4-22-2013

Analysis and optimization of patient bed assignments within a hospital unit while considering isolation requirements

Christina Cignarale

Rochester Institute of Technology

Analysis and Optimization of Patient Bed Assignments within a Hospital Unit while Considering Isolation Requirements

A Thesis

Submitted in partial fulfillment of the

requirements for the degree of

Master of Science in Industrial Engineering

in the

Department of Industrial & Systems Engineering

Kate Gleason College of Engineering

by

Christina Cignarale

April 22, 2013

DEPARTMENT OF INDUSTRIAL AND SYSTEMS ENGINEERING

KATE GLEASON COLLEGE OF ENGINEERING

ROCHESTER INSTITUTE OF TECHNOLOGY

ROCHESTER, NEW YORK

CERTIFICATE OF APPROVAL

M.S. DEGREE THESIS

The M.S. Degree Thesis of Christina Cignarale

has been examined and approved by the

thesis committee as satisfactory for the

thesis requirement for the

Master of Science degree

Approved by:

Dr. Rubén Proaño, Thesis Advisor

Dr. Michael E. Kuhl, Committee Member

ABSTRACT

Healthcare Associated Infections (HAIs) are infections acquired by patients admitted in healthcare facilities. These infections can be contagious or even worse, fatal, which has prompted the medical community to put guidelines in place to isolate patients who may pose a risk of spreading or be susceptible to an infection(s). These isolation practices also have the unwanted consequence of constraining the options for inpatient bed assignments since infected patients cannot be assigned to a room where uninfected patients reside.

Historically, hospitals in the United States have been built with single and double bedrooms to provide inpatient care. As long as demand for a bedroom is less than the number of bedrooms in the hospital, bed assignment is a trivial process. Difficulties for determining bed assignments occur when hospital units operate at full or nearly full bed utilization and must continue to admit new patients. When this occurs, the units' administrators must determine when to admit new patients, whether current patients need to be discharged to make room for new more critical patients, or if there is a need to exchange the rooms of already admitted patients (i.e., internal movement). These decisions are complicated by the limited bedroom capacity (number of rooms and occupancy threshold in rooms) and by the need to implement isolation guidelines necessary to prevent and contain the occurrence of HAIs.

This study presents two optimization models to suggest how to accommodate admitted and incoming patients in a hospital unit to satisfy all isolation requirements, while simultaneously maximizing the total criticality of patients admitted into the unit and minimizing the number of internal movements. These models provide bed assignment recommendations based on available bed demand and patient characteristic information for a current planning period first for current (known) demand, and then considering uncertain future demand.

Additionally, we explore the use of the first model to suggest a methodology for determining the number of single and double bedrooms in a hospital unit as well as the stockpile of spare resources necessary to ensure a desired service level for inpatient hospitalization demand, when isolation requirements are considered. Finally, the second model which considers future demand is explored to examine the effect on bedroom assignments incurred from the number of periods used in the planning horizon given a small subset of hypothetical, yet realistic hospital data.

TABLE OF CONTENTS

1. INTRODUCTION

In recent years, Healthcare Associated Infections (HAIs) rates have become a significant concern in the health care community [1,2]. HAIs are infections that patients acquire while receiving medical care at a healthcare facility most commonly after surgery or during prolonged stays for treatment [3]. In 2002, it was estimated that in United States hospitals 1.7 million patients were infected by HAIs, and 6% of those patients died from complications of the infection [4]. To prevent HAIs the Centers for Disease Control and Prevention (CDC) as well as other health care organizations such as the World Health Organization, the Infectious Disease Society of America, the Society of Healthcare Epidemiology, the Association for Professionals in Infection Control, the European Centre for Disease Prevention and Control, and the Department of Health in the United Kingdom have created isolation guidelines to help hospitals prevent HAIs [5–11].The cost of treating and preventing these HAIs in the United States exceeded $USD 30 billion in 2007[3].

Clinical studies have proven that properly following these HAI guidelines reduces their occurrence [3]. However, despite this improvement in care, the implementation of isolation precautions still presents challenges that affect hospitals' ability to deal with many operational aspects including patient bed assignments. For example, Shenoy et al. indicates that for patients with methicillin-resistant staphylococcus aureus (MRSA) and vancomycin-resistant enterococcus (VRE) infections the time to receiving a bed assignment was longer than that of a patients that did not suffer from these complications [12]. The study attributed the isolation precautions as the sole reason for this delay in bed assignments.

Currently, hospitals make bed assignments primarily relying on the judgment of experienced nurses. A typical nursing process to assign patients to beds is described in detail in Figure 1 [13]. Nurses need to use this process several times a day. Determining the best patient-bed assignments and patient movements among the thousands of potential alternatives becomes difficult for this multi-criteria decision problem. The problem is especially critical in units with high utilization rates. The questions that arise from this process are as follows: Should an incoming patient be admitted? Should a less critical admitted inpatient be released to have more available beds? Should a reassignment of patients within the unit (i.e., internal movement) be performed to facilitate accommodating more patients?

These decisions must be made considering each patient's diagnosis, characteristics (e.g., gender, age, mental status, medical needs), and the isolation requirements necessary to prevent the patient from acquiring HAIs or from infecting other patients in the unit. Therefore, in order to admit a new inpatient to a unit while considering all the aforementioned restrictions, it is often necessary to perform multiple internal movements, in which a pair of patients exchanges their bed assignments. However, these internal movements are expensive to implement since each movement requires two room disinfection procedures and hospital staff must devote time to accommodate the moved patients into their new beds. Internal movements consume valuable nursing time that could be otherwise used to care for patients. Furthermore, internal movements can create a higher risk for infection since the patients are exposed to more people.

A technique to approach this problem is to use a mathematical program to model the bed assignment decision-making process using the objective function to describe the most important goals. For example, assigning the largest number of patients into a unit while mitigating the number of internal movements may be a desired goal for this problem. To ensure all assignments are valid the constraints would consider patient characteristics such as gender and isolation requirements. In this thesis we formulate a methodology to model this decision-making process.

Figure 1: Nurse Bed Assignment Process [13]

Assigning Patient Rooms Using the Steps of the Nursing Process
Room Assignment Process
1. Gather available data using medical diagnosis and pertinent history
2. Determine whether the patient needs require placement close to nurse's station
3. Determine need for isolation or special precautions
4. Identify rooms available
5. Gather information about current patients and circumstances in rooms available
Analysis
1. Analyze the disease processes of the newly admitted patient
2. Evaluate physical layout of the room being considered for the new patient
3. Compare the needs of the new and current patients to room availability
Outcome Indentifications
1. Indentify who the roommate of the new patient will be.
2. Evaluate the expected effects for the new and current patient
Plan
1. Determine what room placemen would be the best for the patient based on his/her functional status
Implementation
1. Place patient in assigned room
Evaluation
1. Continue to monitor, evaluate and reevaluate patient's status based on changes in laboratory data, further diagnostic test findings, and changes in patient conditions
2. New room reassignment may need to be made. For example, diagnostic test indicate the need for isolation or occurrence of confusion indicates the need to move patient close to nurse's station for more careful observation

The research presented in this thesis was developed in collaboration with the Pulmonary and Critical Care Unit (PCCU) at Rochester General Hospital (RGH) in Rochester, NY. The PCCU provided historical data and clinical expertise to support this study. In the remainder of this document we will refer to the PCCU simply as “the unit”. Patients in the PCCU are very ill, need critical respiratory care, require lengthy hospitalization stays, and may be subject to one or multiple isolation requirements. These considerations combined with the number and type of bedrooms available in the unit produce a complex environment for bed assignments. The PCCU is comprised of 10 single-patient and 8 double-patient rooms giving the unit 26 beds in total. Moreover, the unit commonly operates at or near full capacity since typical patients stay for extended periods of time.

2. PROBLEM STATEMENT

It is evident that bed assignments require more than the availability of a bed. Multi-patient rooms, infectious agents, and critical medical needs add complexity to patient care, more specifically to patient bed assignments. The focus of this study is to address the operational and strategic bed management challenges considering isolation requirements by the application of optimization based solutions. To support bed assignment decision-making, two optimization models were developed to maximize the criticality of patients in a hospital unit while minimizing the number of internal movements that occur within the unit; the first model, considers patient information available for a single planning period, and the second model considers likely patient information over multiple planning periods. In this work, criticality is a patient characteristic assigned to all patients to describe their relative medical conditions. In the first model critically is assigned as a numeric value from 0-10 where the value of 10 is considered to be the most critical value. A slight variation on criticality values is made for the second model and is discussed in section 4.4.1. Additionally, the model minimizes the number of internal movements by a penalty expression in the objective function.

Moreover, we integrate the single-period optimization model into a Monte Carlo experiment to strategically determine the optimal number of single and double bedrooms in a particular hospital setting that results in the best tradeoff between occupancy levels and the number of internal movements performed. This analysis was also used to suggest target inventory levels of key resources to accommodate a random inpatient demand for given service levels in the PCCU.

Furthermore the second optimization model proposed in this thesis aims to examine the implications of considering the likelihood of seeking admission by potential inpatients (i.e. patients from the ICU) on current bed assignment decisions. Experimentation with this multi-period model aims to illustrate how to determine the optimal number of periods into the future a hospital unit should observe to determine the best bed assignment for the current planning period.

One benefit of developing these methodologies for the bed decision-making process would be an application hospital assignment staff could use to receive valid bed assignment suggestions with minimal effort and in a relatively small amount of time. We have included an application of this suggestion in section 4.1.1. Furthermore, the experimentation done using the models presented in this study would be advantageous in hospital unit planning. The models provide a general approach for assignments which can be tailored to make decisions based on any target population of a specific unit by altering the data set.

3. LITERATURE REVIEW

This section reviews previous literature on inpatient bed-placement, bed capacity planning, and hospital placement policies that dictate what type of bedrooms each unit should have and the criteria for inpatient admission.

The use of mathematical models for scheduling medical services is not new. Optimization methods have been used in hospitals to schedule surgeries, lab tests, scans, dialysis, and staffing purposes[14–21]. However, despite the benefits of these solutions, most applications have not been implemented across hospitals, due to costs, and the fact that most solutions are highly customized for specific hospital needs.

3.1 Operations Research in Hospital Inpatient Bed Assignments

Demeester et al.[22], and Ceschia and Schaerf [23] propose optimization models that can be used to determine how to assign inpatients to beds across a hospital comprised of six departments and under hypothetical information considering patient gender, diagnosis, room type, age, and patient preference. However, these studies fail to consider the effect of isolation requirements in patient placement which is what this work includes.

Demeester et al.[22] proposes a search method to allocate inpatients to beds, while trying to increase service levels and reduce the number of bed transfers between different hospital rooms, offering gender-specific rooms, and satisfying other specific healthcare needs. In this study, the authors consider a search space of all available beds across different units of a hospital. The service level in this model is measured by how often patients are assigned to appropriate departments based on medical needs and patient preferences while keeping department admissions balanced (i.e. all departments within the hospital have roughly the same number of patients within each departmental unit). Demeester's model assigns as many patients as possible to the rooms of their preference while maintaining hospital constraints such as gender separation, age separation, medical condition separation, among others [22].

Ceshcia and Schaerf [23] extend Demeester et al. [22] work and propose a multi-neighborhood local search method to improve the solution time of assigning patients to subsets of beds. This model applies two pre-processing steps to reduce computational efforts; the first step restricts the search area to a set of rooms instead to a set of beds. The second step uses a penalty matrix to assess the value of assigning a patient to a bed within a group of beds, while considering the suitability of its room to the patient's needs and preferences.

3.2 Simulation Studies for Hospital Bed Management

Simulation models have also been commonly used to deal with patient placement problems [24–28]. For example, Harper and Shahani [24] use discrete event simulation across multiple hospital units to analyze several patient admissions and bed allocation policies used to determine the optimal number of beds in a unit. Similarly, Dumas[25], Vassilacopoulos [26], Khare [27], and Williams [28] propose simulation studies that focus on determining the optimal number of beds per unit while maintaining effective hospital systems. Dumas [25] focuses on assigning patients to the proper units based on their care needs and suggests policies for re-assigning patients when they are "misplaced." Such "misplacement" occurs when patients are not assigned to the most suitable units of care due to lack of capacity. Vassilacopoulos [26] analyzes the appropriate number of beds to have in different hospital units while abiding by a priority policy in which all hospital's emergency patients are assigned to inpatient rooms immediately under limited bed capacity. Through Monte Carlo simulation experiments, Khare [27] and Williams [28] ascertain that increasing the number of beds in the emergency department can reduce the overall length of stay of inpatients.

3.3 Hospital Bed Admission Policies

Hospital bed management also involves determining whom to hospitalize into a unit when patient bed demand exceeds availability. Shmueli [29] proposes a stochastic model that maximizes the overall survival rate in an intensive care unit by recommending a first-come-first-serve admissions policy for patients who exhibit survival benefits that exceed a minimum threshold level. The threshold level results from factoring in the decline in admissions into the ICU and the improved probability of survival of each admitted patient. The survival probability is calculated from mapping the expected improvement of the survival rate that results from admitting a patient into the unit compared to the rate if the patient has not been admitted. The patient's survival score that is used in comparison to the threshold level results from considering the patient's age, gender, general diagnosis, and the number of patients already in the unit.

3.4 Hospital Bed Allocation

Bed assignment is also affected by the way hospitals determine the number and type of rooms available in their units [26–28]. However, these decisions are typically taken using approaches that are tailoring to particular hospital realities. Two of the most common means to determine the number of beds in a unit are the target bed occupancy rate and the ratio-based methodology [30]. However, neither of these approaches takes into consideration that bed demand fluctuates over time. Nguyen et al. [30] propose an alternative model to determine the number of beds needed for surgical and internal medicine departments based on the number of internal patient transfers, the number of days with no possibility for unscheduled admissions, and the number of days having pre-established blocks of unused beds. The model's goal is to minimize the number of unoccupied beds. Similarly, Gong et al.[31] proposes a multi-objective comprehensive learning particle swarm optimization model with a binary search-based representation to maximize admission rates and bed utilization simultaneously.

3.5 Hospital Room Occupancy Studies

There are opposing opinions regarding whether single-patient rooms are more cost effective than multi-occupancy rooms. In 2011, Boardman and Forbes [32] claimed that the net social benefit of a bed in a single-patient room in the USA was $USD 70,000 higher than that for a bed in a multi-patient room. Chaudhury et al.[33] concludes that the net social impact for single-patient rooms is lower than those for multi-occupancy rooms. Factors such as reduced transfers, higher occupancy rates, and fewer medication errors associated with having single patient rooms contribute to cost reductions. Therefore, it is inconclusive at this time to claim which rooms are more cost effective. Further studies need to be performed.

3.6 The Need and Potential Impact of Additional Bed Assignment Optimization Methods

As shown in the literature review, there are several areas of study regarding inpatient bed scheduling. First, there is the process of assigning a patient to an appropriate bed. Furthermore, hospitals must determine how many beds should be assigned to each specialty unit or floor. In addition to these considerations a hospital must determine what admission criteria will be used for qualifying patients to different units within the hospital. Lastly, hospitals must determine the type of rooms that will be included in each unit. Operation research methods, simulation, and surveys have been utilized as methods to solve these problems [11,14,22–33]. Here we also present a methodology that includes operation research methods and simulation. Yet, this work expands upon the studies presented here by including isolation requirement considerations.

This inclusion is paramount since it is law for health care facilities to follow these isolation precautions. Creating a model for assigning patients to beds without this consideration could result in inappropriate bed assignments. Not considering these requirements may also result in oversimplified models that are not reflective of actual hospital scenarios leading to inappropriate conclusions for hospital planning. Therefore, it is in the interest of the medical community to include this feature into patient bed scheduling methodology to create the most applicable suggestions.

The following section describes the methodology proposed in this thesis, which aims to bridge the voids existing in current literature that does not consider isolation requirements in patient bed scheduling, admission policies, bed allocation, and occupancy planning problems.

4. PATIENT SCHEDULING METHODOLOGY

The methodologies presented here are general formulations that are later tested using historical data from a partnership with a local hospital. Detailed information regarding data creation and simulation methods will be included in the appendix section for reference.

4.1 Single Period Methodology

This section describes a single-period deterministic model that suggests how to accommodate groups of admitted and incoming patients within a hospital unit (PCCU). The model captures the decision process a unit administrator would use to determine bed assignments by considering the relative criticality and isolation requirements of each patient. The model aims to maximize the critical level of the unit and minimize the number of internal movements. Additionally, the resulting allocation plan must avoid moving patients with user-specified restrictions (for example, due to the use of specialized equipment, morbidly obese patients, or patients who have recently been moved.) Lastly, the model must respect gender requirements and isolation requirements for double patient rooms. Different genders and isolation requirements cannot be assigned to the same multi-occupancy room.

The model requires several assumptions. First it is assumed that the configuration of the unit (i.e., number of double and single rooms) is known and will not change during the current decision period. It is also assumed that the isolation requirements and criticality of each patient are known and will not change during the current period. Furthermore, if a patient has ended his or her length of stay in the unit he or she will immediately be discharged during the current period. Of course, discharges may not occur instantaneously in a real hospital setting due to other factors such as difficulties arranging transportation for the patient to go home.

To perform all these objectives the model was developed as a binary integer programming model. This type of model was appropriate due to its application in scheduling problems. Furthermore, the instances of this problem contain a small number of variables due to the size of the unit therefore choosing a nonlinear approach does not propagate extensive computational time.

The formulation of the model is described below:

P_a:	set of patients currently occupying beds within the unit
P_n:	set of patients seeking admission
P:	set of all patients (i.e., $P = P_a \cup P_n$)
R:	set of all available rooms
T:	triage room, $T \in R$
D:	discharge room, $D \in R$
B_j:	number of beds available in room $j \in R$
R_1:	set of single-patient rooms.
G_i:	gender of patient $i \in P$
I_i:	isolation type of patient $i \in P$
c_i:	critical state of patient $i \in P$, relative to other patients in *P* (this could be assessed as a number e.g. 1 less critical, 10 more critical).

$$y_{ij}: \begin{cases} 1 & \text{if a patient } i \in P \text{ was assigned to a bed in room } j \in R \text{ the day before} \\ 0 & \text{o.w.} \end{cases}$$

$$f_{ij}: \begin{cases} 1 & \text{if a patient } i \in P \text{ was assigned to a bed in room } j \in R \text{ and should not be moved} \\ 0 & \text{o.w.} \end{cases}$$

The problem decision variables are given by:

$$x_{ij}: \begin{cases} 1 \text{ if a patient } i \in P \text{ is assigned to a bed in room } j \in R \\ 0 \quad \text{o.w.} \end{cases}$$

$$\delta_{gj}: \begin{cases} 1 \text{ if a patient of gender } g \in G \text{ is assigned to a bed in room } j \in R\backslash R_1 \\ 0 \quad \text{o.w.} \end{cases}$$

$$\gamma_{ij}: \begin{cases} 1 \text{ if a patient with isolation needs } i \in I \text{ is assigned to a bed in room } j \in R\backslash R_1 \\ 0 \quad \text{o.w.} \end{cases}$$

Therefore, the resulting optimization problem corresponds to the following binary integer programming problem:

$$\text{Maximize} \qquad \sum_{i\in P}\sum_{j\in R: j\neq T, j\neq D}\left(c_i x_{ij} - \frac{1}{c_i}\left|x_{ij} - y_{ij}\right|\right) \qquad (0)$$

s.t.

$$\sum_{j\in R} x_{ij} = 1 \qquad i \in P \qquad (1)$$

$$x_{ij} \leq \delta_{g_i j} \qquad i \in P, \quad j \in R\backslash(R_1 \cup T \cup D) \qquad (2)$$

$$x_{ij} \leq \gamma_{I_i j} \qquad i \in P, \quad j \in R\backslash(R_1 \cup T \cup D) \qquad (3)$$

$$\sum_{g\in G} \delta_{gj} \leq 1 \qquad j \in R\backslash(R_1 \cup T \cup D) \qquad (4)$$

$$\sum_{i\in I} \gamma_{ij} \leq 1 \qquad j \in R\backslash(R_1 \cup T \cup D) \qquad (5)$$

$$\sum_{i\in P} x_{ij} \leq B_j \qquad j \in R \qquad (6)$$

$$x_{iT} = 0 \qquad i \in P_a \qquad (7)$$

$$x_{iD} = 0 \qquad i \in P_n \qquad (8)$$

$$\sum_{j_2\in R_1: j_2\neq j_1} x_{ij_2} = 0 \qquad i \in P,\ y_{ij_1} = 1 \text{ and } j_1 \in R_1 \qquad (9)$$

$$x_{ij} \geq f_{ij} y_{ij} \qquad i \in P, j \in R\backslash(T \cup D) \qquad (10)$$

The proposed model assumes that on any given day, the hospital unit must manage a set of patients, P including those patients that already have an assigned bed (i.e., patients currently admitted to the unit),P_a, and the group of incoming patients seeking admission into the unit, P_n. The hospital unit has R rooms available to accommodate patients. Two of the rooms, T and D correspond to the triage and discharge areas where patients are stationed before and after entering the unit, respectively. It is assumed that these two areas have ample capacity to accommodate a large number of patients and do not represent any specific hospital area. The remaining rooms in R are single-patient and multi-patient rooms, where the number of beds in room j is given by B_j. Additionally, the model considers that there is a finite set of isolation requirements which must be taken into account to accommodate incoming patients. These isolation requirements dictate that patients who have a particular isolation cannot be assigned a room with another patient who does not require the same isolation precaution.

