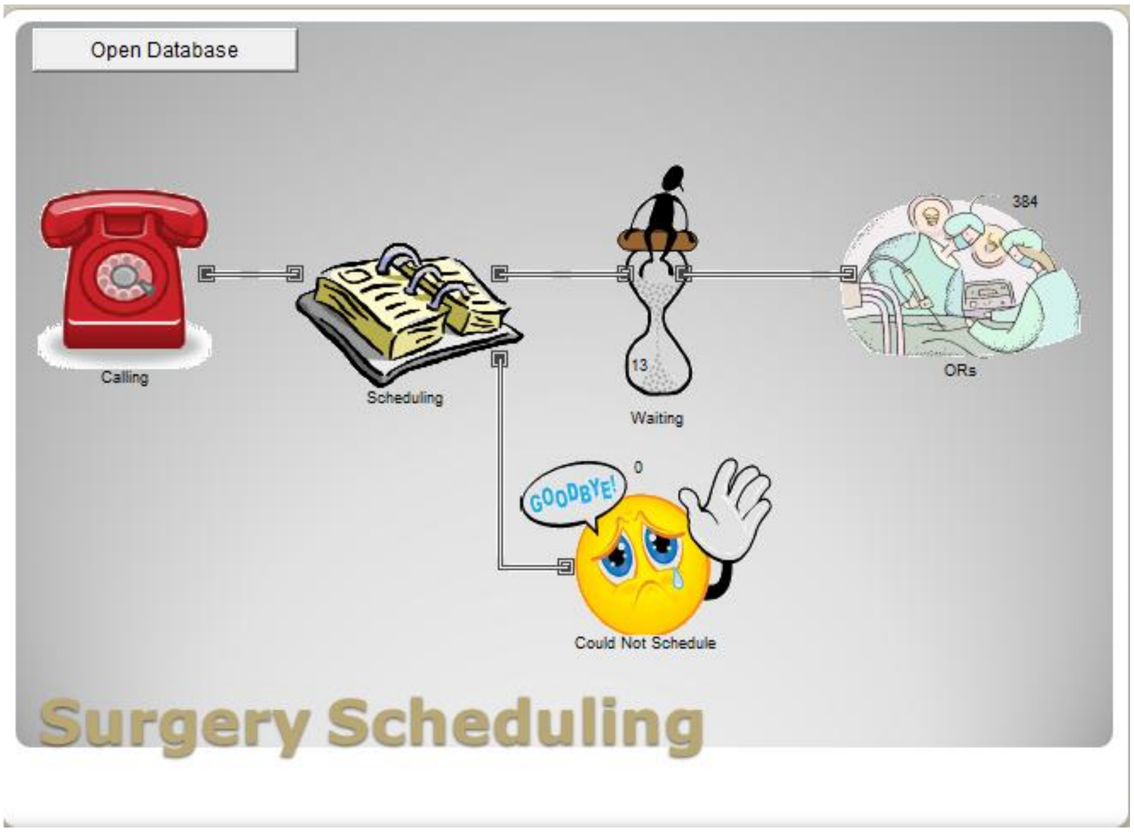
- Vermeulen I., Bohte S., Elkhuizen S., Lameris H., Bakker P., and Poutré H. L., (2009). Adaptive resource allocation for efficient patient scheduling. *Artificial intelligence in Medicine* 46 (1), 67–80.
- Verseput R. (2000). Digging into DOE: Selecting the right central composite Design for response surface methodology applications. *Quality Digest*. QCI International. Available at: http://www.qualitydigest.com/june01/html/doe.html (accessed June 7, 2006).
- Vissers J. (2005). Aggregate hospital production and capacity planning. In: Vissers J., and Beech R., editors. *Health Operations Management: Patient flow logistics in health care*, *New York: Routledge*, 116-45.
- Yaesoubi R. (2006). *A Comparison of Factor Screening Methods for Simulation Models*. Unpublished master's thesis for master's degree, North Carolina State University, Raleigh, NC, USA.
- Zamakhshary M., To T., Guan J., and Langer J. (2008). Risk of incarceration of inguinal hernia among infants and young children awaiting elective surgery. *CMAJ*, 179(10), 1001-5.
- Zhang B., Murali P., Dessouky M. M., and Belson D. (2009). A mixed integer programming approach for allocating operating room capacity. *Journal of the Operational Research Society*, 60, 663-673.

Appendix

Appendix A: The differences between abstract model and real model

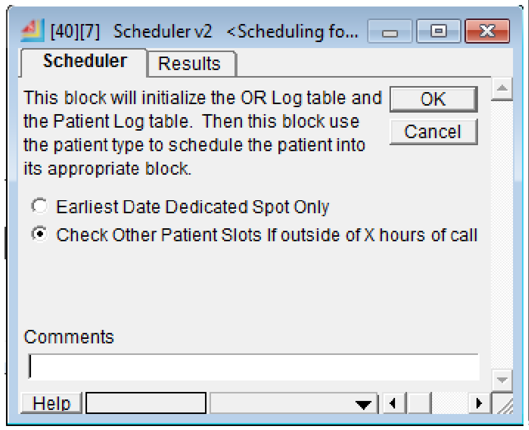
Parameter	Abstract	Real
Hospital size	2 ORs / 4 specialties	4 ORs / 6 specialties
Case duration	1 hour (deterministic)	Lognormal distribution (stochastic)
Unit block cost	3$/hour (same for all surgeon)	Different for each surgeon
OR scheduling policy	Block schedule policy	Modified schedule policy
Patient urgency	All consider urgent	Combination of urgent and non-urgent
Cancellation	There is no cancellation	There is cancellation

Appendix B-1: Surgery scheduling simulation model Main interface



Appendix B-2: Surgery scheduling simulation model code/dialog interface

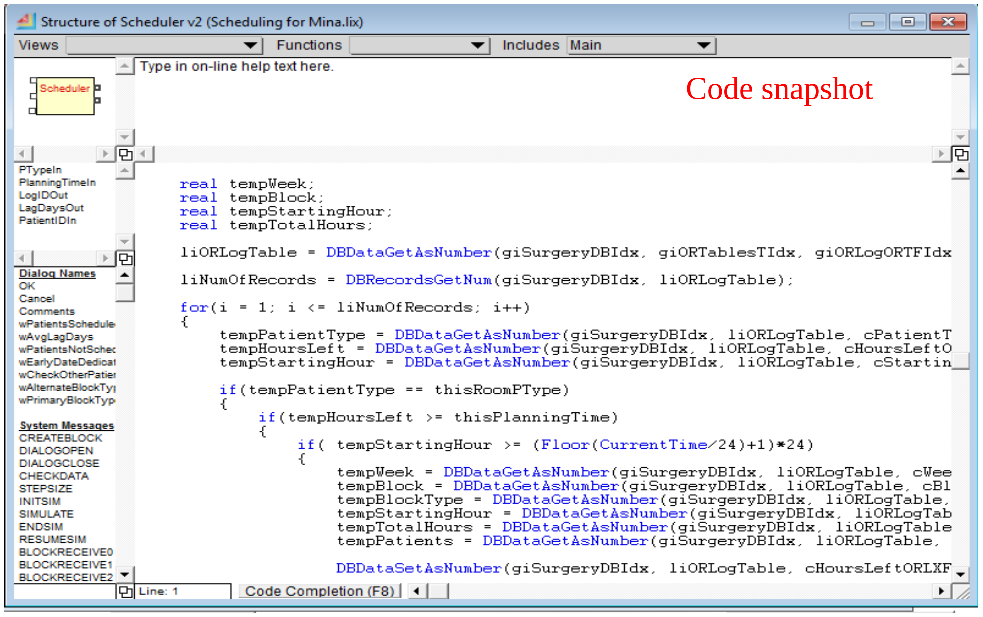
Scenario dialog snapshot



This block will initialize the OR Log table and the Patient Log table. Then this block use the patient type to schedule the patient into its appropriate block.

( ) Earliest Date Dedicated Spot Only
[✓ Check Other Patient Slots If outside of X hours of call]

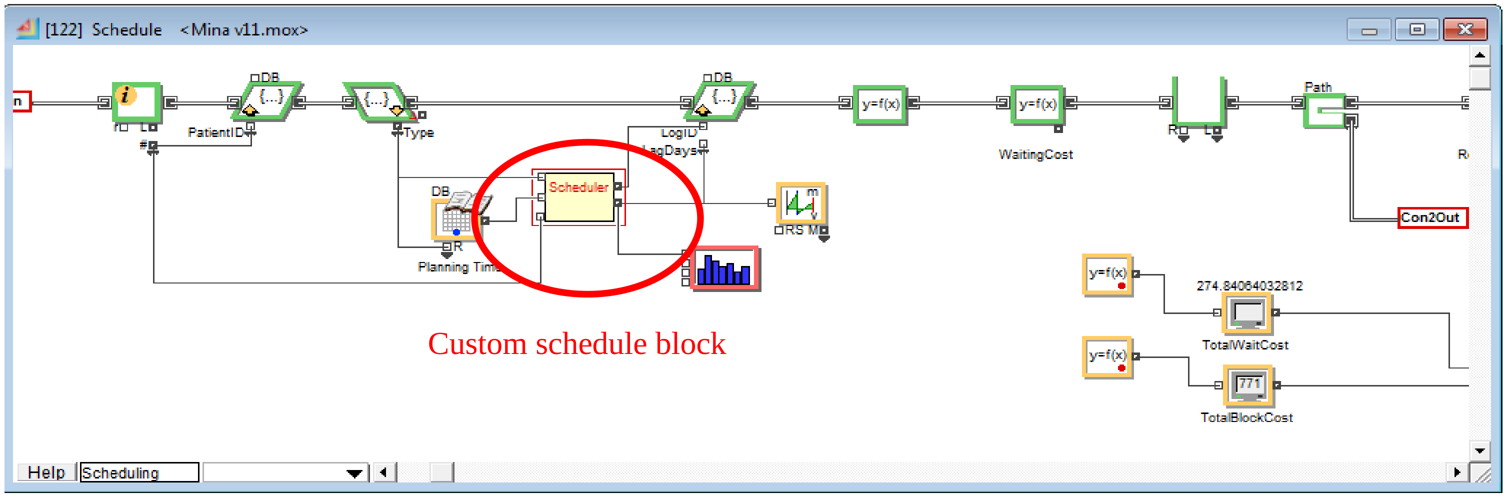
Comments



Type in on-line help text here.

