ABSTRACT

Title of Thesis:	A MODEL-BASED SYSTEMS ENGINEERING SIMULATION: ANALYSIS AND DESIGN OF HOSPITAL BED MAINTENANCE IN CRITICAL HEALTH CARE SYSTEMS
	Arturo J. Davila Andino, Master of Science, 2018
Thesis Directed By:	Professor Michael C. Fu Decision, Operations, and Information Technologies Robert H. Smith School of Business Institute for Systems Research
	Professor of Medicine Kenneth E. Wood Department of Medicine Institute for Systems Research

This thesis summarizes the results of various methodologies integrated to solve a staffing problem when cleaning and maintaining hospital beds. First, a simplified systems engineering design model was developed to translate the need for reducing the total turnaround time of maintaining hospital beds into a performance requirement of the average time a hospital bed waits for service. The tools that were used were queueing analysis, discrete-event simulation modeling, and optimization via simulation. Finally, this work presents the derived staffing requirements from the pertinent measure of effectiveness, the average waiting time.

A MODEL-BASED SYSTEMS ENGINEERING SIMULATION: ANALYSIS AND DESIGN OF HOSPITAL BED MAINTENANCE IN CRITICAL HEALTH CARE SYSTEMS

by

Arturo J. Davila Andino

Thesis submitted to the Faculty of the Graduate School of the
University of Maryland, College Park, in partial fulfillment
of the requirements for the degree of
Master of Science
2018

Advisory Committee:
Professor Michael C. Fu, Chair
Professor Kenneth E. Wood
Professor Jeffrey Herrmann

Dedication

To my friend Emanuel, whose eternal curiosity inspires my drive to research and endeavor for what is true.

Acknowledgements

I would like to thank Dr. Kenneth E. Wood for his continued interest in this work and providing realistic insight into the problems faced by the heathcare industry.

I would like to thank Dr. John E. MacCarthy who relentlessly took the time to teach me, a monumentally stubborn student, the art and science of systems engineering.

Finally, I would like to thank Prof. Michael C. Fu for his guidance and support that made my research possible and the fruits of his teachings are embodied in this thesis.

Table of Contents

List of Figures and Tables

List of Abbreviations

ESD – Environmental Services Department

EOSL – End-of-shift laziness

INCOSE – International Council of Systems Engineering

MLE – Maximum likelyhood estimator

StratBAM – Strategic Bed Allocation Management

UMMC – University of Maryland Medical Center

Chapter 1: Introduction

1.1. Background

1.1.1. Need for Health Care Improvement

Although health care has recently become a highly politicized topic, the Affordable Care Act brought to the center of attention how wasteful the current health care system is. The President's Council of Advisors on Science and Technology (2014) states that a significant portion of the public spending in health care does not lead to better health or better care. The Council also states that the primary barrier precluding the full adoption of systems engineering principles is economical. Namely, the predominant fee-for-service payment system engenders poor quality of care.

Due to the external pressures of cutting costs such as decreased government subsidies, hospitals have taken drastic measures to meet the reduced budgets such as downsizing beds, cutting staff, and merging with other hospitals. Without the help of systems engineering and operations research analyses that are commonly used in other service industries, hospitals deliver a substandard quality of care. Operational waste permeates all levels of care patients receive. The results have been catastrophic, as the waste translates to high patient mortality. Namely, patients experience long admission times, unnecessary treatments, and exorbitant health care costs. The University of Maryland Medical Center, the main subject of this thesis, in collaboration with the Institute for Systems Research, has agreed to address these problems that permeate the hospital's infrastructure to deliver better quality care.

1.1.2. Systems Engineering and Operations Research in Health Care

The disciplines of systems engineering and operations research have provided tools to solve real-world problems. Systems engineering is a perspective, process, and interdisciplinary approach that focuses on the holistic design and realization of successful systems. The goal of systems engineering is to front-load cost and efforts in the design process to provide a seamless and cost-effective implementation of new systems. Conversely, operations research is a collection of techniques used to support decision making. Although systems engineering and operations research tools have been successfully implemented in the past, it is still far from full adoption by the health care industry.

One of the primary tools of interest in health care is optimization. In the field of Operations Research, optimization is a decision-making tool that uses specialized algorithms to quantitatively find the best solution from a large (or infinite) number of options (Capan, et al., 2017). Optimization is an essential tool in health care due to the need of providing the best quality of care at a reasonable cost to patients. According to Capan, et. al. (2017), "[Operations research] models and methods have been successfully applied to decision problems such as treatment modeling, living-donor organ transplantation, disease prevention efforts such as vaccination and screening, the design of health care supply chains, and capacity planning". However, Wagner-Jopling (2017) state that the adoption of operations research in medicine been slow due to lack of data. Insufficient data is prone to endogeneity problems leading to inaccurate conclusions and models.

The International Council of Systems Engineering (INCOSE) defines systems engineering as:

> Systems Engineering is an interdisciplinary approach and means to enable the realization of successful systems. It focuses on defining customer needs and required functionality early in the development cycle, documenting requirements, then proceeding with design synthesis and system validation while considering the complete problem: Operations, Cost & Schedule, Performance, Training & Support, Test, Disposal, and Manufacturing.
>
> Systems Engineering integrates all the disciplines and specialty groups into a team effort forming a structured development process that proceeds from concept to production to operation. Systems Engineering considers both the business and the technical needs of all customers with the goal of providing a quality product that meets the user needs. (INCOSE, 2014)

INCOSE also has a particular interest in the field of health care as it is currently part of the society's 2025 Vision (SE Vision 2025, 2018). Also, INCOSE supports a Healthcare Working Group which promotes the application of systems engineering principles to health care. Similarly to operations research, the health care industry has not adopted systems engineering principles to manage operational or medical complexities effectively. Although still in its infancy, systems engineering will become pivotal in managing clinical complexity in medicine (Kopach-Konrad, 2007) (Khayal & Farid, 2017). Furthermore, Valdez et al. (2010), suggest that the barriers to achieving breakthrough change in health care reduce to miscommunication between health care professionals and systems engineering experts. To elaborate, health care professionals do not have a comprehensive guideline of best practices to implement systems engineering or operations research within health care.

1.2. *Problem Statement*

1.2.1. The University of Maryland Medical Center

The University of Maryland Medical Center (UMMC), the leading subject of this thesis, is a hospital located in Baltimore, Maryland that provides tertiary and quaternary health care services to the western region of Maryland. The hospital has 757 beds for admitted patients serving more than 35,000 patients in 2016. UMMC also serves as an academic institution teaching the next generation of physicians.

Although UMMC has been consistently in the forefront of patient safety, the hospital still struggles to deliver quality care to patients (Willingham, 2018). Similarly to other hospitals, UMMC struggles to cope with the competing objectives of delivering high-quality care to patients and meeting their fiscal goals. To resolve this conflict, the UMMC has partnered with the Institute for Systems Research of the James A. Clark School of Engineering at the University of Maryland, College Park. The primary goal of the partnership is to provide systems solutions to the complex problems that permeate the daily operations of the hospital.

1.2.2. Environmental Services Department

The Environmental Services Department (ESD) within the UMMC has the responsibility of dealing with the maintenance of the hospital. That is, they are responsible for the maintenance of the hospital facilities, premises, patient beds, and offices. This thesis will focus on the patient bed maintenance processes, specifically, the bed cleaning process used to clean hospital beds. After a patient leaves, the bed needs to be cleaned to admit the next available patient.

The bed cleaning process has received urgent attention by upper management due to its direct relationship with patient wait times and operating efficiency of the hospital. The ESD has been studying the bed cleaning process for the year of 2016 and collecting data for their process in attempts to improve the quality of their serivices. Furthermore, the ESD also provided all the pertinent data analyzed throughout the whole thesis. The data provided is a collection of more than 46000 hospital bed that underwent the bed cleaning process.

The data is a collection of timestamps of the discharged or otherwise vacated beds that need cleaning to compute the total turnaround time. The software also records the bed location and priority assigned. The bed location is a unique tag that identifies the hospital unit and room to which a bed needs cleaning. There are three priorities that are assigned to incoming hospital beds: Normal, Next, and Stat. These bed priorities are used to manipulate the order of cleaning when assigning the beds to the janitors. If a bed has a Normal Priority, then the bed is attended in a first-come first-served basis. However, if an emergency situation arises, then the bed can be assigned a Next or Stat priority that, respectively, overrides the order of service assigning the bed to the next available janitor, or interrupts the assignment of a janitor to have the bed cleaned as soon as possible. The data gathering software also records the cleaning protocols as Contact, Droplet, and Airborne. The cleaning protocols determine the precautions needed to avoid the transmission and propagation of diseases within the hospital. (Garner, 2018)

1.2.3. The Bed Cleaning Process

Our goal is to reduce the average time taken from when notification to clean bed is generated to when the bed is available for next patient – termed as the total turnaround time – from 83 minutes to less than 60 minutes. Reducing the total turnaround time will, in turn, increase the availability of hospital beds. Having a slow cleaning process for the hospital beds result in beds being less available to patients. Consequently, this translates into excessive admission times for patients. In critical health care systems, the time a patient waits for treatment is crucial to their survival.

Therefore, the primary objective of this thesis is to determine the optimal factors for the bed cleaning process to reduce the total turnaround time. In this thesis, a functional model of the system was built to analyze the measures of performance of the bed cleaning process. Then, a simplified queueing model was built to determine the optimal allocation of janitors to the process throughout a 24-hour period. However, since the peak hours of the arrival process does not correspond to the working hour shifts, a simulation was built to determine the optimal allocation of janitors during said shifts. Finally, as multiple solutions are possible, the last chapter of the thesis delves into determining the best solution for the University of Maryland Medical Center.

1.3. <u>Literature Review</u>

1.3.1. Queueing Models in Health Care

Queueing models have various factors that determine the appropriate tools to solve congestion (i.e., reducing waiting times) within queues. The factors include the arrival process, service time distributions, and the number of servers. The conventional measures of performance of a queueing system are the average waiting times, average queue length, and server utilization. There are three options to reduce waiting times: reduce the service times, increase the number of servers, change service priorities (e.g. serving customers with shorter service time), or an amalgamation of these three. Generally, these actions result in increased direct operating costs that must be balanced by savings in the indirect cost of waiting times.

In health care, queueing theory has been used extensively to solve congestion problems. Kozumi (2002) provides a comprehensive analysis when modeling and simulating queueing networks with blocking in mental health institutions. Other researchers have proposed mathematical solutions to bed management using queueing theory equations and variability methodologies (Gorunescu, McClean, & Millard, 2002) (Smith, et al., 2013). Finally, Fomundam-Herrmann (2007) provide a comprehensive survey of tools in queueing theory for health care processes.

1.3.2. Simulation in Health Care

Simulation in the general sense has been used in health care as an educational tool to train doctors. Applications of simulation range from actors portraying patient illness to 3D renderings of human anatomy. Most recently, discrete-event simulation has taken the role of process improvement. For example, Steward et al. (2017) use discrete-event simulation to determine a near-optimal allocation of staff for the emergency department to reduce the length of stay of patients. Other applications include nurse scheduling, medical resource planning and allocation, patient appointment problems, hospital collaboration problems, and emergency medical service problems (Chen & Lin, 2017).

Similarly to queueing theory, applications of stochastic discrete-event simulations have also been developed to make managerial decisions in the health care realm. Most discrete-event simulations in health care involve the goal of reducing the length of stay of patients or determine the optimal allocation of medical resources. However, simulations are mostly used a decision support tool rather than a design tool. Furthermore, the results of such simulations are used to compare predetermined system configurations rather than searching for a solution that has the most value in terms of patient satisfaction and fiscal constraints. However there has been recent efforts to implement optimization methodologies in simulation (Pujowidianto, Lee, Pedrielli, Chen, & Li, 2016).

StratBAM is one of the most recent and thorough simulation model of a hospital (Devapriya, et al., 2015). Their simulation model was built in the commercially available discrete-event simulation package *FlexSim HC*. FlexSim HC uses 3D visuals to allow analysts to picture the ongoing simulation. One of the inputs of the software described in the paper is the *Bed Turnover time* which is another name for Total Turnaround Time defined in this thesis. This thesis presents a simulation model of the bed cleaning process which generates and records Total Turnaround Time which in turn can be used as input into the StratBAM simulation.

1.3.3. Research Contribution and Thesis Content

The current state-of-the-art hospital system analyses involve stochastic discrete-event simulation (abbreviated as “simulation” when pertinent for the rest of the thesis). However, simulations are used for decision support rather than decision making (Devapriya, et al., 2015). The most recent attempt to implement optimization in simulations is by using linear regression between the factors and the metrics (Kokangul, Akcan, & Narli, 2017). More sensible approaches to optimize peripheral health care problems have been proposed. Namely, optimization of patient referral in hospital networks (Chen & Lin, 2017), and ambulance deployment and relocation (Zhen, Wang, Hu, & Chang, 2014).

The principal contribution of this thesis is to build, analyze, and simulate the bed cleaning process model in a technically correct manner. The remainder of this thesis will focus in presenting different systemic models used when setting staffing requirements. Chapter 2 documents the SysML architecture model of the bed cleaning process and how it relates to the data provided by the ESD. Chapter 3 introduces a queueing analysis for staffing multiserver queues. Furthermore, Chapter 4 describes the simulation of the queueing model. Chapter 5 analyzes the possible optimal options for the system using statistical ranking and selection. Finally, the Chapter 6 presents a discussion of the results, recommendations, conclusions, and future research options.

Chapter 2: System Architecture Overview

2.1. <u>System Context</u>

2.1.1. System Description

Currently, there is a system in place with the capability to clean hospital beds after a patient has been discharged. The system of interest, *Bed Cleaning System*, is meant to reduce the total turnaround time and provide clean beds for the next waiting patient. Functionally, the system is an aggregation of janitors and Environmental Services Department (*ESD*) Management at UMMC explained in detail in the next section. The purpose of this section is to identify the users and external systems which the system of interest will be interacting with.

The user of the *Bed Cleaning* system is the *Clinical System* as they need clean beds to admit newly arriving patients. Employees within the *Clinical System* can trigger the bed cleaning protocol, so that dirty beds enter the bed cleaning process. The hospital system structure, as shown in Figure 1, is composed of the *Janitors* and *ESD Management* working together using the Epic software (*EPIC Hospital Bed Management System*) to provide clean beds for the hospital and maintaining the premises. Epic is proprietary software used to manage the overall hospital and health care systems and is used by virtually every entity within the hospital. Furthermore, the janitors update the bed cleaning status through the Epic software system to identify and measure total turnaround times and other measures of performance.

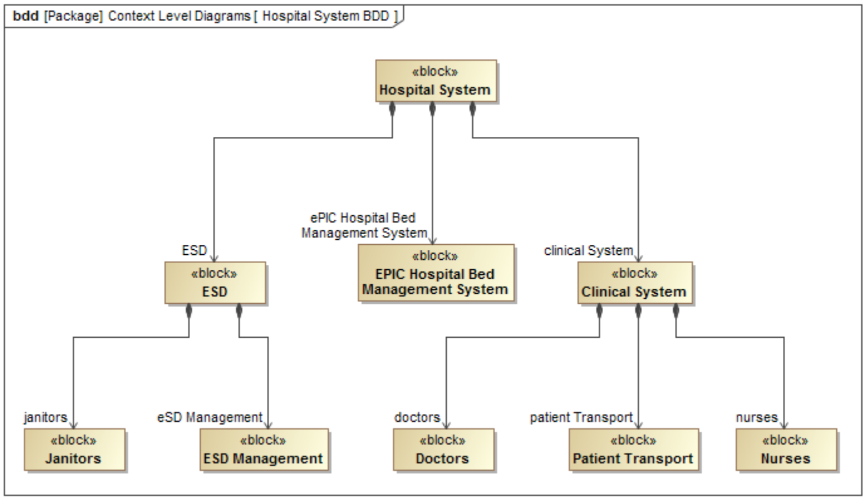


Figure 1 Hospital Hierarchy Block Definition Diagram

2.1.2. Context Structure

The goal of the clinical system is to provide patients with high-quality health care. Generally, the hospital treats sick patients whether by admission, outpatient procedures, or through the emergency room. With their various health units and diverse resources, the University of Maryland Medical Center provides a wide range of health care services specializing in critically ill and injured patients. For the hospital to fulfill their mission, the hospital needs reliable peripheral services to provide a high quality of care. One of these services, which is not directly related to health care, is the cleaning of beds after a patient has been discharged.

The *Bed Cleaning System* uses a portion of the Janitor's time to clean an assigned hospital bed. That is, janitors have other responsibilities to attend to such as cleaning bathrooms, offices and common areas. The janitors are notified through pagers of a bed assignment and then clean their assigned beds. Similarly, the *ESD Management* interacts with the system by assigning dirty hospital beds. To achieve this goal, they assign beds to janitors in the order of arrival unless needed otherwise.