The optimization model assigns patients to beds considering the patient's criticality level, gender, isolation requirements, current placement within the hospital, and if the patient is flagged. The flag is an indicator used to mark if a particular patient cannot be moved to another room in the unit. For example, a flag could be used if the patient is using equipment that is extremely difficult to move or if a patient recently moved into the unit.

The objective function (0) penalizes the number of internal movements needed to accommodate incoming patients, while maximizing the number of highly critical patients in the unit. For implementation purposes, the model's objective function was linearized using a similar approach to the one used by Love and Yerex [34]. To linearize the function, the absolute value expression was altered by introducing two new variables and one new constraint for assigning values to those new variables. This altered form of the objective function and the additional constraint are shown below.

$$\text{Maximize} \qquad \sum_{i\in P}\sum_{j\in R: j\neq T, j\neq D}\left(c_i x_{ij} - \frac{1}{c_i}(s_{ij}^{+} + s_{ij}^{-})\right)$$

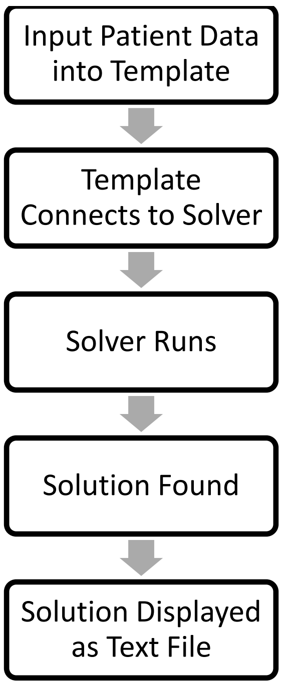
$$x_{ij} - y_{ij} = s_{ij}^{+} - s_{ij}^{-} \qquad i \in P, j \in R\setminus(T \cup D)$$

There are several constraints in place to facilitate that only proper room assignments occur. Constraint (1) ensures that every patient must be assigned a room in the set of *R* rooms (including *T* and *D*). Constraint (2) assigns a value to all the gender variables, δ_{gj}. Constraint (3) assigns a value to all isolation variables, γ_{ij}. Constraints (4) and (5) ensure that each multi-patient room host patients with only one type of isolation requirement at most, and with the same gender. Constraint (6) ensures that the number of patients in a room does not exceed the number of beds available in the room. Constraint (7) prevents moving patients that currently have a room within the unit into the Triage area. Constraint (8) prohibits patients from being directly moved from the triage to discharge area without being assigned a bed in the unit. Constraint (9) prevents moving patients assigned to private rooms to another private room within the unit. Constraint (10) ensures that patients that have a flag are not moved from their rooms.

4.1.1 PCCU Hospital Application

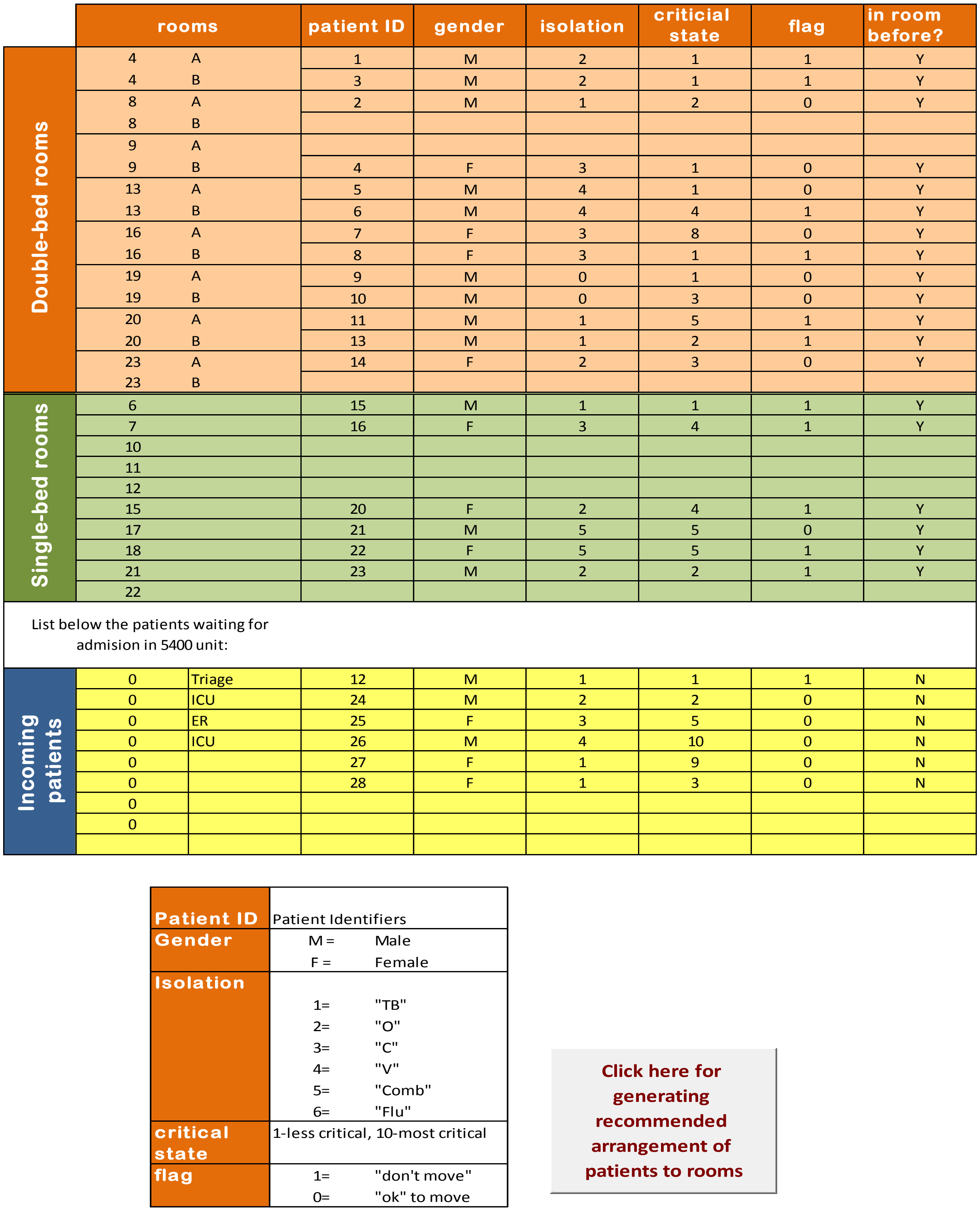
Using the methodology presented in the prior section we created a simple, user friendly tool that hospital bed assignment staff can utilize to facilitate bed assignment decisions. The interface of this tool was created in Microsoft Excel and the solver utilized for computing the assignments was GLPK. GLPK was chosen since it is an open source solver that would not require the hospital to purchase any licenses for operation. The entire tool operates as shown in Figure 2.

Figure 2: Flowchart for Assignment Tool


Input Patient Data into Template
Template Connects to Solver
Solver Runs
Solution Found
Solution Displayed as Text File

The template in Excel serves as the means for bed assignment staff to input patient characteristics. An example of the template is shown in Figure 3. The tool requires as inputs the patient information regarding both the patients in the unit and those seeking admission. When the user requests the tool to generate a bed assignment, the Excel interface internally generates a data file that can be used by GLPK to solve the model presented in 4.1. The solution is automatically read from GLPK and translated into a spreadsheet for the users to view. An example of the displayed results is shown in Figure 4.

Figure 3: Example of Excel User Interface



	rooms		patient ID	gender	isolation	criticial state	flag	in room before?
Double-bed rooms	4	A	1	M	2	1	1	Y
	4	B	3	M	2	1	1	Y
	8	A	2	M	1	2	0	Y
	8	B						
	9	A						
	9	B	4	F	3	1	0	Y
	13	A	5	M	4	1	0	Y
	13	B	6	M	4	4	1	Y
	16	A	7	F	3	8	0	Y
	16	B	8	F	3	1	1	Y
	19	A	9	M	0	1	0	Y
	19	B	10	M	0	3	0	Y
	20	A	11	M	1	5	1	Y
	20	B	13	M	1	2	1	Y
	23	A	14	F	2	3	0	Y
	23	B						
Single-bed rooms	6		15	M	1	1	1	Y
	7		16	F	3	4	1	Y
	10							
	11							
	12							
	15		20	F	2	4	1	Y
	17		21	M	5	5	0	Y
	18		22	F	5	5	1	Y
	21		23	M	2	2	1	Y
	22							

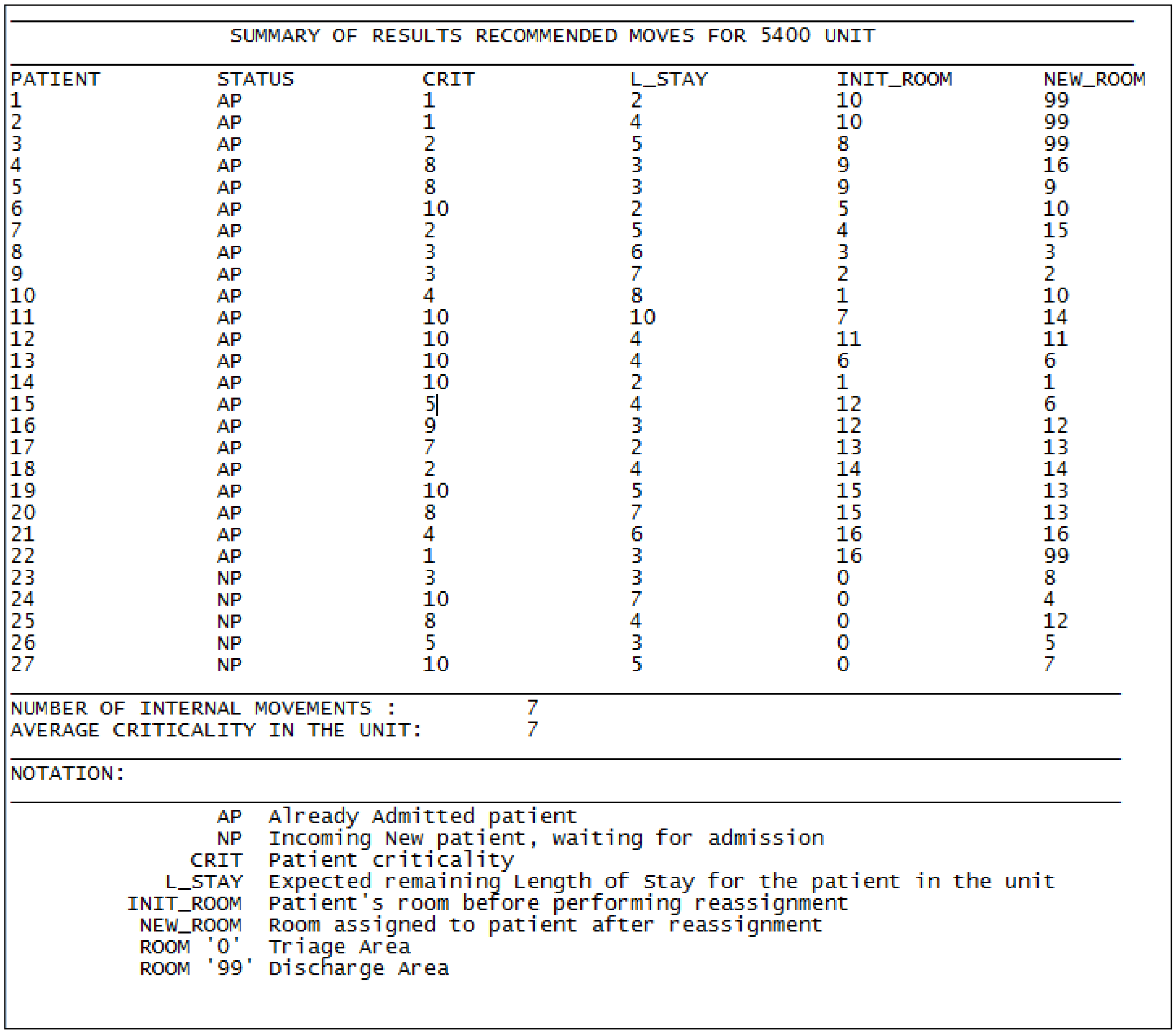
List below the patients waiting for admision in 5400 unit:

Incoming patients	0	Triage	12	M	1	1	1	N
	0	ICU	24	M	2	2	0	N
	0	ER	25	F	3	5	0	N
	0	ICU	26	M	4	10	0	N
	0		27	F	1	9	0	N
	0		28	F	1	3	0	N
	0							
	0							

Patient ID	Patient Identifiers	
Gender	M =	Male
	F =	Female
Isolation	1=	"TB"
	2=	"O"
	3=	"C"
	4=	"V"
	5=	"Comb"
	6=	"Flu"
critical state	1-less critical, 10-most critical	
flag	1=	"don't move"
	0=	"ok" to move

Click here for generating recommended arrangement of patients to rooms

Figure 4: Example Output from Assignment Tool



```
                 SUMMARY OF RESULTS RECOMMENDED MOVES FOR 5400 UNIT

PATIENT        STATUS         CRIT           L_STAY         INIT_ROOM      NEW_ROOM
1              AP             1              2              10             99
2              AP             1              4              10             99
3              AP             2              5              8              99
4              AP             8              3              9              16
5              AP             8              3              9              9
6              AP             10             2              5              10
7              AP             2              5              4              15
8              AP             3              6              3              3
9              AP             3              7              2              2
10             AP             4              8              1              10
11             AP             10             10             7              14
12             AP             10             4              11             11
13             AP             10             4              6              6
14             AP             10             2              1              1
15             AP             5              4              12             6
16             AP             9              3              12             12
17             AP             7              2              13             13
18             AP             2              4              14             14
19             AP             10             5              15             13
20             AP             8              7              15             13
21             AP             4              6              16             16
22             AP             1              3              16             99
23             NP             3              3              0              8
24             NP             10             7              0              4
25             NP             8              4              0              12
26             NP             5              3              0              5
27             NP             10             5              0              7

NUMBER OF INTERNAL MOVEMENTS :          7
AVERAGE CRITICALITY IN THE UNIT:        7

NOTATION:

               AP  Already Admitted patient
               NP  Incoming New patient, waiting for admission
             CRIT  Patient criticality
           L_STAY  Expected remaining Length of Stay for the patient in the unit
        INIT_ROOM  Patient's room before performing reassignment
         NEW_ROOM  Room assigned to patient after reassignment
         ROOM '0'  Triage Area
        ROOM '99'  Discharge Area
```

To examine the benefits of this tool, consider an assignment scenario described in the Table 1 below. This example depicts a typical day at the Pulmonary Critical Care Unit at RGH (PCCU). In this example, the unit has 22 hospitalized patients, 13 and 9 in double and single bed rooms, respectively. The unit can accommodate 4 additional patients, 3 in double bedrooms, and 1 in a single bed room. Eight patients, with different criticalities and isolation requirements are seeking admission from a triage area. Now the questions arise of which patients should be admitted, which patients should be discharged, and should any internal movements be performed? The last three columns of Table 1 show the recommendations from the implementation of assignment tool. The tool suggests that 5 patients can be admitted after performing only 1 internal movement and 1 discharge.

Table 1: PCCU Unit Example

	Initial State							Optimization recommedations	Final State			
	Rooms	(beds)	Patient ID	Gender	Isolation Requirement	Criticality	Flag		Rooms	(beds)	Patient ID	
Double Bedrooms	4	(A)	1	M	2	10	0		4	(A)	1	Double Bedrooms
		(B)	3	M	2	10	0			(B)	3	
	8	(A)	2	M	1	2	0		8	(A)	2	
		(B)								(B)	10	
	9	(A)							9	(A)	25	
		(B)	4	F	3	1	1			(B)	4	
	13	(A)	5	M	4	9	0		13	(A)	5	
		(B)	6	M	4	2	1			(B)	6	
	16	(A)	7	F	3	8	1		16	(A)	7	
		(B)	8	F	3	4	1			(B)	8	
	19	(A)	9	M	1	1	0	Discharge	19	(A)	27	
		(B)	10	M	1	3	0	Move to room 8		(B)	28	
	20	(B)	11	M	1	5	0		20	(B)	11	
		(A)	13	M	1	2	0			(A)	13	
	23	(B)	14	F	2	3	0		23	(B)	14	
		(A)								(A)	29	
Single Bedrooms	6		15	M	1	1	1		6		15	Single Bedrooms
	7		16	F	3	4	1		7		16	
	10		31	F	2	10	0		10		31	
	11		32	M	3	10	0		11		32	
	12								12		26	
	15		20	F	2	4	1		15		20	
	17		21	M	5	6	0		17		21	
	18		22	F	5	5	1		18		22	
	21		23	M	2	2	1		21		23	
	22		33	F	3	7	0		22		33	
Patients Seeking Admission		Triage	12	M	1	1				Triage	12	Patients Still Seeking Admission
		Triage	24	M	2	2				Triage	24	
		Triage	25	F	3	5		admit				
		Triage	26	M	4	10		admit				
		Triage	27	F	1	9		admit				
		Triage	28	F	1	3		admit				
		Triage	29	F	2	10		admit				
		Triage	30	F	1	2				Triage	30	

4.2 Experimenting with the Single Period Model: Addressing Strategic Bed Assignment Challenges

The experimentation conducted for this study was designed to determine the optimal number of single and double bedrooms in the PCCU and to ascertain if an increase in the historical arrival rate to the unit would affect the suggested unit configuration. For implementing this experimentation, a large simulation model was developed in Visual Studio 2010 using C++ and GLPK. The detail of the C++ code is included in Appendices B & C.

In the first set of experiments, the configuration of the unit was analyzed to determine the ideal mix of single and double bedrooms for the PCCU unit. Initially the simulated unit contained 18 single bedrooms. Then successively for each new subset of experiments, one of the single bedrooms was replaced by a double bedroom. This continued until there were only double bedrooms in the unit. A detailed explanation of the simulation is provided in the next section.