Code snapshot

```
real tempWeek;
real tempBlock;
real tempStartingHour;
real tempTotalHours;

liORLogTable = DBDataGetAsNumber(giSurgeryDBIdx, giORTablesTIdx, giORLogORTFIdx

liNumOfRecords = DBRecordsGetNum(giSurgeryDBIdx, liORLogTable);

for(i = 1; i <= liNumOfRecords; i++)
{
    tempPatientType = DBDataGetAsNumber(giSurgeryDBIdx, liORLogTable, cPatientT
    tempHoursLeft = DBDataGetAsNumber(giSurgeryDBIdx, liORLogTable, cHoursLeftO
    tempStartingHour = DBDataGetAsNumber(giSurgeryDBIdx, liORLogTable, cStartin

    if(tempPatientType == thisRoomPType)
    {
        if(tempHoursLeft >= thisPlanningTime)
        {
            if( tempStartingHour >= (Floor(CurrentTime/24)+1)*24)
            {
                tempWeek = DBDataGetAsNumber(giSurgeryDBIdx, liORLogTable, cWee
                tempBlock = DBDataGetAsNumber(giSurgeryDBIdx, liORLogTable, cBl
                tempBlockType = DBDataGetAsNumber(giSurgeryDBIdx, liORLogTable,
                tempStartingHour = DBDataGetAsNumber(giSurgeryDBIdx, liORLogTab
                tempTotalHours = DBDataGetAsNumber(giSurgeryDBIdx, liORLogTable
                tempPatients = DBDataGetAsNumber(giSurgeryDBIdx, liORLogTable,

                DBDataSetAsNumber(giSurgeryDBIdx, liORLogTable, cHoursLeftORLXF
```



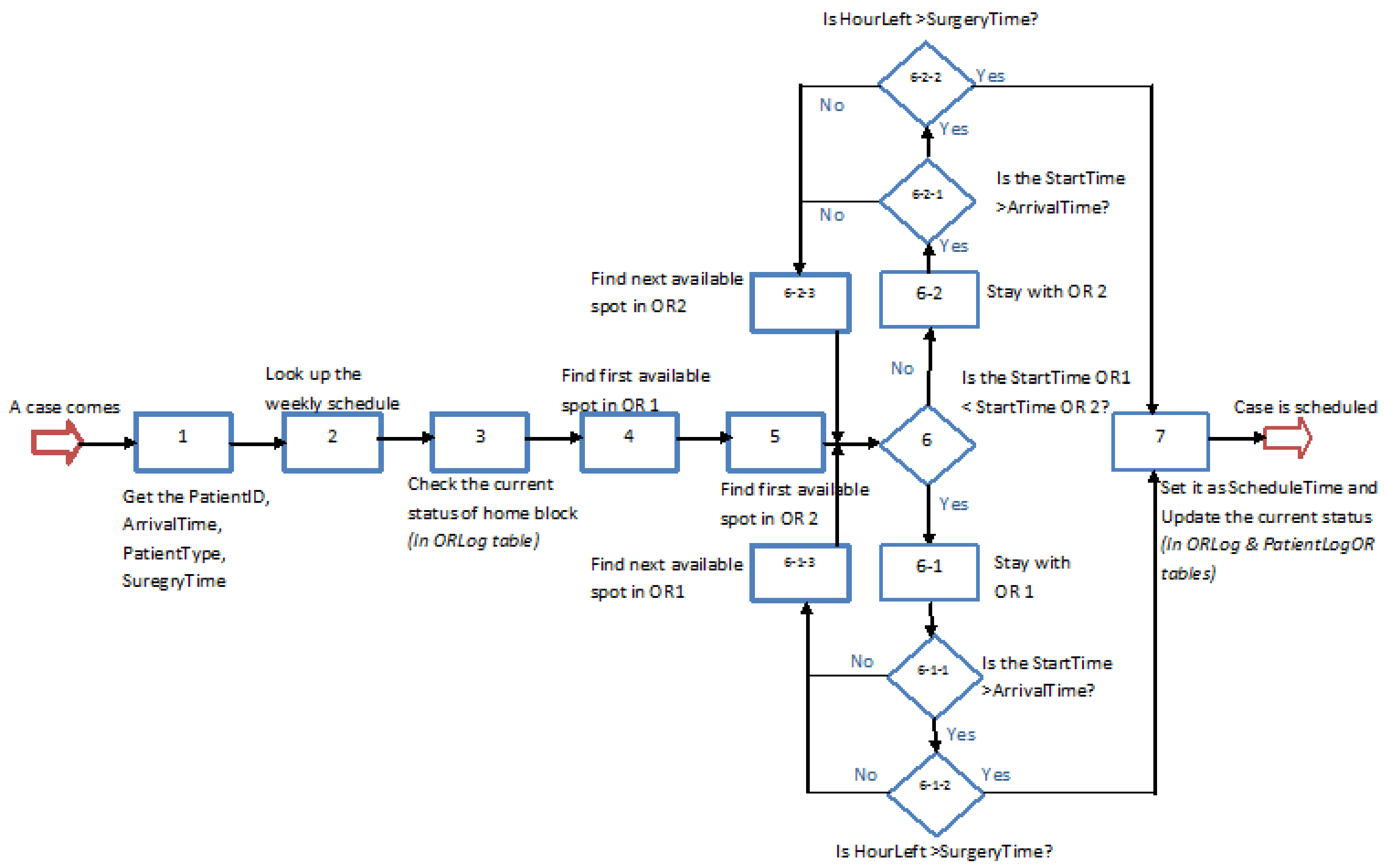
Custom schedule block

Appendix C-1: Simulation algorithm of the scheduling process (Block policy)

Pilot Model

4 Types of patient- 2 ORs

Scenario 1: Each patient just allowed to be scheduled in his home block and not in other blocks



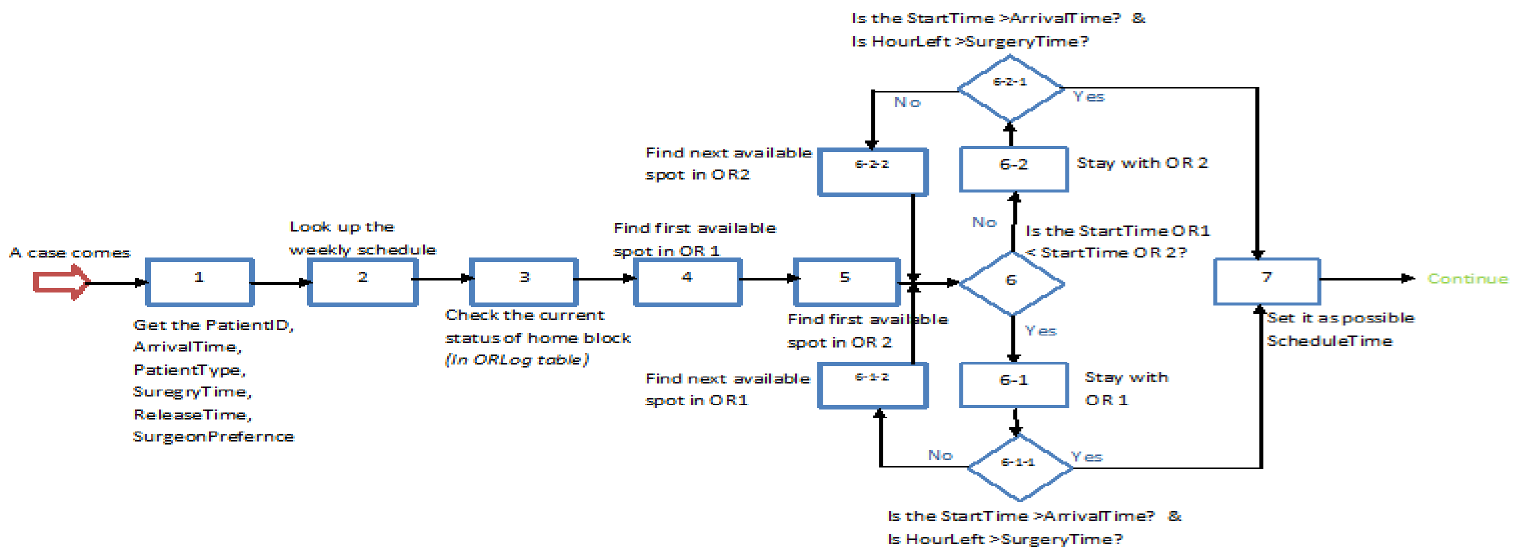
Is HourLeft >SurgeryTime?
6-2-2
Yes
No
Yes
6-2-1
Is the StartTime >ArrivalTime?
No
Yes
Find next available spot in OR2
6-2-3
6-2
Stay with OR 2
No
Is the StartTime OR1 < StartTime OR 2?
Look up the weekly schedule
Find first available spot in OR 1
A case comes
1
2
3
4
5
6
7
Case is scheduled
Get the PatientID, ArrivalTime, PatientType, SuregryTime
Check the current status of home block (In ORLog table)
Find first available spot in OR 2
Yes
Set it as ScheduleTime and Update the current status (In ORLog & PatientLogOR tables)
Find next available spot in OR1
6-1-3
6-1
Stay with OR 1
No
6-1-1
Is the StartTime >ArrivalTime?
Yes
No
6-1-2
Yes
Is HourLeft >SurgeryTime?

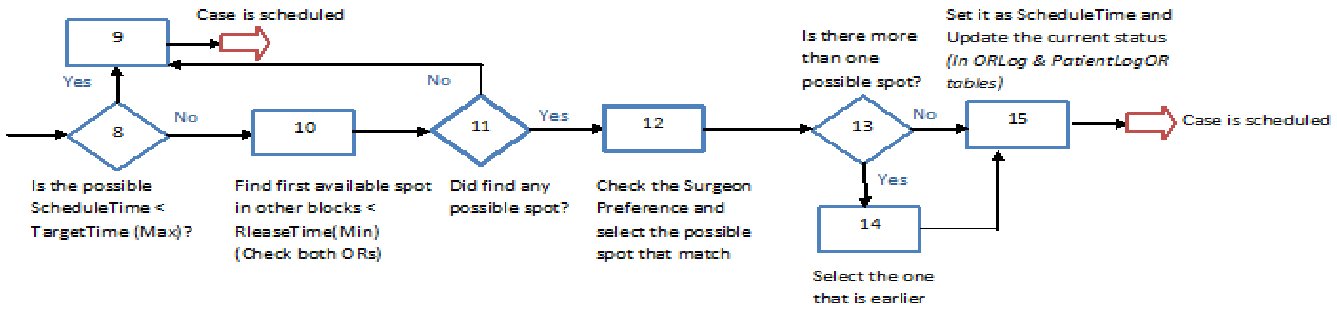
Appendix C-2: Simulation algorithm of the scheduling process (Modify Policy)