The average total turnaround time, the measure of effectiveness of the current system in place, was 140 minutes or 2 hours and 20 minutes for 2016. A SysML representation of the context structure is presented in Figure 2.

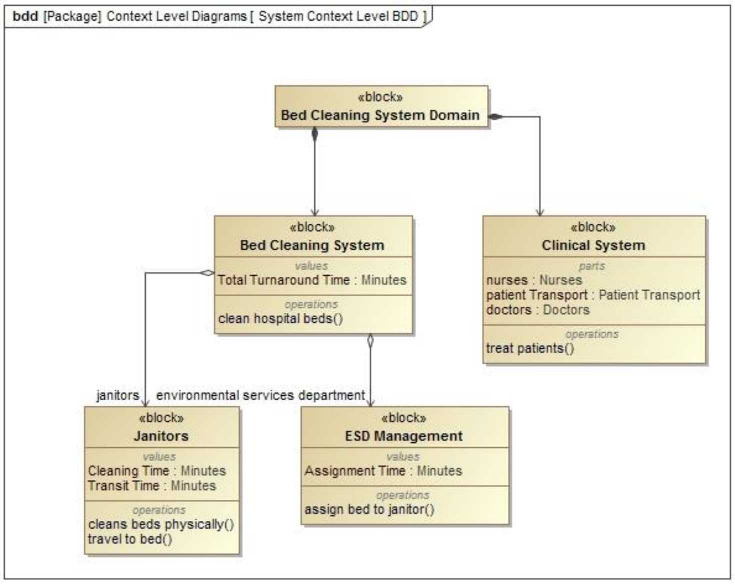


Figure 2 Bed Cleaning System Domain Block Definition Diagram

2.1.3. Context Diagram

The functional purpose of the *Bed Cleaning* system is to provide cleaning services for vacated beds. The Dirty Beds interface is meant to represent that only hospital beds flow through the *Bed Cleaning System* and has specific attributes that characterize the bed, which in turn factors into its total turnaround time. However, this model, as presented in Figure 3, is an oversimplification, as other interfaces flow throughout the system such as cleaning materials, patient transport approval, and even janitors when they change shifts. Moreover, the rationale for this simplification is to present the association that the *Bed Cleaning System* behaves like a queueing system, and therefore, reducing the total time in the queueing system becomes the goal of the analysis presented in this thesis.

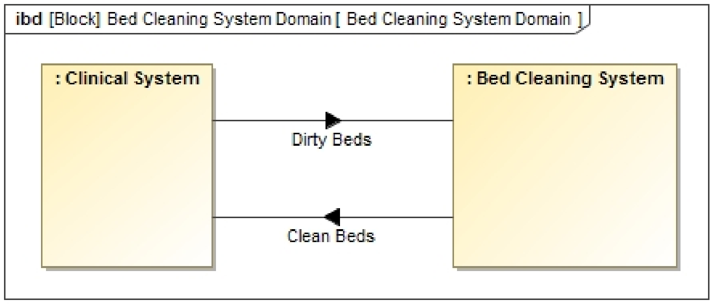


Figure 3 System Context Diagram

Reducing the average total turnaround time of hospital beds will increase the availability of beds for incoming patients. Consequently, higher bed availability translates to reduced waiting times for patients to be admitted and reduced mortality rates due to delayed care. The system will impact the hospital staff and operations, and ultimately the patients. Thus, providing a better quality of care to patients and improves quality of life in the region. Figure 4 presents how the *Clinical System* needs clean beds to admit sick patients.

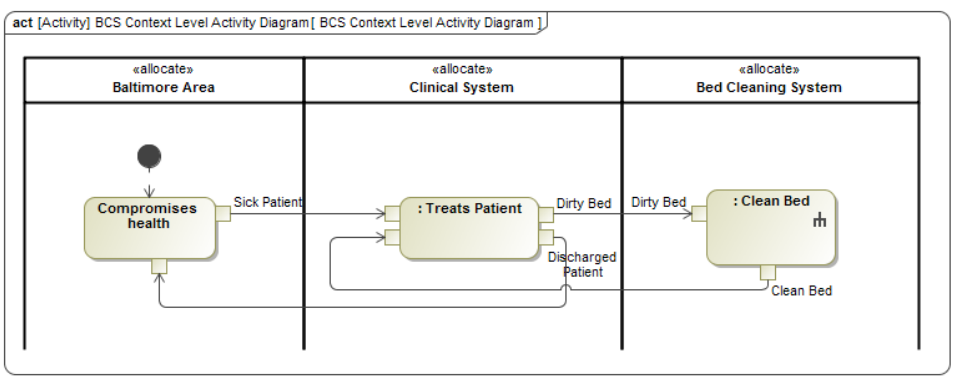


Figure 4 Context Level Activity Diagram

2.2. *System of Interest*

2.2.1. Activity Description

The cleaning process can be divided into the fulfillment of various events. First, the patient gets discharged or otherwise vacates the room. Then, the event onto where the previously occupied bed is input into the system, and the ESD is notified of the bed that needs cleaning. Afterwards, the bed input into the system is assigned to an available janitor for cleaning. Subsequently, the janitor with the assigned bed arrives at the bed and notifes his arrival to the Epic system. Lastly, after finishing cleaning the beds, the janitor sends a notification making the assigned bed available for the next patient. A graphical representation of this process is shown in Figure 5.

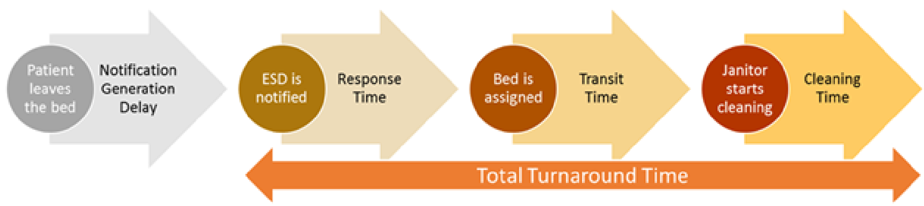


Figure 5 Bed Cleaning Process Flowchart

Between the events explained earlier, there are unique behaviors within the subprocesses between events. These subprocesses' time to completion are measures of performance that, when added together, express the total time it takes to clean a hospital bed. However, only the duration of the last three subprocesses constitutes the measure of effectiveness of the bed cleaning process, the total turnaround time. Although the Notification Generation Delay is not considered part of the cleaning process, the process behind the delay impacts the total turnaround time.

After a patient vacates the hospital bed, a nurse is assigned to input the bed into the system for cleaning. However, to avoid interruptions, the nurse waits until there is enough accumulation of beds to input into the system. This behavior may be justified by the daily routines taken by the medical staff when discharging patients. For example, if most discharges occur into the afternoon, then nurses delay the entries of the dirty hospital beds until it is near the end of their shift to avoid interrupting other responsibilities and better utilize their times. In fact, this collection of behaviors is the source of many delays within the rest of the subprocesses. Since the employees change shifts at the same time, the beds input at the end of a shift do not get assigned to the janitors working on the shift the patient got discharged, which may result in the bed not being cleaned until the next day.

2.2.2. System Design

After the bed is input into the system for cleaning, ESD Management needs to assign the bed to the next available janitor. The time it takes to assign the bed to the next available janitor depends on the order of arrival and other factors. For most beds, the time it waits until it gets assigned to a janitor depends on the number of beds that have been input into the cleaning process and have not yet been assigned. However, as exposed in the first chapter of this thesis, assigning a Next or Stat priority is a mechanism to drastically reduce this waiting time. However, the tradeoff is that the rest of the hospital beds in waiting have to possibly wait a while longer. The measure of performance of the *Assign Bed to Janitor* action, as depicted in Figure 6, is called the response time by the hospital bed managers. However, to avoid terminology confusion with Queueing Theory, this will be called the waiting time in queue or waiting time for short for the rest of the thesis unless specified otherwise. Since the average waiting time (83 minutes) takes the longest of the three subprocesses associated with the bed cleaning activity, it is the topic of most importance for the rest of this thesis. Furthermore, reducing the average waiting time is the sole goal of the system design exposed in this work.

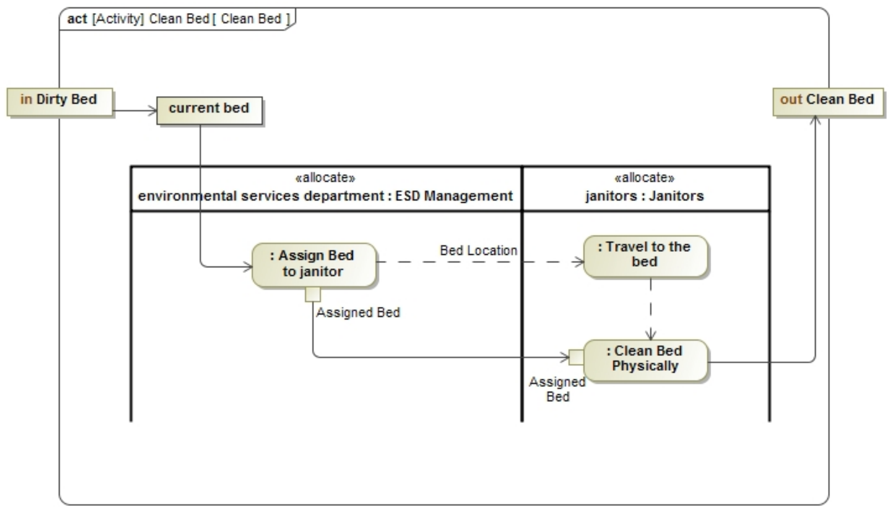


Figure 6 Clean Bed Activity Diagram

The subsequent action, *Travel to the bed*, depicts the activities needed to finish their current assignment or duty, gather the necessary resources to clean the bed, and arrive at the assigned bed. This action may be thought as the first stage of the actual cleaning of the bed that acts as a delay to the physical cleaning of the bed. Generally, janitors may not be nearby hospital beds as they have other duties to attend while working. However, since janitors are not tracked or the nearest bed to be cleaned cannot be identified, the time it takes to travel to the assigned bed could mean there is a stochastic shortest path problem embedded into the cleaning system. Fortunately the author of this thesis, this action's measure of performance, the transit time, contributed 11 minutes to the average total turnaround time, and it is not considered a pressing problem by leaders at UMMC.

The last action within the activity, *Clean Bed Physically,* summarizes the set of steps needed to clean the hospital bed physically. At an average of 46 minutes, this is the second most protracted process that composes the Bed Cleaning activity. As explained in the previous chapter, beds may have cleaning protocols that affect the cleaning times. The next sections expose the factors that might correlate with the associated measure of performance, the cleaning time. Moreover, in Section 2.3, there is an analysis of factors that may contribute to the cleaning time. The factors under study are: Bed Unit, Working Shift, Bed Priority, and Cleaning Protocol

2.2.3. Queueing Model

As stated in the previous sections, the Clean Bed activity can be allocated to the behavior of a queueing system such that the sojourn times can be divided into two stages: A waiting stage and a service stage. When a patient vacates a bed and the bed is input into the system, may be considered an arrival to a queueing system. Then, the bed waits until it is assigned to a janitor, and the janitor cleans the bed. There may be multiple janitors. The service times can be allocated to a 2-phase service process that describes the *Travel to the Bed* and *Clean Bed Physically* action. The last section of this chapter shows an analysis of the service times measured for 2016.

2.3. Cleaning Time Analysis

2.3.1. Unit

Figure 7 and Figure 8 present the boxplots regarding different units within the hospital according to the metric of interest, the bed cleaning time. The x-axis presents the unit abbreviation to represent the general location of the beds under study. The medians for the beds appear to fluctuate between the locations of the units. Notably, units W7A and W7B, corresponding to the Medical Intensive Care Unit, show cleaning times higher than the rest. Although some beds have apparent differences regarding the distribution of the boxplots, most of the cleaning times medians do not exceed 50 minutes. Also, the data used for all the analysis done in the next two sections are not filtered for values at less than or equal to zero. Therefore the analysis done in these next two sections is for illustrative purposes.

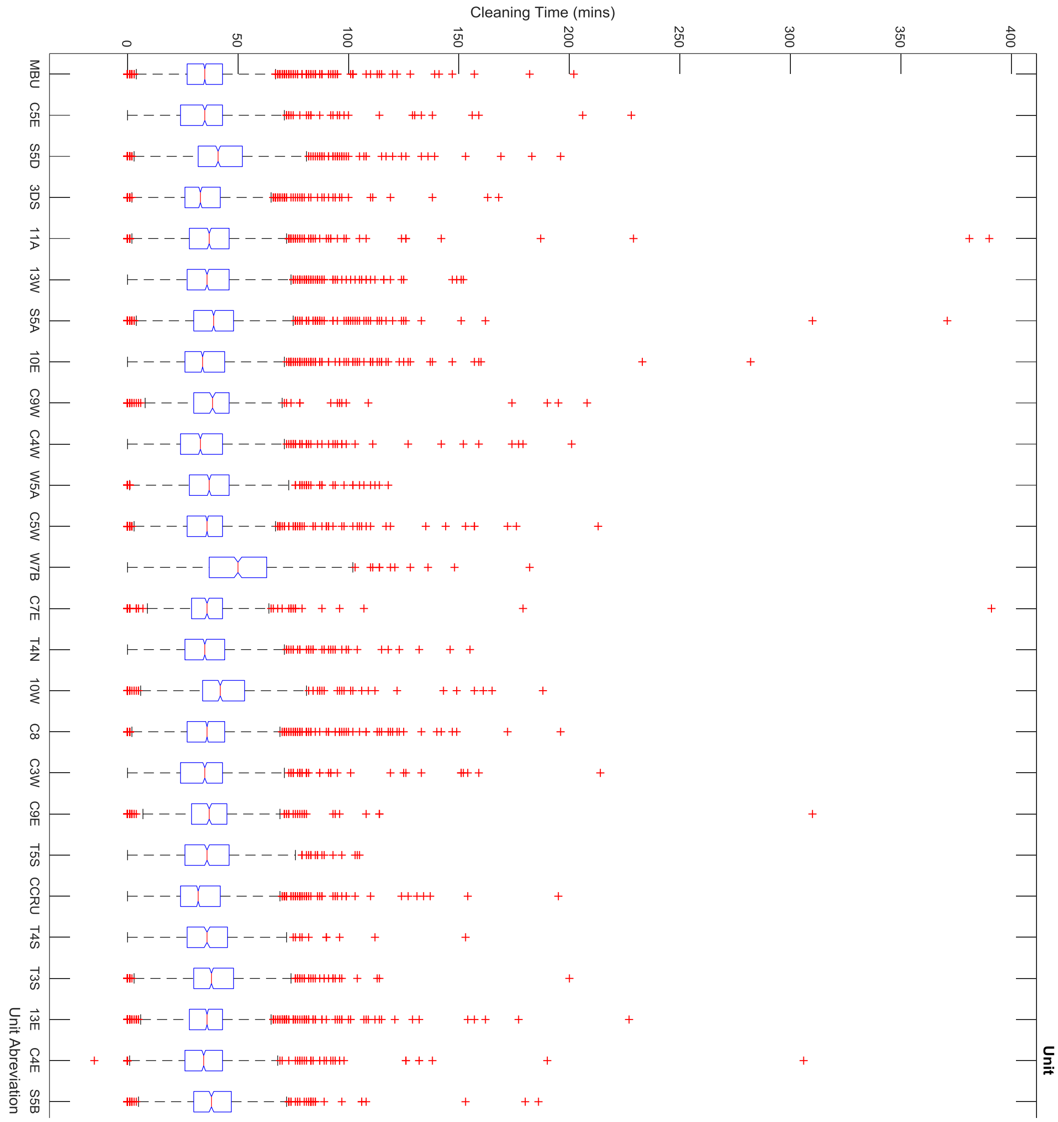


Figure 7 Box plot of cleaning times with respect to the unit

Figure 8 Box plot of cleaning times with respect to the unit (continued)

The first ANOVA shows a table with the relevant measures pertinent to the previous boxplot. The left column of the table shows the labels for the groups under study, and the error with the corresponding total. The labels "Group" denotes the attribute of interest, which in this case is the unit the bed is located. The details on how the analysis is done are lost within the internal mechanics of the Matlab function, *anova1( )*. However, the necessary measures are shown; such as Sum of squares (SS), degrees of freedom (df), Mean Square (MS), F-value (F), and p-value (Prob>F)

ANOVA Table

Source	SS	df	MS	F	Prob>F
Groups	763261.5	48	15901.3	41.46	0
Error	17953772	46807	383.6		
Total	18717033.6	46855			

Figure 9 Unit output data from ANOVA

Since we have a p-value of -practically- zero, then we performed paired t-testing for all categories within the attribute. Since it is humanly impossible to perform t-testing for all 1176 possible combinations, we developed Matlab code to perform the hypothesis testing. (Refer to Appendix C) Out of all combinations, 360 pairs of units did not show any statistically significant difference. Therefore the location of the bed is a factor in the cleaning time distribution.