4.2.1 Simulation

To address the strategic challenge of determining the optimal mix of single and double bedrooms in a hospital unit, we propose a large simulation model that for each possible unit configuration emulates the utilization of rooms over 5 years of simulated inpatient demand.

The simulation model is described by the event graph in Figure 5.To begin, the unit was populated with a set of patients for day 0. Next, a random number of incoming patients was generated following the distribution of inpatient demand per day at the PCCU (see Figure 6). Every patient seeking admission to the unit was characterized by the following characteristics: a unique ID; a randomly generated length of stay (LOS), which followed the distribution described in Figure 7, a discharge day and a criticality level, an isolation category, and a gender randomly generated following discrete uniform distributions. To elaborate, a patient had an equal chance to be assigned any value of 0-7 for isolation, 0 being they did not have an isolation requirement. Next, it was determined if any patient should be discharged from the unit.

Figure 5: Simulation Event Graph

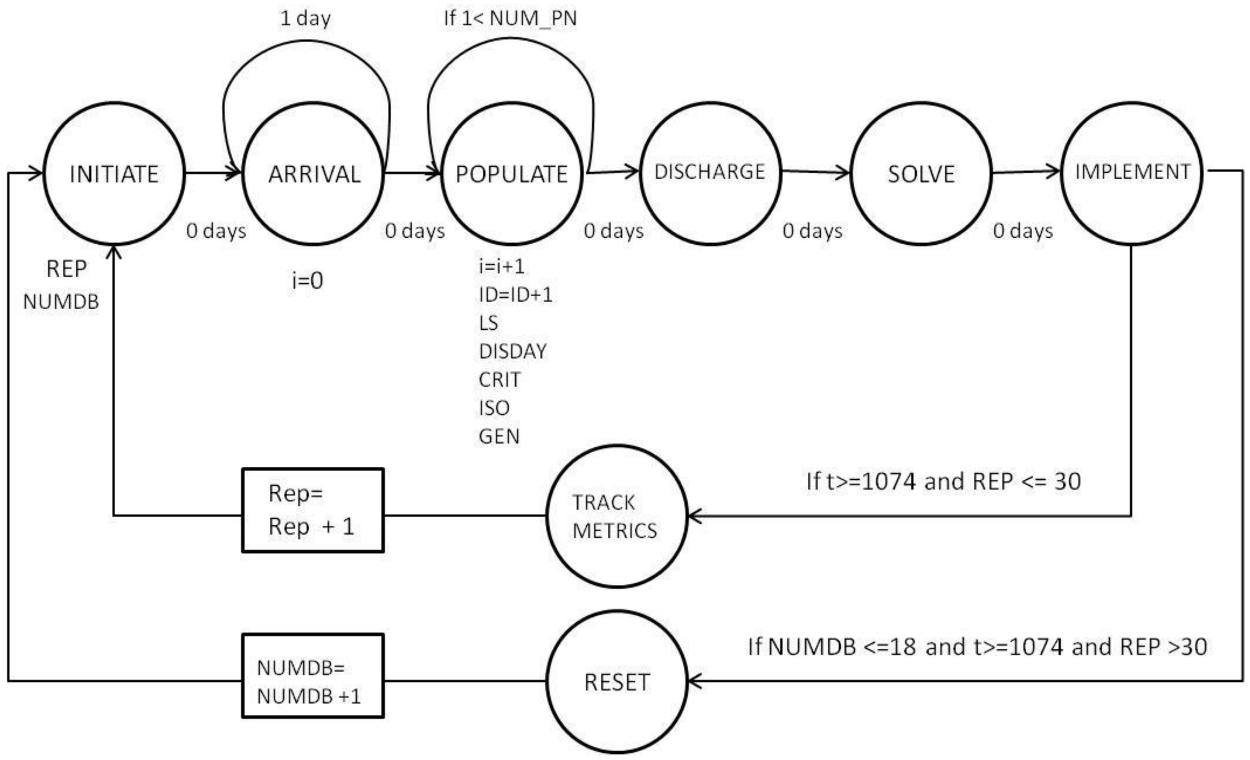


Figure 6: Empirical Distribution of the Number of New Patients Requiring Daily Admission to the PCCU, Based on 3 Years of Historical Information

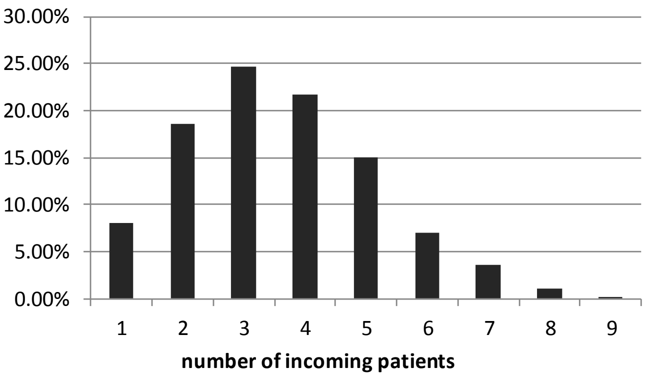
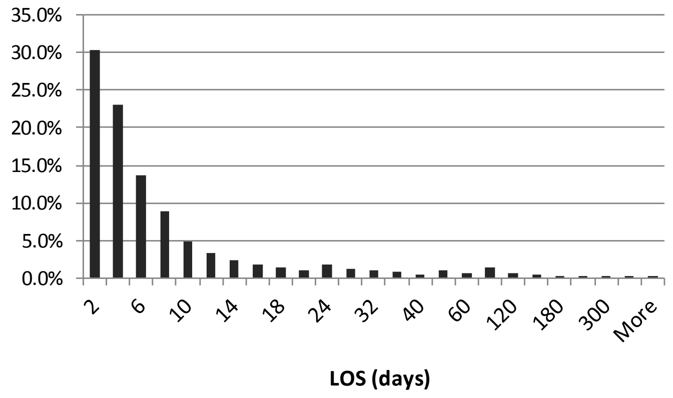


Figure 7: Empirical Distribution of the Length of Stay of Patients Admitted into the PCCU, Based on 3 Years of Historical Information



Once all the characteristics for the set of incoming patients to be considered for admission to the unit were determined then the model presented in Section 4.1 acted as a unit administrator. The model determined the best patient-to-room assignment solution in order to accommodate the random pool of incoming patients while minimizing the number of internal movements and maximizing the unit utilization by admitting the most critical patients. The solution for each problem instance was stored and the following statistics were collected: the iteration number for the simulation, the clock time, the number of internal movements performed, the number of patients assigned to beds in the unit, the average criticality in the unit, the number of internal movements incurred in the day, and the objective value for the optimization model.

For each bedroom configuration, the proposed optimization model solved 30 replications for 5 years of historic data. After thirty replications were run then the conditions of the unit were reset for a different bedroom configuration. The baseline configuration for experimental comparison is eight double bedrooms and ten single bedrooms which was the configuration in the PCCU. Overall this study simulated more than 500,000 days of inpatient arrivals to the unit.

4.2.2 Results & Analysis of the Single-Period Model

As stated earlier, the goal of this experimentation was to determine the optimal mix of single and double bedrooms for the PCCU as well as to determine an appropriate safety stock level of resources to provide a given service level within the unit. The following discussion explains the implications gathered from modifying the number of single and double bedrooms within the unit and practical suggestions for managing bed resources.

Figure 8 shows that there is a strong correlation between the average number of admitted patients ($E[P_a]$) and the average number of internal movements ($E[N]$). The trend line in Figure 8, results from a linear regression model and describes a polynomial of order 3 relationship between $E[N]$ and $E[P_a]$. This trend line can be useful for daily planning in the unit. In particular, it can help determine daily staffing and resources necessary to deal with an expected number of internal movements.

Figure 8: Correlated Fit Between the Average Number of Internal Movements Per Day & the Average Number of Inpatients Per Day

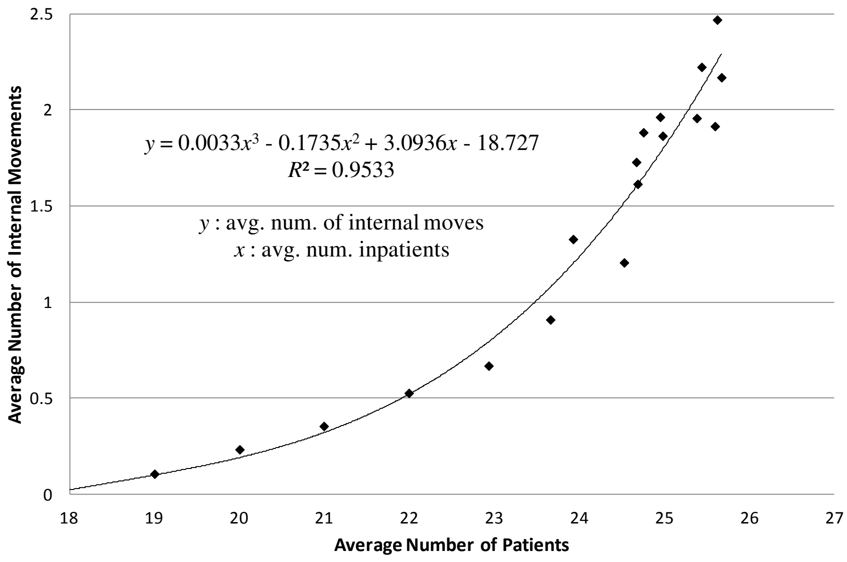
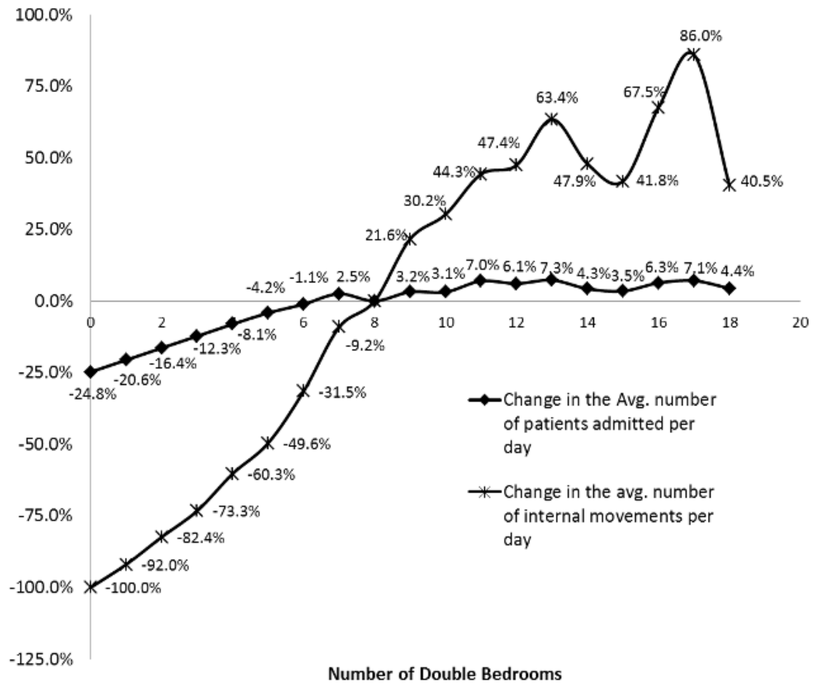


Figure 9 describes the relative change of the average number of internal movements and the relative change in the average number of inpatients admitted to the unit for varying number of double bedrooms in the unit with respect to the base layout of eight double bedrooms. Furthermore, Figure 9 shows that when the unit's capacity is increased, the average number of internal movements increases faster than the growth in capacity.

Considering this information it seems that the optimal number of single and double bedrooms cannot be determined explicitly since different incentives will lead to different conclusions. For example, reviewing Figure 9 shows that reducing the number of double bedrooms to 5 would reduce the number of internal movements by 60% and the number of admitted patients by 8%. Depending on who the decision maker is, it may be worth it to reduce capacity by three beds. Looking at Figure 10(a) shows that the 8% reduction corresponds to an average of admitting one fewer patient to the unit. The results of this experimentation may not state conclusively the number of single and double bedroom to have within the unit but the results can provide the means to develop a business case for a certain unit configuration.

Figure 9: Percent Change in the Average Number of Inpatients Per Day in the Unit and the Average Number of Internal Movements as a Result of Different Double Bedroom Configurations



Furthermore, Figure 10 (a) and 10 (b) show that increasing the number of double beds in the unit not only allows an increase in the average number of admitted inpatients and inpatient movements, but also in their variability. If variability in the number of admitted inpatients was negligible, it will be easy to determine the exact number of additional resources needed to cope with the increasing number of patients. However, in order to mitigate the effects of the increasing variability in the number of inpatients and internal movements, unit administrators can reduce the number of double-patient rooms in the unit or they can rely on securing a buffer (or a "safety stock") of additional resources to cope with potentially high inpatient demand. Therefore, in the latter case, the unit must determine the minimum safety stock level of beds and other resources (e.g., nurses and cleaning personnel) necessary to provide a given service level of α% to patients seeking admission to the unit. A service level of α% implies that there is a (1−α)% probability that the unit will not be able to cope with incoming patient demand due to insufficient resources (e.g., beds). We propose that the safety stock levels of different resources necessary to cope with the increasing variability in demand can be determined using standard manufacturing practices [26]. Assuming a constant lead time for receiving an order of additional resources, L, the safety stock, *ss*, is expressed by $ss = Z_{\alpha}\sqrt{L\,\sigma_d^2}$, where Z_{α} is a standardized normal value for the service level α, and σ_d^2 is the variance of the number of admitted inpatients associated with the unit layout.

For example in the case of beds, Figure 11(a), shows the necessary safety stock of beds (i.e., number of beds to block) to provide a 90% service levels for the unit, as the number of double-bed rooms in the unit increases (for *L*=1). Figures 11(b) and 11(c) report similar results but for inpatient arrival distributions, that have on average 1 and 2 more patients per day than what is currently faced by the unit, respectively. Figures 11 (a), (b), (c) indicate that there are safety stock levels of beds that are robust to changes in the average patient arrival rate, and provide a desired service level. For example, at the current configuration of eight double-patient rooms, the unit would need to ensure access to 3 additional beds to cope with the uncertainty in patient demand.

Figure 10 (a): The Average Number of Inpatients and 2 Standard Deviations from the Mean Versus the Number of Double Bedrooms

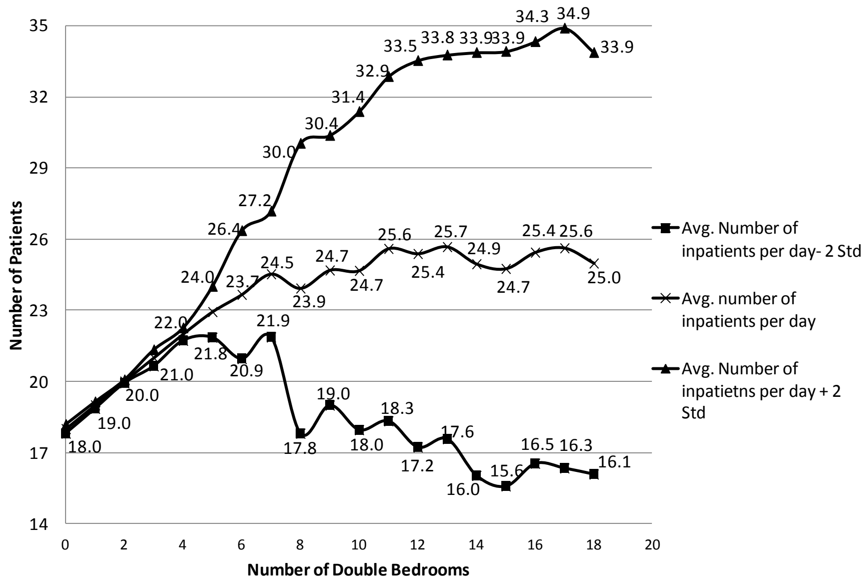


Figure 10 (b): The Average Number of Internal Movements and 2 Standard Deviations from the Mean versus the Number of Double Bedrooms

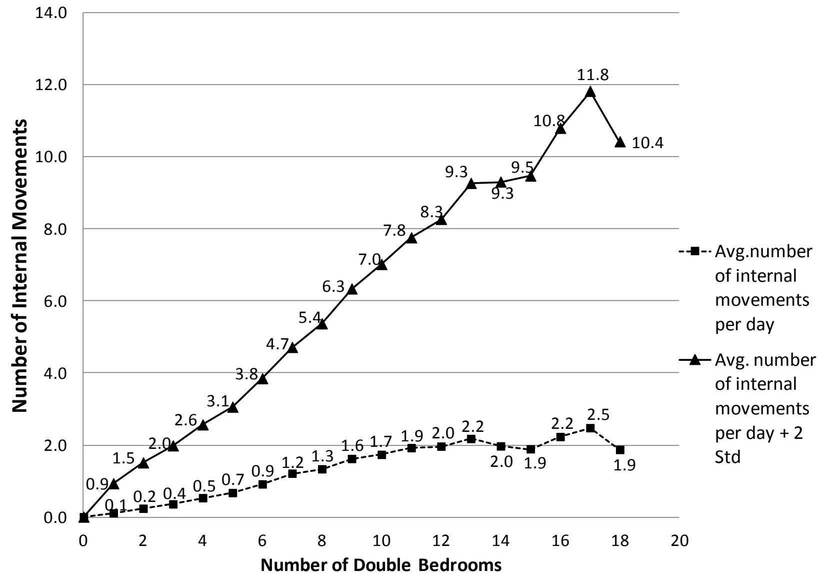


Figure 11 (a): The Number of Additional Beds Needed to Provide a 90% Service Level at the Historic Patient Demand Level

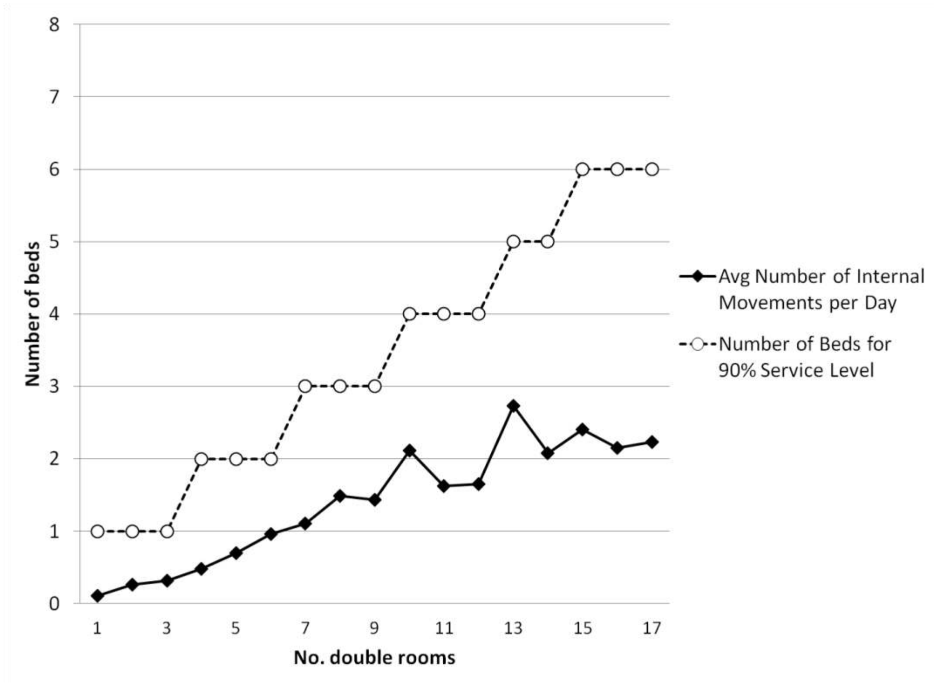


Figure 11(b): The Number of Additional Beds Needed to Provide a 90% Service Level at the Historic Patient Demand Level Plus one Additional Patient on Average