Pilot Model

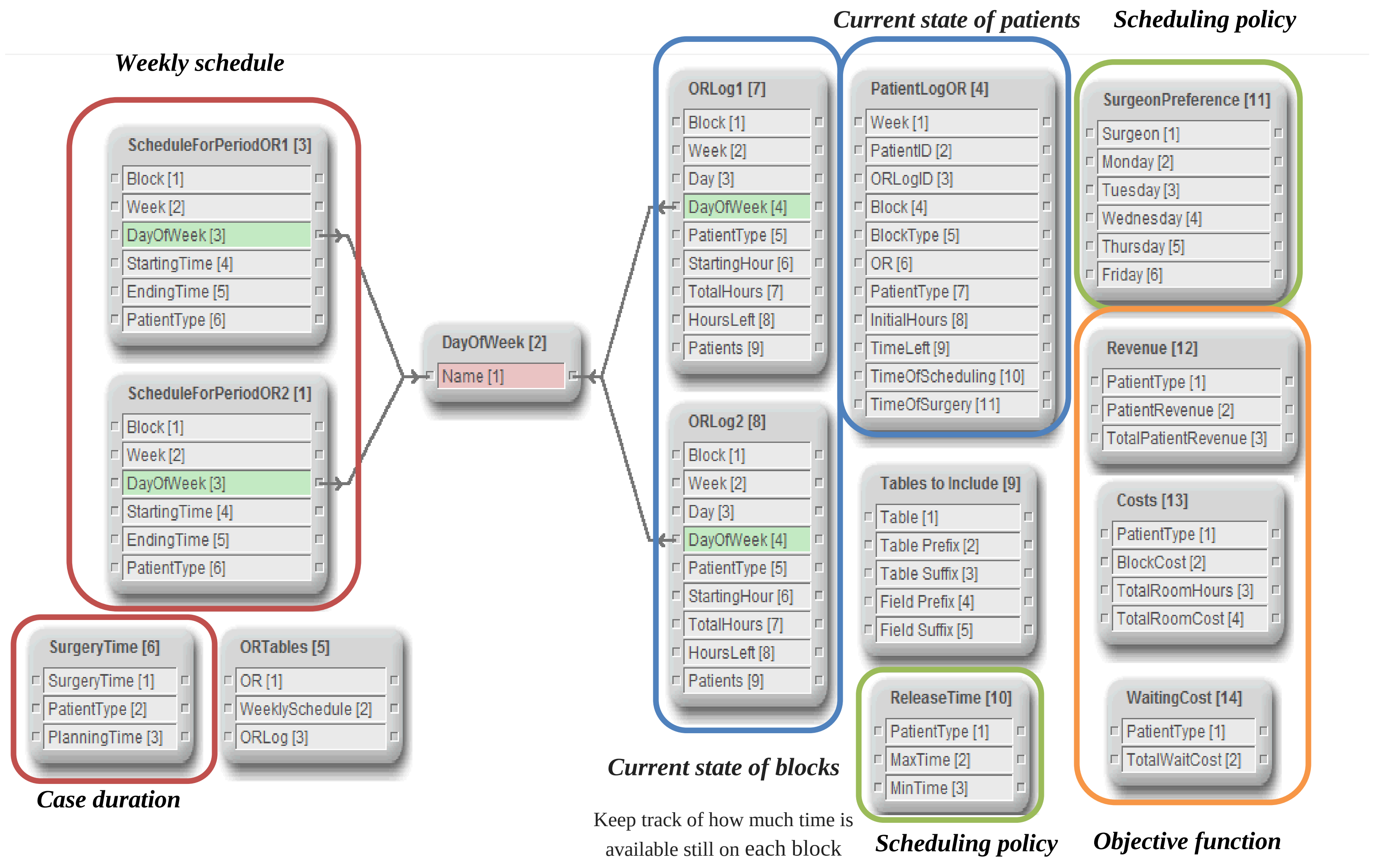
4 Types of patient- 2 ORs

Scenario 2: Each patient allowed to be scheduled in any block at long as it satisfy the logic

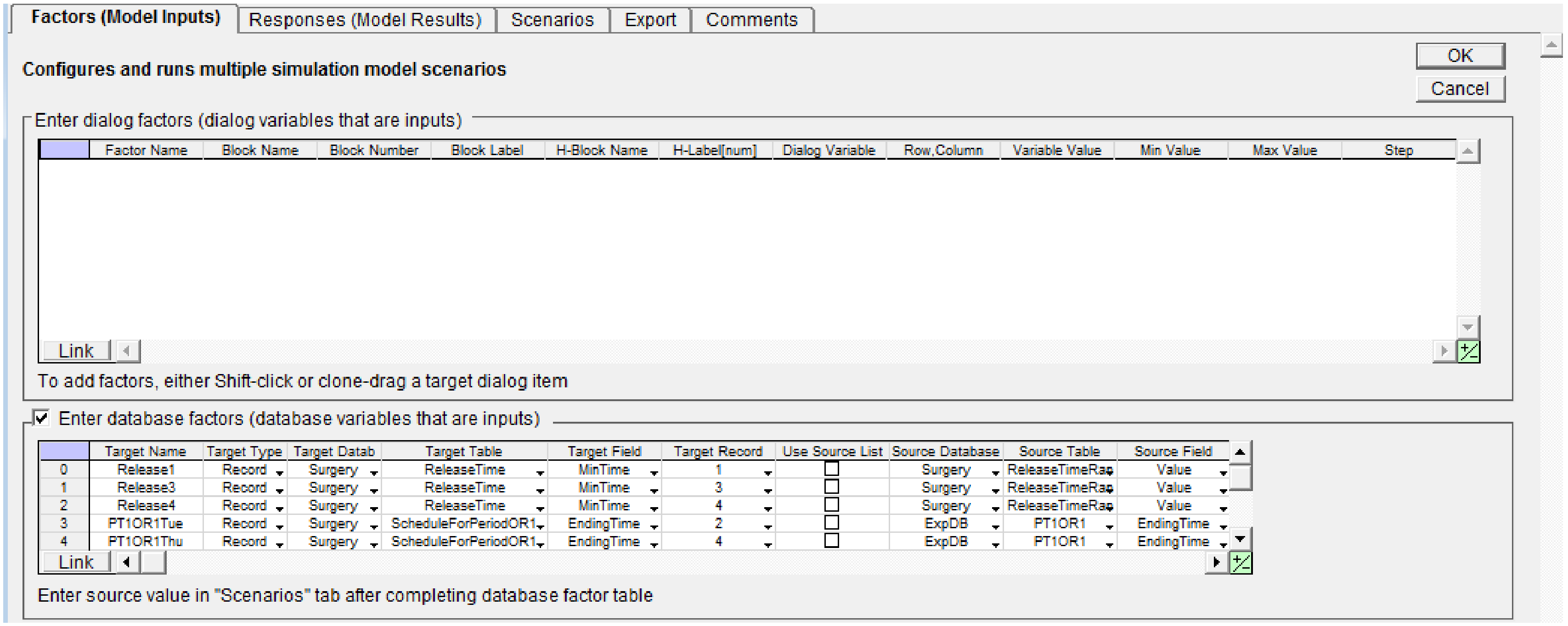

A case comes
1
Get the PatientID, ArrivalTime, PatientType, SuregryTime, ReleaseTime, SurgeonPreferncе
Look up the weekly schedule
2
3
Check the current status of home block (In ORLog table)
Find first available spot in OR 1
4
5
Find first available spot in OR 2
6
Is the StartTime OR1 < StartTime OR 2?
No
Yes
6-2
Stay with OR 2
6-2-1
Is the StartTime >ArrivalTime? & Is HourLeft >SurgeryTime?
No
Yes
6-2-2
Find next available spot in OR2
6-1
Stay with OR 1
6-1-1
Is the StartTime >ArrivalTime? & Is HourLeft >SurgeryTime?
No
Yes
6-1-2
Find next available spot in OR1
7
Set it as possible ScheduleTime
Continue


9
Yes
Case is scheduled
No
8
Is the possible ScheduleTime < TargetTime (Max)?
No
10
Find first available spot in other blocks < RleaseTime(Min) (Check both ORs)
11
Did find any possible spot?
Yes
12
Check the Surgeon Preference and select the possible spot that match
13
Is there more than one possible spot?
No
Yes
14
Select the one that is earlier
15
Set it as ScheduleTime and Update the current status (In ORLog & PatientLogOR tables)
Case is scheduled

Appendix D: Simulation Database



Appendix E: ExtendSim's Scenario manager factors (Model Inputs) tab snapshot



Configures and runs multiple simulation model scenarios

Enter dialog factors (dialog variables that are inputs)

	Factor Name	Block Name	Block Number	Block Label	H-Block Name	H-Label[num]	Dialog Variable	Row,Column	Variable Value	Min Value	Max Value	Step

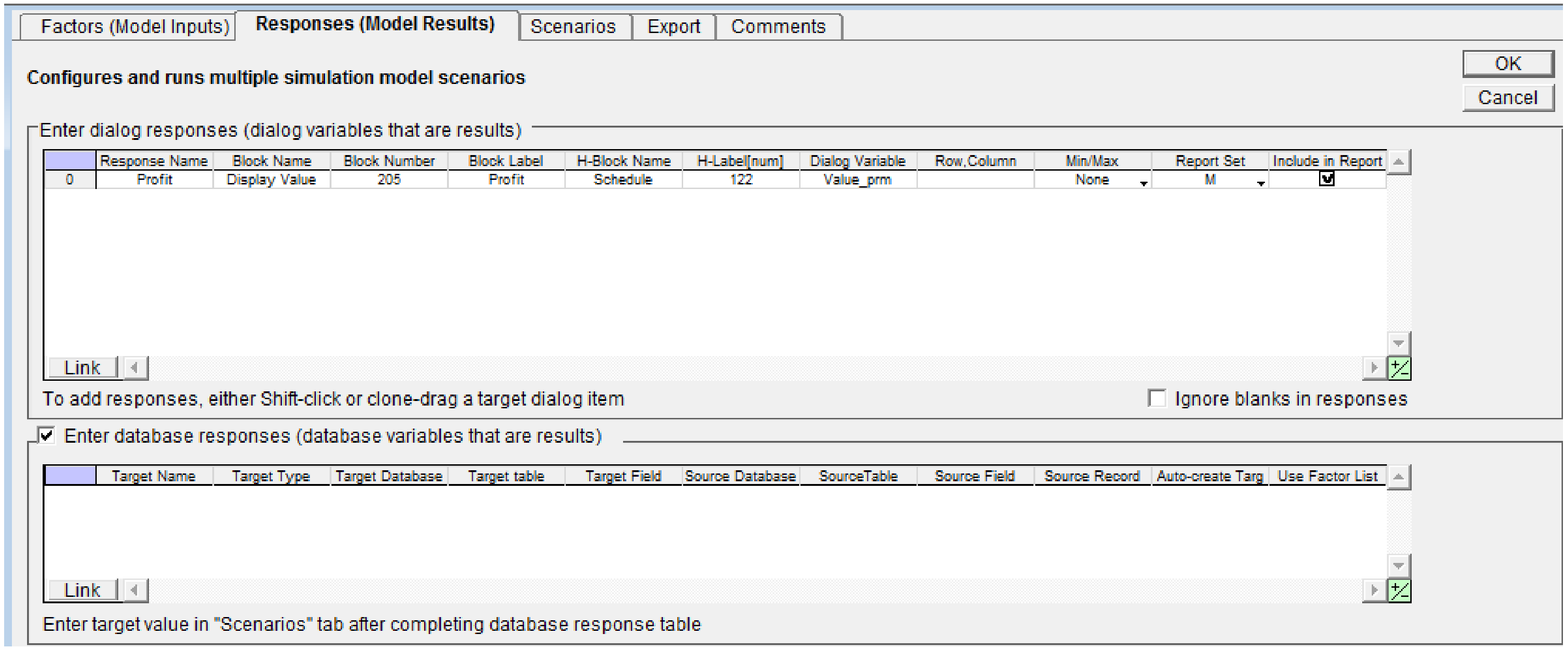
To add factors, either Shift-click or clone-drag a target dialog item

☑ Enter database factors (database variables that are inputs)

	Target Name	Target Type	Target Datab	Target Table	Target Field	Target Record	Use Source List	Source Database	Source Table	Source Field
0	Release1	Record	Surgery	ReleaseTime	MinTime	1	☐	Surgery	ReleaseTimeRa	Value
1	Release3	Record	Surgery	ReleaseTime	MinTime	3	☐	Surgery	ReleaseTimeRa	Value
2	Release4	Record	Surgery	ReleaseTime	MinTime	4	☐	Surgery	ReleaseTimeRa	Value
3	PT1OR1Tue	Record	Surgery	ScheduleForPeriodOR1	EndingTime	2	☐	ExpDB	PT1OR1	EndingTime
4	PT1OR1Thu	Record	Surgery	ScheduleForPeriodOR1	EndingTime	4	☐	ExpDB	PT1OR1	EndingTime

Enter source value in "Scenarios" tab after completing database factor table

Appendix F: ExtendSim's Scenario manager response (Model Results) tab snapshot



Configures and runs multiple simulation model scenarios

Enter dialog responses (dialog variables that are results)

	Response Name	Block Name	Block Number	Block Label	H-Block Name	H-Label[num]	Dialog Variable	Row,Column	Min/Max	Report Set	Include in Report
0	Profit	Display Value	205	Profit	Schedule	122	Value_prm		None	M	✓