2.3.2. Shift Analysis

Figure 10 presents the boxplots regarding different shifts according to the metric of interest, the bed cleaning time. The x-axis presents the shift to represent the general time of day of the beds under study. Shift 1 represents the time from 7 AM to 3 PM. Shift 2 represents the time from 3 PM to 11 PM. Otherwise, the time of day is the third shift. The medians for the beds appear to fluctuate between the shifts. Namely, it seems as if the cleaning time takes longer during the second shift. Also, it appears as if the interquartile ranges of the third shift is higher than the other two shifts. Therefore, the most important conclusion from this analysis is that the cleaning time is not stationary.

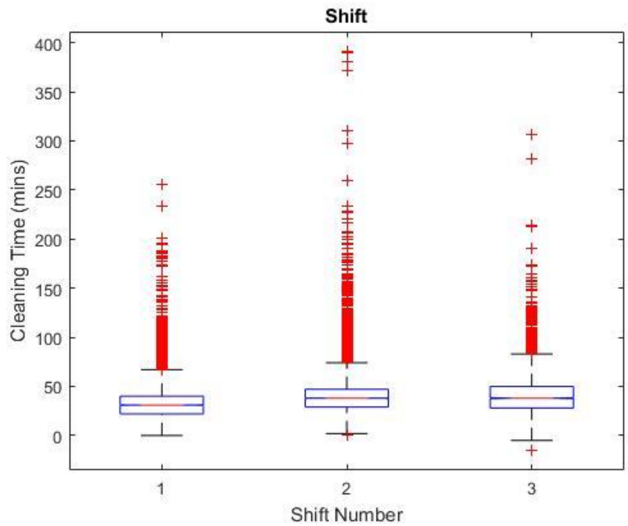


Figure 10 Box plot of cleaning times with respect to the shift

2.3.3. Bed Priority Analysis

Figure 11 presents the boxplots regarding different Bed Priority within the hospital according to the metric of interest, the bed cleaning time. The x-axis presents the bed priorities of the beds under study. As mentioned in the first chapter of this work, there are three bed priorities: Normal, Next, Stat; corresponding respectively to 1, 2 and 3 in Figure 11. However, the medians for the beds cleaning times appear to be slightly different, and the data shows there is a statistically significant difference between the data sets.

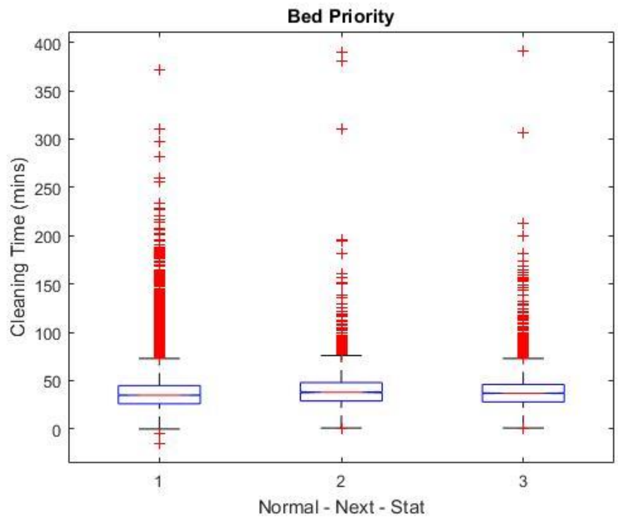


Figure 11 Box plot of cleaning times with respect to the cleaning protocol

2.3.4. Bed Priority Analysis

Figure 12, Figure 13, and Figure 14 present the comparison between boxplots when a bed has the attribute present. Beds can have a combination of these three cleaning protocols. Since statistically significant difference was measured for each of the cleaning protocols, a 3-way analysis of variance was not considered. In each of the plots, 0 denoted the absence of the cleaning protocol, and 1 denotes the presence of the protocol.

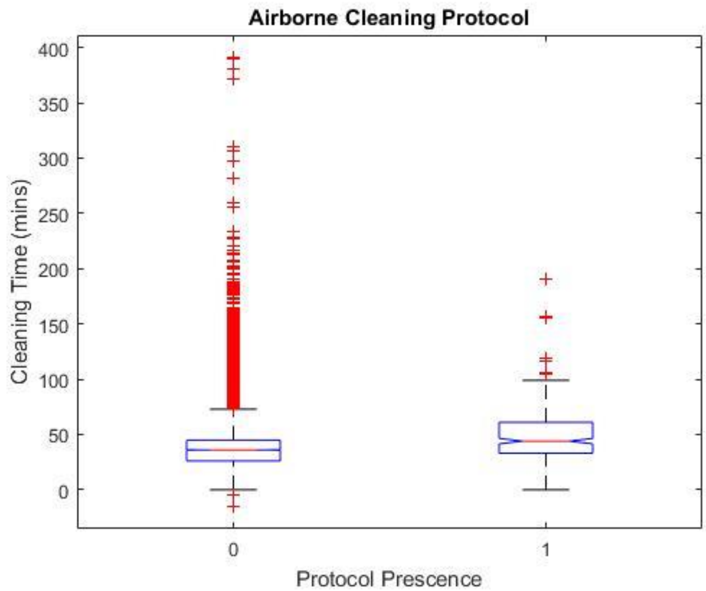


Figure 12 Box Plots for the Airborne Cleaning Protocol

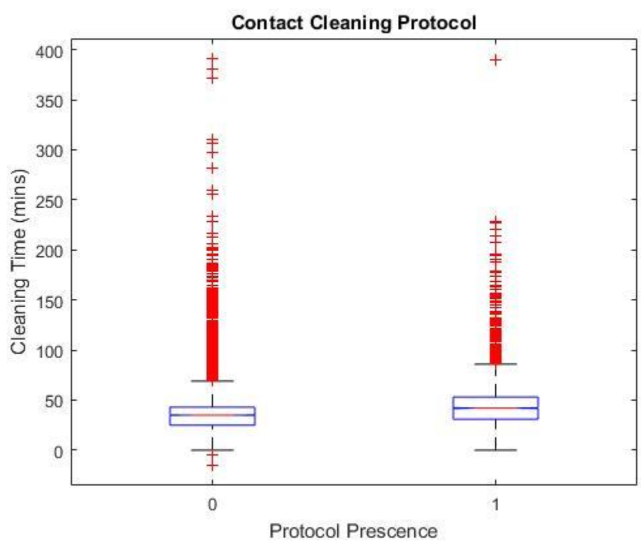


Figure 13 Box Plots for the Contact Cleaning Protocol

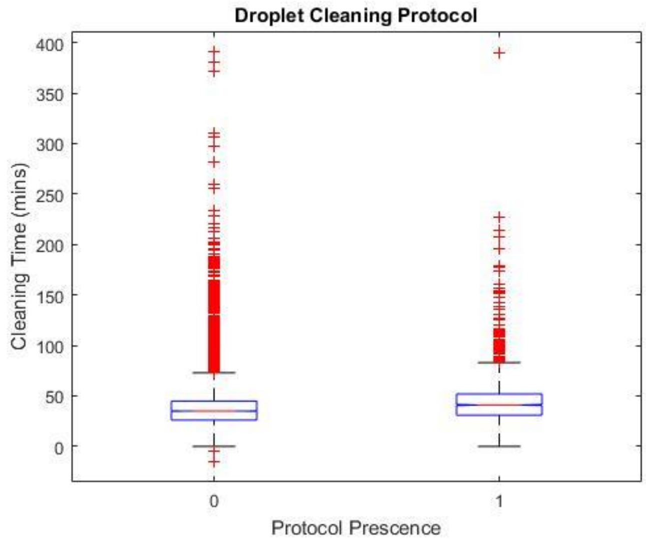


Figure 14 Box Plots for the Droplet Cleaning Protocol

Chapter 3: Queueing Analysis

3.1. Arrival Process

3.1.1. Non-stationary Poisson Process

An initial survey of the data suggests that the bed inter-arrival times approximate an exponential distribution. Figure 15 shows the comparison between the interarrival times collected from the data provided by the hospital and the exponential distribution fitting. The first queueing model of the process will be an M/M/s queue with a mean inter-arrival time of 7.865 minutes. As it is the case with most abundantly available data, the data for the arrival process did not undergo a fitting process because the Kolmogorov-Smirnov test will almost certainly reject all hypothesized parameters that involve an exponential distribution.

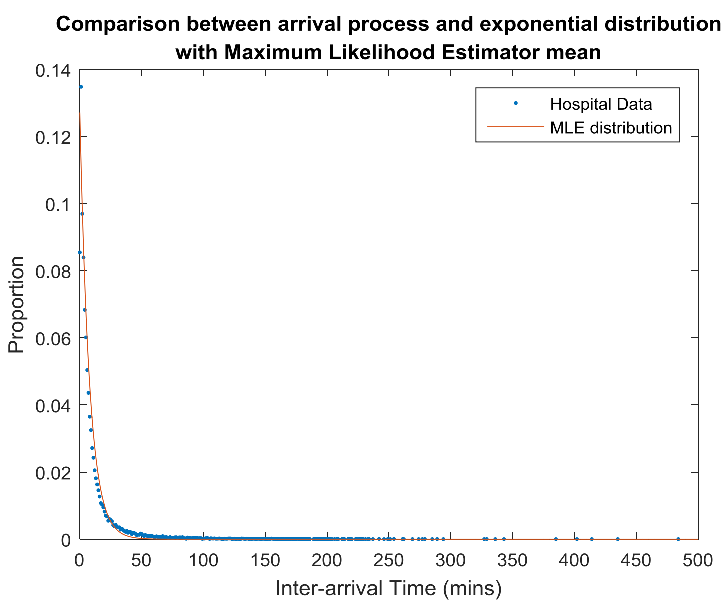


Figure 15 Inter-arrival times vs. the proportion of arrivals comparisons between the hospital data and the distribution of the maximum likelihood estimator

However, the data has also shown that the arrival process for the hospital beds is time-dependent, meaning the arrival rate may be higher than usual at certain times during the course of the day. Figure 16 presents the various arrival rates used for each queueing model built. The blue function represents the hourly arrival rate according to the data for the year of 2016 of the hospital subject. The red function is the computed average arrival rate (5.152 beds per hour) of the non-stationary curve as it is assumed that the same pattern repeated throughout the rest of the days of the year. The arrival rate from the red function differs from the arrival rate mentioned in the previous paragraph due to the shifting of the weights when computing the average of averages. Finally, the yellow curve shows the peak hour's period of the arrivals to the hospital bed cleaning process. The peak hours last from 11:00 AM to 9:00PM and the arrival rate is 9.639 beds per hour. For the rest of the day, the arrival rate is 2.161 beds per hour.

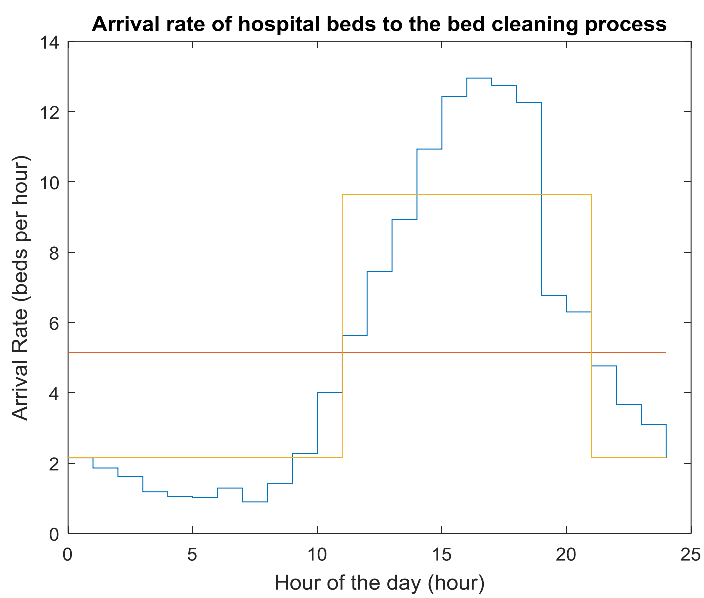


Figure 16 Arrival rate as a function of time

The queueing system exhibits time-varying behavior in its arrival process, meaning most tools used for stationary models (e.g., Markov chain flow equations) cannot be used. Therefore, a finite multi-server queueing system with a non-stationary arrival process is analytically intractable. However, the one approach is to solve two separate M/M/s queueing models for the two periods, obviating the transiency and disparity between the models. One of the queueing models has a higher arrival rate corresponding to the peak-hours, or the period of time the arrival rate is higher than usual. The second queueing model is meant to represent the system off-peak-hours, meant to represent the system throughout the rest of the day when the arrival rate is lower. The caveat of solving queueing systems this manner is the propensity of arriving at erroneous conclusions due to the significant difference between the relative amplitude to its average (Green & Kolesar, 1991).

3.1.2. Cyclical Behavior

Another approach to solve $M_t/GI/s$ queues is build the analogous $M_t/GI/\infty$ model to determine the number of servers needed (elaboration of the rationale for this model is explained in another section). Therefore, a Fourier series approximates the arrival intensity as a continuous periodic function. Figure 17 illustrates the fitting results of Fourier series into the intensity function. Equation 1 shows the intensity distribution fitting for the first three terms of the Fourier series. The detailed values can be found in Appendix B.

$$\lambda(t) = a_0 + \sum_{n=1}^{3} \mathrm{a_n} \sin(wnt) + b_n \cos(wnt)$$

Equation 1 Fourier series approximation of the arrival rate

The coefficients a_n and b_n were the terms fitted to the data and *w* is the period frequency. The term w, being the frequency, is set to 2π/24 as one period of the cyclical function is meant to represent one day in terms of hours.

Data suggests that the arrival intensity is not continuous due to the sharp decrease of arrivals from 6PM to 7PM (or hour 18-19 in Figure 17) caused by the change of medical and supporting staff. However, the continuity assumption is kept to simplify the queueing model within this thesis. Previous authors have attempted to simplify cyclical queueing models utilizing a positive sinusoidal function (Green, Kolesar, & Whitt, 2007). However, to the author's knowledge, a Fourier series or sensible truncation of terms has never been published before.

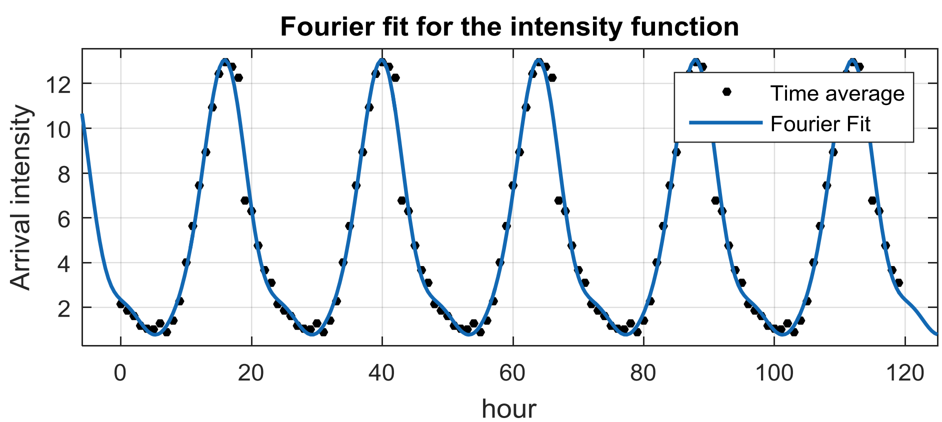


Figure 17 Fourier approximation of the intensity function with respect to time

3.2. Service Time Distribution

3.2.1. Exponential service time

A queueing model with an exponentially distributed service time was first attempted to build the three stationary M/M/s models mentioned in the previous section. For the purposes of this study, the mean service time of each of the M/M/s model will be the same as the average observed for all the beds cleaned in 2016. The rationale behind approximating the queueing models as M/M/s is to implement what is sometimes referred to as Kingman's law of congestion approximation (Gans, Koole, & Technion, 2003). The Kingman's law of congestion, as presented in Equation 2, states that the expected waiting time in an M/G/s queue is proportional to the expected waiting time of an M/M/s queue. The coefficient factor c_s is the coefficient of variation for the service time distributions of an M/G/s queue. Furthermore, $W_{M/*/s}$ refers to the random variable of the waiting time of each queueing system.

$$E\left[W_{M/G/s}\right] = \frac{c_s^2 + 1}{2} E[W_{M/M/s}]$$

Equation 2

3.2.2. Total Service Time

As discussed in the second chapter of this thesis, the service time distribution varies by hospital and by shift. Since time to completion distribution of the bed cleaning process is conjectured to be normally distributed, then the stationary service time distribution will be assumed to be a mixture of an unknown number normal distributions. Therefore, the mixture of two normal distributions was first used to fit the data as presented in Figure 18.