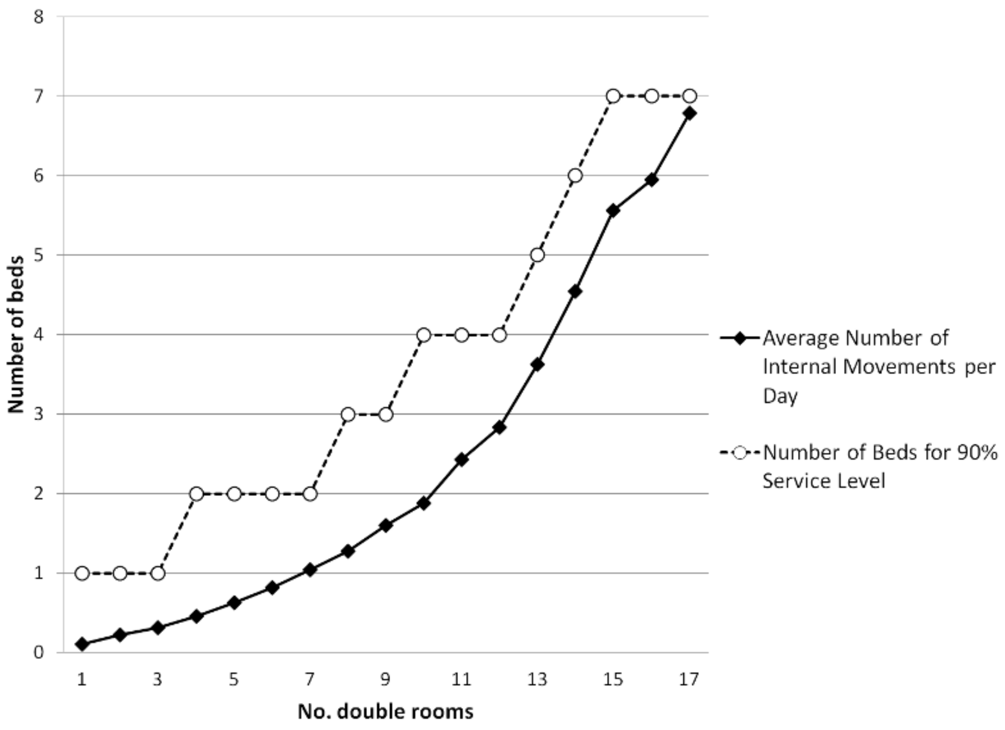
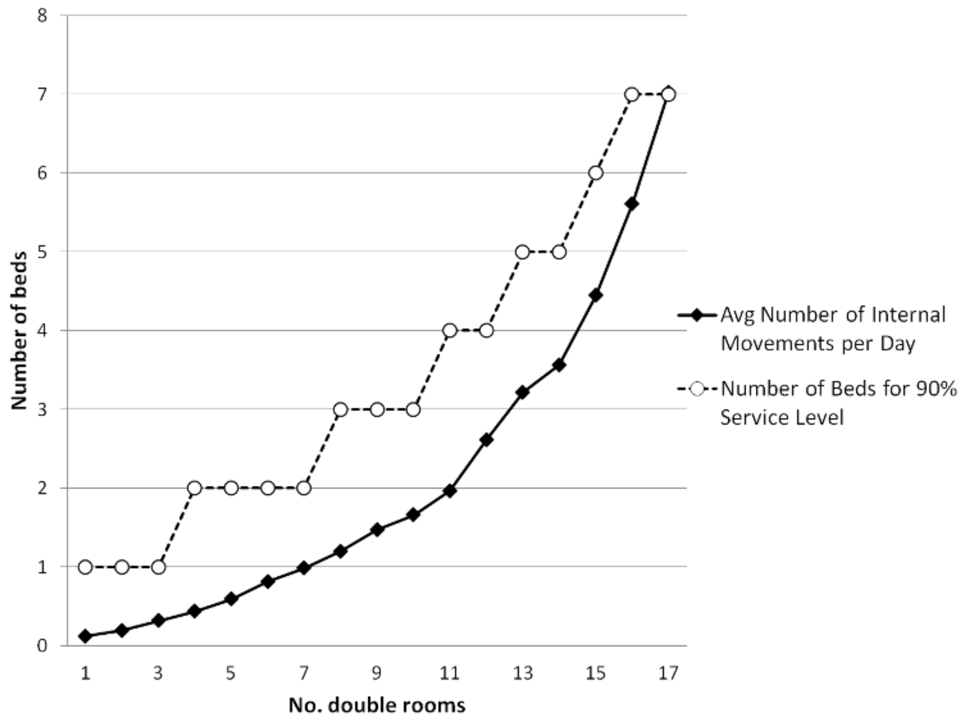


Figure 11(c): The Number Additional of Beds Needed to Provide a 90% Service Level at the Historic Patient Demand Level Plus Two Additional Patients on Average



4.3 Future Knowledge Consideration Methodology

The single period model was designed for use when inpatient demand is known as well as all patients' characteristics. However, hospital units may also have some additional information about patients who may seek admission in the future. For example, there could be patients in the ICU or in the Emergency Room who are expected to seek a bed in the PCCU by the next day. Thus, in this section we propose an extension to the bed assignment model provided in section 4.1 that incorporates when the likelihood of seeking admission is known for patients currently not requiring admission into the unit but who may do so in the upcoming days. Incorporating future knowledge in today's bed assignment decision-making may be advantageous, if for example, keeping a bed unused allows for admitting a highly critical patient tomorrow, without increasing the expected number of internal movements.

This model considers a current planning period (period 1), where a hospital unit has a number of patients already admitted in the unit. At this time, the unit contemplates a set of patients seeking admission P_n, and a third set of patients, P_T, who may seek admission one or more periods into the future. Without loss of generality, all P_T patients are assumed to be ICU patients. The possibility of seeking admission for patients in P_T is modeled by the parameter, p_i, which describes the likelihood that a patient i in the ICU will be discharged from the ICU on any given day and requests admission to the PCCU. For sets P_a and P_n it was assumed that the probability that these patients require a bed in the PCCU is one since these patients are known to demand a bed in the unit, respectively. Further explanation on the generation of this probability parameter is described in section 4.3.1. Moreover, a series of weights, w_k are used to describe the relative importance of the expected bed assignments in day k towards the overall bed assignment plan. For example, information available for period 1 in the planning horizon would be given the largest weight since it is the period during which the assignments are actually made and the available information is the most accurate. The information of subsequent periods would be given decreasing weights to reflect the growing uncertainty and their decreasing importance towards the overall bed assignment plan.

There are particular assumptions in this model that differ from the implementation discussed in section 4.1. First, patients already admitted in the unit can either remain in their original assignments or can be moved to another bed within the unit, but they cannot be discharged. This assumption results from the fact that admitted patients to the PCCU have a longer LOS than a practical planning horizon for bed managing purposes. Second, patients who may seek admission in the future can be assigned to a bed in the PCCU only in period 2 and onwards since these periods represent future assignments. Third, it is assumed that the ICU is fully occupied at the time of making a bed assignment decision (i.e., at period 1). We also assumed that the ICU is always at full capacity. Finally, it is assumed that if a patient in P_T is recommended for admission in a future period of the planning horizon, he or she cannot be sent back to the ICU in any subsequent period.

The configuration of the unit (eight double bedrooms and 10 single bedrooms) is kept consistent over all planning periods. Moreover, the criticality and isolation requirements assigned in period 1 remain constant over all planning periods. Additionally, patients in set P_T are assigned criticalities higher than patients in sets P_a and P_n since being in the ICU indicates a more critical condition. Moreover, the criticalities for ICU patients are assumed static (i.e., they do not change after a patient has been assigned a bed in the PCCU) since patients' future health conditions are uncertain.

The resulting model provides a bed assignment schedule for the current planning period while inducing the lowest expected number of internal movements. The model considers all restrictions used in the single-period model of section 4.1, plus additional constraints necessary to consider future events.

The notation and formulation of the multi-period model is described as follows:

P_a:	set of patients currently admitted to the unit
P_n	set of patient currently seeking admission to the unit

P_T:	set of patients who are currently not seeking admission and remain in the ICU but may seek admission during the planning horizon
P:	set of all patients (i.e., $P = P_a \cup P_n \cup P_T$)
Q:	set of patients in the unit and seeking admission (i.e., $P = P_a \cup P_n$)
R:	set of all available rooms
T:	triage room $T \in R$
D:	discharge room, $D \in R$
U:	ICU, $U \in R$
B_j:	number of beds available in room $j \in R$
H:	set of planning periods (H=1…t)
R_1:	set of single-patient rooms.
G_i:	gender of patient $i \in P$
I_i:	isolation type of patient $i \in P$
c_i:	the assessed level of how much patient $i \in P$ needs intensive medical care (it can take a value 0-10)
w_k:	the smoothing weight factor assigned for each planning period $k \in H$

$$\mu: \begin{cases} 1 \text{ if discharges are allowed} \\ 0 \quad\quad\quad\quad\quad\quad \text{o.w.} \end{cases}$$

p_i: the probability associated with each patient $i \in P_T$ seeks a room in the unit

$$x_{ij0}: \begin{cases} 1 \text{ if a patient } i \in P_a \text{ was in a bed in room } j \in R \text{ before the initial planning period} \\ 0 \quad\quad\quad\quad\quad\quad \text{o.w.} \end{cases}$$

Decision variables:

$$x_{ijk}: \begin{cases} 1 \text{ if a patient } i \in P \text{ is assigned to a bed in room } j \in R \text{ in period k} \in \text{H} \\ 0 \qquad\qquad\qquad \text{o.w.} \end{cases}$$

$$\delta_{gjk}: \begin{cases} 1 \text{ if a patient of gender } g \in G \text{ is assigned to a bed in room } j \in R\backslash R_1 \text{ in period } k \in H \\ 0 \qquad\qquad\qquad \text{o.w.} \end{cases}$$

$$\gamma_{ijk}: \begin{cases} 1 \text{ if a patient with isolation needs } i \in I \text{ is assigned to a bed in room } j \in R\backslash R_1 \\ \quad \text{in the period before } k \\ 0 \qquad\qquad\qquad \text{o.w.} \end{cases}$$

Maximize

$$\sum_{k\in H} w_k \left[\sum_{i \in Q} \sum_{j\in R, j\neq T j\neq U j\neq D} c_i * x_{ijk} - \frac{1}{2} * c_i^{-1} \left|x_{ijk} - x_{ij(k-1)}\right|\right] +$$

$$\sum_{k\in H} w_k \left[\sum_{i \in P_T} \sum_{j\in R, j\neq T, j\neq U j\neq D} \left(c_i * x_{ijk} - \frac{1}{2} * c_i{}^{-1} \left|x_{ijk} - x_{ij(k-1)}\right|\right) * p_i (1-p_i)^{k-1}\right] \quad (0)$$

s.t.

$$\sum_{j \in R} x_{ijk} = 1 \qquad i \in P, k \in H \qquad (1)$$

$$x_{ijk} \leq \delta_{g_i jk} \qquad i \in P, \quad j \in R\backslash(R_1 \cup T \cup D \cup U), \quad k \in H \qquad (2)$$

$$x_{ijk} \leq \gamma_{I_i jk} \qquad i \in P, \quad j \in R\backslash(R_1 \cup T \cup D \cup U), \quad k \in H \qquad (3)$$

$$\sum_{g\in G} \delta_{gjk} \leq 1 \qquad j \in R\backslash(R_1 \cup T \cup D \cup U), \quad k \in H \qquad (4)$$

$$\sum_{i\in I} \gamma_{ijk} \leq 1 \qquad j \in R\backslash(R_1 \cup T \cup D \cup U), \quad k \in H \qquad (5)$$

$$\sum_{i \in P} x_{ijk} \leq B_j \qquad j \in R, k \in H \qquad (6)$$

$$\sum_{j_2\in R_1: j_2\neq j_1} x_{ij_2} \leq (1 - x_{ij1(k-1)}) \qquad i \in P, \quad j_1 \in R_1, \quad k \in H\backslash 0 \qquad (7)$$

$$x_{iTk} = 0 \qquad i \in P_a, \quad k \in H \tag{8}$$

$$x_{iUk} = 0 \qquad i \in Q, \quad k \in H \tag{9}$$

$$\sum_{j \in R: j \neq U} \sum_{k=0}^{1} x_{ij0} = 0 \qquad i \in P_T \tag{10}$$

$$x_{iT0} = 1 \qquad i \in P_n \tag{11}$$

$$x_{ij0} = y_{ij} \qquad i \in P_a, \quad j \in R \tag{12}$$

$$x_{iUk} \leq 1 - \sum_{j \in R} x_{ij(k-1)} \qquad i \in P_t, \quad k \in H \backslash 0 \tag{13}$$

$$\mu * x_{iDk} = 0 \qquad i \in P, \quad k \in H \tag{14}$$

As stated earlier, this model focuses on incorporating the use of future knowledge regarding patients who may seek admission into the unit. A set, P_T of patients is added to the original formulation having the additional parameter, p_i. The set H, was added to describe the number of periods for which the decision maker will look into the future to make a decision now. Furthermore, as mentioned earlier, a series weights, w_k is included to describe the relative importance of periods to the schedules.

It should also be noted in this multi-period bed assignment model, criticality is defined as in the single-period model representation on a scale from (1-10) for patients seeking admission and for those patients within the unit. A value of 10 is interpreted as a patient who is highly critical. However, patients in P_T, who are currently being treated in the ICU are assigned a criticality value higher than those of admitted patients (P_a) and those currently seeking admission into the unit (P_n).

The objective function in (0) consists of two weighted sum expressions. The first summation maximizes the expected criticality in the unit while penalizing the expected number of internal movements for patients currently in the unit and for those seeking admission. The second summation corresponds to the expected criticality and penalty resulting from future admission of any ICU patients who do not seek admission in the unit during period 1.

Constraint (1) ensures that all patients are assigned to a room in each period. Constraint (2) ensures that for every period, a patient is assigned to a double room only if the room already has a patient with the same gender. Constraint (3) ensures that for every period, a patient can be assigned to an occupied double room if both patients have the same isolation requirements. Constraint (4) prohibits more than one gender in a double bedroom for each period. Constraint (5) prohibits more than one isolation condition in a double bedroom for each period. Constraint (6) ensures room capacity is not exceeded. Constraint (7) ensures that for each planning period, patients assigned to a single-bedroom cannot be moved to another single-bedroom. Constraint (8) ensures that admitted patients are not moved to the Triage area. Constraint (9) ensures that admitted patients and those seeking admission to the unit are not assigned to the ICU during any period. Constraint (10) ensures that patients in P_T are not given a room in period 1. Constraint (11) and Constraint (12) are necessary to set initial conditions for patients seeking admission and those already admitted. Constraint (11) assigns every patient in set P_n to the "Triage" area during period 0. Constraint (12) assigns the initial bed assignments to P_a patients which can be interpreted as where the patients are assigned for period 0. Constraint (13) does not allow patients who have been admitted from the ICU to be sent back to the ICU in subsequent periods.

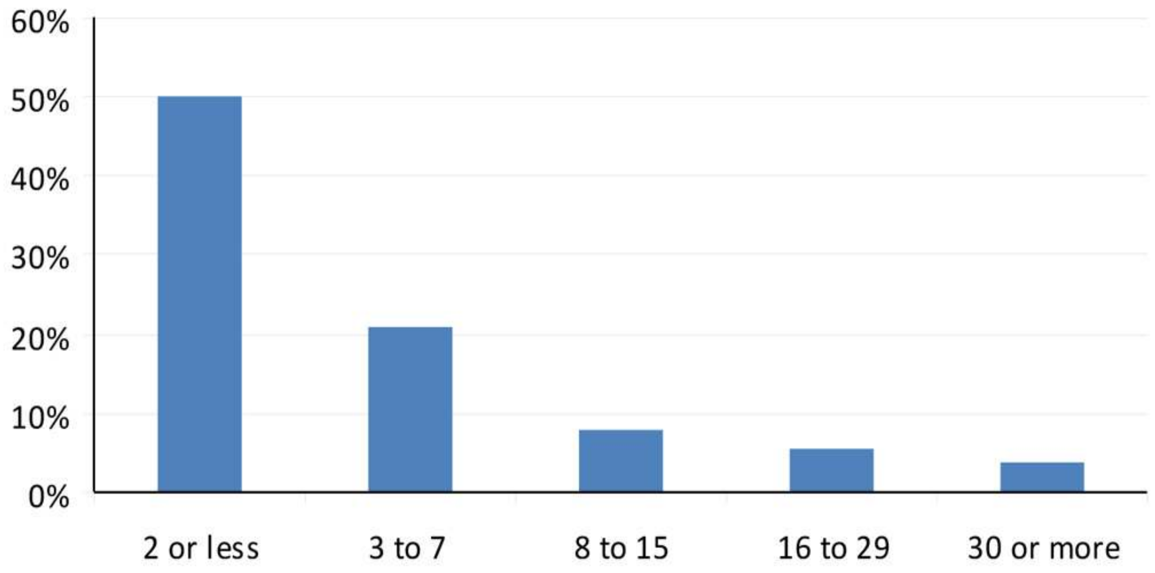
Lastly, Constraint 14 controls if patients can be discharged from the unit during any period. This constraint can be modified by changing the μ parameter.

4.3.1 Estimating the Probabilities of Requiring a Bed in the PCCU in a Given Day

This section describes how to estimate the parameter p_i used in the model to describe the probability that a patient *i* (in the ICU but not seeking admission to PCCU today) is discharged from the ICU and requires a bed in the PCCU during a later day.

Unfortunately, historical data regarding LOS and hospitalization rates in the ICU at our partner hospital were not available for this part of the study; hence, this section relies on a study available in literature to develop our estimates. In particular, we take advantage of Ong et al. [35] who surveyed the LOS of 4902 ICU patients over 7 years in a United States Hospital. As a result, Ong et al. [35] categorizes ICU patients according to the LOS distribution as presented in Figure 12.

Figure 12: LOS Distribution from ICU Patients



X axis = length of stay (days)
Y axis = percentage of all intensive care unit (ICU) patients.

The demand for a bed in the PCCU by an ICU patient *i* for a given day is a Bernoulli random variable with probability p_i. Therefore, the number of days until an ICU patient requires a bed in the PCCU, *N*, follows a geometric distribution with parameter *p*. Therefore, the probability that an ICU patient requires a bed in the PCCU *x* days after the day of planning is given by P(*N*=*x*) $= p(1-p)^{x-1}$, which indicates that the patient stays (*x*-1) consecutive days in the ICU before requiring admission to the PCCU in day *x*.

Since the expected value of a geometrically distributed random variable corresponds to the reciprocal of its parameter *p*, we can estimate the value of *p* by the reciprocal of the expected number of days for a patient to first seek a bed in the PCCU (i.e., $p=1/E[N]$). Since ICU patients have different LOSs in the ICU, based on Ong et al [35] categorization, the value of *p* for a given ICU patient is estimated by the average of the minimum and maximum calculated *p* values corresponding to the patient's LOS category. For example, consider a category 1 patient expected to stay in the ICU for 1 or 2 days. Thus the probability that this patient seeks a bed in the PCCU is estimated by the average of ½ and 1, which results in ½ (½ + 1/1)= ¾. The resulting estimated values for *p* for all ICU patients according to their LOS are given in Table 2.