To add responses, either Shift-click or clone-drag a target dialog item

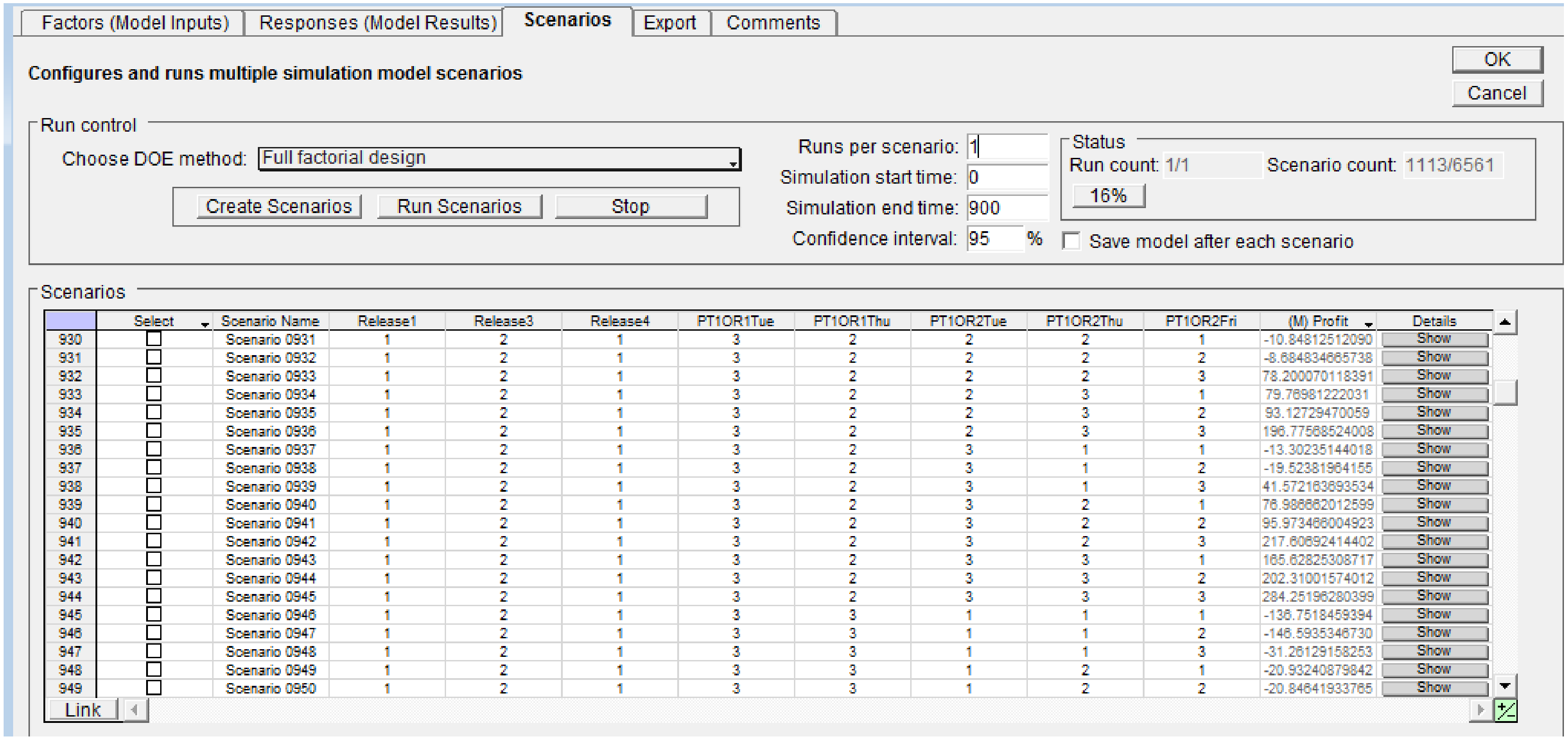
☐ Ignore blanks in responses

☑ Enter database responses (database variables that are results)

	Target Name	Target Type	Target Database	Target table	Target Field	Source Database	SourceTable	Source Field	Source Record	Auto-create Targ	Use Factor List

Enter target value in "Scenarios" tab after completing database response table

Appendix G: ExtendSim’s Scenario manager scenario (upon DOE method) tab snapshot



Configures and runs multiple simulation model scenarios

Run control

Choose DOE method: Full factorial design

Runs per scenario: 1
Simulation start time: 0
Simulation end time: 900
Confidence interval: 95 %

Status
Run count: 1/1 Scenario count: 1113/6561
16%

☐ Save model after each scenario

Scenarios

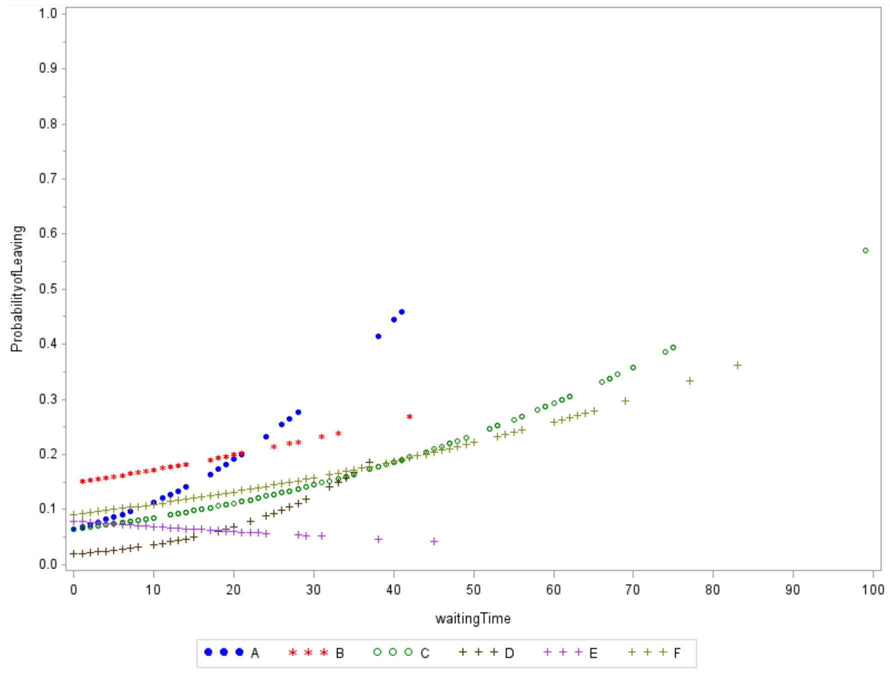
	Select	Scenario Name	Release1	Release3	Release4	PT1OR1Tue	PT1OR1Thu	PT1OR2Tue	PT1OR2Thu	PT1OR2Fri	(M) Profit	Details
930	☐	Scenario 0931	1	2	1	3	2	2	2	1	-10.84812512090	Show
931	☐	Scenario 0932	1	2	1	3	2	2	2	2	-8.684834665738	Show
932	☐	Scenario 0933	1	2	1	3	2	2	2	3	78.200070118391	Show
933	☐	Scenario 0934	1	2	1	3	2	2	3	1	79.76981222031	Show
934	☐	Scenario 0935	1	2	1	3	2	2	3	2	93.12729470059	Show
935	☐	Scenario 0936	1	2	1	3	2	2	3	3	196.77568524008	Show
936	☐	Scenario 0937	1	2	1	3	2	3	1	1	-13.30235144018	Show
937	☐	Scenario 0938	1	2	1	3	2	3	1	2	-19.52381964155	Show
938	☐	Scenario 0939	1	2	1	3	2	3	1	3	41.57216393534	Show
939	☐	Scenario 0940	1	2	1	3	2	3	2	1	76.986662012599	Show
940	☐	Scenario 0941	1	2	1	3	2	3	2	2	95.973466004923	Show
941	☐	Scenario 0942	1	2	1	3	2	3	2	3	217.60692414402	Show
942	☐	Scenario 0943	1	2	1	3	2	3	3	1	165.62825308717	Show
943	☐	Scenario 0944	1	2	1	3	2	3	3	2	202.31001574012	Show
944	☐	Scenario 0945	1	2	1	3	2	3	3	3	284.25196280399	Show
945	☐	Scenario 0946	1	2	1	3	3	1	1	1	-136.7518459394	Show
946	☐	Scenario 0947	1	2	1	3	3	1	1	2	-146.5935346730	Show
947	☐	Scenario 0948	1	2	1	3	3	1	1	3	-31.26129158253	Show
948	☐	Scenario 0949	1	2	1	3	3	1	2	1	-20.93240879842	Show
949	☐	Scenario 0950	1	2	1	3	3	1	2	2	-20.84641933765	Show

Appendix H: Logistic regression for 6 cluster (Stafford)

Model Fit Statistics		
Criterion	Intercept Only	Intercept and Covariates
AIC	836.705	811.626
SC	841.776	872.485
-2 Log L	834.705	787.626

Type 3 Analysis of Effects			
Effect	DF	Wald Chi-Square	Pr > ChiSq
waitingTime	1	5.6642	0.0173
Group	5	6.8214	0.2343
waitingTime*Group	5	4.8985	0.4284

Analysis of Maximum Likelihood Estimates						
Parameter		DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept		1	-2.3045	0.2728	71.3442	<.0001
waitingTime		1	0.0209	0.00878	5.6642	0.0173
Group	A	1	-0.3613	0.5330	0.4596	0.4978
Group	B	1	0.5531	0.6216	0.7917	0.3736
Group	C	1	-0.3694	0.4857	0.5783	0.4470
Group	D	1	-1.6538	0.7627	4.7016	0.0301
Group	E	1	-0.1603	0.4684	0.1171	0.7322
waitingTime*Group	A	1	0.0401	0.0291	1.8996	0.1681
waitingTime*Group	B	1	-0.00298	0.0345	0.0075	0.9310
waitingTime*Group	C	1	0.00898	0.0129	0.4824	0.4873
waitingTime*Group	D	1	0.0463	0.0359	1.6641	0.1971
waitingTime*Group	E	1	-0.0355	0.0360	0.9712	0.3244



Appendix I: Logistic regression SAS code

```
/* Read Data */
Proc Import out= Data Datafile= "H:\my documents\logistic\semiurgent11j.xlsx"
DBMS = xlsx REPLACE;
Run;

/* define input and output data sets */
proc logistic data=data;
class Group/param =ref;
model status(event='1')= waitingTime |Group;
output out = result
p=pred;
run;

/* define symbol characteristics */
symbol1  value=dot   color=Blue height=0.7;
symbol2  value=star color=red height=0.7;
symbol3  value=circle color=green height=0.7;

/* define legend characteristics */
legend1 label=none frame;

/* define axis characteristics */
axis1 label=("waitingTime") minor=none offset=(1,1);
axis2 label=(angle=90 "ProbabilityofLeaving")
      order=(0 to 1 by 0.1) minor=(n=1);

/* Plot result using gplot function */
proc gplot data= result;
   plot pred*waitingtime=Group  / overlay legend=legend1
 haxis=axis1 vaxis=axis2 ;
run;
```

Appendix J: Cluster Analysis SAS code

```
Proc Import out= Data Datafile= "H:\my documents\logistic\semiurgent11.xlsx"
DBMS = xlsx REPLACE;
Run;

ods graphics on;
proc cluster data=Data method=ward ccc pseudo trim=10 k=50 print=25;
var waitingTime status;
copy Group;
run;
ods graphics off;

ods graphics on;
proc cluster data=Data method=average ccc pseudo trim=10 k=50 print=25;
var waitingTime status;
copy Group;
run;
ods graphics off;
```