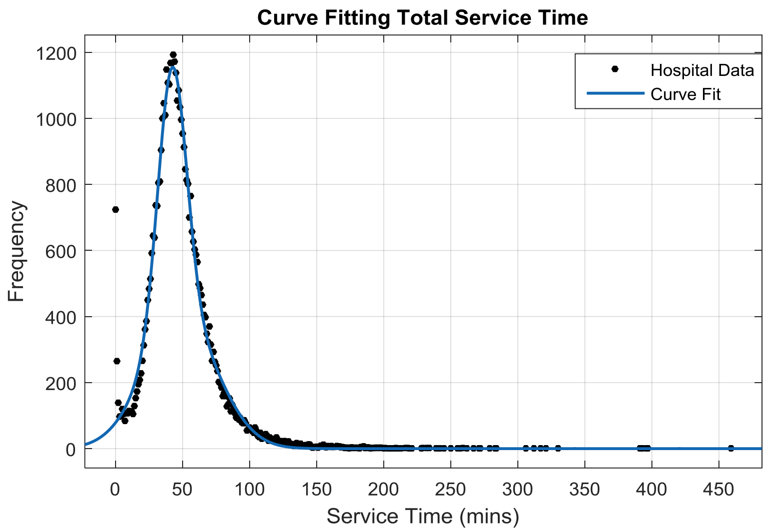


Figure 18 Frequency of service times with respect to time with a fitted distribution

To assure the service time is positive, the distribution is defined as a piecewise function, expressed in Equation 3, where the probability of having a negative service time is zero. The term ξ is the corrective coefficient for "cutting" the left tail of the fitted density distribution, ρ_i is the mixing proportion, and φ_i is a non-standard normal density distribution ($\varphi_i(x) = \varphi(x|\mu_i, \sigma_i)$). Although the goodness of fit will be discussed in Chapter 4, for the rest of this chapter the service time density distribution will be assumed to be g(x). The values that will be used for each of the factors will be:

$$g(x) = \begin{cases} \xi\rho_1\varphi_1(x) + \xi\rho_2\varphi_2(x) & x \geq 0 \\ 0 & o.w. \end{cases}$$

Equation 3

3.3. Number of Servers

3.3.1. Hospital Data

The number of assigned beds to janitors for cleaning varies throughout the day. Therefore, the number of servers in our queueing model vary throughout the day as well to achieve acceptable performance. Although the number of janitors per shift is constant, they have other duties to attend, and the number of beds assigned to janitors is not constant throughout the shift. For the queueing models that were built in this chapter, the janitors were represented as servers in the system and the number of beds cleaned throughout the day was considered as the number of available janitors per hour. For the stationary queueing models, the number of servers will be minimized subject to having an average waiting time of fewer than 15 minutes. For the non-stationary queueing model in section 3.4.2, the average number of servers function will be discussed and compared to the curve in Figure 19.

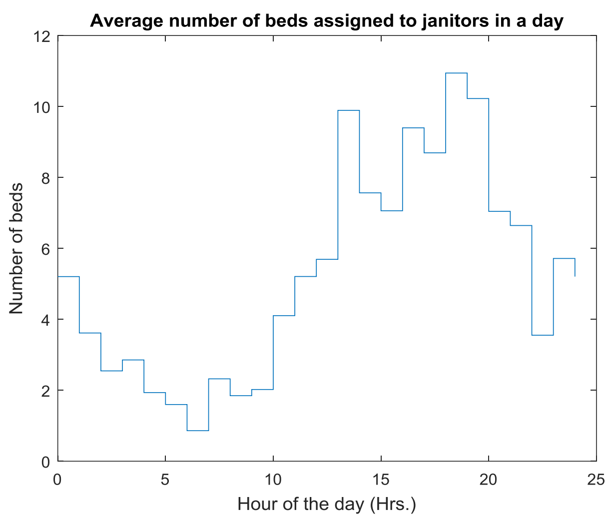


Figure 19 Average number of beds assigned to janitors in a day with respect to time

3.3.2. Shift Observations

Every 8 hours starting at 6:00AM, there is a dip in assigned beds to janitors. Although the effect may be minimal during throughout the day, there is a change in shift near the maxima of the peak-hours. The hour before the change in shift, management stops assigning incoming dirty beds to the leaving janitors to avoid the interruption of a bed cleaning or risking paying overtime on beds that take longer than the average time to clean. The reduced assignment phenomenon causes an accumulation of dirty beds in the system before the start of the next shift; which is termed as "stacking" and defined in the next subsection. Using queueing theory is inadequate to solve this specific problem when incorporating the described reduced assignment phenomenon, making simulation the best tool to solve this particular staffing problem.

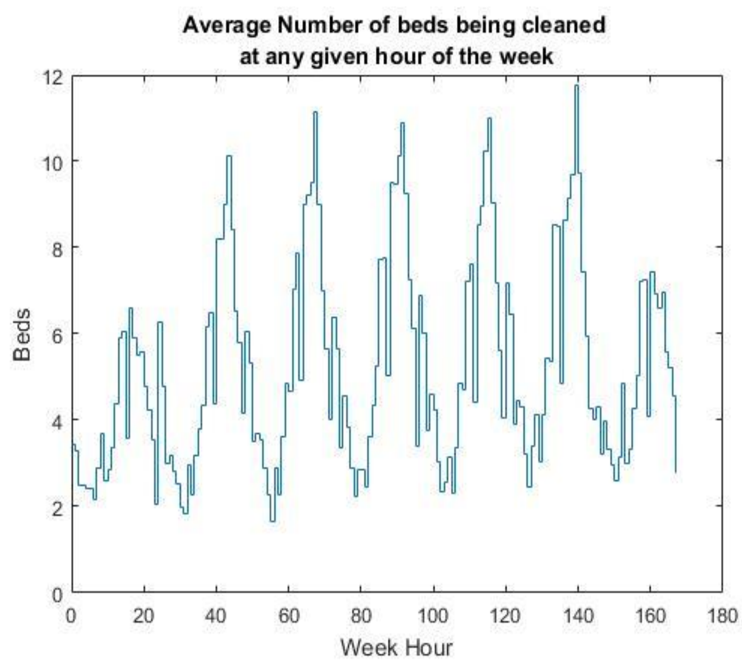


Figure 20 Average number of beds being cleaned each hour throughout the week

3.3.3. Observed Hourly Utilization

When the hospital stops assigning beds to janitors during the last hour compounded with being in the peak-hours of the day, results in temporary overutilization of the queueing system which may result in the “stacking” phenomenon described by the ESD management (as shown in Figure 21). The “stacking” phenomenon is used to describe the accumulation of hospital beds waiting to be assigned a janitor. When the arrival rate is higher than the product of the service rate and the number of servers, the utilization of the queueing system is higher than one. To validly apply queueing theory equations to make predictions on the behavior of a queueing system it is necessary to presume that the queueing model does not have a utilization higher than one. This presents a problem for the hospital because the janitors and managers alike are overwhelmed by the virtual work accumulated at the start of their shift resulting in the high turnover of employees and compounding the penalty of not addressing the bed cleaning process appropriately.

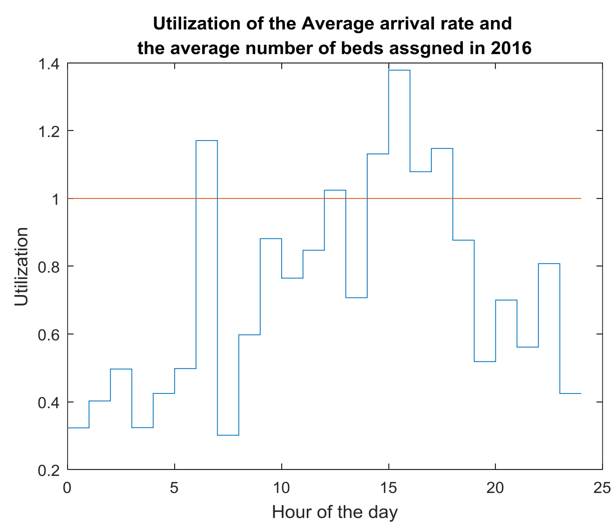


Figure 21 Average utilization curve with respect to time

3.4. Queueing Analysis Results

3.4.1. M/M/s and M/G/s Queueing Results

This section focuses on applying stationary queueing analysis to various non-stationary queueing systems. Various values for Equation 4 was computed iterating the number of servers to determine the highest tolerable average waiting time in queue for the queueing systems of interest. Equation 4 is used to determine W_q, average waiting time in queue; where λ is the arrival rate, μ is the service rate, c is the number of servers, and $C(c, \lambda/\mu)$ is the Erlang-C equation.

$$W_q = \frac{C(c, \lambda/\mu)}{c\mu - \lambda}$$

Equation 4

The service rate of all three queueing models is 1.277 beds/hr. Table 1 shows, shaded in grey, the number of servers necessary to achieve an average waiting time of fewer than 5 minutes. Since the Kingman's coefficient of the service time distribution is 3.175 and the goal is to reduce waiting times to less than 15 minutes, then the resulting average waiting time of the corresponding M/M/s queue has to be less than 4.7 minutes. The arrival rates stated in each column respectively correspond to the MLE of the stationary surmised inter-arrival times, the average of the intensity function, or time average of the arrivals (TA), low-arrival rate (LAR), and peak-hour rate (PR). The average arrival rate for 2016 was 5.3 beds/hour, and the service rate is 1.1 beds/hour, when there are 3 servers, the average waiting time should be 24 minutes. However, data suggests that the average waiting time for 2016 was 1.4 hours exposing further the disparity between the stationary queueing model and time-varying model. Further investigation on the matter revealed the arrivals and numbers of servers varied through each hour of the day to cope with the peak hours. Thus, there is a need for a more complex model and corresponding stochastic discrete-event simulation to cope with the transiency of the queueing system.

The second model for an M/M/c queue includes a peak-hours period of arrivals where the arrival rate is above average. Then, setting staffing requirements for both periods and assigning more staff when the system is in its peak-hours to reduce the average waiting time for customers.

		MLE	TA	LAR	PR
Arrival Rate (beds/hr)		7.865	5.152	2.161	9.639
Number of Servers	3	overload	overload	11.1	overload
	4	overload	overload	2.16	overload
	5	overload	27.54	0.48	overload
	6	overload	7.02	0.12	overload
	7	37.26	2.22	0	overload
	8	10.08	0.72	0	85.38
	9	3.66	0.24	0	16.86
	10	1.44	0.06	0	6.06
	11	0.54	0	0	2.46
	12	0.24	0	0	1.02
	13	0.06	0	0	0.42

Table 1 Average waiting times (in minutes) of an M/M/s queue

3.4.2. $M_t/G/\infty$ Queueing Results

The focus of this section is to solve for the $M_t/GI/\infty$ as a proxy to the $M_t/GI/s$ queueing model. The rationale is to set staffing requirements throughout time and observe behavior given infinite resources (Green, Kolesar, & Whitt, 2007). The time average behavior can be described in Equation 5.

$$m_\infty(t) = \int_{-\infty}^{t} [1 - G(t-u)]\lambda(u)du \,,$$

Equation 5

where $m_\infty(t)$ is the average number of servers attending customers in a queueing system with infinite resources, $G(x)$ is cumulative distribution function of Equation 3, and $\lambda(u)$ is the intensity function defined in Equation 1. However to set staffing requirements it may be required to identify the pertinent confidence interval to determine the ballpark figure of the probability of a customer waiting. The purpose of this analysis is to perform a qualitative comparison to the bed assignment behavior for the data collected for 2016. As presented in Figure 22, it seems as if the bed assignment average roughly corresponds to the analytical curve. However, it is evident that the second and third shift struggle with the bed assignment. Management of the UMMC have corroborated that there is a high turnover of employees due to the pressures of managing the bed cleaning process.

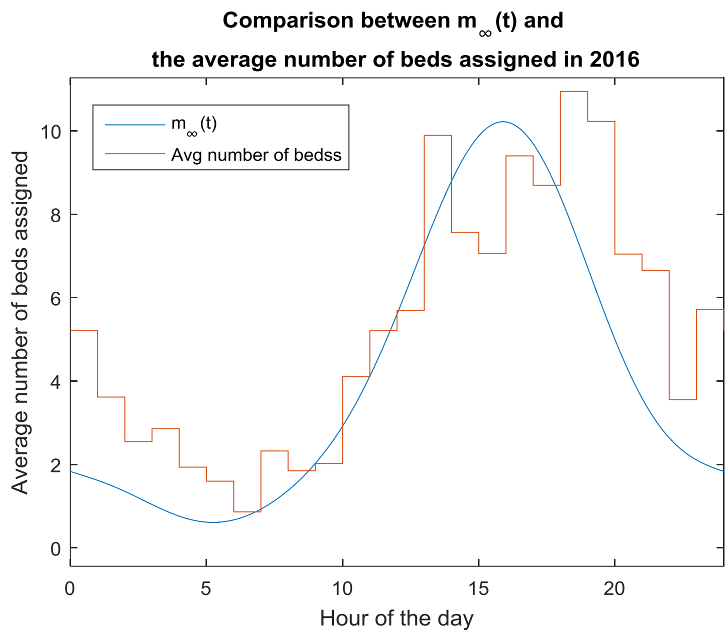


Figure 22 Comparison between the infinite resource queue and hospital data

Chapter 4: Discrete-Event Simulation

4.1. Simulated System

4.1.1. Simulation Model

The framework of the process is modeled as a first-come-first-served queueing system. Explicitly, a discrete-event simulation was scripted using the Matlab software describing a multi-server queue incorporating some events to emulate behaviors seen in the system. As a simplifying assumption, the service time is stationary, identically distributed, and independent of the wing, working shift, and janitor it was assigned to. It was shown in Chapter 3 that these assumptions do not hold and the service time, in fact, depends at the very least on shift and location. However, to reduce the development time of the simulation, the service time was sampled from the same distribution throughout the day.

The simulation model assumes a non-stationary Poisson arrival process consistent with the analysis provided in Chapter 3. The intensity function is modeled as piecewise constant rates at one-hour intervals for every hour of the week. The simulation takes as input a 168-element array as input for the intensity function and changes according to the simulation epoch. The simulation uses a different algorithm than the traditional ones presented in the Law (2013) textbook. Instead of using the thinning algorithm or the inverse of the intensity function, it uses an algorithm presented in Strömblad and Devapriya (2012) explained in further detail in Section 4.3. One period of the intensity function used in the simulation study is presented in Figure 23.

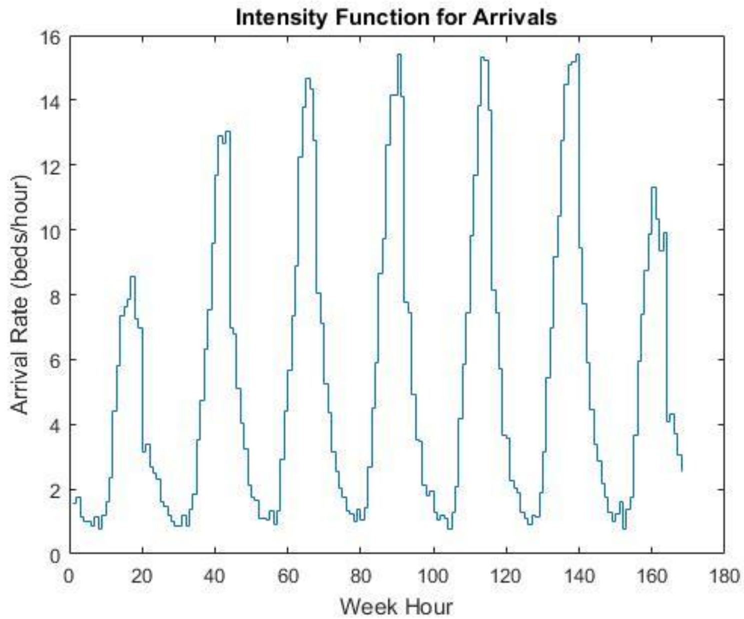


Figure 23 Intensity function of one week of the nonstationary Poisson process

This simulation study compared three methods to sample service time distributions. Since the service times are composed of a two-stage service, then service times can be sampled from the convolution distribution of the transit time and the cleaning time or the independent sampling of each of the phases. When sampling from each of the phases independently, this simulation study included the sampling from a distribution fitted to the data and the distributions with conjectured distributions and applying maximum likelihood estimators.