Table 2: Estimated Probability Parameters for Each LOS Category

LOS Categories (days)	Calculation of p	Estimated p
1-2	1/2(1/2 +1)	0.75
3-7	1/2(1/3 +1/7)	0.23
8-15	1/2(1/8 + 1/15)	0.10
16-29	1/2(1/16 + 1/29)	0.05
30+	1/2(1/30)	0.02

4.4 Experimentation of Future Knowledge Consideration Model

We use the future knowledge model described in section 4.3 to explore, whether considering information about the future results in improved bed assignment plans compared to plans resulting from considering patient information only for the day in which planning occurs. Therefore, we propose an experimental setting in which the future knowledge model is solved using a 2^2 factorial design, for which each problem instance is solved over different planning horizons. A full description of this experimental approach is described in the following two sections. Patient characteristics for the elements of these problem sets (i.e., each patient's gender, criticality, and isolation requirement) are randomly generated similarly to the single- period experimentation. A detailed explanation on how these patient sets and their characteristics were randomly simulated is included in the Appendix F.

In a preliminary analysis we aim to determine the most effective experimental setting for evaluating the performance of applying the future knowledge model over different planning horizons. This exploratory exercise, evaluated the bed assignments resulting from the solution of our model for a small set of problem instances that differ in the number of patients in their sets P_n and P_a, and changes in the criticality values for patients in the ICU (P_T). Given this understanding of the model, a set of narrowly scoped experiments were designed considering different numbers of patients in sets P_n and P_a, where ICU patients (P_T) have criticality values of 11-15.

4.4.1 Exploratory Sampling

We consider a baseline configuration in the PCCU with eight double bedrooms and ten single bedrooms according to the current arrangement used in the single period model experimentation. To explore the effect on bed assignments from the criticality values assigned to the ICU patients, we constructed three distinct problem conditions with differing numbers of patients in sets P_a, P_n, and P_T that resulted in differing number of beds available per condition. For each of these problem conditions all patient characteristics were randomly generated with the exception of the relative criticality of P_T patients (ICU patients). From each problem condition we created two problem instances (or data sets), one having a criticality value of 20 for all ICU patients (data8a, data10a, data14a), and the other set having randomly generated criticalities ranging from 1-10 for the ICU patients (data8b, data10b, and data14b). The future knowledge model was then applied to each of these six data sets for five different planning horizons (i.e., 1 day to 5 days), and the number and type of patients admitted in each period were recorded. These results were used to determine how the different criticality values assigned to ICU patients influences bed assignment plans.

The results of the application of the future model in this experiment are presented in tables 4-9, which show that the number of patients admitted was the same for every period in each data set (See columns "# Admit" in tables 4-9). The differing criticality values did not have an effect on the number of patients admitted or on the types (P_n vs P_T) of patients admitted into the unit.

Table 3: Data Sets used for Comparing Different Assigned Criticality Values for P_T Patients

	PA(admitted)	PN(seeking)	PT (ICU)	Beds Open
data8	18	4	18	8
data 10	19	3	18	7
data 14	20	5	18	6

<u>the columns display the following information</u>

PA (admitted):	Patients assigned a bed in the unit
PN (admitted):	Patients seeking a bed in the unit during period 1
PT (admitted):	Patients in the ICU seeking a bed in period 2 or later
Beds Open:	The number of beds unassigned in the unit before period 1

Table 4: Data Set 8a Results Using Assigned Criticality (1-10)

H (planning horizon)	obj	Data	**Period**	**# Admit**	# int mov	Assigned Crit
1	118.405	8	**1**	**4 (PN)**	2	(1-10)
2	119.671	8	**1**	**4 (PN)**	3	(1-10)
		8	**2**	**4 (PT)**	2	(1-10)
3	120.089	8	**1**	**4 (PN)**	3	(1-10)
		8	**2**	**4 (PT)**	2	(1-10)
		8	**3**	**0**	0	(1-10)
4	120.445	8	**1**	**4 (PN)**	3	(1-10)
		8	**2**	**4 (PT)**	2	(1-10)
		8	**3**	**0**	0	(1-10)
		8	**4**	**0**	0	(1-10)
5	120.706	8	**1**	**4 (PN)**	3	(1-10)
		8	**2**	**4 (PT)**	2	(1-10)
		8	**3**	**0**	0	(1-10)
		8	**4**	**0**	0	(1-10)
		8	**5**	**0**	0	(1-10)

the columns display the following information

H:	(planning horizon): number of periods planned for
Obj:	the objective value function
Data:	the label for the data set used
Period:	the specific period in the planning horizon
# admit:	the number and type of patients admitted each period admitted
# int movements:	the number of internal movements that occurred during each period
Assigned Crit:	the criticality values assigned for the ICU patients

Table 5: Data Set 8b Results Using Assigned Criticality 20

H (planning horizon)	obj	Data	**Period**	**# Admit**	# int mov	Assigned Crit
1	118.405	8	**1**	**4 (PN)**	2	20
2	121.616	8	**1**	**4 (PN)**	2	20
		8	**2**	**4 (PT)**	5	20
3	122.859	8	**1**	**4 (PN)**	2	20
		8	**2**	**4 (PT)**	5	20
		8	**3**	**0**	0	20
4	123.913	8	**1**	**4 (PN)**	2	20
		8	**2**	**4 (PT)**	5	20
		8	**3**	**0**	0	20
		8	**4**	**0**	0	20
5	124.649	8	**1**	**4 (PN)**	2	20
		8	**2**	**4 (PT)**	5	20
		8	**3**	**0**	0	20
		8	**4**	**0**	0	20
		8	**5**	**0**	0	20

Table 6: Data Set 10a Results Using Assigned Criticality (1-10)

H (planning horizon)	obj	Data	**Period**	**# Admit**	# int mov	Assigned Crit
1	136.532	10	**1**	**3 (PN)**	2	(1-10)
2	137.913	10	**1**	**3 (PN)**	2	(1-10)
		10	**2**	**4 (PT)**	2	(1-10)
3	138.152	10	**1**	**3 (PN)**	2	(1-10)
		10	**2**	**4 (PT)**	2	(1-10)
		10	**3**	**0**	0	(1-10)
4	138.367	10	**1**	**3 (PN)**	2	(1-10)
		10	**2**	**4 (PT)**	2	(1-10)
		10	**3**	**0**	0	(1-10)
		10	**4**	**0**	0	(1-10)
5	138.495	10	**1**	**3 (PN)**	2	(1-10)
		10	**2**	**4 (PT)**	2	(1-10)
		10	**3**	**0**	0	(1-10)
		10	**4**	**0**	0	(1-10)
		10	**5**	**0**	0	(1-10)

Table 7: Data Set 10b Results Using Assigned Criticality 20

H (planning horizon)	obj	Data	**Period**	**# Admit**	# int mov	Assigned Crit
1	136.532	10	**1**	**3 (PN)**	2	20
2	139.957	10	**1**	**3 (PN)**	2	20
		10	**2**	**4 (PT)**	4	20
3	141.188	10	**1**	**3 (PN)**	2	20
		10	**2**	**4 (PT)**	4	20
		10	**3**	**0**	0	20
4	142.229	10	**1**	**3 (PN)**	2	20
		10	**2**	**4 (PT)**	4	20
		10	**3**	**0**	0	20
		10	**4**	**0**	0	20
5	142.952	10	**1**	**3 (PN)**	2	20
		10	**2**	**4 (PT)**	4	20
		10	**3**	**0**	0	20
		10	**4**	**0**	0	20
		10	**5**	**0**	0	20

Table 8: Data Set 14a Results Using Assigned Criticality (1-10)

H (planning horizon)	obj	Data	**Period**	**# Admit**	# int mov	Assigned Crit
1	141.066	14	**1**	**5 (PN)**	3	(1-10)
2	141.55	14	**1**	**5 (PN)**	3	(1-10)
		14	**2**	**1 (PT)**	0	(1-10)
3	141.681	14	**1**	**5 (PN)**	3	(1-10)
		14	**2**	**1 (PT)**	0	(1-10)
		14	**3**	**0**	0	(1-10)
4	141.798	14	**1**	**5 (PN)**	3	(1-10)
		14	**2**	**1 (PT)**	0	(1-10)
		14	**3**	**0**	0	(1-10)
		14	**4**	**0**	0	(1-10)
5	141.897	14	**1**	**5 (PN)**	3	(1-10)
		14	**2**	**1 (PT)**	0	(1-10)
		14	**3**	**0**	0	(1-10)
		14	**4**	**0**	0	(1-10)
		14	**5**	**0**	0	(1-10)

Table 9: Data Set 14b Results Using Assigned Criticality 20

H (planning horizon)	obj	Data	**Period**	**# Admit**	# int mov	Assigned Crit
1	141.066	14	**1**	**5 (PN)**	3	20
2	141.993	14	**1**	**5 (PN)**	3	20
		14	**2**	**1 (PT)**	0	20
3	142.289	14	**1**	**5 (PN)**	3	20
		14	**2**	**1 (PT)**	1	20
		14	**3**	**0**	0	20
4	142.631	14	**1**	**5 (PN)**	3	20
		14	**2**	**1 (PT)**	1	20
		14	**3**	**0**	0	20
		14	**4**	**0**	0	20
5	142.893	14	**1**	**5 (PN)**	3	20
		14	**2**	**1 (PT)**	1	20
		14	**3**	**0**	0	20
		14	**4**	**0**	0	20
		14	**5**	**0**	0	20

The reason why the ICU criticalities do not affect assignments significantly is because the objective function of the future knowledge model (i.e., eq (0)) and data available for ICU discharge make it hard for an ICU patient to compete for a bed with a patient in sets P_n or P_a. The contribution of an ICU patient to the objective function (eq. (0)) is factored twice. First, by the probability that the patient is discharged from the ICU and seeks a bed in the PCCU, $(1\text{-}p)^n\ p$, and second, by the importance given to information for such day, w_k (where k is the day of discharge and p is the probability of being discharged in any given day). Consider the values of p for the cohort of patients most likely to be discharged from the ICU (p = 0.75), the weighting factors ($(1\text{-}p)^n\ p$) for the contribution of admitting an ICU patient in days 2, 3 and 4 are 0.1875, 0.047, 0.012, respectively. In other words, the criticality of an ICU patient (from the aforementioned cohort, ie., p=0.75) would need to be more than 20 times higher than that of a patient already admitted in the unit to have the ICU patient admitted in day 3 (even when there is no doubt of the importance of the information associated with day 3,(i.e. w_3= 1). Therefore, the criticality value of ICU patients is important only for deciding *which* ICU patient should go in the unit if any bed remains unoccupied in period 2 or later.

Three additional experiments were also run to understand the effect on the bed assignment plans from the number of patients seeking admission (set P_n) and the number of beds available. Two of the data sets used in this experimentation had more patients seeking admission into the unit than beds available, whereas the third data set had fewer patients seeking admission into the unit than beds available. Further explanation regarding the details of the data sets used for this experimentation is included in Appendix H. In almost all problem instances, our experiments showed that when the number of patients seeking admission exceeded the number of beds available during period 1, considering information about future periods is irrelevant since the unit reached full capacity in period 1.

One factor that still needs to be discussed is the manner in which weights for the importance of information of future days were chosen. The weighting factors, w_k are arbitrary and can take any value depending on the importance that the PCCU's decision makers give to the information of day *k*, therefore we chose not to consider this factor into the experimentation. In these experiments, we assume a set of daily weights that monotonically decrease for larger planning horizons. The fixed set of values for each planning horizon was defined and used for all experimentation (see Table 10).

The results of our preliminary experimentation suggest that the number of patients in the unit and the number of patients seeking admission during period 1 most directly influenced a bed assignment plan. We have reached this conclusion since these two factors seem to affect the utilization of the unit during period 1. If the unit reaches full utilization during period 1 then considering any additional future information becomes unnecessary since no beds are available for those patients. Considering these insights, in the next section we propose an experimental design to better understand the effect of considering future information in today's bed assignment decision for multiple planning horizons, while controlling these two critical factors.

4.4.2 Experimentation: Exploring the Effect of Different Planning Horizons in Bed Assignment Decisions

The experimental results summarized in this section illustrate a hypothetical, yet realistic hospital setting in which decision makers give decreasing importance to information regarding bed demand of future days. We propose an experiment to evaluate the effectiveness of scheduling for planning horizons of up to 5 days, where such horizons are understood as the number of future days a decision maker would consider when making bed assignments. The future knowledge model was applied over 4 problem sets, each having 5 different problem instances. In each problem set, the five instances have the same number of $P_{a,}$ P_n , and P_T patients, yet different (randomly generated) patient characteristics. In total, our analysis is based on the results of applying the future knowledge model to 100 different experiments.

The four problem sets were developed from a 2^2 factorial design, where the two factors of interest result from the preliminary experimentation described in the previous section. The number of patients originally admitted (set P_a) and the number of patients seeking admission (set

P_n) into the unit significantly affected bed assignments and therefore are the factors to be blocked in this experimentation. The levels chosen for the number of patients originally admitted into the unit are minimally 20 and a maximum of 24, which equivalently results in having 6 and 2 free beds, respectively. The levels for the number of patients seeking admission were chosen to be 2 as a minimum and 4 as a maximum level.

For each problem instance, the solution of applying the future knowledge model to data sets for planning horizons longer than 1 day were compared with solutions generated from planning with only day 1. The planning period extended to only 5 periods since that is a traditional work week in our partner hospital. The choice of smoothing weights selected for each planning period is shown in Table 10. The weights were chosen to depict the growing uncertainty of information surrounding future periods. Hence, period 1 is always weighted the highest and all weights following period 1 are assigned monotonically decreasing values.

Table 10: Weights Assigned to each Period in the Planning Horizon for Experimentation Done for 1-5 Periods

Number of Periods	Weights Assigned
1	1
2	0.8, 0.2
3	0.7, 0.2, 0.1
4	0.6, 0.2, 0.15, 0.05
5	0.5, 0.2, 0.15, 0.1, 0.05

The following information was collected from each of the 100 problem instances: the objective function value of the resulting assignment, the number of internal movements per period, the percent change in the objective function value for periods with movements, and the unit's utilization. This information was used to compare schedules using future information with the schedule made considering available information for only day 1. The tables presented are grouped by experimental scenario. When reviewing these tables, it is important to note during which periods internal movements occurred and during which period the unit reached 100% utilization. Furthermore, the percent increase was only calculated for periods where a patient(s) was accepted into the unit after period 1 to help evaluate the value of including future information into planning for today. Perhaps a unit administrator would think a 2% increase in the objective function translates into a tangible increase in care therefore he or she would attribute value in considering future information for planning. This can apply conversely as well. Two percent may be perceived as too trivial to consider future information.

Tables 11-15: Results for 20 PA, 2 PN

Table 11

	Num. of Periods in the Planning Horizon	Objective Value	Number of Internal Movements per Period					Percent Increase	Utilization of the Unit
data 45			**1**	**2**	**3**	**4**	**5**		
	1	92.41	2					NA	92%
	2	94.61	2	2				2.33	100%
	3	95.29	2	4					100%
	4	96.01	2	7					100%
	5	96.54	3	5					100%

Table 12

	Num. of Periods in the Planning Horizon	Objective Value	Number of Internal Movements per Period					Percent Increase	Utilization of the Unit
data 38			**1**	**2**	**3**	**4**	**5**		
	1	109.50	1					NA	92%
	2	111.88	1	2				2.12	100%
	3	112.53	1	3					100%
	4	113.27	1	3					100%
	5	113.76	1	3					100%

Table 13

	Num. of Periods in the Planning Horizon	Objective Value	Number of Internal Movements per Period					Percent Increase	Utilization of the Unit
data 49			1	2	3	4	5		
	1	114.25	2					NA	92%
	2	116.71	3	1				2.10	100%
	3	117.61	2	4					100%
	4	118.40	2	4					100%
	5	118.95	2	4					100%

Table 14

	Num. of Periods in the Planning Horizon	Objective Value	Number of Internal Movements per Period					Percent Increase	Utilization of the Unit
data 27			1	2	3	4	5		
	1	125.68	1					NA	92%
	2	127.81	1	4				1.66	100%
	3	128.43	1	4					100%
	4	128.97	1	4					100%
	5	129.32	1	4					100%

Table 15

	Num. of Periods in the Planning Horizon	Objective Value	Number of Internal Movements per Period					Percent Increase	Utilization of the Unit
data 24			1	2	3	4	5		
	1	136.66	1					NA	92%
	2	138.96	1	1				1.65	100%
	3	139.63	1	3					100%
	4	140.21	1	3					100%
	5	141.73	1	3					100%

Tables16-20: Results for 20 PA, 4 PN

Table 16

	Num. of Periods in the Planning Horizon	Objective Value	Number of Internal Movements per Period					Percent Increase	Utilization of the Unit
data 17			**1**	**2**	**3**	**4**	**5**		
	1	119.22	2					NA	92%
	2	116.35	3	2				-2.47	100%
	3	116.71	3	2					100%
	4	117.13	3	2					100%
	5	117.47	3	2					100%

Table 17

	Num. of Periods in the Planning Horizon	Objective Value	Number of Internal Movements per Period					Percent Increase	Utilization of the Unit
data 18			**1**	**2**	**3**	**4**	**5**		
	1	135.18	3					NA	92%
	2	136.46	3	1				0.94	100%
	3	136.84	3	4					100%
	4	137.28	3	4					100%
	5	137.60	3	4					100%

Table 18

	Num. of Periods in the Planning Horizon	Objective Value	Number of Internal Movements per Period					Percent Increase	Utilization of the Unit
data 37			**1**	**2**	**3**	**4**	**5**		
	1	121.00	3					NA	92%
	2	122.73	4	4				1.41	100%
	3	122.81	4	4					100%
	4	123.27	4	5					100%
	5	123.63	4	3					100%

Table 19

	Num. of Periods in the Planning Horizon	Objective Value	Number of Internal Movements per Period					Percent Increase	Utilization of the Unit
data 26			1	2	3	4	5		
	1	126.52	3					NA	88%
	2	127.86	1	6				1.05	100%
	3	128.38	1	6					100%
	4	128.99	1	5					100%
	5	129.40	1	6					100%

Table 20

	Num. of Periods in the Planning Horizon	Objective Value	Number of Internal Movements per Period					Percent Increase	Utilization of the Unit
data 39			1	2	3	4	5		
	1	99.17	2					NA	92%
	2	100.28	3					1.10	100%
	3	100.61	3						100%
	4	100.95	1	3					100%
	5	101.24	2	3					100%

Tables21-25: Results for 24 PA, 2 PN

Table 21

	Num. of Periods in the Planning Horizon	Objective Value	Number of Internal Movements per Period					Percent Increase	Utilization of the Unit
data 48			1	2	3	4	5		
	1	125.46	2					NA	100%
	2	125.54	2						100%
	3	125.60	2						100%
	4	125.66	2						100%
	5	125.73	2						100%

Table 22

data 47	Num. of Periods in the Planning Horizon	Objective Value	Number of Internal Movements per Period					Percent Increase	Utilization of the Unit
			1	2	3	4	5		
	1	137.11	3					NA	100%
	2	137.29	3						100%
	3	137.38	3						100%
	4	137.47	3						100%
	5	137.55	3						100%

Table 23

data 46	Num. of Periods in the Planning Horizon	Objective Value	Number of Internal Movements per Period					Percent Increase	Utilization of the Unit
			1	2	3	4	5		
	1	148.14	4					NA	100%
	2	148.46	4						100%
	3	148.53	4						100%
	4	148.60	4						100%
	5	148.66	4						100%

Table 24

data 40	Num. of Periods in the Planning Horizon	Objective Value	Number of Internal Movements per Period					Percent Increase	Utilization of the Unit
			1	2	3	4	5		
	1	128.54	3					NA	100%
	2	128.83	3						100%
	3	128.98	3						100%
	4	129.12	3						100%
	5	129.27	3						100%

Table 25

	Num. of Periods in the Planning Horizon	Objective Value	Number of Internal Movements per Period					Percent Increase	Utilization of the Unit
			1	2	3	4	5		
data 44	1	128.80	1					NA	96%
	2	129.38	1	2				0.45	100%
	3	129.61	1	2					100%
	4	129.80	1	2					100%
	5	129.94	1	2					100%

Tables26-30: Results for 24 PA, 4 PN

Table 26

	Num. of Periods in the Planning Horizon	Objective Value	Number of Internal Movements per Period					Percent Increase	Utilization of the Unit
			1	2	3	4	5		
data 35	1	158.58	2					NA	100%
	2	158.67	2						100%
	3	158.71	2						100%
	4	158.75	2						100%
	5	158.79	2						100%

Table 27

	Num. of Periods in the Planning Horizon	Objective Value	Number of Internal Movements					Percent Increase	Utilization of the Unit
			1	2	3	4	5		
data 2	1	81.03	2					NA	100%
	2	81.22	2						100%
	3	81.32	2						100%
	4	81.42	2						100%
	5	81.51	2						100%

Table 28

data 3	Num. of Periods in the Planning Horizon	Objective Value	Number of Internal Movements					Percent Increase	Utilization of the Unit
			1	**2**	**3**	**4**	**5**		
	1	84.1	2					NA	100%
	2	84.28	2						100%
	3	84.37	2						100%
	4	84.46	2						100%
	5	84.55	2						100%

Table 29

data 4	Num. of Periods in the Planning Horizon	Objective Value	Number of Internal Movements					Percent Increase	Utilization of the Unit
			1	**2**	**3**	**4**	**5**		
	1	111.47	2					NA	100%
	2	111.57	2						100%
	3	111.63	2						100%
	4	111.68	2						100%
	5	111.73	2						100%

Table 30

data 5	Num. of Periods in the Planning Horizon	Objective Value	Number of Internal Movements					Percent Increase	Utilization of the Unit
			1	**2**	**3**	**4**	**5**		
	1	96.521	2					NA	100%
	2	96.6168	2						100%
	3	96.6647	2						100%
	4	96.7126	2						100%
	5	96.7605	2						100%

The above experiments indicate that including future knowledge did not affect the number of internal movements performed during period 1. Only in one data set (26) did the number of internal movements reduce after period 1 once future knowledge was considered. For the remaining 19 problem instances the number of internal movements remained the same as occurred in the solution for period 1. Furthermore, if there are not many beds available during period 1, as for scenarios with high number of admitted patients (i.e., 2 free beds) including future information seems irrelevant since the unit reaches full utilization during period 1. Therefore, our experiments suggest that it is most advantageous for the type of unit modeled in this work to use the single-period model as needed to support bed assignment problems.