Appendix K: Visual Basic code, used to generate the best sequential decisions

```
/*write output file, including "Appointment Time, Patient Id, Patient Type, Arrival Time, OR Type"*/

Imports System.IO
Imports System.Text

Public Class Processor
   Public Sub Process(inputFile As String, outputFile As String, Type1Cost As Integer, Type2Cost As Integer,
NumberofAppointment As Integer)
      Dim inputResult As InputPatients = ReadInputPatient(inputFile)
      Dim output As Result = AssignPatientToRoom(inputResult, Type1Cost, Type2Cost, NumberofAppointment)

      Dim ora As New StringBuilder
      ora.AppendLine("Appointment Time, Patinet Id, Patient Type,Arrival Time,OR Type")
      For Each key In output.ORA.PatientList.Keys
         ora.AppendLine(key.ToString & "," & output.ORA.PatientList(key).Id & "," &
output.ORA.PatientList(key).PatientType & "," & output.ORA.PatientList(key).ArrivalTime & ",1")
      Next

      Dim orb As New StringBuilder

      For Each key In output.ORB.PatientList.Keys
         orb.AppendLine(key.ToString & "," & output.ORB.PatientList(key).Id & "," &
output.ORB.PatientList(key).PatientType & "," & output.ORB.PatientList(key).ArrivalTime & ",2")
      Next

      File.WriteAllText(outputFile, ora.ToString & orb.ToString)
   End Sub

   Private Function ReadInputPatient(path As String) As InputPatients
      Dim inputLines() As String = File.ReadAllLines(path)
      Dim output As New InputPatients With {.Patients = New List(Of Patient)}

      For i As Integer = 0 To inputLines.Count - 1
         Dim line() As String = inputLines(i).Split(",")
         output.Patients.Add(New Patient With {.Id = i, .ArrivalTime = line(0), .PatientType = line(1)})
      Next

      Return output
   End Function


/* Initialize: Read input file (input file including Arrival time and Patient Type)*/



   Private Function AssignPatientToRoom(ByVal input As InputPatients, Type1Cost As Integer, Type2Cost As Integer,
NumberofAppointment As Integer) As Result
      Dim output As New Result With {.ORA = New ORRoom With {.PatientList = New Dictionary(Of Integer, Patient)},
.ORB = New ORRoom With {.PatientList = New Dictionary(Of Integer, Patient)}}

      For i As Integer = 0 To NumberofAppointment
         FindPatientForAppointment(i, input, output, 2)
      Next

      For i As Integer = 0 To NumberofAppointment
```

```
            FindPatientForAppointment(i, input, output, 1)
        Next

        Return output
    End Function

    Private Sub FindPatientForAppointment(timeId As Integer, inputPatient As InputPatients, output As Result, fillType As
Integer)
        Console.WriteLine(timeId)

        'first find patient type, fill type which has not been assigned yet
        Dim typeBefore As List(Of Patient) = inputPatient.Patients.Where(Function(x) x.ArrivalTime <= timeId AndAlso
x.PatientType = fillType AndAlso
                                                                    output.ORB.PatientList.Where(Function(y) y.Value.Id = x.Id).Count = 0
AndAlso
                                                                    output.ORA.PatientList.Where(Function(y) y.Value.Id = x.Id).Count = 0
                                                                    ).OrderBy(Function(x) x.ArrivalTime).ToList

        If typeBefore.Count = 0 Then
            Return
        End If


        For Each p As Patient In typeBefore

            'fill the first available slot for type
            Dim hasTime As Boolean = False
            Dim currentTime As Integer = timeId

            While Not hasTime
                If Not output.ORA.PatientList.ContainsKey(currentTime) Then
                    output.ORA.PatientList.Add(currentTime, p)
                    hasTime = True
                End If

                If Not hasTime AndAlso Not output.ORB.PatientList.ContainsKey(currentTime) Then
                    output.ORB.PatientList.Add(currentTime, p)
                    hasTime = True
                End If

                currentTime += 1
            End While
        Next

    End Sub


End Class
```

Appendix L: Optimality proof of 2-room scheduling policy

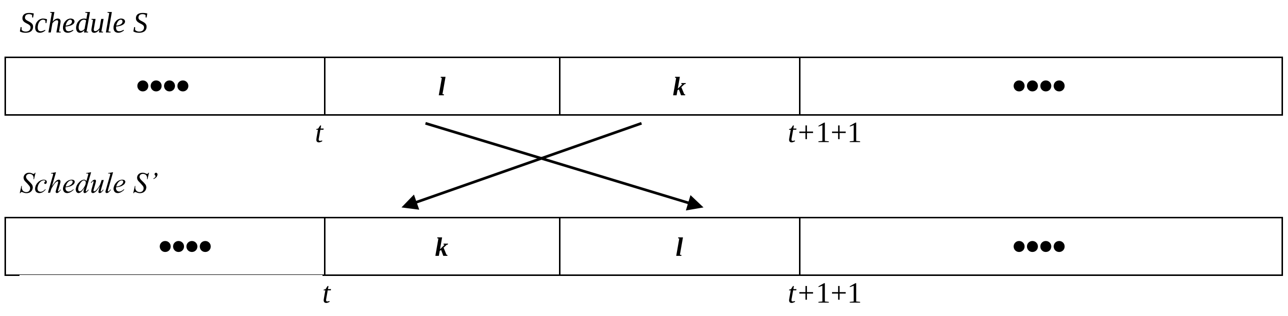
The objective is to find the minimum waiting cost across all patients. The weight p_j of case j represents a waiting penalty per unit time. Theorem 1 formalizes the optimality of sequencing the cases where the high priority patients are assigned first and then the low priority patients are assigned in non-decreasing order of arrival time in remaining spots. In other words, the goal is to sequence the cases such that all high priority patients that arrived before lower priority patients are assigned before any lower priority patients (referred to BA sequencing). The proof of optimality is based on a useful technique called the method of adjacent pairwise interchange,

Theorem 1: The total waiting time is minimized by BA sequencing

Proof: By contradiction, suppose a schedule S that is not BA sequence is optimal. In this schedule, there must be at least two adjacent cases-say case *l* followed by case *k*, such that

$$p_l < p_k$$

Under the original schedule S, job *l* starts its processing at time t and is followed by job *k*. All other jobs remain in their original positions. Refer to the new schedule S'. The total waiting cost of cases processed before case l and k is not affected by the interchange. Neither is the total waiting cost of cases processed after case *l* and *k*. Thus, the difference in the waiting cost under schedules *S* and *S'* is due only to cases *l* and *k* (see following figure)



Under *S*, the total waiting cost is, $(t + 1)p_l + (t + 1 + 1)p_k$

Whereas under *S'* and assuming can be started at time t, the cost is,$(t + 1)p_k + (t + 1 + 1)p_l$

It is easily verified that if $p_l < p_k$, the sum of the two waiting cost functions under *S'* is strictly less than under *S*. This contradicts the optimality of S and completes the proof of the theorem.