Finally, the simulation takes into consideration the varying number of servers per shift to represent the changing number of available janitors to clean hospital beds. The simulation also considers an end-of-shift behavior of not assigning a bed to janitors during the last hour of the shift.

4.1.2. Measures of Performance

The simulation gathers various statistics for the output. Specifically, he average waiting time of departed customers, the average time the queue is empty, and the average time the system is empty. These statistics are used in Chapter 5 to evaluate the system.

4.1.3. Simulation System Events

There are five events in this stochastic discrete-event simulation model. The events have allocated algorithms described in more detail in the next section. These events change the state of the system by either changing the number of beds in the cleaning process, the arrival process, or the number of servers available in the queueing model. The arrival event is used to simulate hospital bed arrivals and increasing the number of beds in the cleaning process modeled. an arrival rate change event was implemented to simulate the change of arrival rate at the corresponding hour. A departure event is used to simulate when a hospital bed leaves the cleaning process, and the clean bed is available for the next patient. Finally, the server shift change event and the end-of-shift laziness (EOSL) event which respectively increases or reduces the number of servers available to simulate characteristic behavior by the hospital.

4.1.4. Initialization Criteria

The simulation program starts by gathering all user-defined inputs into the simulation. Service times are generated from a function external to the simulation file and changed manually to generate service times in different manners. Another input of the simulation is a 168 positive valued vector to represent the arrival rates the simulation has to cycle through the sample run. These values are used in the computation to schedule the next arrival. Therefore, if the discharge discipline is changed in the future, the simulation can still be used to analyze and design the bed cleaning process if the other behaviors do not change. Meanwhile, the simulation also takes a 3 positive valued vector representing the number of servers available at each shift.

The simulation initializes with the system empty at midnight. The first arrival is scheduled stochastically with the appropriate arrival rate. The number of servers in the simulation starts at the amount set for the third shift, the first change of shift is scheduled at 7:00 AM in the simulation. The first EOSL event is scheduled at 6:00 AM in the simulation. Since initialization bias is not taken into consideration in the simulation, the estimation of the average waiting time is biased.

4.2. Event Algorithms

4.2.1. Arrival Algorithm

The arrival algorithm was adapted from the textbook Law (2013) with the example queueing simulation in its first chapter. When the simulation runs the arrival algorithm, the simulation schedules the next arrival. If there are no servers available, the simulation adds 1 to the number of customers in a queue and stores the incoming customer’s time of arrival. However, if there are servers available (i.e., the number of customers being served is less than the number of servers), then the simulation adds 1 to the number of customers being served, stores the arrival time and (lack of) delay and schedules the departure time for the serving customer. Since the departure time can be lesser than the previous earliest departure, then the simulation implements a simple bubble sorting algorithm to order the departure times list. Figure 24 presents a graphical representation of the algorithm described in this paragraph.

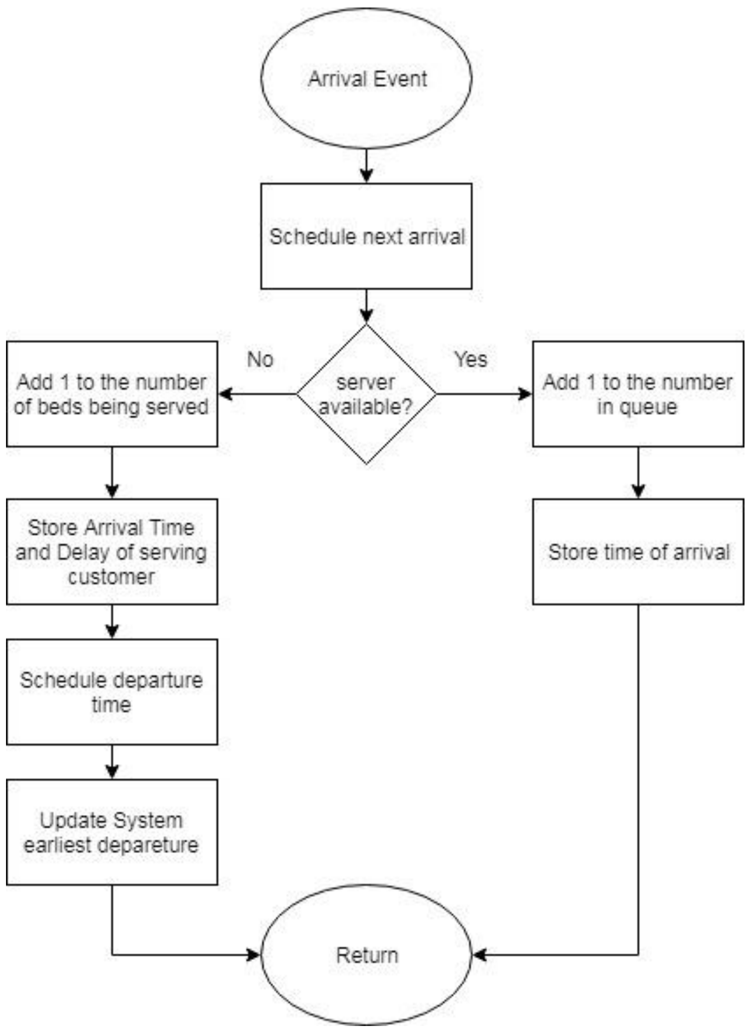


Figure 24 Flowchart for arrival routine

4.2.2. Arrival Rate Change Algorithm

The arrival rate change routine implements the algorithm presented in Strömblad and Devapriya (2012). However, instead of incorporating the algorithm into the arrival routine, the simulation treats the arrival rate change as an event within the simulation. That is, the event is scheduled with the other events in the simulation. When an arrival rate change event occurs, the arrival rate changes and reschedules the next arrival event appropriately. Afterwards, the simulation discards the next arrival scheduled and schedules a new arrival with the updated arrival rate. Figure 25 presents the algorithm flowchart pertaining to the arrival rate change event.

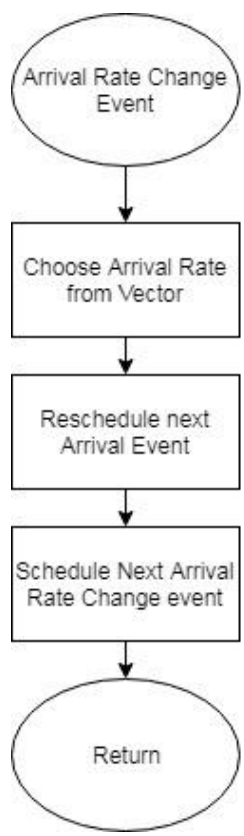


Figure 25 Flowchart for the arrival rate change routine

4.2.3. Departure Algorithm

Similar to the arrival event algorithm, the departure algorithm was adapted from the textbook by Law (2013) with the example queueing simulation in its first chapter. When the simulation runs the departure routine, the simulation increases the counter of the total number of customers served by one and gathers statistics from the departing customer. Then, the simulation eliminates the departing customer from the departures list and moves the departures up in the list. If there aren't any customers in the queue, the simulation subtracts the number of customers being served. Then, if there are no customers left, the simulation routine sets the earliest departure at infinity.

However, if there are customers in the queue, then the simulation first subtracts 1 from the number in the queue. Then the simulation stores the arrival time and delay of the serving customer. Then, the simulation generates a departure time for the customer entering service. Finally, the simulation moves each customer up one space in the queue and updates the schedule of the earliest departing customer. Figure 24 presents a graphical representation of the departure event algorithm described in this paragraph.

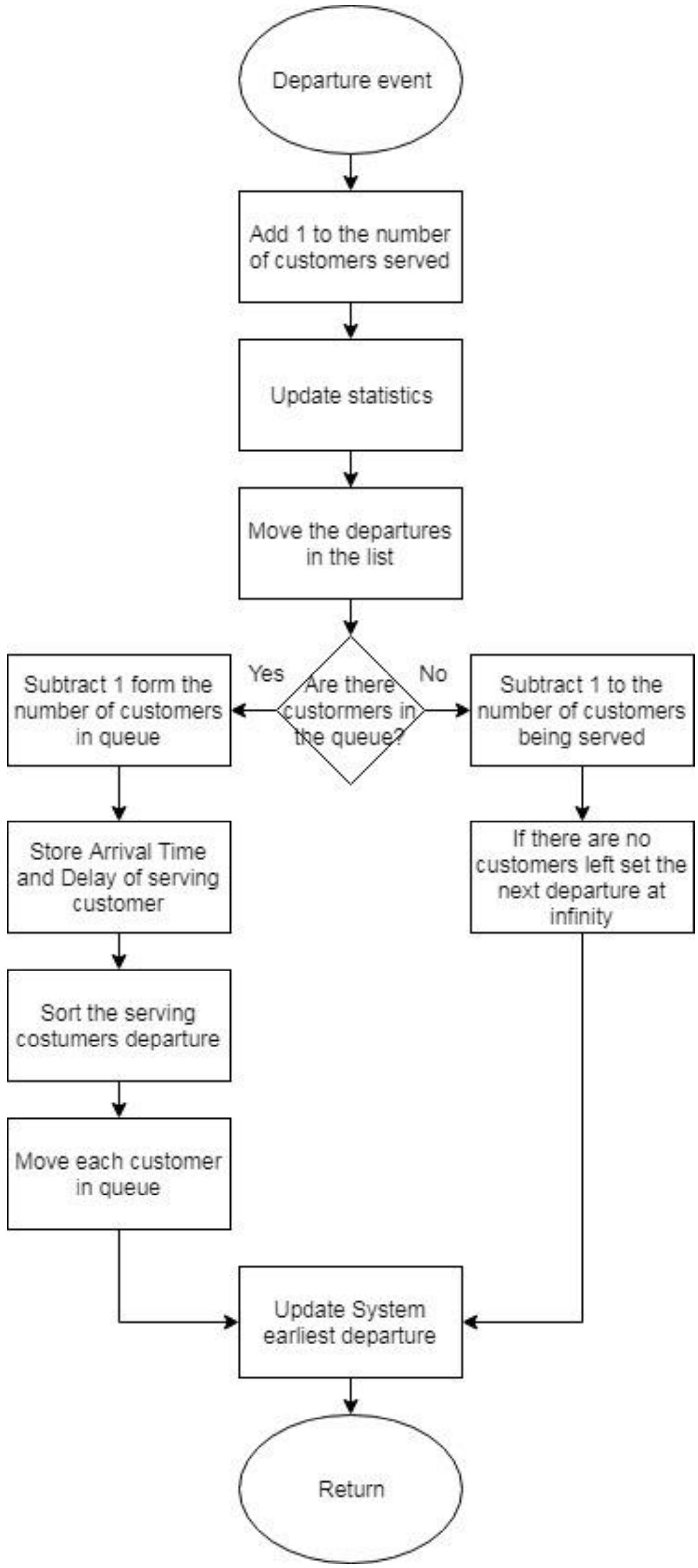


Figure 26 Flowchart for arrival routine

4.2.4. End-of-shift Algorithm

At the end of the shift, the simulation stops assigning customers to the janitors to emulate the corresponding behavior of the system. The behavior is affectionately called the End-of-Shift Laziness (EOSL) routine, and the simulation schedules the event one hour before the server shift change event. The routine changes the number of available servers without interrupting the customers already assigned to servers. This algorithm causes attrition of beds in service in the queueing system and accumulating beds for the next shift. Furthermore, Figure 27 presents the flowchart for the EOSL event.

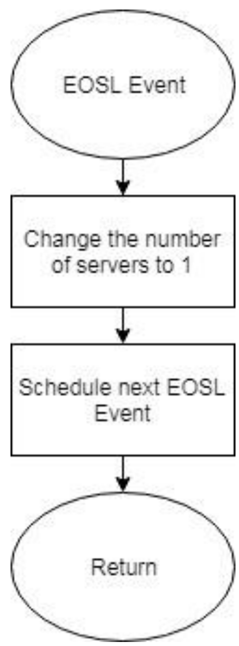


Figure 27 s Event Flowchart for end-of-shift laziness routine

4.2.5. Shift Change Algorithm

The server shift change event changes the number of servers every eight hours coinciding with the working shifts at the University of Maryland Medical Center. If a server is busy with a customer, the simulation does not interfere until the server finishes. However, if the number of servers increases and there are customers in the queue, the algorithm assigns customers to each server when possible. Figure 28 shows the flowchart regarding the server shift change for the simulated queueing system.

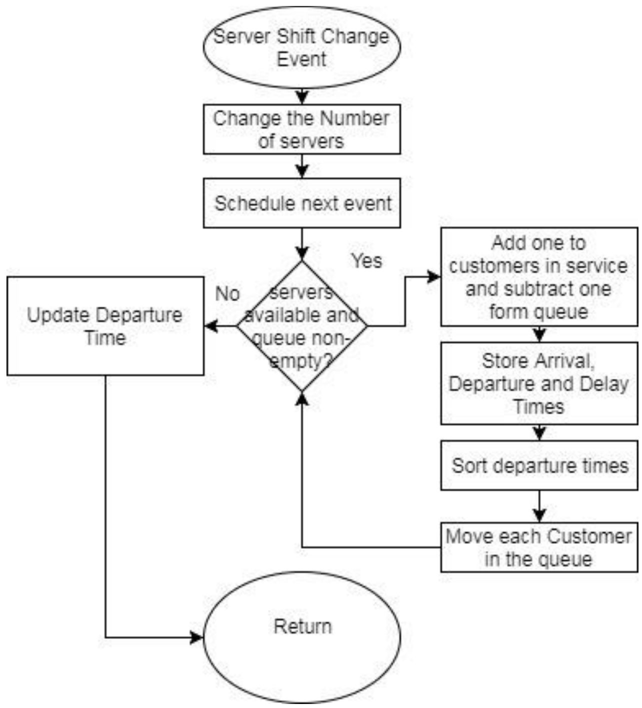


Figure 28 Flowchart for server shift change routine

4.3. <u>Input Analysis</u>

4.3.1. Input Distributions

The simulation uses an external function to generate random variates for the service time. This facilitates the use of different methods and distributions to generate service times. The service time is composed of two stages: the transit time and the cleaning time. First, the time it takes the janitor to reach the bed, which is denoted as the transit time. Then, the time it takes a janitor to clean the hospital bed, which is denoted as the cleaning time. In this study, we focused on three methods of generating service times: Convolution of independent sampling for each stage and sampling from the total service time distribution. Since most of the distributions fitted in the rest of this section have a negative part, the service time function rejects all negative variates and resamples them appropriately to ensure the service times are always positive.

In section 3.3.2, Figure 18, the fitting of the total service time distribution was presented stating the futility of attempting to find a “simple” distribution to fit large amounts of data. The total service time was fitted with a mixture of two normal distribution. One of the normal distributions has a mean of 48.55 minutes, a standard deviation of 27.21 minutes, and a proportion coefficient of 0.56. The second normal distribution has a mean of 42.24 minutes, a standard deviation of 10.6 minutes, and a proportion coefficient of 0.44. The fitted distribution has a root mean squared error of 48.66. More information about this and the rest of the fittings done in this section can be found in Appendix B.

For the convolutional independent sampling method, we sample independently from two distributions corresponding to each stage of the service process. For the transit time sampling distribution, Figure 29 presents the fitted curve of the mixture of two exponential distributions and a normal distribution. The normal distribution has a mean of 6.31 minutes and a standard deviation of 3.186 minutes with a mixture coefficient of 0.18. The first exponential distribution has a mean of

13.86 minutes and a mixture coefficient of 0.44. The second exponential distribution has a mean of 0.87 minutes and a mixture coefficient of 0.38. The fitted distribution has a root mean squared error of 84.66.

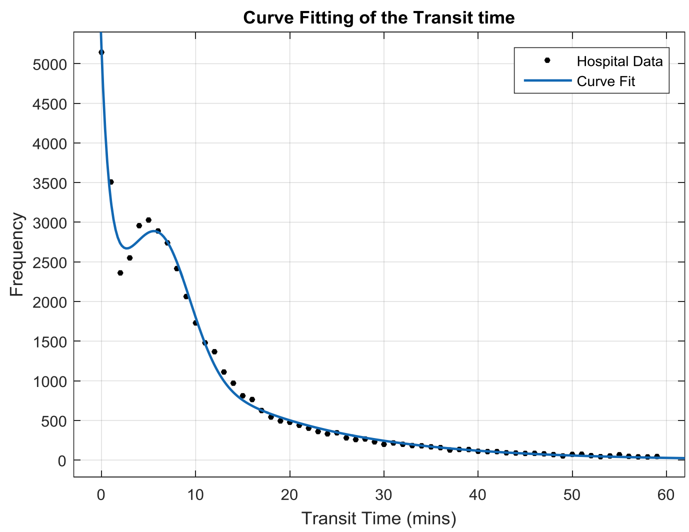


Figure 29 Curve fit of the transit time data

For the cleaning time sampling distribution, Figure 30 presents the fitted curve of the mixture of two normal distributions. One of the normal distributions has a mean of 35.48 minutes, a standard deviation of 9.381 minutes, and a proportion coefficient of 0.53. The second normal distribution has a mean of 36.59 minutes, a standard deviation of 24.52 minutes, and a proportion coefficient of 0.47. The fitted distribution has a root mean squared error of 43.45. Also, for simulation validation, we generate service time random variates from the sum of a normally distributed random variable for the cleaning time and an exponentially distributed random variable. The location and scale parameters are estimated using the maximum likelihood estimator (MLE) of the data.