However, the future knowledge model may be useful for other areas of the hospital where admission into a hospital unit will be more certain for future patients, especially for elective procedures. For example, an adaptation of this model might be useful on a surgical floor where a bed may be scheduled weeks in advance. Of course, some of the model may need to be adjusted to meet the needs for that type of patient. Most likely the isolation requirements would be defined differently to represent patient needs (e.g. bariatric, dialysis, HIV, etc) instead of isolations. In this example, the likelihood parameter could describe the likelihood a patient will require a bed on that floor due to a scheduled surgery on a given day.

Moreover, considering longer planning horizons for bed assignments may be more relevant if the objective function also captures the financial incentives of admitting and discharging patients. Our partner hospital for this study did not disclose any financial information. It is common in hospitals to charge higher costs for admitting and discharging patients than the cost of care during hospitalization, as a consequence, such behavior could dramatically influence the results of the analysis of this section.

Additionally, this model could be applied using a rolling horizon to facilitate bed assignment planning. One glaring limitation of this approach is the lack of updated patient information into the model after period 1. By updating the sets of patients and patient characteristics, the model may behave differently. However, the approach used for data generation in this study does not facilitate changes to the data sets after period 1. Specifically, this study would need criteria to update criticality values daily as well as determine the arrival rate into the ICU.

5. CONCLUSIONS AND FUTURE WORK RECOMMENDATIONS

Through the use of mathematical programming and simulation, this thesis proposes a methodology to facilitate the implementation of isolation requirements in bed patient placement under stochastic inpatient demand. The proposed single-period mathematical optimization model can help unit managers mitigate the impact of internal movements on a day-to-day basis, while maximizing unit utilization. By integrating the proposed optimization model with a Monte Carlo simulation experiment, we have shown that having more double-bed rooms in a hospital unit can be counterproductive, since it increases the expected number of internal movements much faster than the expected capacity increase. Additionally, we have shown that increasing a hospital unit capacity through multi-occupancy rooms will amplify the variability of the number of internal movements and admissions. To cope with the effect of such variability the use of simple safety stock modeling can help the unit to provide a desired service level. Additionally we have shown for the scenarios considered in this study that it is not worth considering likely and uncertain demand information for future days into the current bed assignment planning.

Extensions to the proposed study must consider the effect of complying with isolation requirements across multiple units. Additionally, the proposed methodology could be adapted to facilitate multi-criteria analysis to determine the unit configuration that provides the best trade-off in terms of hospital capacity, revenue, and quality of care. Additionally, more work should be done to generalize the conclusion made from the future knowledge experimentation. The sample of patient scenarios should be broadened to include different bedroom configurations and more randomized patients in sets P_a and P_n.

REFERENCES

[1] E. R. Sherman, K. H. Heydon, K. H. St John, E. Teszner, S. L. Rettig, S. K. Alexander, T. Z. Zaoutis, and S. E. Coffin, “Administrative Data Fail to Accurately Identify Cases of Healthcare-Associated Infection.,” *Infection control and Hospital Epidemiology : the Official Journal of the Society of Hospital Epidemiologists of America*, vol. 27, no. 4, pp. 332–337, April 2006.

[2] J. P. Burke, “Infection Control — A Problem for Patient Safety,” *New England Journal of Medicine*, vol. 348, no. 7, pp. 651–656, February 2003.

[3] “Health Care-Associated Infections Declined in 2010,” *Centers for Disease Control and Prevention*, 2011. [Online]. Available: http://www.cdc.gov/media/releases/2011/p1019_healthcare_infections.html. [Accessed: 16-Jan-2012].

[4] “Clean, Safe Care: Reducing Infections and Saving Lives,” *Department of Health*, 2008. [Online]. Available: http://webarchive.nationalarchives.gov.uk/+/http://www.dh.gov.uk/en/Publicationsandstatistics/Publications/PublicationsPolicyAndGuidance/DH_081650. [Accessed: 30-Jan-2012].

[5] C. A. Umscheid, R. K. Agarwal, and P. J. Brennan, “Updating the Guideline Development Methodology of the Healthcare Infection Control Practices Advisory Committee (HICPAC).,” *American Journal of Infection Control*, vol. 38, no. 4, pp. 264–273, May 2010.

[6] “HHS Action Plan to Prevent Healthcare-Associated Infections: AMBULATORY SURGICAL CENTERS,” *Office of Public Health Science*. [Online]. Available: http://www.hhs.gov/ash/initiatives/hai/tier2_ambulatory.html. [Accessed: 16-Jan-2012].

[7] “Council Recommendation of 9 June 2009 on Patient Safety, Including the Prevention and Control of Healthcare Associated Infections,” *The Council of the European Union*, 2009. [Online]. Available: http://ec.europa.eu/health/patient_safety/docs/council_2009_en.pdf . [Accessed: 05-Mar-2012].

[8] “Practical Guidelines for Infection Control in Health Care Facilities,” *World Health Organization*, 2003. [Online]. Available: http://whqlibdoc.who.int/wpro/2003/a82694.pdf . [Accessed: 05-Mar-2012].

[9] J.S. Garner., “Guidelines for Isolation Precautions in Hospital,” *Infectious Control Hospital Epidemiology,* vol. 17, no. 1, pp. 53-80, January 1996.

[10] D. M. Shlaes, D. N. Gerding, J. F. John, W. A. Craig, D. L. Bornstein, R. A. Duncan, M. R. Eckman, W. E. Farrer, W. H. Greene, V. Lorian, S. Levy, J. E. McGowan, S. M. Paul, J. Ruskin, F. C. Tenover, and C. Watanakunakorn, “Society for Healthcare Epidemiology

of America and Infectious Diseases Society of America Joint Committee on the Prevention of Antimicrobial Resistance: guidelines for the prevention of antimicrobial resistance in hospitals.," *Clinical Infectious Diseases : an Official Publication of the Infectious Diseases Society of America*, vol. 25, no. 3, pp. 584–599, September 1997.

[11] D. Gupta and B. Denton, "Appointment Scheduling in Health Care: Challenges and Opportunities," *IIE Transactions*, vol. 40, no. 9, pp. 800–819, July 2008.

[12] E. S. Shenoy, R. P. Walensky, H. Lee, B. Orcutt, and D. C. Hooper, "Resource Burden Associated with Contact Precautions for Methicillin-Resistant Staphylococcus aureus and Vancomycin-Resistant Enterococcus: The Patient Access Managers' Perspective," *Infection Control and Hospital Epidemiology*, vol. 33, no. 8, June 2012.

[13] S. B. Saundra and K. Lipe, "Critical Thinking in Nursing: A Cognitive Skills Workbook", Illustrated. *Lippincott Williams & Wilkins*, 2004 , p. 337

[14] G. Mageshwari and E. G. M. Kanaga, "Literature Review on Patient Scheduling Techniques," vol. 4, no. 03, pp. 397–401, March 2012.

[15] I. B. Vermeulen, S. M. Bohte, S. G. Elkhuizen, H. Lameris, P. J. M. Bakker, and H. La Poutré, "Adaptive Resource Allocation for Efficient Patient Scheduling.," *Artificial Intelligence in Medicine*, vol. 46, no. 1, pp. 67–80, May 2009.

[16] B. Cardoen, E. Demeulemeester, and J. Beliën, "Operating room planning and scheduling: A Literature Review," *European Journal of Operational Research*, vol. 201, no. 3, pp. 921–932, March 2010.

[17] M. Vanhoucke and B. Maenhout, "On the Characterization and Generation of Nurse Scheduling Problem Instances," *European Journal of Operational Research*, vol. 196, no. 2, pp. 457–467, July 2009.

[18] U. Aickelin and K. A. Dowsland, "An Indirect Genetic Algorithm for a Nurse-Scheduling Problem," *Computers & Operations Research*, vol. 31, no. 5, pp. 761–778, April 2004.

[19] P. F. Clerkin D, Fos PJ, "A Decision Support System for Hospital Bed Assignment," *Hospital Health Service Administration*, vol. 40, no. 3, pp. 386–400, Fall 1995.

[20] F. Guerriero and R. Guido, "Operational Research in the Management of the Operating Theatre: a Survey.," *Health Care Management Science*, vol. 14, no. 1, pp. 89–114, March 2011.

[21] D. Conforti, F. Guerriero, R. Guido, M. M. Cerinic, and M. L. Conforti, "An Optimal Decision Making Model for Supporting Week Hospital Management.," *Health Care Management Science*, vol. 14, no. 1, pp. 74–88, March 2011.

[22] P. Demeester, W. Souffriau, P. De Causmaecker, and G. Vanden Berghe, “A Hybrid Tabu Search Algorithm for Automatically Assigning Patients to Beds.,” *Artificial Intelligence in Medicine*, vol. 48, no. 1, pp. 61–70, January 2010.

[23] S. Ceschia and A. Schaerf, “Local Search and Lower Bounds for the Patient Admission Scheduling Problem,” *Computers & Operations Research*, vol. 38, no. 10, pp. 1452–1463, October 2011.

[24] P. R. Harper and A. K. Shahani, “Modeling for the Planning and Management of Bed Capacities in Hospitals,” *The Journal of the Operational Research Society*, vol. 53, no. 1, pp. 11–18, January 2002.

[25] M. B. Dumas, “Simulation Modeling for Hospital Bed Planning ,” *SIMULATION* , vol. 43 , no. 2 , pp. 69–78, August 1984.

[26] G. Vassilacopoulos, “A Simulation Model for Bed Allocation to Hospital Inpatient Departments ,” *SIMULATION* , vol. 45 , no. 5 , pp. 233–241, November 1985.

[27] R. K. Khare, E. S. Powell, G. Reinhardt, and M. Lucenti, “Adding More Beds to the Emergency Department or Reducing Admitted Patient Boarding Times: Which Has a More Significant Influence on Emergency Department Congestion?,” *Annals of Emergency Medicine*, vol. 53, no. 5, pp. 575–585, May 2009.

[28] S. V. Williams, “How Many Intensive Care Beds are Enough?,” *Critical Care Medicine.*, vol. 11, no. 6, pp. 412–416, June 1983.

[29] A. Shmueli, C. L. Sprung, and E. H. Kaplan, “Optimizing Admissions to an Intensive Care Unit.,” *Health Care Management Science*, vol. 6, no. 3, pp. 131–136, August 2003.

[30] J. M. Nguyen, P. Six, D. Antonioli, P. Glemain, G. Potel, P. Lombrail, and P. Le Beux, “A Simple Method to Optimize Hospital Beds Capacity.,” *International Journal of Medical Informatics*, vol. 74, no. 1, pp. 39–49, January 2005.

[31] Y. Gong, “Multi-Objective Comprehensive Learning Particle Swarm Optimization with a Binary Search-Based Representation Scheme for Bed Allocation Problem in General Hospital,” *Systems Man and Cybernetics (SMC)*, 2010 IEEE Internal Conference Proceeding, pp. 1083–1088, October 2010.

[32] A. E. Boardman and D. Forbes, “A Benefit-Cost Analysis of Private and Semi-Private Hospital Rooms,” *Journal of Benefit-Cost Analysis*, vol. 2, no. 1, January 2011.

[33] H. Chaudhury, A. Mahmood, and M. Valente, “Nurses’ Perception of Single-Occupancy Versus Multioccupancy Rooms in Acute Care Environments: An Exploratory Comparative Assessment.,” *Applied Nursing Research : ANR*, vol. 19, no. 3, pp. 118–125, August 2006.

[34] R. F. Love and L. Yerex, “An Application of a Facilities Location Model in the Prestressed Concrete Industry,” *Interfaces*, vol. 6, no. 4, pp. 45–49, August 1976.

[35] A. W. Ong, L.A. Omert, D. Vido, B. M. Goodman, J. Protetch, A. Rodriguez, and E. Jeremitsky, “Characteristics and Outcomes of Trauma Patients with ICU Lengths of Stay 30 days and Greater: A Seven-Year Retrospective Study.,” *Critical Care (London, England)*, vol. 13, no. 5, p. R154, January 2009.

APPENDICES

Appendix A: Single-Period Math Programming Model

```
#sets
set PA;                                          # subset of patients admitted in the unit
set PN;                                          # subset of incoming patients requiring admission in  the unit
set P:= PA union PN ;                            # set of all patients in the system
set ISOLATION;                                   # set of isolation needs
set GENDER;                                      # set of genders
 set R;                                          # set of available rooms

#parameters
param B {j in R};                                # number of beds available in each room j in R
param G {i in P};                                # gender of each patient i in P
param I {i in P};                                # isolation requirement of patient i in P
param c {i in P};                                # relative criticality of patient i in P compared
                                                   with all other patients
param y{i in P, j in R}binary, default 0;        # binary parameter that is 1 if patient i was
                                                   in room j the day before, and 0 o.w.
param flag {i in P,j in R} binary, default 0;    # binary parameter that is 1 if patient i cannot be moved from
                                                   current bed assignment room, and 0 o.w.

#variables
var X {i in P, j in R} binary;                          # binary variable: 1 if patient i is moved to room
                                                          j, and 0 o.w.
var delta {g in GENDER, j in R: B[j] > 1} binary;       # binary variable : 1 if there is at least one
                                                           patient with gender g in room j, and 0 o.w.
var gamma {i in ISOLATION, j in R: B[j] > 1} binary;    # binary variable: 1 if there is at least one patient
                                                          with isolation i in room j, and 0 o.w.

#variables for linearization
var splus{i in P, j in R} >=0;
var sminus{i in P, j in R} >=0;

maximize Goal:sum{i  in P, j in R: j <>0 and j<>99} c[i]*X[i,j] - sum{p in P, q in R: q<>0 and
            q<>99}(1/c[p])*(splus[p,q]+sminus[p,q]);

subject to const0 {i in P, j in R: j <> 0 and j <> 99}:X[i,j]-y[i,j] = splus[i,j]-sminus[i,j];
subject to const1{i in P}: sum{j in R} X[i,j]=1;
subject to const2 {i in P, j in R: j <> 0 and j <> 99 and B[j] <>1}: X[i,j] <= delta[G[i],j];
subject to const3 {i in P, j in R: j <> 0 and j<> 99 and B[j] <>1}: X[i,j] <= gamma[I[i],j];
subject to const4{j in R: j <> 0 and j <> 99 and B[j] <>1}: sum {g in GENDER} delta[g,j] <=1;
subject to const5 {j in R: j <> 0 and j <> 99 and B[j] <>1}: sum {i in ISOLATION} gamma[i,j] <=1;
subject to const6 {j in R}: sum {i in P} X[i,j] <= B[j];
subject to const7 {i in PA,j in R: j=0}: X[i,j] = 0;
subject to const8 {i in PN,j in R: j=99 }: X[i,j] =0;
subject to const9 {i in P, j1 in R: B[j1]=1 and y[i, j1]=1}:sum {j2 in R: B[j2] = 1 and j1 <>j2} X[i,j2]=0;
subject to const10 {i in P, j in R: j <> 0 and j <> 99 }: X[i,j] >= flag[i,j]*y[i,j];

end;
```

Appendix B: Code Development for Experimentation

This appendix describes in full detail the rationale behind the code developed to run the simulation for the single-period model experimentation described in section 4.2.1.

1. Class Descriptions

Two classes were defined for this code to describe specific structures from the model. A patient class was defined that included the following identifiers: ID, isolation, criticality, lengthstay, dischargeday, room, and gender. The ID was the unique identifier for each patient. “Isolation” was the isolation requirement assigned to each patient. “Criticality” was the critical value assigned to each patient. “Lengthstay” was the length of stay for each patient if they were assigned a bed within the unit. “Dischargeday” is the day the patient would leave the unit if the patient was staying in the unit. This day was calculated by adding the current day in the simulation and the “lengthstay” value if the patient was admitted to the unit. The room was the current room location of the patient. Lastly, the gender described if the patient was male or female. The second class was defined for the rooms described in the model. Each room had a “name” which was a numeric identifier and a “numbeds.” The “numbeds” is the capacity of the room. For the rooms in the unit, the “numbeds” were one or two whereas for the “Triage” and “Discharge” areas that number was much larger.

2. Function Descriptions

Eight functions were created to run the simulation. The first function (ReadExperimentalCondition) reads a text file that lists the experimental conditions for the set of experiments that will be run. The text file includes the “endtime” of the simulation. The “endtime” refers to the number of days which the simulation will run. For example if the “endtime” value is 365 then this infers that the program is running for one year. Next, the file contains the number of repetitions for each experimental day. Therefore, the total simulated days would be calculated by multiplying the “endtime” by the number of repetitions. Next, the file contains the type of experimental arrival rate that will be used. In this experiment the historical arrival rate, the historical arrival rate plus one additional patient per day, and the historical arrival rate plus two additional patients per day were used. The text file would indicate which of these three arrival rates would be used for the particular experiment being run. Lastly, the file contained the number of double bedrooms for the particular set of experiments. As described earlier, the goal of this experimentation was to determine how many double bedrooms should be in the unit to meet the demand of the historical arrival rate.

Following this function, the code proceeded to read the initial conditions of the unit. This function (ReadInitConditions) opened a text file that contained patient IDs, length of stays, isolation requirements, criticalities, rooms, and genders for those patients who would be in the unit at the beginning of the experiment. Furthermore, from the length of stay, the day of discharge from the unit was calculated in this function. All of this patient information was stored in a list construction. Immediately following this function, a function (DischargePatients) was executed to determine if any patient should be discharged from the unit. This function compared the current time(experimental day) of the experiment with the value listed for discharge day for each patient. If these values matched, the patient would be erased from the unit list.