Appendix M: ExtendSim Code for optimal scheduling

```
// Declare constants and static variables
Real		grTotalLagDays;
Integer	giTotalPatients;
Real		EarliestDate[5][13]; // room / PT Type
Real		EarliestRecord[5][13]; // room / PT Type
Integer	possiblePType[12];//this is holding the competetor to the spot, this means if we have a spot for ptype 1 what other types can be there
Integer	giMaxORRooms;
Integer	giMaxPatientTypes;
Integer	giSurgeryDBIdx;
Integer	giPatientLogORTIdx;
Integer	giORLogIDPLORFIdx;
Integer	giWeekPLORFIdx;
Integer	giBlockPLORFIdx;
Integer	giBlockTypePLORFIdx;
Integer	giORPLORFIdx;
Integer	giPatientTypePLORFIdx;
Integer	giInitialHoursPLORFIdx;
Integer	giTimeLeftPLORFIdx;
Integer	giTimeOfSchedulingPLORFIdx;
Integer	giPatientIDPLORFIdx;
Integer	giTimeOfSurgeryPLORFIdx;
Integer	giDaysOutToSchedulePLORFIdx;
Integer	giPlanningTimePLORFIdx;
Integer	giORTablesTIdx;
Integer	giWeeklyScheduleORTFIdx;
Integer	giORLogORTFIdx;
Integer	giReleaseTimeTIdx;
Integer	giPatientTypeRTFIdx;
Integer	giMaxTimeRTFIdx;
Integer	giMinTimeRTFIdx;
Integer	giSecondaryArrivalRateTIdx;
Integer	giSurgeonPreferenceTIdx;
Integer	giMondaySPFIdx;
Integer	giTuesdaySPFIdx;
Integer	giWednesdaySPFIdx;
Integer	giThursdaySPFIdx;
Integer	giFridaySPFIdx;
Integer	giRevenueTIdx;
Integer	giTotalPatientRevenueFIdx;
Integer	giCostsTIdx;
Integer	giRevenueTFIdx;
Integer	giPatientRevenueFIdx;
Integer	giBlockCostFIdx;
Integer	giTotalRoomHoursFIdx;
Integer	giTotalRoomCostFIdx;
Integer	giScheduleType;
Integer	giSurgeryTimeTIdx;
Integer	giPlanningtimeFIdx;
Integer	giSecArrivalRateFIdx;
Constant		cBlockSPORXFIdx is 1;
ConstantcWeekSPORXFIdx is 2;
```

```
Constant		cDayOfWeekSPORXFIdx is 3;
Constant		cStartingTimeSPORXFIdx is 4;
Constant		cEndingTimeSPORXFIdx is 5;
Constant		cPatientTypeSPORXFIdx is 6;
Constant		cBlockORLXFIdx is 1;
Constant		cWeekORLXFIdx is 2;
Constant	cDayORLXFIdx is 3;
Constant	cDayOfWeekORLXFIdx is 4;
Constant		cPatientTypeORLXFIdx is 5;
Constant		cStartingHourORLXFIdx is 6;
Constant		cTotalHoursORLXFIdx is 7;
Constant		cHoursLeftORLXFIdx is 8;
Constant		cPatientsORLXFIdx is 9;
Constant	cBlockWSORFIdx is 1;
Constant	cDayOfWeekWSORFIdx is 2;
Constant	cStartingTimeWSORFIdx is 3;
Constant	cEndingTimeWSORFIdx is 4;
Constant	cPatientTypeWSORFIdx is 5;
Constant	cPatientType0 is 0;


//	This procedure will capture all of the database variables needed during the simulation run.  This will get
called at the beginning of the run.

Procedure GetDBVariables()
{
	giSurgeryDBIdx = DBDatabaseGetIndex( "Surgery" );
	giPatientLogORTIdx = DBTableGetIndex( giSurgeryDBIdx, "PatientLogOR" );
	giORLogIDPLORFIdx = DBFieldGetIndex( giSurgeryDBIdx, giPatientLogORTIdx, "ORLogID" );
	giWeekPLORFIdx = DBFieldGetIndex( giSurgeryDBIdx, giPatientLogORTIdx, "Week" );
	giBlockPLORFIdx = DBFieldGetIndex( giSurgeryDBIdx, giPatientLogORTIdx, "Block" );
	giBlockTypePLORFIdx = DBFieldGetIndex( giSurgeryDBIdx, giPatientLogORTIdx, "BlockType" );
	giORPLORFIdx = DBFieldGetIndex( giSurgeryDBIdx, giPatientLogORTIdx, "OR" );
	giPatientTypePLORFIdx = DBFieldGetIndex( giSurgeryDBIdx, giPatientLogORTIdx, "PatientType" );
	giInitialHoursPLORFIdx = DBFieldGetIndex( giSurgeryDBIdx, giPatientLogORTIdx, "InitialHours" );
	giTimeLeftPLORFIdx = DBFieldGetIndex( giSurgeryDBIdx, giPatientLogORTIdx, "TimeLeft" );
	giTimeOfSchedulingPLORFIdx = DBFieldGetIndex( giSurgeryDBIdx, giPatientLogORTIdx,
	"TimeOfScheduling" );
	giTimeOfSurgeryPLORFIdx = DBFieldGetIndex( giSurgeryDBIdx, giPatientLogORTIdx,
	"TimeOfSurgery" );
	giPatientIDPLORFIdx = DBFieldGetIndex( giSurgeryDBIdx, giPatientLogORTIdx, "PatientID" );
	giDaysOutToSchedulePLORFIdx = DBFieldGetIndex( giSurgeryDBIdx, giPatientLogORTIdx,
	"DaysOutToSchedule" );
	giPlanningTimePLORFIdx = DBFieldGetIndex( giSurgeryDBIdx, giPatientLogORTIdx, "PlanningTime"
	);
	giORTablesTIdx = DBTableGetIndex( giSurgeryDBIdx, "ORTables" );
	giWeeklyScheduleORTFIdx = DBFieldGetIndex( giSurgeryDBIdx, giORTablesTIdx, "WeeklySchedule" );
	giORLogORTFIdx = DBFieldGetIndex( giSurgeryDBIdx, giORTablesTIdx, "ORLog" );
	giReleaseTimeTIdx = DBTableGetIndex( giSurgeryDBIdx, "ReleaseTime" );
	giPatientTypeRTFIdx = DBFieldGetIndex( giSurgeryDBIdx, giReleaseTimeTIdx, "PatientType" );
	giMaxTimeRTFIdx = DBFieldGetIndex( giSurgeryDBIdx, giReleaseTimeTIdx, "MaxTime" );
	giMinTimeRTFIdx = DBFieldGetIndex( giSurgeryDBIdx, giReleaseTimeTIdx, "Mintime" );
	giSurgeonPreferenceTIdx = DBTableGetIndex( giSurgeryDBIdx, "SurgeonPreference" );
```

```
          giMondaySPFIdx = DBFieldGetIndex( giSurgeryDBIdx, giSurgeonPreferenceTIdx, "Monday" );
          giTuesdaySPFIdx = DBFieldGetIndex( giSurgeryDBIdx, giSurgeonPreferenceTIdx, "Tuesday" );
          giWednesdaySPFIdx = DBFieldGetIndex( giSurgeryDBIdx, giSurgeonPreferenceTIdx, "Wednesday" );
          giThursdaySPFIdx = DBFieldGetIndex( giSurgeryDBIdx, giSurgeonPreferenceTIdx, "Thursday" );
          giFridaySPFIdx = DBFieldGetIndex( giSurgeryDBIdx, giSurgeonPreferenceTIdx, "Friday" );
          giRevenueTIdx = DBTableGetIndex( giSurgeryDBIdx, "Revenue" );
          giPatientRevenueFIdx = DBFieldGetIndex( giSurgeryDBIdx, giRevenueTIdx, "PatientRevenue" );
          giTotalPatientRevenueFIdx = DBFieldGetIndex( giSurgeryDBIdx, giRevenueTIdx, "TotalPatientRevenue"
          );
          giCostsTIdx = DBTableGetIndex( giSurgeryDBIdx, "Costs" );
          giBlockCostFIdx = DBFieldGetIndex( giSurgeryDBIdx, giCostsTIdx, "BlockCost");
          giTotalRoomHoursFIdx = DBFieldGetIndex( giSurgeryDBIdx, giCostsTIdx, "TotalRoomHours");
          giTotalRoomCostFIdx = DBFieldGetIndex( giSurgeryDBIdx, giCostsTIdx, "TotalRoomCost");
          giMaxORRooms = DBRecordsGetNum(giSurgeryDBIdx, giORTablesTIdx);
          giMaxPatientTypes = -1 + DBRecordsGetNum(giSurgeryDBIdx, giCostsTIdx);
          giSurgeryTimeTIdx = DBTableGetIndex( giSurgeryDBIdx, "SurgeryTime");
          giPlanningtimeFIdx = DBFieldGetIndex( giSurgeryDBIdx,
          giSurgeryTimeTIdx, "PlanningTime");
        giSecondaryArrivalRateTIdx = DBTableGetIndex( giSurgeryDBIdx,     "secondaryArrival");
        giSecArrivalRateFIdx = DBFieldGetIndex( giSurgeryDBIdx,    giSurgeryTimeTIdx, "secondaryArrival");
        giSecondaryArrivalRateTIdx = DBTableGetIndex( giSurgeryDBIdx, "secondaryArrival");
        giSecArrivalRateFIdx = DBFieldGetIndex( giSurgeryDBIdx,  giSurgeryTimeTIdx, "secondaryArrival");

}

//        This function is testing to see if the specified patient type has been already scheduled in another OR at the
same time.

Integer TestIfPatientTypeIsScheduledAtTheSameTimeSomeWhereElse(integer thisOR, integer thisPType, real
thisSimulationTime, real thisPlanningTime)
{
          integer liORToTest;
          integer i;
          integer liNumOfRecords;
          integer tempPatientType;
          integer liORLogTable;
          integer liDrNotUsedElseWhere;

          real      tempHoursLeft;
          real      tempSimulationTime;
          real      tempTotalHours;
          real      tempBeginTimeORBlock;
          real      tempEndTimeORBlock;
          real      tempTestBeginTime;
          real      tempTestEndTime;

          for(liORToTest = 1; liORToTest <= giMaxORRooms; liORToTest++)
          {
                    if(liORToTest == thisOR)
                              Continue;

                    liORLogTable = DBDataGetAsNumber(giSurgeryDBIdx, giORTablesTIdx, giORLogORTFIdx,
liORToTest);
```

```
                    liNumOfRecords = DBRecordsGetNum(giSurgeryDBIdx, liORLogTable);

                    for(i = 1; i <= liNumOfRecords; i++)
                    {
tempPatientType = DBDataGetAsNumber(giSurgeryDBIdx, liORLogTable, cPatientTypeORLXFIdx, i);
tempHoursLeft = DBDataGetAsNumber(giSurgeryDBIdx, liORLogTable, cHoursLeftORLXFIdx, i);
tempSimulationTime = DBDataGetAsNumber(giSurgeryDBIdx, liORLogTable, cStartingHourORLXFIdx, i);
tempTotalHours = DBDataGetAsNumber(giSurgeryDBIdx, liORLogTable, cTotalHoursORLXFIdx, i);

tempBeginTimeORBlock = tempSimulationTime + tempTotalHours - tempHoursLeft;
tempEndTimeORBlock = tempSimulationTime + tempTotalHours;

          if( tempPatientType != thisPType)
                              Continue;

// If the block of time is in the future then the Begin time of the OR Block will be larger then the ending time of the patient scheduled time.
          if (tempBeginTimeORBlock > thisSimulationTime + thisPlanningTime)
                              Continue;

// If the block of time is in the past then the Ending time of the OR Block will be smaller then the starting time of the patient scheduled time. At this point I can pass back the false because the rest of the records will be in the future
          if(tempEndTimeORBlock < thisSimulationTime)
                              Return False;

// Finally, it is the same patient, it is not TOO early and not TOO late.
                         return True;
                    }
          }

// If it hasn't hit the return TRUE option then it is in the same block of time.
          return False;
}
// this function will search the OR Log table for the next spot in the specified OR for a specified Patient Type. If it doesn't find a spot it will return -1.  If it does then it will return the time that it could          be scheduled.  If the actual patient type is not the block type you are looking for then another test will be done to make sure the doctor isn't already simultaniously being used.

// The thisPType variable is the patient being scheduled
// The thisPossiblePType variable is the spot we want to test to see if the current patient can be scheduled in.

Real TestForNextAvailabileSpotForType(integer thisOR, integer thisPossiblePType, real thisPlanningTime, integer thisPType)
{
          integer i;
          integer liNumOfRecords;
          integer tempPatientType;
          integer liORLogTable;
          integer liPatientTypeIsAlreadyScheduledElseWhere;
          integer tempDay;
          integer tempTest;
          integer parentArray[3];
          integer tempDayOfWeek;
```

```
        real    tempHoursLeft;
        real    tempSimulationTime;
        real    tempTotalHours;
        real    tempNextActualAvailableTime;
        real    tempValue;


        liORLogTable = DBDataGetAsNumber(giSurgeryDBIdx, giORTablesTIdx, giORLogORTFIdx, thisOR);
        liNumOfRecords = DBRecordsGetNum(giSurgeryDBIdx, liORLogTable);

        for(i = 1; i <= liNumOfRecords; i++)
        {
                tempPatientType = DBDataGetAsNumber(giSurgeryDBIdx, liORLogTable,
cPatientTypeORLXFIdx, i);
                tempHoursLeft = DBDataGetAsNumber(giSurgeryDBIdx, liORLogTable, cHoursLeftORLXFIdx,
i);
                tempSimulationTime = DBDataGetAsNumber(giSurgeryDBIdx, liORLogTable,
cStartingHourORLXFIdx, i);
                tempTotalHours = DBDataGetAsNumber(giSurgeryDBIdx, liORLogTable,
cTotalHoursORLXFIdx, i);
                tempDay = DBDataGetAsNumber(giSurgeryDBIdx, liORLogTable, cDayORLXFIdx, i);
                DBDataGetParent(giSurgeryDBIdx, liORLogTable, cDayOfWeekORLXFIdx, i, parentArray);
                tempDayOfWeek = parentArray[2];


                tempTest = true;

                if( tempPatientType != thisPossiblePType )
                        tempTest = False;

                if(tempHoursLeft < thisPlanningTime )
                        tempTest = False;

                tempNextActualAvailableTime = tempSimulationTime + tempTotalHours - tempHoursLeft;
                if(tempNextActualAvailableTime < CurrentTime )
                        tempTest = False;

                tempValue = Floor(CurrentTime/24) + 1 + wDaysOutToSchedule;
                if( tempDay <  tempValue)