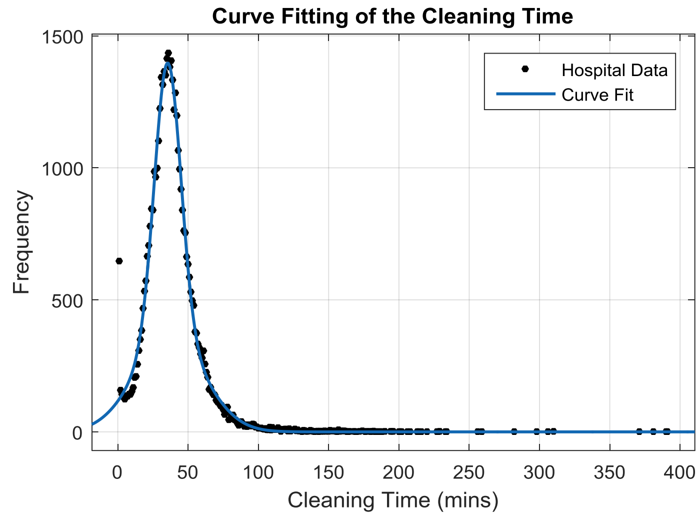


Figure 30 Curve fit of the cleaning time data

4.3.2. Arrival Process

The arrival process was modeled as exponentially distributed inter-arrival times with periodic changes of the arrival rate as shown in Figure 23.

4.4. Output Analysis

4.4.1. Simulation Verification

To verify the simulation, we compared and theoretical values of an analogous M/M/s queue to the output of the simulation using stationary arrival rates and exponentially distributed service times. Table 2 provides a summary of the test cases that were being used in the verification test. The test scenario consists of 5 test cases of 10 replications each. Unofficial test scenarios included sample path inspection and tests designed to output simulation errors such as overloaded queues.

Test Case	Name	# of Servers	Arrival Rate (1/Hr)	Service Rate (1/Hr)
1	Reference	13	10	1.09
2	Low Serv	12	10	1.09
3	High Serv	14	10	1.09
4	Low Arrive	13	8	1.09
5	High Arrive	13	12	1.09

Table 2 Verification Test Case Tables

The measure of performance of interest is average waiting time (or Response Time as shown in Table 3). The simulation was determined to be verified due to its success in predicting the average waiting times of various queueing models.

Test Case	Name	Expected Response Time (mins)	Average Waiting time (mins)	Standard Error (mins)	Verified
1	Reference	2.52	2.53	0.05	YES
2	Low Serv	5.7	5.8	0.18	YES
3	High Serv	1.14	1.12	0.02	YES
4	Low Arrive	0.42	0.4	0.02	YES
5	High Arrive	12.96	13.21	0.34	YES

Table 3 Verification Test results

4.4.2. Simulation Validation

The simulation was run using analogous inputs to the hospital data. Table 4 shows the input values for the simulation. The service distributions vary as specified in the previous section. The average number of servers available in each of the shifts were 6, 8, and 5 starting at the first shift at 7:00AM. The simulation was executed for one year, and 10 independent replications were run.

Test Factor/Input Values			
Intensity Function	Service time distribution	Number of servers per shift	simulation duration
Figure 23	Varies by test case	6, 8, 5	1 year

Table 4 Test Table

Table 5 shows the results of the test cases described in the previous paragraph. The first test case was the closest to behave on average as the sample data of the hospital and was given the comment of “Close but no cigar.” The second test case corresponding to the convolution of random variates generated from the fitted distributions, was the least successful because the average had the largest difference between the simulation and observation data. Therefore, it was given a comment of “Not even close.” Finally, the third test case corresponding to the convolution of random variates generated from the MLE distributions were better than the averages for the fitted distributions. The difference in behavior may be due to the lack of evidence when assuming that the transit time and cleaning time data is uncorrelated.

Although the simulation did not pass validation testing, it was accepted and deemed credible by UMMC leadership under current assumptions. Moreover, there are some discrepancies between the simulation and the hospital data, which constrains the utility of the simulation. Most of the differences may be the effect of lack of granularity or depth of the simulation. For example, the simulation considers the hospital as one queueing system without taking into consideration the allocation of janitors to different wings of the hospital or the working shift.

Test Case	Name	Expected Waiting Time (mins)	Average Waiting time (mins)	Standard Error (mins)	Validated	Comment
1	Service	82	83.4	0.4	NO	Close but no cigar
2	Convolution	82	66.6	0.3	NO	Close
3	MLE	82	98.5	0.4	NO	A bit better

Table 5 Validation Test Results

The limited depth of the simulation is the most obvious deficiency in the simulation analysis. The lack of depth limits the possible recommendations the simulation can provide. That is, this simulation can be used only to determine how many janitors may be needed throughout the week but not how to allocate them to each unit. Another impact is the accuracy of the outputs the simulation may have when the goals are too extreme or alien to the waiting times. For example, if the goal is to reduce the cleaning time, the simulation can only predict the effect of such reduction to the overall system. To correct this deficiency, the next iteration of the simulation should contain the individual performance of each wing per shift and incorporate it into an overarching discrete-event simulation such as the StratBAM.

4.4.3. Comparison with data

Figure 31 shows the comparison of the data gathered from the simulation model and the hospital data. The plot shows the simulation models roughly corresponds to the hospital data. However, there is behavior that is not captured within the simulation justifying the need of either increasing the depth of the simulation, or using software that is commercially available.

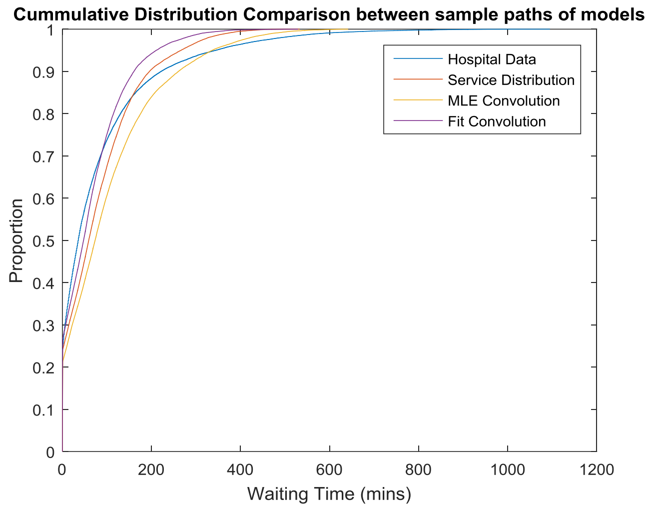


Figure 31 Sample Path Comparison between simulation models

Chapter 5: Optimization via Simulation

5.1. <u>Simulation Study</u>

5.1.1. Simulation Assumptions

Since most simulation models have multiple factors influencing a given behavior of interest, there are multiple ways to search for an optimal set of options to get a definitive (or approximate) optimal behavior. The field Simulation Optimization offers various tools to either search for an optimal set of parameters (optimization via simulation) or converging to a solution faster from a set of possible options (ranking and selection, and variance reduction). Optimization via simulation requires an objective cost function to perform a trade-off between parameters to search for the solution with the lowest cost. In the case of the cleaning system simulation presented in this thesis, there is an implicit cost of delaying hospital beds for the next patient and a direct cost of committing resources to improve the cleaning process. However, this work has not presented a cost model that accurately describes the cleaning system. Therefore, the focus of this chapter is to search for an optimal number of servers that minimizes the number of servers in the first and second shift, subject to having a waiting time of fewer than 15 minutes.

Per the results in the M/M/s model expressed in Chapter 3, the range for the number of servers in either the first or second shift will vary discretely from 5 to 14. Since the arrival rate throughout the third shift is relatively low, the simulation model assumes 5 servers are fixed for the third shift in all the iterations of the system. Also, as discussed in Chapter 4, there are various ways to describe the service times that show similar results. However, since sampling from the total service time distribution generates average waiting times closest to the average measured in the data provided, then the rest of this chapter will optimize the model sampling from the total service time distribution. The simulation study will consider an initial 10 replications for each parameter within the set of possible values with simulation duration of one year; where each replication serves in the order of 46,000 customers. In Chapter 3 it was shown that the standard error of 10 simulation runs yield standard errors of less than 1 minute.

The principal measure of performance under study is the average waiting time. However, there are other measures of performance that may differentiate between systems that have acceptable waiting times. As mentioned in the first paragraph of this section, there is no objective function developed for this simulation study. Therefore, the effects of the queue and server utilization are not be objectively described. Specifically, queue and server utilization needs may differ between shifts and the distinction or preference are not shown in the results.

5.2. Response Output

5.2.1. Average Waiting Time in Queue

Assigning more servers to the second shift has a higher impact on the queueing system's average waiting times. As shown in Figure 32, the system achieves minimum accepted behavior when the number of servers available is higher than 10. When assuming that the cost of janitors in the first and second shifts are equal, however, the system achieves the optimization criteria in three different configurations at the same total number of servers. Therefore, there are multiple options to consider in other measures of performance; which motivates the rest of the analysis done in this chapter.

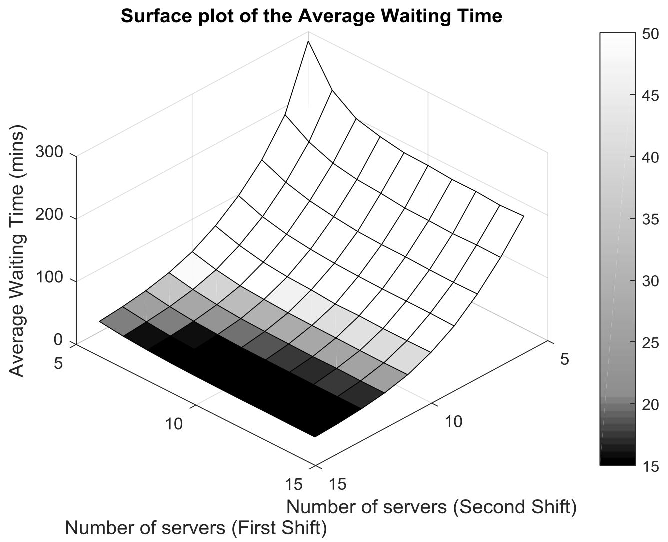


Figure 32 Response time with respect to the number of servers in the first and second shift

With the exception of the lowest amount of servers possible, the standard errors for all other average waiting times are less than one minute (Figure 33).

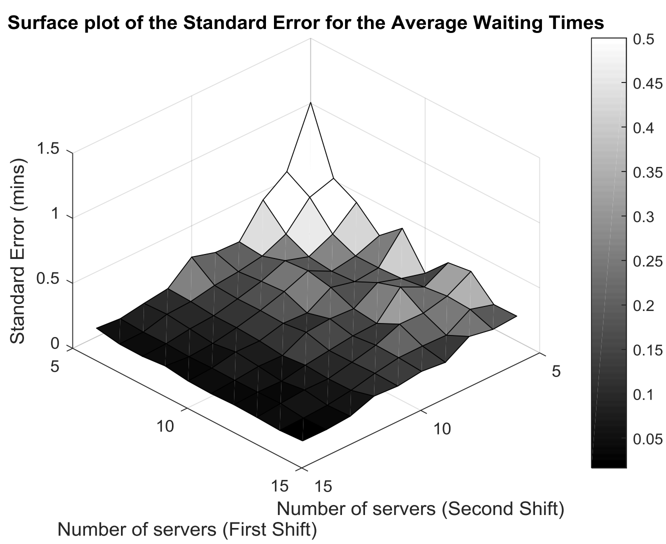


Figure 33 Standard Error for Average Response Time

5.2.2. Server Utilization

In this thesis, server utilization is defined as the percentage of time the system is not empty. As shown in Figure 34, the utilization averages do not drastically change after there are more than 10 servers on the second shift. Meaning, that after assigning more than 10 servers, the system idleness behaves as if it had close to infinite resources.

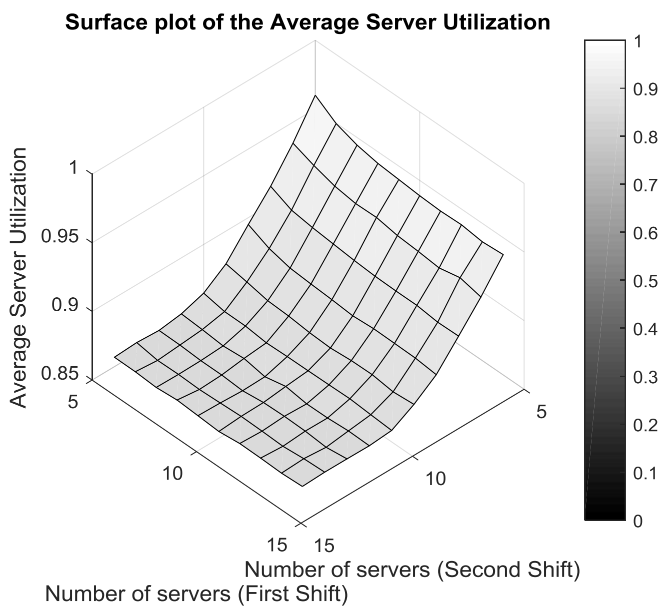


Figure 34 Response time with respect to the number of servers in the first and second shift

Figure 35 Standard Error for the Average Server Utilization

5.2.3. Queue Utilization

Figure 36 shows the time average of the queue utilization for the simulation replications. In this thesis, queue utilization is defined as the percentage of time the queue is not empty. In the region where the system meets minimum performance, the queue utilization is approximately less than 22%.

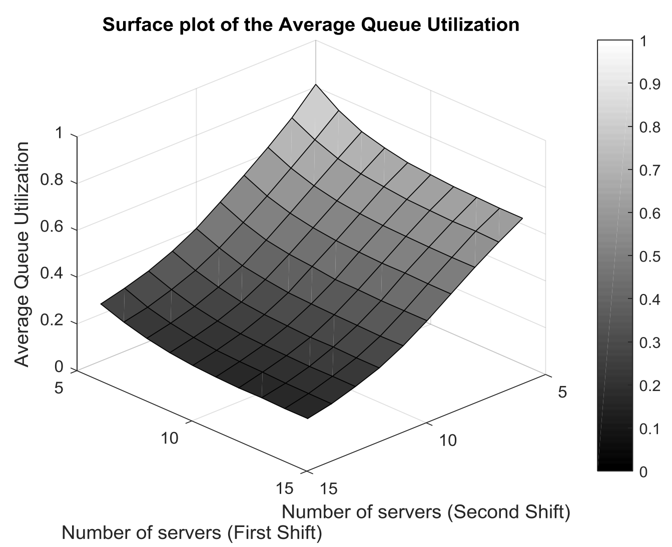


Figure 36 Waiting time with respect to the number of servers in the first and second shift

Figure 37 Standard Error for the Average Queue Utilizations

5.3. <u>*Optimization*</u>

5.3.1. Possible Solutions

There are three possible solutions that satisfy the optimization constraints. The system configurations will be named: Whiskey, Tango, and Foxtrot. Each of the systems has a total of 20 servers working 8-hour shifts. However, each system has allocated the servers differently. Table 6 shows the configuration summary and output performance; where the uncertainty expressed is the standard error.

System Name	Servers in the first shift	Servers in the Second shift	Average Waiting Time (mins)	Average Server Utilization	Average Queue Utilization
Whiskey	6	14	12.51±0.04	0.8567±0.0025	0.2094±0.0004
Tango	7	13	12.19±0.04	0.8555±0.0014	0.2062±0.0004
Foxtrot	8	12	13.95±0.05	0.8557±0.0026	0.2175±0.0003

Table 6 System configurations and performance outputs based on

5.3.2. Ranking and Selection Methodology

In this section, we present an implementation of a statistical procedure developed by Dudewicz and Dalal (1975) and presented in the textbook by Law (2003). In the procedure, there is a two-stage sampling of the possible systems, where the algorithm attempts to minimize the number of additional samples to guarantee a predetermined probability of correct selection (P(CS)) for a given "indifference" value (d*).