Once the configuration of the unit was finalized, the next function (NewPatients) determined the number of patients who would be seeking admission into the unit and their respective length of stays. To determine the number of patients who would seek admission for oa given experimental day, the code would randomly generate a number between 1 and 50. This number would indicate to the code which file of arrival values it should open to read how many patients would be seeking admission into the unit. Next, the code would open the file and read in the first value in the file. This value is interpreted as the number of patients seeking admission for the current day. Next, the code repeated this random number generation to determine which file to open to for the length of stay data. Then for each patient seeking admission, the length of stay value was read from the file and stored as a vector. This method was used to determine the number of patients seeking admission and the length of stay values so that these attributes could follow the historical distributions of the respiratory unit in RGH. Merely using a RAND function would not achieve this.

Following this function, the patients seeking admission are assigned their remaining attributes. In this function (PopulateNewPatients), each patient seeking admission was given an ID, length of stay, a discharge day, a critical value, an isolation requirement, a gender, and a room.

Once the data for the experiment is compiled, the data file can be generated using the “GenerateGLPKfile” function. This function writes a data file according to the syntax of the Math programming language MATHPROG, which is used by GLPK. Next, the code calls the stand alone solver in GLPK to run the outcome of that experimental day (using SolveGLPK function). The solver will stop once the solver reaches its maximum runtime which was defined in the code. The solution is exported to a text file which is read in the final function

(ReadSolution). In the last function, the output file is opened so that the code can read the patient information. Most importantly this file indicates the new configuration of the unit which is indicated by the room attribute of each patient. Furthermore, this function adjusts the discharge day since one experimental day would have elapsed.

3. Main Function

The overall code executes through the eight functions described above in that order then enters into a loop to proceed through the entire experiment. For each repetition the code will perform the following functions until the "endtime" is reached:

- PopulateNewPatients
- GenerateGLPKfile
- SolveGLPK
- ReadSolution
- DischargePatients

If the experiment is designed to run more than one repetition then the current day will reset to zero and the initial conditions of the unit will be reinstated. After the unit is essentially reset, the loop described above is repeated again until the "endtime" is reached. Once the entire experiment is completed then the text file which has been storing all experimental results is renamed for the particular experiment that was run.

Appendix C: Simulation C++ Code

```
// LINUX VERSION
// Dec 17, 2011

#include <iostream>
#include <list>
#include <algorithm>
#include <functional>
#include <iterator>
#include <cstdlib>
#include <vector>
#include <fstream>
extern "C"{
       #include "glpk.h"
}
#include <stdio.h>
#include <stdlib.h>
#include <string>
#include <sstream>
#include <time.h>

// used here for convenience, use judiciously in real programs.
using namespace std;

// definition patient class
class patient {
  public:
    int Id;
    int Isolation;
    int Criticality;
    int LenghtStay;
    int DischargeDay;
    int Room;
    int Gender;
};

bool GetDischargeDay(patient pat, int clktime)
{
       return(pat.DischargeDay<= clktime);
}

class room{
public:
       int name;
       int numbeds;
};
// function declarations

// reads initial experimental conditions (strings: test, numb of dbrms, endtime,
repetition, numtest, numdbrms)
void ReadExperimentCondition(vector<string> &, vector<string> &, int&, int&, int&, int&);

// reads number of new patients per day and their respective length of stay
void NewPatients(vector<int>&, vector<int> &, vector<int> &, int&, string & );

// for each new patient object it populates its features
```

```
void PopulateNewPatients(list<patient> &, vector<int>&, vector<int>&, vector<int>&, int,
int& );

// reads the initial list of patients and conditions in the 5400 unit
void ReadInitConditions(list<patient> &, int);

// removes from the unit, patients who have been discharged
void DischargePatients(list<patient> &, int);

// generates the input file for glpsol
void GenerateGLPKfile(list<patient>&, list<room>&, vector<int>&, int&, int&, int&);

// calls glpsol to solve problem instance
int SolveGLPK(list<patient> &, list<room>&, int&);

// uploads glpsol solution into the unit
void ReadSolution(list<patient> &, int );

// uploads initial room conditions at the 5400 unit
void ReadRoomConditions(list<room>& , int & , int &, int &, string & );


void Welcoming(string&, string&, int&, int&);

// this function declaration is necessary for controlling the mip_gap tolerance in glpk.
The function is called in SolveGLPK function
void cb_func(glp_tree *tree, void *info)
{       if(glp_ios_reason(tree)==GLP_IBINGO)
        { if(glp_ios_mip_gap(tree)<=0.20)
                glp_ios_terminate(tree);
        }
        return;
}
// global constants. number of critical levels and isolation levels used to clasify
patients
int Maxcriticality=10;
int Maxisolation=6;


int main()
{
        //initial random seed
        srand(unsigned(time(NULL)));

        int repetition=0;

        // patients in the 5400 unit
        list<patient> unit;

        // rooms in the 5400 unit
        list<room> u5400;

        //vector with the daily number of new inpatients in the unit
        vector<int> inpatients;

        //vector with the length of stay of the new inpatients
        vector<int> lengthinpatients;
```

```
//vector with the position lengthinpatients corresponding to a new inpatients in
the unit
vector<int> indexlength;

int clktime=0;       // execution time
int endtime=0;       // number of days for each replication ( 1823:= approx. 5
                        years of information)
int numnewpat=0;     //temp integer
int intmovement=0;  //temp integer
int nsinglerooms=0;
int ndoublerooms=0;
int totalrooms=0;
int idctr=0;
//string fldrpath= "/home/rpmeie/CRGHPatFlow/R1"; // in windows:
"C:\\Users\\rpmeie\\Documents\\Visual Studio 2008\\Projects\\RGH5400\\RGH5400\\" ;
string strnumdbrms;
string expname;
string smryname;

//initial welcoming message in prompt

int NUMTESTS=0;
int NUMDBBRMS=0;
vector<string> tests;
vector<string> strarraydbrms;

ReadExperimentCondition(tests, strarraydbrms, endtime, repetition, NUMTESTS,
NUMDBBRMS);

for(int i=0; i<NUMTESTS; i++)
{
       expname=tests[i];

       for(int k=0; k <NUMDBBRMS; k++) // change 5 when instruction in green
                                           before loop is activated
       {
              clktime=0;
              strnumdbrms=strarraydbrms[k];

              ofstream outresults;
              outresults.open("smry.txt", ios::out);
              outresults<<"NUMITERATIONS\t"<<
           "CLKTIME\t"<<"NUM_INT_MOVS\t"<<"NUMPATIENTS\t"<<"AVG_CRIT\t"<<"GOAL\n";
              outresults.close();

              ReadRoomConditions(u5400,nsinglerooms,ndoublerooms,totalrooms,
              strnumdbrms);

              ReadInitConditions(unit, clktime);

              DischargePatients(unit, clktime);

              NewPatients(inpatients,lengthinpatients,indexlength,numnewpat,
              expname);

              idctr=unit.size();

              for(int j=1; j<=repetition; j++)
```

```
                        {
                                while(clktime< endtime)
                                {
                                        cout<<"Running replication: "<<j<<", day:
                                        "<<clktime<<"...\n";
                                        PopulateNewPatients(unit, inpatients, lengthinpatients,
                                        indexlength, clktime, idctr);
                                        GenerateGLPKfile(unit, u5400, inpatients, clktime,
                                        Maxisolation, j);
                                        intmovement=SolveGLPK(unit, u5400, clktime);
                                        ReadSolution(unit, clktime);
                                        clktime++;
                                        DischargePatients(unit,clktime);
                                }
                                clktime=0;
                                ReadInitConditions(unit, clktime);
                        }
                        smryname+=expname;
                        smryname+="_db_";
                        smryname+=strnumdbrms;
                        smryname+=".txt";
                        rename("smry.txt",smryname.c_str());
                        smryname="";
                }

        }

        return 1;
}

// function definition:

void ReadExperimentCondition(vector<string> & tests, vector<string> & strarraydbrms, int&
endtime, int& repetition, int& numtest, int& numdbrms)
{
        vector<string> test0;
        vector<string> strarraydbrms0;
        string tmpstr;

        ifstream indata;
        indata.open("./ExperimentalConditions.txt");

        if(!indata)
        {
                cerr<<"Error: file could not be opened"<<endl;
                exit(1);
        }

        while(!indata.eof())
        {
                indata >> endtime;
                indata >> repetition;
                indata >> numtest;
                indata >> numdbrms;

                for(int i=1; i<= numtest; i++)
                {
                        indata >> tmpstr;
                        test0.push_back(tmpstr);
                }

                for(int j=1; j<=numdbrms; j++)
                {
                        indata >> tmpstr;
                        strarraydbrms0.push_back(tmpstr);
                }
        }

        tests=test0;
        strarraydbrms=strarraydbrms0;
        indata.close();
}




void ReadInitConditions(list<patient> & unit, int currentime)
{
        list<patient> unittemp;
        patient CurrentPatient;
        ifstream indata; // indata is like cin

        indata.open("../PATINIT.txt"); // opens the file
        if(!indata) { // file couldn't be opened
        cerr << "Error: file could not be opened" << endl;
        exit(1);
        }

        while (!indata.eof()) { // keep reading until end-of-file
          indata >> CurrentPatient.Id;
                indata >> CurrentPatient.LenghtStay;
                CurrentPatient.DischargeDay=CurrentPatient.LenghtStay+currentime;
                indata >> CurrentPatient.Isolation;
                indata >> CurrentPatient.Criticality;
                indata >> CurrentPatient.Room;
                indata >> CurrentPatient.Gender;

                unittemp.push_back(CurrentPatient);// sets EOF flag if no value found
   }

        unit=unittemp;
   indata.close();

}


void DischargePatients(list<patient> & unit, int currentime)
{
        //unit.erase(remove_if(union.begin(),union.end(),GetDischargeDay()),union.end());
        for(list<patient>::iterator it=unit.begin();it != unit.end();)
        {
                if(GetDischargeDay(*it,currentime))
                {
                        it=unit.erase(it);
                }else
                {
                        it++;
                }
        }

}


void GenerateGLPKfile(list<patient> &unit, list<room> &u5400, vector<int> &inpatients,
int& clktime, int& maxisolation, int& repetition)
{
        int tmpctr=0;
        for(list<patient>::iterator it = unit.begin(); it != unit.end(); ++it)
         {
                        if((*it).Room != 0)
                        {
                                tmpctr++;
                        }
         }

        ofstream outdata;
        outdata.open("glpk_input.dat",ios::out);
        outdata<<"param numPA:="<<tmpctr<<";\n"; //unit.size()-inpatients[clktime]<<";\n";
        outdata<<"param numPN:="<<inpatients[clktime]<<";\n";
        outdata<<"param numISO:="<<maxisolation<<";\n";
        outdata<<"param repetition:="<< repetition<<";\n";
        outdata<<"param clktime:="<<clktime<<";\n";
        outdata<<"set PA:=";

        for(list<patient>::iterator it = unit.begin(); it != unit.end(); ++it)
         {
                        if((*it).Room != 0)
                        {
                                outdata << (*it).Id << " ";
                        }
         }

        outdata<<";\n";

        outdata<<"set PN:=";

        for(list<patient>::iterator it = unit.begin(); it != unit.end(); ++it)
         {
                        if((*it).Room==0)
                        {
                                outdata << (*it).Id << " ";
                        }
         }
        outdata<<";\n";

        outdata<<"set ISOLATION:=";
        for (int i =1; i <= maxisolation; i++)
        {
                outdata<<i<<" ";
        }
        outdata<<";\n";

        outdata<<"set GENDER:= 0 1;\n";
        outdata <<"set R:=";
        for(list<room>::const_iterator it=u5400.begin(); it!=u5400.end(); ++it)
        {
               outdata<<(*it).name<<" ";
        }
        outdata<<";\n";

        outdata<< "param B:=\n";
        for(list<room>::const_iterator it=u5400.begin(); it!=u5400.end(); ++it)
        {
               outdata<<(*it).name<<" "<<(*it).numbeds<<"\n";
        }
        outdata<<";\n";

        outdata<<"param: "<<" G\t"<<"I\t"<<"c\t"<<"flag\t"<<"lstay:=\n";
        for(list<patient>::const_iterator it = unit.begin(); it != unit.end(); ++it)
     {
               outdata << (*it).Id << "\t
"<<(*it).Gender<<"\t"<<(*it).Isolation<<"\t"<<(*it).Criticality<<"\t"<<
0<<"\t"<<(*it).LenghtStay<<"\n";
     }
        outdata<<";\n";

        outdata<<"param y:=\n";
        for(list<patient>::const_iterator it = unit.begin(); it != unit.end(); ++it)
     {
               outdata << (*it).Id << "\t "<<(*it).Room<<"\t"<<1<<"\n";
     }
        outdata<<";\n";
        outdata<<"end;";

        outdata.close();
};
void ReadSolution(list<patient> & unit, int clktime)
{
        list<patient> glpunit;
        patient CurrentPatient;

        int unitsize=unit.size();


        ifstream indata;
        indata.open("gout.dat");
        if(!indata) { // file couldn't be opened
      cerr << "Error: file could not be opened" << endl;
      exit(1);
        }


        for(int i =1; i<=unitsize; i++)
        {
               indata>>CurrentPatient.Id;
               indata>>CurrentPatient.LenghtStay;
               indata>>CurrentPatient.Isolation;
               indata>>CurrentPatient.Criticality;
               indata>>CurrentPatient.Gender;
               indata>>CurrentPatient.Room;
               if(CurrentPatient.Room!=0)
```

```
				{
					CurrentPatient.LenghtStay=CurrentPatient.LenghtStay-1;

					for(list<patient>::iterator it = unit.begin(); it != unit.end();
++it)
					{
						if((*it).Id==CurrentPatient.Id)
						{
							CurrentPatient.DischargeDay=(*it).DischargeDay;
						}
					}

				}else
				{
					CurrentPatient.DischargeDay=CurrentPatient.LenghtStay+clktime;
				}
				glpunit.push_back(CurrentPatient);
		}
		indata.close();
		unit=glpunit;
}

void NewPatients(vector<int> & vnum0, vector<int> & vlen0, vector<int> & vindx0, int&
numarrivals,  string & expname)
{
		vector<int> vnum;
		vector<int> vlen;
		vector<int> vindx;

		ifstream inarrivals;
		ifstream inlength;
		int num;
		int len;
		int indx=0;
		int rndAR=1+rand()%50;
		int rndLS=1+rand()%50;
		string str_rndAR;
		string str_rndLS;
		stringstream stAR_out;
		stringstream stLS_out;
		stAR_out << rndAR;
		stLS_out << rndLS;
		str_rndAR = stAR_out.str();
		str_rndLS = stLS_out.str();


		string str0="../Experiment";
		string str1="/incomingpatient";// windows: "\\incomingpatient";
		string str2="/lengthpatient"; // windows\\lengthpatient";
		string str3=".txt";
		string inpatient_filename;
		string lengthstay_filename;

		if(expname=="A")
		{
			inpatient_filename=str0+expname+str1+str3;
			lengthstay_filename=str0+expname+str2+str3;
		}else{
```

```
		inpatient_filename=str0+expname+str1+str_rndAR+str3;
		lengthstay_filename=str0+expname+str2+str_rndLS+str3;
	}

	inarrivals.open(inpatient_filename.c_str(), ios::in);
	inlength.open(lengthstay_filename.c_str(),ios::in);
	vindx.push_back(0);

	while(!inarrivals.eof())
	{
		inarrivals>>num;
		vnum.push_back(num);
		if(num!=0)
		{
			for(int i=1;i<=num;i++)
			{
				inlength>>len;
				vlen.push_back(len);

			}
		}
		indx=indx+num;
		vindx.push_back(indx);
	}
	inarrivals.close();
	inlength.close();
	numarrivals=vnum.size();
	vnum0=vnum;
	vlen0=vlen;
	vindx0=vindx;

}

void PopulateNewPatients(list<patient> &unit, vector<int> &vnum,  vector<int> &vlen,
vector<int> &vindex, int curtime, int &ctr )
{
	patient NewPat;

	if (vnum[curtime]!=0)
	{

		for(int i=1; i<= vnum[curtime]; i++)
		{   ctr++;
			NewPat.Id=ctr;
			NewPat.LenghtStay=vlen[vindex[curtime]+i-1];
			NewPat.DischargeDay=vlen[vindex[curtime]+i-1]+curtime;
			NewPat.Criticality= 1+ rand()%Maxcriticality;
			NewPat.Isolation=1+rand()%Maxisolation;
			NewPat.Gender=rand()%2;
			NewPat.Room=0; //0:= TRIAGE
			unit.push_back(NewPat);
		}
	}                                                   }



int SolveGLPK(list<patient> & unit, list<room> & u5400, int& clktime){
        //problem object for glpk:see glpk manual for more info
        glp_prob *mip;//defines type of problem
        glp_tran *tran;
        glp_iocp parm;

        int ret;//"return"
        glp_mem_limit(3584);
        mip = glp_create_prob();
        tran = glp_mpl_alloc_wksp();
        ret = glp_mpl_read_model(tran, "RGH_simulation.mod", 1);
        if (ret != 0)
        {
               fprintf(stderr, "Error on translating model\n");
               goto skip;
        }
        ret = glp_mpl_read_data(tran, "glpk_input.dat");

        if (ret != 0)
        {
               fprintf(stderr, "Error on translating data\n");
               goto skip;
        }

        ret = glp_mpl_generate(tran, NULL);
        if (ret != 0)
        {
               fprintf(stderr, "Error on generating model\n");
               goto skip;
        }
        glp_mpl_build_prob(tran, mip);
        glp_simplex(mip, NULL);

        glp_init_iocp(&parm); // for controling MIP_gap
        parm.cb_func=cb_func; // for controling MIP_gap
        glp_intopt(mip, &parm); // for controling MIP_gap
        //glp_intopt(mip, NULL);  // function used in original code instead of the above
        ret = glp_mpl_postsolve(tran, mip, GLP_MIP);
        if (ret != 0)
               fprintf(stderr, "Error on postsolving model\n");
        skip: glp_mpl_free_wksp(tran);
        glp_delete_prob(mip);


        return 0;
}

void ReadRoomConditions(list<room>& u5400, int & numsinglerooms, int &numdoublerooms, int
&numrooms, string& strnumdbrms)
{
        list<room> u54temp;
        room ReadRoom ;
        ifstream inroom;
        string str1="../RoomExpConfig/room_info"; // windows:"RoomExpConfig\\room_info";
        string str2=strnumdbrms;
        string str3=".dat";
        string roominfo_filename= str1 + str2 + str3;
        inroom.open(roominfo_filename.c_str(),ios::in);
        inroom>> numsinglerooms>>numdoublerooms >> numrooms;
        for (int i=1; i<=numrooms; i++)
        {
               inroom>> ReadRoom.name >> ReadRoom.numbeds;
               u54temp.push_back(ReadRoom);
        }

        inroom.close();
        ReadRoom.name=0;
        ReadRoom.numbeds=3000;  // initializing TRIAGE AREA as the first room in the list
        of rooms
        u54temp.push_front(ReadRoom);
        ReadRoom.name=99;
        ReadRoom.numbeds=3000; // initializing DISCHARGE AREA as the last room in the list
        of rooms
        u54temp.push_back(ReadRoom);
        u5400=u54temp;
}

void Welcoming(string& strnumdblrms, string& expname, int& repetition, int& nsdays)
{
        cout<< "+++++++++++++++++++++++++++++++++++++++++++++++++++++++++\n";
        cout<< "+++            RGH 5400 UNIT SIMULATION              +++\n";
        cout<< "+++                                                  +++\n";
        cout<< "+++    Copyright Dr. Ruben Proano, Feb 2011 (RIT)    +++\n";
        cout<< "+++    rpmeie@rit.edu                                +++\n";
        cout<< "+++++++++++++++++++++++++++++++++++++++++++++++++++++++++\n";
        cout<<"\n";
        cout<<" This simulation has been written in C++ and aims to determine\n";
        cout<<" the effect of the number of single and double rooms at the \n";
        cout<<" 5400 unit at RGH.\n";
        cout<<" The optimization engine of this simulation was built on \n";
        cout<<" GLPK optimization software, and it has been used under the\n";
        cout<<" GNU license agreements. \n";
        cout<<"\n";
        cout<<"++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++\n";
        cout<<"PLEASE INPUT THE NUMBER OF REPETITIONS FOR THIS SIMULATION: ";
        cin>> repetition;
        cout<<"\n";
        cout<< "PLEASE INPUT THE NUMBER OF DOUBLE ROOMS IN THIS EXPERIMENT: ";
        cin>> strnumdblrms;
        cout<<"\n";
        cout<<"PLEASE INPUT THE EXPERIMENT NAME (choose from: A, B0, B1, B2): ";
        cin>> expname;
        cout<<"\n";
        cout<<" This experiment will simulate the functioning of the \n";
        cout<<" 5400 unit for batches of "<<nsdays<<" working days. Each\n";
        cout<<" of such batches will be randomly replicated "<< repetition <<" times\n";
        cout<<" Please refer to the readme.txt file for details on the output\n ";
        cout<<" data generated\n";
        cout<<"\n";
        cout<<"+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++\n";
        cout<<" STARTING EXPERIMENTATION \n";
        cout<<"+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++\n";

}
```