                        tempTest = False;

                tempValue = DBDataGetAsNumber(giSurgeryDBIdx, giSurgeonPreferenceTIdx,
tempDayOfWeek+1, thisPType);
                if(thisPType != thisPossiblePType and tempValue == False)
                        tempTest = False;

                if(tempTest)
                {
                        liPatientTypeIsAlreadyScheduledElseWhere = False;

// Check if patient type is schedule elsewhere at the same time if I am trying to schedule in an alternate block.

                        if(thisPossiblePType != thisPType and wCheckOtherPatientSlots == True)
```

```
                                        liPatientTypeIsAlreadyScheduledElseWhere =

        TestIfPatientTypeIsScheduledAtTheSameTimeSomeWhereElse(thisOR, thisPType, tempSimulationTime,
thisPlanningTime);

                                if(liPatientTypeIsAlreadyScheduledElseWhere == False)
                                {
                                        EarliestDate[thisOR][thisPossiblePType] = tempNextActualAvailableTime;
                                        EarliestRecord[thisOR][thisPossiblePType] = i;
                                        return 1;
                                }
                        }
        }
        return -1;
}


// Once the OR room and Block is chosen, this function will place the appropriate data in the OR Log table. After it
places the data in the table it will return the record number it placed the data in.

integer ScheduleToUseOR(integer thisPatientID, integer thisPType, integer thisOR, integer thisRoomPType, real
thisPlanningTime)
{
        integer i;
        integer liNumOfRecords;
        integer tempPatientType;
        integer tempPatients;
        integer tempPatientLogID;
        integer liORLogTable;
        integer tempBlockType;
        integer tempSurgeonPreference;
        real tempSimulationTime;
        real tempHoursLeft;
        real tempWeek;
        real tempBlock;
        real tempStartingHour;
        real tempTotalHours;

        liORLogTable = DBDataGetAsNumber(giSurgeryDBIdx, giORTablesTIdx, giORLogORTFIdx, thisOR);

        liNumOfRecords = DBRecordsGetNum(giSurgeryDBIdx, liORLogTable);

        for(i = 1; i <= liNumOfRecords; i++)
        {
                tempPatientType = DBDataGetAsNumber(giSurgeryDBIdx, liORLogTable,
cPatientTypeORLXFIdx, i);
                tempHoursLeft = DBDataGetAsNumber(giSurgeryDBIdx, liORLogTable, cHoursLeftORLXFIdx,
i);
                tempStartingHour = DBDataGetAsNumber(giSurgeryDBIdx, liORLogTable,
cStartingHourORLXFIdx, i);

                if(tempPatientType == thisRoomPType)
                {
```

```
                              if(tempHoursLeft >= thisPlanningTime)
                              {
                                        if( tempStartingHour >= (Floor(CurrentTime/24)+1)*24)
                                        {
                                                  tempWeek = DBDataGetAsNumber(giSurgeryDBIdx, liORLogTable,
cWeekORLXFIdx, i);
                                                  tempBlock = DBDataGetAsNumber(giSurgeryDBIdx, liORLogTable,
cBlockORLXFIdx, i);
                                                  tempBlockType = DBDataGetAsNumber(giSurgeryDBIdx,
liORLogTable, cPatientTypeORLXFIdx, i);
                                                  tempStartingHour = DBDataGetAsNumber(giSurgeryDBIdx,
liORLogTable, cStartingHourORLXFIdx, i);
                                                  tempTotalHours = DBDataGetAsNumber(giSurgeryDBIdx,
liORLogTable, cTotalHoursORLXFIdx, i);
                                                  tempPatients = DBDataGetAsNumber(giSurgeryDBIdx,
liORLogTable, cPatientsORLXFIdx, i);

                                                  DBDataSetAsNumber(giSurgeryDBIdx, liORLogTable,
cHoursLeftORLXFIdx, i, tempHoursLeft - thisPlanningTime);
                                                  DBDataSetAsNumber(giSurgeryDBIdx, liORLogTable,
cPatientsORLXFIdx, i, tempPatients+1);

                                                  tempPatientLogID = DBRecordsGetNum(giSurgeryDBIdx,
giPatientLogORTIdx) + 1;
                                                  DBRecordsInsert(giSurgeryDBIdx, giPatientLogORTIdx,
tempPatientLogID, 1);

                                                  DBDataSetAsNumber(giSurgeryDBIdx, giPatientLogORTIdx,
giORLogIDPLORFIdx, tempPatientLogID, i);
                                                  DBDataSetAsNumber(giSurgeryDBIdx, giPatientLogORTIdx,
giWeekPLORFIdx, tempPatientLogID, tempWeek);
                                                  DBDataSetAsNumber(giSurgeryDBIdx, giPatientLogORTIdx,
giBlockPLORFIdx, tempPatientLogID, tempBlock);
                                                  DBDataSetAsNumber(giSurgeryDBIdx, giPatientLogORTIdx,
giBlockTypePLORFIdx, tempPatientLogID, tempBlockType);
                                                  DBDataSetAsNumber(giSurgeryDBIdx, giPatientLogORTIdx,
giORPLORFIdx, tempPatientLogID, thisOR);
                                                  DBDataSetAsNumber(giSurgeryDBIdx, giPatientLogORTIdx,
giTimeOfSurgeryPLORFIdx, tempPatientLogID, tempStartingHour + (tempTotalHours - tempHoursLeft));
                                                  DBDataSetAsNumber(giSurgeryDBIdx, giPatientLogORTIdx,
giPatientTypePLORFIdx, tempPatientLogID, thisPType);
                                                  DBDataSetAsNumber(giSurgeryDBIdx, giPatientLogORTIdx,
giInitialHoursPLORFIdx, tempPatientLogID, tempTotalHours);
                                                  DBDataSetAsNumber(giSurgeryDBIdx, giPatientLogORTIdx,
giTimeLeftPLORFIdx, tempPatientLogID, tempHoursLeft - thisPlanningTime);
                                                  DBDataSetAsNumber(giSurgeryDBIdx, giPatientLogORTIdx,
giTimeOfSchedulingPLORFIdx, tempPatientLogID, CurrentTime);
                                                  DBDataSetAsNumber(giSurgeryDBIdx, giPatientLogORTIdx,
giPatientIDPLORFIdx, tempPatientLogID, thisPatientID);
                                                  DBDataSetAsNumber(giSurgeryDBIdx, giPatientLogORTIdx,
giPlanningTimePLORFIdx, tempPatientLogID, thisPlanningTime);

                                                  return tempPatientLogID;
                                        }
                                }
                        }
                }
                return -1;
}


//Find competetors for a recorded (appointment) under multi-priority patient types

Integer  FindCompetetorForDay(integer thisPatientType,integer thisOR,integer thisRecordId)
{
        integer i;
        integer liNumOfRecords;
        integer tempPatientType;
        integer liORLogTable;
        integer liPatientTypeIsAlreadyScheduledElseWhere;
        integer tempDay;
        integer tempTest;
        integer parentArray[3];
        integer tempDayOfWeek;
        integer tempPTType;
        real        tempHoursLeft;
        real        tempSimulationTime;
        real        tempTotalHours;
        real        tempNextActualAvailableTime;
        real        tempValue;
        real  lrPlanningTime;
        real  thisPlanningTime;

        for(i = 1; i <= giMaxPatientTypes; i++){
        possiblePType[i]=0;
        }

        liORLogTable = DBDataGetAsNumber(giSurgeryDBIdx, giORTablesTIdx, giORLogORTFIdx, thisOR);
        liNumOfRecords = DBRecordsGetNum(giSurgeryDBIdx, liORLogTable);


                tempPatientType = DBDataGetAsNumber(giSurgeryDBIdx, liORLogTable,
cPatientTypeORLXFIdx, thisRecordId);
                tempHoursLeft = DBDataGetAsNumber(giSurgeryDBIdx, liORLogTable, cHoursLeftORLXFIdx,
thisRecordId);
                tempSimulationTime = DBDataGetAsNumber(giSurgeryDBIdx, liORLogTable,
cStartingHourORLXFIdx, thisRecordId);
                tempTotalHours = DBDataGetAsNumber(giSurgeryDBIdx, liORLogTable,
cTotalHoursORLXFIdx, thisRecordId);
                tempDay = DBDataGetAsNumber(giSurgeryDBIdx, liORLogTable, cDayORLXFIdx,
thisRecordId);
                DBDataGetParent(giSurgeryDBIdx, liORLogTable, cDayOfWeekORLXFIdx, thisRecordId,
parentArray);
                tempDayOfWeek = parentArray[2];
```

```
			for(tempPTType = 1; tempPTType <= giMaxPatientTypes; tempPTType++)
			{

			tempTest=True;
			if(thisPatientType!=tempPTType){
				lrPlanningTime = DBDataGetAsNumber(giSurgeryDBIdx, giSurgeryTimeTIdx,
giPlanningtimeFIdx, tempPTType);

				lrPlanningTime = Ceil(lrPlanningTime);

				if(tempHoursLeft < lrPlanningTime )
				  tempTest = False;

				tempValue = DBDataGetAsNumber(giSurgeryDBIdx, giSurgeonPreferenceTIdx,
tempDayOfWeek+1, tempPTType);
				if(tempValue==False)
				 tempTest=False;

				if(tempTest)
					{
						possiblePType[tempPTType]=1;
					}

			}
			}
			return -1;
}



//	This function will set the earliest date for the specified patient.  This would be for a fixed patient type.

Integer	SetEarliestDateForSpecificBlock(integer thisPatientType, real thisPlanningTime, integer thisPatientID)
{
	integer liMinOR;
	integer liMinRecord;
	integer liRoomPtType;
	integer i;
	integer liLogID;
	integer thisRecord;
	integer tempRecord;
	integer tempOR;
	real		tempDate;
	real	lrMinDate;

	liMinOR = -1;
	liMinRecord = -1;
	lrMinDate = 99999;
	for(i = 1; i <= giMaxORRooms; i ++)
	{
		tempDate = EarliestDate[i][thisPatientType];
		tempRecord = EarliestRecord[i][thisPatientType];
```

```
				tempOR = i;
				if(tempDate < lrMinDate)
				{
					lrMinDate = tempDate;
					liMinRecord = tempRecord;
					liMinOR = tempOR;
					liRoomPtType = thisPatientType;
				}

				tempDate = EarliestDate[i][cPatientType0];
				tempRecord = EarliestRecord[i][cPatientType0];
				tempOR = i;
				if(tempDate < lrMinDate and lrMinDate -tempDate > 120)
				{
					lrMinDate = tempDate;
					liMinRecord = tempRecord;
					liMinOR = tempOR;
					liRoomPtType = 0;
				}
	}

	if(liMinOR >= 0)
	{
				thisRecord = EarliestRecord[liMinOR][thisPatientType];
				liLogID = ScheduleToUseORGivenRecordNumber(thisPatientID, thisPatientType, liMinOR,
liRoomPtType, thisPlanningTime, thisRecord );
	}
	else
				liLogID = -1;

	return liLogID;
}

//	This function will search for the earliest date but only looking at the blocks for the specified patient type
and place the data in that date.  This function will return the record number in the log table it placed the data in.

Integer   TestForEarliestDateAndIncludeAllTypesUnderConditions2(integer thisPatientType, real thisPlanningTime,
integer thisPatientID)

{
	integer i, j;
	integer   liLogID;
	integer liORID;
	integer tempType;
	integer liMinRecord;
	integer liMinOR;
	integer tempRecord;
	integer tempOR;
	integer liRoomPtType;
	real tempDate;
	real lrMinDate;
	real tempTestingPatientTypeMinTime;
	real tempOriginalPatientTypeMaxTime;
	real	lrTimeWaiting;
	integer tempPTType;
	real PTypeCost[13][2];

	for(i = 0; i <= giMaxPatientTypes+1; i++){
		PTypeCost[i][0]=-1;
		PTypeCost[i][1]=-1;
	}

	liMinOR = -1;
	liMinRecord = -1;
	lrMinDate = 99999;
	for(i = 1; i <= giMaxORRooms; i ++)
	{
		tempDate = EarliestDate[i][thisPatientType];
		tempRecord = EarliestRecord[i][thisPatientType];
		tempOR = i;
		if(tempDate < lrMinDate)
		{
			lrMinDate = tempDate;
			liMinRecord = tempRecord;
			liMinOR = tempOR;
			liRoomPtType = thisPatientType;
		}

		tempDate = EarliestDate[i][cPatientType0];
		tempRecord = EarliestRecord[i][cPatientType0];
		tempOR = i;
		if(tempDate < lrMinDate and lrMinDate -tempDate > 120)
		{
			lrMinDate = tempDate;
			liMinRecord = tempRecord;
			liMinOR = tempOR;
			liRoomPtType = 0;
		}
	}

	tempOriginalPatientTypeMaxTime = DBDataGetAsNumber(giSurgeryDBIdx, giReleaseTimeTIdx,