In the first stage of the procedure, estimated variances based on the initial number of replications (n_0) are used to compute the additional number of replications needed to select the best system. We let X_{ij} denote the output performance of interest (in this case the average waiting time) from the *j*th replication of the *i*th system. Then the sample averages of the first stage of the procedure are given by

$$\bar{X}_i^{(1)}(n_0) = \frac{\sum_{j=1}^{n_0} X_{ij}}{n_0},$$

and the sample variances are given by

$$S_i^2(n_0) = \frac{\sum_{j=1}^{n_0}\left[X_{ij} - \bar{X}_i^{(1)}(n_0)\right]^2}{n_0 - 1}.$$

The total number of required replications (N_i) was computed using Equation 10.3 in Law (2003), given by

$$N_i = \max\left\{n_0 + 1\,, \left\lceil \frac{h_1^2\, S_i^2(n_0)}{(d^*)^2} \right\rceil\right\}$$

where h_1, in Dudewicz and Dalal (1975), depends on the number of systems being compared, the probability of correct selection, and the initial number of samples.

The second stage of the procedure involves simulating the additional number of replications and performing a weighted sum of the averages in the first and second set of replications. After simulating the second number of samples, the second-stage averages are given by

$$\bar{X}_i^{(2)}(N_i - n_0) = \frac{\sum_{j=n_0+1}^{N_i} X_{ij}}{N_i - n_0}.$$

Then defining the weights as

$$W_{1i} = \frac{n_0}{N_i}\left\{1 + \sqrt{1 - \frac{N_i}{n_0}\left[1 - \frac{(N_i - n_0)(d^*)^2}{h_1^2 S_i(n_0)}\right]}\right\}$$

for the first-stage average and $W_{2i} = 1 - W_{1i}$ for the second-stage average, to finally perform the weighted sum given by

$$\bar{X}_i(N_i) = W_{1i}\bar{X}_i^{(1)}(n_0) + W_{2i}\bar{X}_i^{(2)}(N_i - n_0)$$

and selecting the system with the lowest weighted average $\bar{X}_i(N_i)$.

5.3.3. Results

For this study, we choose the probability of correct selection to be more than 0.95 ($P(CS) = 0.95$), an indifference value of 0.1 minutes ($d^* = 0.1$), and ten initial replications ($n_0 = 10$). The results of this multiple system comparison and selection are summarized in Table 7, where the Tango system performs best.

System Identifier (i)	$\bar{X}_i^{(1)}(n_0)$	$S_i^2(n_0)$	N_i	$\bar{X}_i^{(2)}(N_i - n_0)$	W_{1i}	W_{2i}	$\bar{X}_i(N_i)$
W	12.51	0.13	81	12.43	0.10	0.90	12.44
T	12.19	0.18	116	12.27	0.12	0.88	12.26
F	13.95	0.23	144	14.01	0.07	0.93	14.01

Table 7 Ranking and Selection results ($n_0 = 10$)

Chapter 6: Conclusions and Future Research

6.1. Recommendations

6.1.1. Staffing Requirements

To satisfy the need for reduced waiting times, we set the staffing requirements and allocate it to an element level component. Therefore the stakeholder requirement of "The system shall assign a bed to a janitor after no more than 15 minutes on average" can be derived into the following system level requirements:

1. The system shall have at least 7 janitors available for bed cleaning on the first shift
2. The system shall have at least 13 janitors available for bed cleaning on the second shift
3. The system shall have at least 5 janitors available for bed cleaning on the third shift

6.1.2. System Integration and Testing

Optimizing the simulation and queueing models are excellent decision making tools. To solve the delay problem within the hospital context, however, there is a need to develop test procedures to verify that the recommendations really satisfies the needs of the hospital. That is, the delay problem can only be considered solved when the system of interest, the Bed Cleaning System, really does have average waiting times of less than 15 minutes.

6.2. Conclusions

6.2.1. Queueing Model

Overall, limitations in queueing theory prevent the development of taking into consideration all the factors that have a measurable impact on the cleaning process. Solving for the average staffing period (shift) may not give accurate results when the intensity function varies significantly throughout the period. An analytically tractable method to set staffing requirements $M_t/GI/s$ has not been developed yet. However, using the $M_t/GI/\infty$ model as proxy gives us insight into how the staffing requirement should look.

6.2.2. Discrete-Event Simulation

Developing discrete-event simulations of a system is an iterative process limited by the number of resources or time to produce the most accurate results. Although the discrete-event simulation built for the system of interest took into consideration the driving aspects, there are still other factors that were not taken into consideration. Lack of depth and fidelity of the model prevents us from making pinpoint recommendations, and appropriate testing may be required.

6.2.3. Simulation Optimization

The simulation optimization chapter presented a decision-making tool when presented with various systems with similar performance and similar perceived cost.

Mainly, the use of Dudewicz-Dalal procedure was implemented and for the staffing requirements for the Bed Cleaning System.

6.3. Future Work

6.3.1. StratBAM Integration

There are various options to develop this work depending on the field of interest. From a health care perspective, there may be interest in using the simulation tool to make decisions. Therefore, as mentioned in the first Chapter of this thesis, there is a need to simulate the turnaround times accurately in StratBAM simulations. The simulation model can be integrated as a subroutine within an overarching simulation. Therefore adapting the simulation code into the StratBAM simulation is the preferred next course of action.

6.3.2. Systems Engineering

From a systems engineering perspective, there is the need to implement testing for verification of the staffing requirements presented in the first section of this chapter. Moreover, the development and execution of testing procedures is another area of possibility for future work.

6.3.3. Operations Research

From an operations research point of view, the author needs to familiarize himself with the field of operations research. There are still many problems within health care that go unsolved due to the unfamiliarity with operations research tools.

Therefore, another direction of this thesis research is to apply queueing theory and other operations research to other problems within the health care domain.

Appendices

Appendix A – Main Code

```
% Hospital Bed Cleaning Process Simulation
% Thesis Simulation
% Author: Arturo J. Davila Andino

clc, clear, close all
% Inputs
sim_mode = 1;% input(''); % 0 analytical, 1 empirical
if sim_mode == 0
    ARva = 10; % input(' ');
    SR = 1.09; % Static Factor
elseif sim_mode == 1
    ARva = 0;
    SR = 1/47;
end

ARv = arv(ARva,sim_mode);   %Arrival Rate Vector (constant, 0cons
1empirical)
load('fc.mat')
% fils=2;
% cols=4;
Servers = [fils+4 cols+4 5];       %Servers per shift starts at 7am

% Initialization
% Define Stopping Time - The system will enter the loop at least
once
SD = 365*24*1; % The simulation runs for a whole year

% Specify the number of events for the timing function
ne = 6; % Five Events: Arrival, Departure, Server Shift, Rate
change, End
V = zeros(1,ne);

%Initialize the event schedule
for ii=1:ne
    %Set all Values to infinity
    V(ii)=inf;
end

N_max = 300; %No more than 300 beds on system

%Initialize System States
SS   = 0;                % Server Status Initialization
N_q = 0;                 % Number In Queue Initialization
T_arr=zeros(N_max,1);    % Times of arrival (Queue) Initialization
sim_clock=0;             % System time initialization
AR = ARv(1);             % First arrival time block
```

```
ser = Servers(3);        % Initialization of number of servers
max_s = max(Servers);    % Initialization of the server vector length
sto_D = zeros(1,max_s+1); % Initialization of the server vector
storage
sto_A = zeros(1,max_s+1); % Initialization of the associated
arrivals
sto_E = inf*ones(1,max_s+1); % Initialization of the associated
departure

% Initialize statistical counters
A_n = 0; % number of customers served Initialization
D_T = 0; % Total Delay Initialization
S_T = 0;
N_t = 0; % N(t) Initialization
T_45= 0; % ****NOTE: we can install an MOP for the system
N_0 = 0; % Time system empty Initialization
Q_0 = 0; % Time Empty queue Initialization


% Schedule Events
V(1) = sim_clock + -log(rand())/AR; % Schedule First Arrival
            % V(2) is the departure schedule (already at infinity)
V(3) = 7;   % Schedule First Shift Change
V(4) = 1;   % Schedule First Arrival Rate Change
V(5) = SD;  % Schedule Simulation End
V(6) = 7 - 1;   % Schedule First inefficiency
if sim_mode == 0
    V(6) = inf;
end

% Sample Path

%FCFS
D=[];
ijk=1;

% Run the simulation while more customers are still needed.
while sim_clock <= SD
    % Determine the next event
    dt=min(V);
    next_event_type=find(V==dt,1);


    % Update Statistical Counters

    N_t = N_t + (SS+ N_q)*(dt-sim_clock); % Area under N
    if SS == 0
        % Time the system is empty
        N_0 = N_0 + (dt-sim_clock);
    end
    if N_q == 0
        % Time the queue is empty
        Q_0 = Q_0+ (dt-sim_clock);
    end
```

```
% Advance Simulation Clock
sim_clock=dt;

% Invoke the appropriate event function.

switch next_event_type

    case 1
        % Arrival Routine

        % Schedule Next arrival
        V(1) = sim_clock + -log(rand())/AR;

        % Determine if a server is available
        if SS < ser
            % If there are servers available
            % Add 1 to the number of beds being served
            SS = SS + 1;

            % Store Arrival Time and Delay of serving customer
            sto_A(SS) = sim_clock;
            sto_D(SS) = 0;
            sto_E(SS) = rvgser(SR,sim_mode, sim_clock);

            % Order the departures
            for kk=max_s+1:-1:2
                if sto_E(kk)<sto_E(kk-1)
                    token_1 = sto_E(kk);
                    token_2 = sto_A(kk);
                    token_3 = sto_D(kk);
                    sto_E(kk) = sto_E(kk-1);
                    sto_A(kk) = sto_A(kk-1);
                    sto_D(kk) = sto_D(kk-1);
                    sto_E(kk-1) = token_1;
                    sto_A(kk-1) = token_2;
                    sto_D(kk-1) = token_3;
                end
            end
            % Update System earliest depareture
            V(2) = sto_E(1);

        elseif SS >= ser
            % Add 1 to the number in queue
            N_q = N_q + 1;

            if N_q >= N_max
                % Kill-switch if the system is unstable
                disp('Arturo you are stupid! Queue is full!')
                D_T=2;
                A_n=1;
                break
            end
```

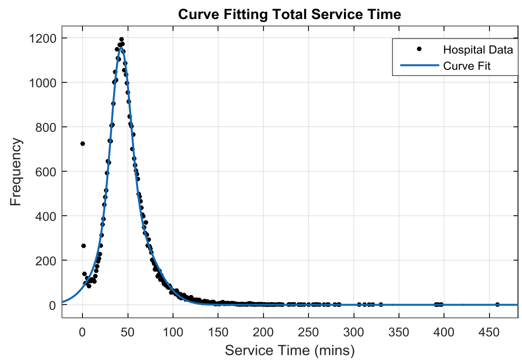
```
                % Store time of arrival
                T_arr(N_q) = sim_clock;
            end

        case 2
            % Departure Routine

            % Add 1 to the number of customers served
            A_n = A_n + 1; %


            % Update statistics
            D_T = D_T + sto_D(1);   % Sum of delay time
            D(ijk) = sto_D(1);
            ijk = ijk + 1;
            S_T = S_T + sto_E(1) - sto_A(1);
            if sim_clock - sto_A(1) > 45/60
                T_45 = T_45 + 1;    % Fraction of customers spent
more than
                                    % *** minutes in system
            end

            % Move the departures
            for ii=2:max_s+1
                sto_E(ii-1) = sto_E(ii);
                sto_A(ii-1) = sto_A(ii);
                sto_D(ii-1) = sto_D(ii);
            end
            if N_q == 0 && SS>0
                % Queue is empty
                % Subtract the number of servers
                SS = SS-1;

            elseif SS==ser
                % Queue is not empty and there are too many
customers being
                % served
                % Note: For further development implement a reneging
alg.
                % Subtract 1 form the number in queue
                N_q = N_q - 1;

                % Store Arrival Time and Delay of serving customer
                sto_A(SS) = T_arr(1);
                sto_D(SS) = sim_clock - T_arr(1);
                sto_E(SS) = rvgser(SR,sim_mode, sim_clock); %
Generate Departure
                % Order the departures
                for kk=max_s+1:-1:2
                    if sto_E(kk)<sto_E(kk-1)
                        token_1 = sto_E(kk);
                        token_2 = sto_A(kk);
                        token_3 = sto_D(kk);
                        sto_E(kk) = sto_E(kk-1);
                        sto_A(kk) = sto_A(kk-1);
```

```
                    sto_D(kk) = sto_D(kk-1);
                    sto_E(kk-1) = token_1;
                    sto_A(kk-1) = token_2;
                    sto_D(kk-1) = token_3;
                end
            end

            % Move each customer in queue
            for jj=1:sum(T_arr>0)
                T_arr(jj) = T_arr(jj+1);
            end
        else
            SS=SS-1;
        end
         % Schedule next departure
         V(2) = sto_E(1);
    case 3
        % Server Shift Routine

        % Change the Number of Servers
        nn = mod((sim_clock-7)/8,3)+1;
        ser = Servers(nn);
        % Schedule the next shift change
        V(3) = sim_clock + 8;

        if ser > SS && N_q > 0
            while SS < ser && N_q > 0
                %PUT PEOPLE TO WORK
                %Add one to customers being served and subtract
one from
                %queue
                SS = SS + 1;
                N_q = N_q - 1;
                % Schedule Departure of the customer
                % Store Arrival Time and Delay of serving
customer
                sto_A(SS) = T_arr(1);
                sto_D(SS) = sim_clock - T_arr(1);
                sto_E(SS) = rvgser(SR,sim_mode, sim_clock); %
Generate Departure

                % Order the departures
                for kk=max_s+1:-1:2
                    if sto_E(kk)<sto_E(kk-1)
                        token_1 = sto_E(kk);
                        token_2 = sto_A(kk);
                        token_3 = sto_D(kk);
                        sto_E(kk) = sto_E(kk-1);
                        sto_A(kk) = sto_A(kk-1);
                        sto_D(kk) = sto_D(kk-1);
                        sto_E(kk-1) = token_1;
                        sto_A(kk-1) = token_2;
                        sto_D(kk-1) = token_3;
                    end
                end
```

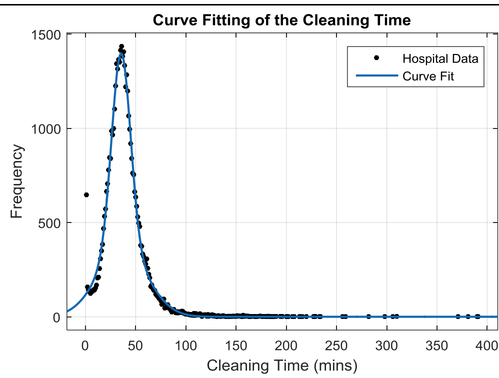
```
                    % Move each customer in queue
                    for jj=1:sum(T_arr>0)
                        T_arr(jj) = T_arr(jj+1);
                    end
                end
            end
            V(2)=sto_E(1);
        case 4
            % Arrival Rate Routine

            % Change the Arrival Rate
            nn1 = mod(sim_clock,168)+1;
            AR = ARv(nn1);

            % Schedule NEW next arrival - Reference Stromblad
            V(1) = sim_clock - log(rand())/AR;

            % Schedule Next Rate Change
            V(4) = sim_clock + 1;

        case 5
            % Simulation End
%             disp('simulation end')
%             disp('Press [ENTER] to generate Report')
            % Break the loop
            break
        case 6
            % Server End Shift Routine

            % Change the Number of Servers
            ser = 1;
            % Schedule the next shift change
            V(6) = sim_clock + 8;

    end
end
%pause
%disp(num2str(D_T*60/A_n))

% Write report heading and input parameters.
% clc
% disp('SIMULATION REPORT')
% disp(' ')
% disp('INPUTS')
% disp('Multiple-server queueing system')
% disp(['Queueing Discipline:     ','FCFS'])
%
% if sim_mode == 0
%     disp('The arrival Process and service time distributions are
Markovian')
%     disp(['Mean Arrival Rate:
',num2str(round(A_n/(sim_clock)),2),'  beds per minutes'])
%     disp(['Mean Service Time:       ',num2str(round(1/SR*60)),'
minutes'])
%     disp(['Number of Servers first shift: ',num2str(Servers(1))])
```

```
%     disp(['Number of Servers second shift: ',num2str(Servers(2))])
%     disp(['Number of Servers third shift: ',num2str(Servers(3))])
% elseif sim_mode == 1
% disp('The arrival Process and service time distributions are 
empirical')
% disp(['Mean interarrival time: 
',num2str(round((365*24)*60/A_n),2),' minutes'])
% disp(['Mean service time:       ',num2str(1/SR),' minutes'])
% disp(['Number of Servers first shift: ',num2str(Servers(1))])
% disp(['Number of Servers second shift: ',num2str(Servers(2))])
% disp(['Number of Servers third shift: ',num2str(Servers(3))])
% end
% 
% disp(' ')
% disp(' ')
% disp('OUTPUT')
% disp(' ')
% disp('Customer Averages')
%disp(['Average time that a job waits in queue: 
',num2str(D_T*60/A_n,4),' minutes'])
% %disp(['Fraction of jobs that spent more than 4.5 minutes in the 
system: ', num2str(T_45/A_n)])
% disp(' ')
% disp('Time Averages')
% disp(['Time-average number in the system: 
',...
%     num2str(N_t/sim_clock)])
% % disp(['Fraction of time that the system is not idle 
(utilization):  ',...
%     % 12num2str(1-(N_0/sim_clock))])
% disp(['Fraction of time that the queue is nonempty: 
',...
%     num2str(1-(Q_0/sim_clock))])
% disp(' ')
```