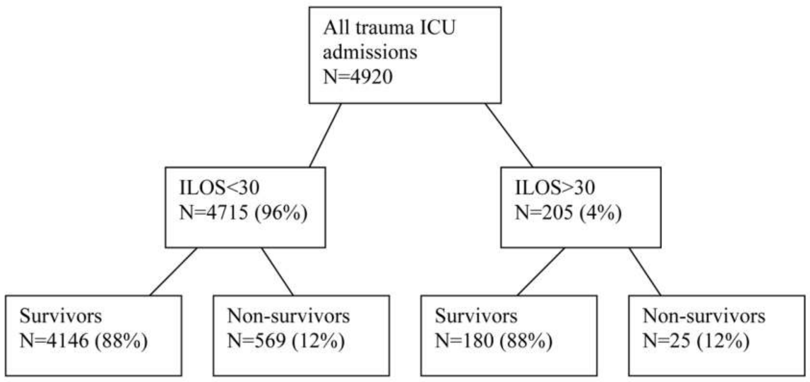
Appendix D: Future Knowledge Math Programming Model

```
#sets
set PA;                                  #subset of patients admitted in the unit
set PN;                                  #subset of incoming patients requiring admission in the unit
set PT;                                  #subset of patient who are in the ICU not seeking admission
                                           during the initial planning period 1
set P:= PA union PN union PT;            #set of all patients in the system
set Q:= PA union PN;                     #set of patients who are admitted in the unit and those patients
                                           requiring admission
set ISOLATION;                           #set of isolation needs
set GENDER;                              #set of genders
set R;                                   #set of available rooms
set H;                                   #the set of periods for the planning horizon

#parameters
param B {j in R};                        # number of beds available in each room j in R
param G {i in P};                        # gender of each patient i in P
param I {i in P};                        # isolation requirement of patient i in P
param c {i in P};                        # current relative criticality of patient i in P compared to all
                                           patients
param w {k in H};                        # the weight assigned with period
param y {i in PA, j in R} binary, default 0;   # binary parameter that is 1 if patient i was in room j the day
                                           before, and 0 o.w.
param p {i in PT};                       # probability that patient i in PT may be discharged from the
                                           ICU in a given period

#variables
var X {i in P, j in R, k in H} binary;                            # binary variable: 1 if patient i is in room j
                                                                    during period k, and 0 o.w.
var delta {g in GENDER, j in R, k in H: B[j] > 1} binary;         # binary variable: 1 if there is at least one
                                                                    patient with gender g in room j in period k,
                                                                    and 0 o.w.
var gamma {i in ISOLATION, j in R,k in H: B[j] > 1} binary;       # binary variable: 1 if there is at least one
                                                                     patient with isolation i in room j in period
                                                                     k, and 0 o.w.

#variables for linearization
var splus{i in P, j in R, k in H} >=0;
var sminus{i in P, j in R, k in H} >=0;

maximize Goal:
sum{k in H:k<>0} w[k]*(sum{i in Q, j in R: j <>0 and j<>99} c[i]*X[i,j,k]
- (1/2)*sum{o in Q, b in R: b <>0 and b<>99}(1/c[o])*(splus[o,b,k]+sminus[o,b,k]))
+sum{k in H:k<>0} w[k]*(sum{i in PT, j in R: j <>0 and j<>99} c[i]*X[i,j,k]
-(1/2)*sum{u in PT, d in R: d <>0 and d<>99}(1/c[u])*(splus[u,d,k]+sminus[u,d,k])*((1-p[u])^(k-1)*p[u]));

subject to const0 {i in P, j in R, k in H: j <> 0 and j <> 99 and k<>0}:X[i,j, k]-X[i,j,k-1] = splus[i,j,k]-sminus[i,j,k];
subject to const1{i in P, k in H}: sum{j in R} X[i,j,k]=1;
subject to const2 {i in P, j in R, k in H: j <> 0 and j<>50 and j <> 99 and B[j] <>1}: X[i,j,k] <= delta[G[i],j,k];
subject to const3 {i in P, j in R, k in H: j <> 0 and j<>50 and j<> 99 and B[j] <>1}: X[i,j,k] <= gamma[I[i],j,k];
subject to const4 {j in R, k in H: j <> 0 and  j<> 50 and j <> 99 and B[j] <>1}: sum {g in GENDER} delta[g,j,k] <=1;
subject to const5 {j in R, k in H: j <> 0 and j<>50 and j <> 99 and B[j] <>1}: sum {i in ISOLATION} gamma[i,j,k] <=1;
subject to const6 {j in R, k in H}: sum {i in P} X[i,j,k] <= B[j];
subject to const7
     {i in P, j1 in R, k in H: B[j1]=1 and k<>0}:sum {j2 in R: B[j2] = 1 and j1 <>j2} X[i,j2,k] <= (1-X[i,j1,(k-1)]);
subject to const8 {i in PA, k in H}: X[i,0,k] = 0;
subject to const9 {i in Q, k in H}: X[i,50,k] = 0;
subject to const10 {i in PT }: sum {j in R, k in 0..1: j<>50} X[i,j,k] =0;
subject to const11 {i in PN}: X[i,0,0] = 1;
subject to const12{ i in PA, j in R}: X[i,j,0]=y[i,j];
subject to const13 {i in PT, k in H}: X[i,0,k] =0;
subject to const14{i in PT, k in H:k<>0}:  X[i,50,k]<= 1- sum {j in R:j<> 50} X[i,j,(k-1)];
subject to const15 {i in PN, k in H: k<>0}:X[i,99,k]=1-X[i,0,k-1];
subject to constbanddischarged {i in P, k in H: k>0}: X[i,99,k]=0;

end;
```

Appendix E: Determining Number of Patients per LOS Category

In this appendix an example calculation is provided to understand the logic behind determining the proportion of the number of patients from the ICU study done by Ong et al[35] that comprise each LOS category. In this study, nearly 5000 patients were considered to determine what characteristics could be used to determine LOSs greater than 30 days. However, this thesis is only concerned with patients who had a LOS less than 30 days. Therefore, the following calculations were performed to separate the data relevant for patients with a LOS less than 30 days.

Figure 13: Composition of the Study Groups.



ILOS<30 = patients with intensive care unit (ICU) length of stay less than 30 days
ILOS>30 = patients with ICU length of stay greater than or equal to 30 days.

Example:LOS category 1

By looking at Figure 12 in the main body of this paper which shows the percentage of patients in each LOS category it was determined that 50% of the admitted patients were characterized as LOS category 1. To determine the number of patients in category 1 the total number of patients was multiplied by 50% (see equation 1). Next, to determine how many patients were in category 1 and had a LOS < 30 days the calculation in equation 2 was performed. Lastly, to determine the percentage of patients who survived their ICU stay, fell into category 1, and had a LOS < 30 days the calculation in equation 3 was performed. We can now conclude that about 56.8% of the patients considered in this study had a LOS of 2 days or less (category 1 patient).

$$50\%(Category\ 1) \times 4920(total\ number\ of\ ICU\ admissions) \tag{1}$$
$$= 2460\ patients\ in\ category\ 1$$

$$2460\ (category\ 1\ patients) \times 96\%(percentage\ of\ LOS < 30) \tag{2}$$
$$= 2362\ category\ 1\ patients\ with\ LOS < 30$$

$$\frac{2362\ (category\ 1\ patients\ with\ LOS < 30)}{4146(number\ of\ survivors)} \approx 0.568 \tag{3}$$

Appendix F: Data File Creation for Future Knowledge Consideration Model

To create the data sets to experiment for the future knowledge consideration model a combination of Microsoft Excel 2010 and Microsoft Visual Studios 2010 were used. Listed below is pseudocode that describes the basic process for creating a single data file with a given set of weights and a given planning horizon.

PSEUDOCODE:

```
1) Generate data for set $P_a$
    a) Randomly determine the number (between 18 and 26) of patients in the set $P_a$
    b) Randomly assign genders, isolation requirements, and criticality values
    c) Make room assignments randomly to determine initial condition
    d) Review assignments to ensure no constraints would be violated
    e) Perform any modifications to parameters so that all room assignments are valid
2) Generate data for set $P_n$
    a) Randomly determine number of patients in the set $P_n$
    b) Assign genders, isolation requirements, and criticality values
3) Generate data for set $P_t$
    a) Randomly assign patients in the ICU to different LOS cohorts
    b) Assign probabilities of being discharged in any given day to ICU patients based on their LOS
       cohort
    c) Randomly assign genders, isolation requirements, criticality values
```

Create Set P_a

First the number of patients in set P_a was chosen randomly by the experimenter. Next the gender values were ascertained by using the RAND() function and a logic formula in Excel. This procedure worked as follows,if the random number generated by the RAND() function was less than 0.5 then the value was a determined to be a “1” and “0” otherwise by implementing an IF() function. For this data set, male gender was associated with a value of “1” and female gender a value of “0.”Isolation requirement values were determined by the RANDBETWEEN() function with the parameters 0 and 6. The RANDBETWEEN() function returns an integer value 0-6 to signify the 6 different isolation requirements and the status of having no isolation requirements, “0.” The criticality values were generated utilizing the same method as for isolation requirements except different parameters were used. Criticality values range from 1-10 for patients in the unit thus the parameters 1 and 10 were used.

Create set P_N

Next, the set of patients, P_N, who were seeking admission to the unit was determined using the C++ code in Appendix G . This code selects a random integer from a stream of integers that follows the distribution of the patient demand for the PCCU. Next, a function assigns all patient characteristics for each patient.

Create set P_T

It was assumed in this problem that the ICU would always be fully utilized therefore the number of patients in that unit did not need to be generated. Additionally, these patients did not need to have randomly assigned criticality values. A value of 20 was assigned to all patients to signify that they were much more critical than the patients in the unit or patients seeking admission into the unit.

However, for the set of patients in the ICU the p_i parameter needed to be determined. Excel was used to generate the probability, p_i that the patient would leave the ICU in the current planning period. The procedure for assigning a probability to a patient in the ICU involves a three step process. First the RAND() function is instated to generate a random decimal. Then nested formulas of IF() functions are used to determine which LOS category the patient should be assigned. The nested IF() functions compare the random decimal to each value in the cumulative probability column in Table 31 until the decimal is less than the cumulative probability. Now, the patient can be assigned a LOS category. Next, a VLOOKUP() function is used to determine the proper probability to assign the patient based on what LOS category was assigned.

Table 31: Table used to Assign Probabilities

LOS Categories	Proportion	Probability	Cumulative Probality
1	0.568	0.75	0.568
2	0.25	0.321428571	0.818
3	0.09	0.104761905	0.908
4	0.057	0.050574713	0.965
5	0.034	0.002867106	0.999

All other parameters such as y_{ij}, w_k, and set H were assigned purposefully by the experimenter to create specific scenarios that are outlined in the beginning of Section 4.4.

Appendix G: Code Used for Generating PN Patients

```
/*Generates Data for ICU patients
  Author: Christina Cignarale
  Date:7/20/2012
  Using 5400 Unit Data:arrival distributions and unit configuration (10 single bedroom &
8 double bedrooms)
*/

#include <iostream>
#include <list>
#include <algorithm>
#include <functional>
#include <iterator>
#include <cstdlib>
#include <vector>
#include <fstream>
#include <stdio.h>
#include <stdlib.h>
#include <string>
#include <sstream>
#include <time.h>


// used here for convenience, use judiciously in real programs.
using namespace std;

// definition patient class
class patient {
  public:
       int ID;
       int ISO;
       int Crit;
       int Room;
       int Gender;
       string type;

};

//definition of the room:the number of beds in the room and the room number
class room{
public:
       int numberbeds;
       int roomnumber;
};

/*bool GetDischargeDay(patient pat, int curtime)
{
       return(pat.DischargeDay<= curtime);
}*/


/**** Function Definitions ****/
void NumberSeekAdmission(int&);
int MakeSeekAdmissionList(list<patient> &, int &);// adding the parameter planninghorizon
void FindFile(string &);
void AssignCharacteristics(patient &, int , int &,int&, list<patient> &);
void ReadInitialConditions(list<patient> & , int &, int&, int&);
int MakeDataFile(list<patient> &, list<room> &,list<patient> &,int&, int&, int&, int&);

// global constants
int Maxcrit=10;
int Maxiso=6;
int numberbedsICU=18;

//Variables
int numberseek=0;//the number of patients who seek admission
int ctr=0;//counter used for patient ID
int planninghorizon=2;
string inpatient_filename=" ";
string lengthstay_filename=" ";
string expname="Test";

//the vector that lists the weight for each planning period
vector<double> weight;

//list of all the patients in the system:ICU, unit, seekingadmission
list<patient> unit;

//the patients who will be seeking admission to the unit today
list<patient> seeking;

//rooms in the 5400 unit
list<room>  bedrooms;

//MAIN FX
int main(){

       int curtime=0;//the time the computer is running
       int numICUPat=0;//number of patients who are in the ICU and not seeking admission

       srand (time(NULL));
              }


       //Determine the number of patients who will seek admission in the current period
by selecting value from arrival file
       NumberSeekAdmission(numberseek);
       //Make Data file
       MakeDataFile(unit, bedrooms,seeking, numberseek, curtime, Maxiso, numberbedsICU);


return 0;

}

/*********************************************************************************/
void ReadInitialConditions(list<patient> & unit, int &curtime,int &planninghorizon,int
&ctr)
{
       list<patient> unittemp;
       //double prob;
       vector<double> initprob;
       patient CurrentPatient;
       ifstream indata; // indata is like cin
       indata.open("PATINIT.txt"); // opens the file
       if(!indata) { // file couldn't be opened
     cerr << "Error: file could not be opened" << endl;
     exit(1);
       }

       while (!indata.eof()) { // keep reading until end-of-file
         indata >> CurrentPatient.ID;
               //indata >> CurrentPatient.LOS;
              // CurrentPatient.DischargeDay=CurrentPatient.LOS+curtime;
               indata >> CurrentPatient.ISO;
               indata >> CurrentPatient.Crit;
               indata >> CurrentPatient.Room;
               indata >> CurrentPatient.Gender;
               CurrentPatient.type="A";
                     //assigns probabilities to patients currently in the unit, these are
all "1" because they are in the unit
                     /*for(int i=1;i<=planninghorizon;i++)
                            {
                                   prob=1;
                                   initprob.push_back(prob);
                            }
              CurrentPatient.probabilities=initprob;*/
              unittemp.push_back(CurrentPatient);// sets EOF flag if no value found
   }

       unit=unittemp;
       //ctr is used for assigning patients IDs, initialized for assignment here
       ctr=unit.size();
       indata.close();

}


/*************************************************************************************/

/*************************************************************************************/
void NumberSeekAdmission(int & numberseek)
{
       ifstream inarrivals;
       FindFile(expname);
       inarrivals.open(inpatient_filename.c_str(),ios::in);

              //in case of error
              if(!inarrivals)
              {
                     cerr<<"Error: file could not be opened"<<endl;
                     exit(1);
              }

       //select an element from the vector
       inarrivals >> numberseek;

       //close the LOS file
       inarrivals.close();

}
```

```
/*************************************************************************************/

/*************************************************************************************/
void FindFile(string &expname)
{
	int randAR= 1+rand()%50;
	//int randLS= 1+rand()%50;

	string str_randAR;
	//string str_randLS;
	//stringstream used to convert an integer to a string
	stringstream strAR_out;
	//stringstream strLS_out;
	//assigns random integer to variable str**_out
	strAR_out<<randAR;
	//strLS_out<<randLS;
	//assigns string to str_rand**
	str_randAR=strAR_out.str();
	//str_randLS=strLS_out.str();

	string str0="RGH";
	string str1="\\incomingpatient";//windows: "\\incomingpatient";
	//string str2="\\lengthpatient";

	inpatient_filename=str0+expname+str1+str_randAR+ ".txt";
	//lengthstay_filename=str0+expname+str2+str_randLS+ ".txt";

	//inpatient_filename="incomingpatient1.txt";
	//lengthstay_filename= "lengthpatient1.txt";

	}



/*************************************************************************************/

/*************************************************************************************/
void AssignCharacteristics(patient &Pat, int curtime, int &i,int &ctr,list<patient>
&unit)
{

			ctr++;
			Pat.ID=ctr;
			//Pat.LOS=vlen[curtime+i];
			//Pat.DischargeDay=vlen[curtime+i]+curtime;
			Pat.Crit= 1+ rand()%Maxcrit;
			Pat.ISO=1+rand()%Maxiso;
			Pat.Gender=rand()%2;
			Pat.Room=0; //0:= TRIAGE
			//Pat.probabilities=GenerateProbabilities(planninghorizon);
			Pat.type="N";
			unit.push_back(Pat);//adds patients to the entire system (ICU, unit,
seeking admission)
}
/*************************************************************************************/
```

Appendix H: Data Sets Used in Exploratory Sampling

These two tables display the summary data for the three data sets used to determine if the patient configuration of the unit (where patients are assigned in the unit) affected how many days the unit should use for planning with future information. Data sets “Ndata” and “N2data” had more patients seeking admission into the unit than beds available during period 1 while data set “Ndata10” had fewer patients seeking admission into the unit than beds available. In the second table it is clear that no one was admitted during period 2 for data sets “Ndata” and “N2data” since full utilization of the unit was achieved in period 1. Therefore ,we concluded that the number of patients seeking and the number of beds available may affect how many days the unit should use for planning with future information.

Table 32:Data Summary Tables for Patient Configuration Sampling

Data file	PA(admitted)	PN(seeking)	PT (ICU)	Beds Open	H	w
Ndata	23	4	18	3	2	0.7, 0.3
N2data	23	5	18	3	2	0.7, 0.3
Ndata10	21	3	18	5	2	0.7, 0.3

Table 33:Results for Patient Configuration Sampling

	Period 1		Period 2	
Data file	Admitted	Utilization	Admitted	Utilization
Ndata	4 (PN)	100%	0	100%
N2data	5 (PN)	100%	0	100%
Ndata10	3 (PN)	92%	2 (PT)	100%