giMaxTimeRTFIdx, thisPatientType);

	// within X days of call
	if(lrMinDate - CurrentTime < tempOriginalPatientTypeMaxTime and liMinRecord > 0 and liMinRecord <
99999)
	{
		liLogID = ScheduleToUseORGivenRecordNumber(thisPatientID, thisPatientType, liMinOR,
liRoomPtType, thisPlanningTime, liMinRecord );
		return liLogID;
	}

// These are outside X days of call
// These are patients who are eligible to assign to open/release hours

	tempType = 0;
```

```
            liMinOR = -1;
            liMinRecord = -1;
            lrMinDate = 99999;
            for(i = 1; i <= giMaxPatientTypes; i++) // Patient Types
            {
                        if (i != thisPatientType )
                        {
                                    for(j = 1; j <= giMaxORRooms; j++) // OR Rooms
                                    {
                                                tempDate = EarliestDate[j][i] - CurrentTime;
                                                tempRecord = EarliestRecord[j][i];
                                                tempOR = j;
                                                tempTestingPatientTypeMinTime = DBDataGetAsNumber(giSurgeryDBIdx,
giReleaseTimeTIdx, giMinTimeRTFIdx, i);
                                                if(tempDate < lrMinDate and tempDate < tempTestingPatientTypeMinTime)
                                                {
                                                            tempType = i;
                                                            lrMinDate = tempDate;
                                                            liMinRecord = tempRecord;
                                                            liMinOR = tempOR;
                                                }
                                    }
                        }
            }

// If I found an available block and it is withing the Alternate block X day call window,
// first, check if matches with surgeons preference
// second, find all patienttype whom surgeons can operate on this weekday
// calculate waiting cost of selected patient types and sort them

            real lrTimeOfScheduling;
            real lrTimeOfSurgery;
            real lrWaitingCost;
            integer flag1;
            integer temp[2];
    integer numLength;
    real lrPatientRevenue;
    if(     liMinOR > 0)
            {
                        FindCompetetorForDay(thisPatientType,liMinOR,liMinRecord);


     //liPatientType = DBDataGetAsNumber(giSurgeryDBIdx, giPatientLogORTIdx, giPatientTypePLORFIdx, i);
                        lrTimeOfScheduling = DBDataGetAsNumber(giSurgeryDBIdx, giPatientLogORTIdx,
giTimeOfSchedulingPLORFIdx, liMinRecord);
                        lrTimeOfSurgery = DBDataGetAsNumber(giSurgeryDBIdx, giPatientLogORTIdx,
giTimeOfSurgeryPLORFIdx, liMinRecord);
                        //liDaysOutToSchedule = DBDataGetAsNumber(giSurgeryDBIdx, giPatientLogORTIdx,
giDaysOutToSchedulePLORFIdx, i);
                        //liCanceled = DBDataGetAsNumber(giSurgeryDBIdx, giPatientLogORTIdx,
giCanceledPLORFIdx, i);

                        lrTimeWaiting = (lrTimeOfSurgery - lrTimeOfScheduling)/1.5;
```

```
                    for(tempPTType = 1; tempPTType <= giMaxPatientTypes; tempPTType++)
                    {

                    if(possiblePType[tempPTType]==1){


// Calculate waiting cost for multi-priority patient

if(tempPTType == 1)
          {lrWaitingCost = exp(-2.665+(0.061*lrTimeWaiting))/(exp(-2.665+(0.061*lrTimeWaiting))+1);

                              }
if(tempPTType == 2 or tempPTType == 5 or tempPTType == 11)
          {lrWaitingCost = exp(-1.751+(0.0179*lrTimeWaiting))/(exp(-1.751+(0.0179*lrTimeWaiting))+1);
                              }
if(tempPTType == 3 or tempPTType == 6 or tempPTType == 7 or tempPTType == 8 or tempPTType == 10)
          {lrWaitingCost = exp(-2.45+(0.0248*lrTimeWaiting))/(exp(-2.45+(0.0248*lrTimeWaiting))+1);

                              }
if(tempPTType == 4 or tempPTType == 9 )
          {lrWaitingCost = exp(-3.958+(0.0672*lrTimeWaiting))/(exp(-3.958+(0.0672*lrTimeWaiting))+1);

                              }



                              lrPatientRevenue = DBDataGetAsNumber(giSurgeryDBIdx, giRevenueTIdx,
giPatientRevenueFIdx, tempPTType);
                              PTypeCost[tempPTType][1] =  (lrWaitingCost*lrPatientRevenue) ;
                              PTypeCost[tempPTType][0]=tempPTType;
                    }

                              }
                    }



          flag1=1;
      numLength = 12;
      for(i = 1; (i <= numLength && flag1==1); i++)
     {
         flag1 = 0;
         for (j=1; j < (numLength -1); j++)
        {
            if (PTypeCost[j+1][1] > PTypeCost[j][1]) // ascending order simply changes to <
           {
               temp[0] = PTypeCost[j][0];
               temp[1] = PTypeCost[j][1];           // swap elements
               PTypeCost[j][0] = PTypeCost[j+1][0];
                 PTypeCost[j][1] = PTypeCost[j+1][1];
               PTypeCost[j+1][0] = temp[0];
               PTypeCost[j+1][1] = temp[1];
               flag1 = 1;            // indicates that a swap occurred.
            }
         }
     }

      if(thisPatientType==PTypeCost[1][0])
            {
          liLogID = ScheduleToUseORGivenRecordNumber(thisPatientID, thisPatientType, liMinOR, tempType,
thisPlanningTime, liMinRecord );
          return liLogID;
          }

     real lrPlanningTime;
     real totalTime;
     totalTime=0;

     integer index;
     index=0;

     for(i=1;i<12;i++){
     if(PTypeCost[i][0]==thisPatientType){
       index=i;
       break;
     }
     }

     for(i=1;i<=index;i++){
                    lrPlanningTime = DBDataGetAsNumber(giSurgeryDBIdx, giSurgeryTimeTIdx,
giPlanningtimeFIdx, PTypeCost[i][0]);
                 totalTime=totalTime+lrPlanningTime;
       //if(PTypeCost[i][1]==thisPatientType){
         //   break;
       //}
          }

integer liORLogTable;
real tempHoursLeft;
          liORLogTable = DBDataGetAsNumber(giSurgeryDBIdx, giORTablesTIdx, giORLogORTFIdx, liMinOR);
          //tempPatientType = DBDataGetAsNumber(giSurgeryDBIdx, liORLogTable, cPatientTypeORLXFIdx,
liMinRecord);
          tempHoursLeft = DBDataGetAsNumber(giSurgeryDBIdx, liORLogTable, cHoursLeftORLXFIdx,
liMinRecord);


          if(tempHoursLeft>totalTime){
                    liLogID = ScheduleToUseORGivenRecordNumber(thisPatientID, thisPatientType, liMinOR,
tempType, thisPlanningTime, liMinRecord );
                    return liLogID;
          }



 real mycost;
```

```
  for(i=1;i<=numLength;i++){
   if(PTypeCost[i][0]==thisPatientType){
   mycost=PTypeCost[i][1];
   break;
   }
  }

        real totalDelay;

        for(i=1;i<=numLength;i++){
   if(PTypeCost[i][0]==thisPatientType){
   mycost=PTypeCost[i][1];
   break;
   }
  }


// If No other room fit (available), First search in OPEN room and then schedule the patient in thier proper patient
block
        liMinOR = -1;
        liMinRecord = -1;
        lrMinDate = 99999;
        for(i = 1; i <= giMaxORRooms; i ++)
        {
                tempDate = EarliestDate[i][thisPatientType];
                tempRecord = EarliestRecord[i][thisPatientType];
                tempOR = i;
                if(tempDate < lrMinDate)
                {
                        lrMinDate = tempDate;
                        liMinRecord = tempRecord;
                        liMinOR = tempOR;
                        liRoomPtType = thisPatientType;
                }

                tempDate = EarliestDate[i][cPatientType0];
                tempRecord = EarliestRecord[i][cPatientType0];
                tempOR = i;
                if(tempDate < lrMinDate and lrMinDate -tempDate > 120)
                {
                        lrMinDate = tempDate;
                        liMinRecord = tempRecord;
                        liMinOR = tempOR;
                        liRoomPtType = 0;
                }
        }

        if( liMinRecord > 0 and liMinRecord < 99999 )
                liLogID = ScheduleToUseORGivenRecordNumber(thisPatientID, thisPatientType, liMinOR,
liRoomPtType, thisPlanningTime, liMinRecord);
        else
                liLogID = -1;
        return liLogID;
```

```
}

real FindArrivalRate(integer paitientType)
{
	//integer i;
	  real rate;
	//integer liMaxrows;


	//liMaxrows = DBRecordsGetNum(giSurgeryDBIdx, giSecondaryArrivalRateTIdx);
			rate = DBDataGetAsNumber(giSurgeryDBIdx, giSecondaryArrivalRateTIdx, 
giSecArrivalRateFIdx, paitientType);

	return rate;
}

procedure InitializeEarliestDateArray()
{
	integer i, j, k;

	for(i = 0; i <= giMaxORRooms; i++) // OR Rooms
	{
		for(j = 0; j <= giMaxPatientTypes; j++) // Patient Types
		{
				EarliestDate[i][j] = 99999;
				EarliestRecord[i][j] = 99999;
		}
	}
}


//	This message handler will first capture the data for all patient types and all ORs.  Then it will call different 
functions to test different logic for choosing the right block.
on PTypeIn
{
	real tempScheduleTime;
	real tempSurgeryTime;
	real lrPlanningTime;

	integer tempDaySurgery;
	integer tempDayScheduled;
	integer tempORRoom;
	integer tempPTType;

	InitializeEarliestDateArray();

	lrPlanningTime = DBDataGetAsNumber(giSurgeryDBIdx, giSurgeryTimeTIdx, giPlanningtimeFIdx, 
PTypeIn);

	lrPlanningTime = Ceil(lrPlanningTime);

	for(tempORRoom = 1; tempORRoom <= giMaxORRooms; tempORRoom++)
	{
			if( wCheckOtherPatientSlots == True  )
				TestForNextAvailabileSpotForType(tempORRoom, cPatientType0, lrPlanningTime,
PTypeIn);

			for(tempPTType = 1; tempPTType <= giMaxPatientTypes; tempPTType++)
			{
				if( wCheckOtherPatientSlots == True or PTypeIn == tempPTType )
					TestForNextAvailabileSpotForType(tempORRoom, tempPTType,
lrPlanningTime, PTypeIn);
			}
		}

		if(wEarlyDateDedicatedSpotRBtn == True)
			LogIDOut = SetEarliestDateForSpecificBlock(PTypeIn, lrPlanningTime, PatientIDIn);
		else
			LogIDOut = TestForEarliestDateAndIncludeAllTypesUnderConditions2(PTypeIn,
lrPlanningTime, PatientIDIn);

		giTotalPatients ++;

		if(LogIDOut > 0)
		{
			wPatientsScheduled++;

			tempSurgeryTime = DBDataGetAsNumber(giSurgeryDBIdx, giPatientLogORTIdx,
giTimeOfSurgeryPLORFIdx, LogIDOut);
			tempScheduleTime = DBDataGetAsNumber(giSurgeryDBIdx, giPatientLogORTIdx,
giTimeOfSchedulingPLORFIdx, LogIDOut);

			tempDaySurgery = Floor(tempSurgeryTime/24);
			tempDayScheduled = Floor(tempScheduleTime/24);
			LagDaysOut = tempDaySurgery - tempDayScheduled - 1;
			grTotalLagDays += LagDaysOut;
			wAvgLagDays = grTotalLagDays / wPatientsScheduled;
			SendMsgToInputs(LagDaysOut);
		}

		wPatientsNotScheduled = giTotalPatients - wPatientsScheduled;
}
//		Initialize any simulation variables.
on initsim
{
		integer liNumOfRecords;
		integer liNumOfWeeks;
		integer i;
		wPatientsScheduled = 0;
		wAvgLagDays = 0;
		wPatientsNotScheduled = 0;
		grTotalLagDays = 0;
		giTotalPatients = 0;
		GetDBVariables();
		InitializeEarliestDateArray();
}
```

Vita

Mina Loghavi was born and raised in Iran. She received her Bachelor of Science degree in Industrial Management from Shiraz University, Iran in 2003. She continued her study in the same school and received Master degree of Industrial Management in 2005. She later attended University of Dayton, Ohio where she graduated with Master of Business Administration (MBA) in 2008. In January 2009, she entered the PhD program in the department of Statistics, Operations, and Management Science in the University of Tennessee, Knoxville where she graduated with a PhD in Management science, in December of 2015.