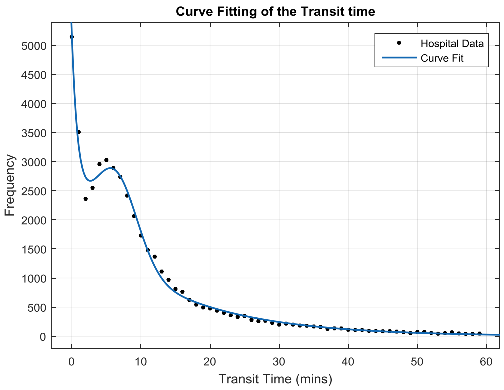
Appendix B



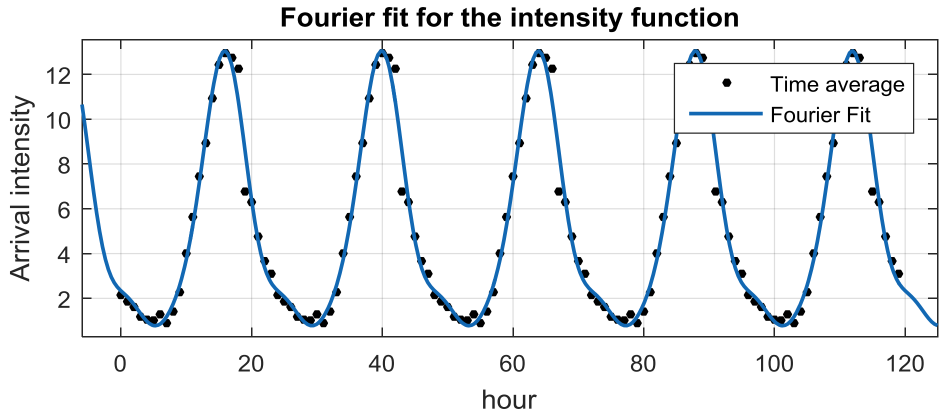
General model:
f(x) = a1*exp(-((x-b1)/c1)^2/2)/sqrt(2*pi*c1^2)
+ a2*exp(-((x-b2)/c2)^2/2)/sqrt(2*pi*c2^2)
Coefficients (with 95% confidence bounds):
a1 = 2.629e+04 (2.289e+04, 2.968e+04)
a2 = 2.072e+04 (1.727e+04, 2.416e+04)
b1 = 48.55 (46.4, 50.7)
b2 = 42.24 (41.67, 42.81)
c1 = 27.21 (24.42, 30)
c2 = 10.6 (9.734, 11.47)
Goodness of fit:
SSE: 5.612e+05
R-square: 0.9772
Adjusted R-square: 0.9767
RMSE: 48.66



General model:
f(x) = a1*exp(-((x-b1)/c1)^2/2)/sqrt(2*pi*c1^2)
+ a2*exp(-((x-b2)/c2)^2/2)/sqrt(2*pi*c2^2)
Coefficients (with 95% confidence bounds):
a1 = 2.458e+04 (2.112e+04, 2.805e+04)
a2 = 2.163e+04 (1.85e+04, 2.477e+04)
b1 = 35.48 (35.14, 35.83)
b2 = 36.59 (34.82, 38.35)
c1 = 9.381 (8.777, 9.985)
c2 = 24.52 (20.98, 28.07)
Goodness of fit:
SSE: 3.852e+05
R-square: 0.9873
Adjusted R-square: 0.987
RMSE: 43.45

 	General model: f(x) = a1*exp(-x/b1)/b1+ a2*exp(-x/b2)/b2+ a3*exp(-((x-b3)/c3)^2/2)/sqrt(2*pi*c3^2) Coefficients (with 95% confidence bounds): a1 = 2503 (1518, 3488) a2 = 2.942e+04 (2.524e+04, 3.36e+04) a3 = 1.201e+04 (7743, 1.627e+04) b1 = 0.8786 (0.6466, 1.111) b2 = 13.86 (11.25, 16.47) b3 = 6.31 (5.864, 6.757) c3 = 3.186 (2.613 , 3.758) Goodness of fit: SSE: 3.798e+05 R-square: 0.9948 Adjusted R-square: 0.9942 RMSE: 84.66

Appendix C



General model Fourier3:

f(x) = a0 + a1*cos(x*w) + b1*sin(x*w) +
a2*cos(2*x*w) + b2*sin(2*x*w) + a3*cos(3*x*w) + b3*sin(3*x*w)

Coefficients (with 95% confidence bounds):

a0 = 5.277 (5.174, 5.38)
a1 = -2.848 (-3.084, -2.612)
b1 = -4.823 (-5.011, -4.635)
a2 = -0.5277 (-0.727, -0.3284)
b2 = 1.693 (1.539, 1.848)
a3 = 0.4379 (0.2916, 0.5842)
b3 = 0.1265 (-0.02721, 0.2803)
w = 0.2618 (0.2612, 0.2624)

Goodness of fit:

SSE: 36
R-square: 0.983
Adjusted R-square: 0.982
RMSE: 0.567

Appendix D – ANOVA Code

```
clc, clear, close all

load('6sp.mat')

% anovan(y,{Air,Drp,Con},'model','full')
%[p,tbl,stats,terms]=anovan(y,{Air,Con,Drp},'model',3,'varnames',...
%    {'Air','Con','Drp'})


% anova1(y,Air,'group',{'Airborne'}), title('Airborne Cleaning
Protocol')...
%     ,ylabel('Cleaning Time (mins)'),xlabel('Protocol Prescence');
% anova1(y,BP), title('Bed Priority')...
%     ,ylabel('Cleaning Time (mins)'),xlabel('Normal - Next -
Stat');
% anova1(y,Con), title('Contact Cleaning Protocol')...
%     ,ylabel('Cleaning Time (mins)'),xlabel('Protocol Prescence');
% anova1(y,Drp), title('Droplet Cleaning Protocol')...
%     ,ylabel('Cleaning Time (mins)'),xlabel('Protocol Prescence');
% anova1(y,Sh), title('Shift'),xlabel('Shift Number')...
%     ,ylabel('Cleaning Time (mins)');
% anova1(y,Unit), title('Unit')...
%     ,ylabel('Cleaning Time (mins)'),xlabel('Unit Abreviation');


Label=unique(Unit);
k=1;
for ii=1:numel(Label)-1
    x=y(find(strcmp(Unit,Label(ii))));
    for jj=ii+1:numel(Label)
        z=y(find(strcmp(Unit,Label(jj))));
        [h,p,ci,stats]=ttest2(x,z);
%         M(k)=p;
%         if p>0.05
%             disp(['p-value: ',num2str(p)])
%             disp([Label(ii),Label(jj)])
%             disp('')
%         end
        k=k+1;
    end
end

Label=unique(BP);
k=1;
for ii=1:numel(Label)-1
    x=y(find(BP==Label(ii)));
    for jj=ii+1:numel(Label)
        z=y(find(BP==Label(jj)));
        [h,p,ci,stats]=ttest2(x,z);
        M1(k)=p;
```

```
        if p>0.05
            disp(['p-value: ',num2str(p)])
            disp([Label(ii),Label(jj)])
            disp('')
        end
        k=k+1;
    end
end

Label=unique(Air);
k=1;
for ii=1:numel(Label)-1
    x=y(find(Air==Label(ii)));
    for jj=ii+1:numel(Label)
        z=y(find(Air==Label(jj)));
        [h1,p1,ci1,stats1]=ttest2(x,z);
%         M2=[h,p,ci,stats];
%         if p>0.05
%             disp(['p-value: ',num2str(p)])
%             disp([Label(ii),Label(jj)])
%             disp('')
%         end
        k=k+1;
    end
end

Label=unique(Con);
k=1;
for ii=1:numel(Label)-1
    x=y(find(Con==Label(ii)));
    for jj=ii+1:numel(Label)
        z=y(find(Con==Label(jj)));
        [h2,p2,ci2,stats2]=ttest2(x,z);
%         M3=[h,p,ci,stats];
%         if p>0.05
%             disp(['p-value: ',num2str(p)])
%             disp([Label(ii),Label(jj)])
%             disp('')
%         end
        k=k+1;
    end
end

Label=unique(Drp);
k=1;
for ii=1:numel(Label)-1
    x=y(find(Drp==Label(ii)));
    for jj=ii+1:numel(Label)
        z=y(find(Drp==Label(jj)));
        [h3,p3,ci3,stats3]=ttest2(x,z);
%         M4=[h,p,ci,stats];
%         if p>0.05
%             disp(['p-value: ',num2str(p)])
%             disp([Label(ii),Label(jj)])
%             disp('')
%         end
```

```
        k=k+1;
    end
end

Label=unique(Sh);
k=1;
for ii=1:numel(Label)-1
    x=y(find(Sh==Label(ii)));
    for jj=ii+1:numel(Label)
        z=y(find(Sh==Label(jj)));
        [h,p,ci,stats]=ttest2(x,z);
        M5(k)=p;
%         if p>0.05
%             disp(['p-value: ',num2str(p)])
%             disp([Label(ii),Label(jj)])
%             disp('')
%         end
        k=k+1;
    end
end
```

Appendix E – Optimization

```
clc, clear, close all
M=zeros(10,10);
S=M;
Mq=M;
Sq=M;
Ms=M;
Ss=M;
save('M.mat','M','S','Mq','Sq','Ms','Ss')
fils=0;
cols=0;
reps=10;
for fil=1:10
    fils=fils+1;
    cols=0;
    for col=1:10
        cols=cols+1;
        save('fc.mat','fils','cols')
        Pa=zeros(10,3);
        counter=0;
        save('PQ.mat','Pa','counter');

        for ii=1:10
        main
            load('PQ.mat')
            counter=counter+1;
            Pa(counter,1)=D_T*60/A_n;
            Pa(counter,2)=1-(Q_0/sim_clock);
            Pa(counter,3)=1-(N_0/sim_clock);
```

```
            save('PQ.mat','Pa','counter');
        end
        load('fc.mat')
        load('M.mat')
        M(fils, cols)=mean(Pa(1:10,1));
        S(fils, cols)=std(Pa(1:10,1));
        Mq(fils, cols)=mean(Pa(1:10,2));
        Sq(fils, cols)=std(Pa(1:10,2));
        Ms(fils, cols)=mean(Pa(1:10,3));
        Ss(fils, cols)=std(Pa(1:10,3));
        save('M.mat','M','S','Mq','Sq','Ms','Ss')
    end
end
```

Bibliography

Capan, M., Khojandi, A., Denton, B. T., Williams, K. D., Ayer, T., Chhatwal, J., . . . Schwartz, J. S. (2017). From Data to Improved Decisions: Operations Research in Healthcare Delivery. *Medical decision making: an international journal of the Society for Medical Decision Making, 272989X17705636*.

Chen, P.-S., & Lin, M.-H. (2017). Development of simulation optimization methods for solving patient referral problems in the hospital-collaboration environment. *Journal of Biomedical Informatics*, 148-158.

Devapriya, P., Strömblad, C. T., Bailey, M. D., Frazier, S., Bulger, J., Kemberling, S. T., & Wood, K. E. (2015). StratBAM: A Discrete-Event Simulation Model to Support Strategic Hospital Bed Capacity Decisions. *Journal of Medical Systems*.

Dudewicz, E. J., & Dalal, S. R. (1975). Allocation of observations in ranking and selection with unequal variances. *Sankhyā: The Indian Journal of Statistics, Series B*, 28-78.

Fomundam, S., & Herrmann, J. (2007). *A survey of queuing applications in healthcare.* Greenbelt, MD: Institute for Systems Research.

Fu, M. C. (2004). Optimization via simulation: A review. *Annals of Operations Research, 53*, 199-247.

Fu, M. C. (2016). *Handbook of simulation optimization.* Springer.

Gans, N., Koole, G., & Technion, A. M. (2003). Call Center Measurements, Data Models and Data Analysis. *Manufacturing and Service Operations Management (M&SOM).*

Garner, J. S. (2018, March 12). Retrieved from Center for Disease and Control Prevention: https://wonder.cdc.gov/wonder/prevguid/p0000419/p0000419.asp#head003004000000000

Gorunescu, F., McClean, S. I., & Millard, P. H. (2002). A queueing model for bed-occupancy management and planning of hospitals. *Journal of the Operational Research Society, 53(1)*, 19-24.

Green, L., & Kolesar, P. (1991). The pointwise stationary approximation for queues with nonstationary arrivals. *Management Science, 37(1)*, 84-97.

Green, L., Kolesar, P., & Whitt, W. (2007). Time-Varying Demand and Staffing Requirements. *Production and Operations Management*, 13-39.

INCOSE. (2014). *INCOSE Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities.* International Council on Systems Engineering.

Khayal, I. S., & Farid, A. M. (2017). The Need for Systems Tools in the Practice of Clinical Medicine. *Systems Engineering*.

Koizumi, N. (2002). *Koizumi, N. (2002). A Queueing Network Model with Blocking: Anaylsis of Congested Patients Flows in Mental Health Systems.* (Doctoral dissertation, University of Pennsylvania).

Kokangul, A., Akcan, S., & Narli, M. (2017). Optimizing nurse capacity in a teaching hospital neonatal intensive care unit. *Health care management science*, 276-285.

Kopach-Konrad, R. e. (2007). Applying Systems Engineering Principles in Improving Heath Care Delivery. *Journal of general internal medicine*, 431-437.

Law, A. M. (2003). *Simulation Modeling and Analysis.* Tucson, AZ: McGraw Hill.

Lemoine, A. J. (1981). On Queues with Periodic Poisson Input. *Journal of Applied Probability, Vol. 18, No.4*, 889-900.

President's Council of Advisors on Science and Technology (US). (2014). *Report to the President, Better Health Care and Lower Costs: Accelerating Improvement Through Systems Engineering.* Executive Office of the President, President's Council of Advisors on Science and Technology.

Pujowidianto, N. A., Lee, L. H., Pedrielli, G., Chen, C. H., & Li, H. (2016). Constrained optimizaton for hospital bed allocation via discrete event simulation with nested partitions. *Proceedings of the 2016 Winter Simulation Conference*, 1916-1925.

SE Vision 2025. (2018, 01 13). Retrieved from INCOSE: https://www.incose.org/AboutSE/sevision

Smith, C. D., Spackman, T., Brommer, K., Stewart, M. W., Vizzini, M., Frye, J., & Rupp, W. C. (2013). Re-engineering the operating room using variability methodology to improve health care value. *Journal of the American College of Surgeons*, 559-568.

Stewart, D., Glass, T. F., & Ferrand, Y. B. (2017). Simulation-Based Design of ED Operations with Care Streams to Optimize Care Delivery and Reduce Length of Stay in the Emergency Department. *Journal of medical systems*, 162.

Valdez, R. S., Brennan, P. F., & Ramly, E. (2010). *Industrial and systems engineering and health care: critical areas of research.* Agency for Healthcare Research and Quality, US Department of Health and Human Services.

Wagner, T. H., & Jopling, J. K. (2017). Déjà Vu: Introducing Operations Research to Health Care. *Society for Medical Decision Making*.

Whitt, W. (2016). Queues with time-varying arrival rates: A bibliography. *Working paper*.

Willingham, A. (2018, January 13). *Video shows patient left at cold bus stop at night, wearing only a hospital gown*. Retrieved from CNN: http://www.cnn.com/2018/01/11/health/baltimore-hospital-patient-video-bus-stop-trnd/index.html

Zhen, L., Wang, K., Hu, H., & Chang, D. (2014). A simulation optimization framework for ambulance deployment and relocation problems. *Computers & INdustrial Engineering*, 12-23.