Optimizing Intensive Care Unit Discharge Decisions Policy for Burn Units

By Nate Witte & Zitao Shen

April 2019

Abstract

As the most critical unit for a hospital, the Intensive Care Unit (ICU)'s beds are extremely valuable, especially when there is high demand for admission into the ICU. Therefore, the demand-driven discharge on ICU patients has become a method for maintaining the ICU's efficiency. However, to discharge a patient can bring some trade-offs to hospitals and patients. Based on this phenomena, this paper wants to provide some new perspectives on designing an ICU discharge policy for a burn center, by considering and comparing different policies, which emphasized on either ICU operational efficiency or patients' welfare or both. These policies included ranking patients based on costs of QALY, mortality, and readmission and discharging the patient with lowest cost. The team designed a mathematical model and performed a simulation study to analyze the situation, and discovered some heuristics the optimal policy should have.

1 Introduction

The Intensive Care Unit (ICU) is an incredibly critical department for a hospital because the ICU accommodates the most acute inpatients who require 24 hours continued care with advanced equipment and staff. Hence, due to the high demand for the equipment and highly experienced team, the cost per bed in ICU can be the highest among all types of wards. For maintaining the efficiency of ICU, the hospital needs to discharge patients and transfer them to other wards for making space for new patients. To remove a patient from the ICU, it will depend on the recovery status of this patient, and sometimes on the demand on ICU.

However, discharging a patient can bring some trade-offs to hospitals and the patient. For instance, if the decision on releasing the patient is made too early, then the probability for this patient to be readmitted will increase [**1**]. Also, if readmission happens, patients' expected length of stay (LOS) can be longer than their first time's length of stay [**2**]. On the other hand, although the hospital can keep a patient longer to make sure she/he has a higher chance to get recovered, it will decrease ICU's efficiency. Furthermore, there is evidence to

show the longer patients have spent in the hospital, the lower patients' likelihood of departure will be [**1**]. This is because some kinds of patients, such as patients affected by burn injuries, have a higher chance to contract an infection due to being exposed in an environment with multiple sources of infections, such as an ICU. Hence, having an operationally efficient discharge policy is exceptionally critical to control the patient flow in hospitals.

Also, unlike other kinds of wards, the ICU's management has a lot of uncertainties. For instance, because patients can be directly admitted to an ICU, diverted by ambulances, or returned from surgeries, it can be challenging to forecast the patient arrival patterns. Additionally, other uncertainty can be the patients' length of stay (LOS). The inpatients' acute LOS depends on their health status and ICU's demand, which is affected by uncertainty under a stochastic setting. Based on the above conditions, making an ICU discharge policy design is difficult in terms of modeling and calculation.

For this paper, the research team wants to provide insights on designing an ICU discharge policy for a specific case, i.e., a burn center's ICU, throughout demonstrating the trade-offs between ICU operational efficiency and the patients' welfare. Also, the paper proposes several implementable heuristics for hospitals with burn units to consider in the end. The technical approach can be concluded as followed: By assuming that ICUs can practice a demand-driven discharge policy, the paper shed some light on the required decision making for admitting/rejecting an incoming burn patient; and selecting the inpatient to be discharged from the ICU before completion of his/her treatment. We outline a Markov Decision Problem to model an actual ICU where the ICU manager can decide whether to accept a patient or not; and which inpatient should be prematurely discharged when the demand-driven discharge cannot be prevented. Every decision will incur several costs associated with the related medical metric, such as quality-adjusted life years (QALY), mortality risk, and readmission risk. The amount of those patients' cost depends on that patient's health status. However, due to the "curse of the dimensionality," the team realized that in this case solving a real sized problem is almost unrealistic in any reasonable time frame. Therefore, instead of solving the MDP directly, the rest of the analysis will compare various policies, which are proposed and emphasized on different metrics, by using an extensive simulation study to make admission decisions and nominate a patient for demand-driven discharge, if necessary.

In the next section, the paper presents a review of the related literature. Section 3 formulates the problem as an MDP problem. Section 4 introduces the cost metrics which will be used in the analysis. Section 5 explains different discharge policies. In Section 6, we carry out an extensive simulation study considering different costs, different levels of occupancy and policies discussed in Section 5. In Section 7, the paper summarizes the results of the simulation study. Section 8 discusses the results. Finally, Section 9 offers a conclusion and future opportunities.

2 Literature Review

Since ICU discharge policy design is a significant and challenging problem, many OR/OM papers attempt to address this issue. Therefore, in this section, this project will give a brief review on selected papers The selected literature can be grouped into the following categories: 1) papers emphasized on ICU modeling and discharge policy designing, and 2) papers emphasized on burn related clinical characteristics.

2.1 Methodology Comparison on Selected Papers

All selected papers focus on designing an ICU discharge related policy for controlling and optimizing patient flow in the ICU. Based on their research questions, their policy designs can be divided into two categories:

Pure Premature Discharge Strategy (PPD Strategy): Under this strategy, researchers are only concerned about premature discharge. Because of this, researchers usually focus more on readmission risk. Since the inpatients' readmission risk will be determined by the severities of the inpatients' diseases and the actual LOS.

Admission Discharge Strategy (AD Strategy): Unlike the Pure Premature Discharge Strategy, people consider their admission controls and early discharge policies together. The cost/reward can be generated from the remaining LOS or survival benefits.For the survival benefit, it can be intuitively understood as the benefit of admitting a patient into the ICU. Therefore, the readmission related variables are not emphasized in their models. Furthermore, unlike papers which only concentrate on the admission control, they can apply a premature discharge policy to control the patient flow on the other side.

All selected models are stochastic models. Most of them formulate this ICU problem as an MDP with finite time horizon, and one uses a discrete time Markov chain. Most papers do not explain why they use finite time horizon in their MDP, but this setting makes the computation and the accompanying simulation easier. The following parts in this section will compare those models in terms of assumptions, states, actions, value functions, data, and solvability. Table 1 lists the summarized information on the selected literature.

Among those papers, common critical assumptions are made on arrival patterns and the distribution of LOS. For arrival patterns, most papers assume that at most one new patient can arrive in each period, except Dobson et al. [**3**], which allow multiple people to arrive at one time, and allow for a non-stationary arrival pattern. The rest of the papers assume that patients' arrival patterns follow a stationary Poisson distribution. In the terms of the distribution of LOS, most of the papers, excluding Dobson et al. [**3**], assume LOS's distribution should have memoryless properties at the start, so they believe that the exponential distribution or geometric distribution can be a good choice to consider. However, several publications determine it might not be the case from their empirical studies and change their assumptions in their simulation. For example, Chan et al. [**1**] and Li et al. [**5**] find that the LOS is log-normal

Table 1: Summary table of the selected publications on ICU policy design

Publications	**Assumptions on Arrival Pattern**	**Assumptions on Patient LOS Distribution**	**Cost Metric**	**Strategy Type**	**Model Type**
Chan et al. [**1**]	Stationary Poisson	Log-normal	Δ-readmission load	PPD Strategy	MDP
Hosseinifard et al. [**2**]	Stationary Poisson	Weibull	Patients' Readmission LOS	PPD Strategy	MDP
Dobson et al. [**3**]	Non-stationary Poisson	Non-memoryless distributed	Patients' Remaining LOS	AD Strategy	Markov Chain
Li et al. [**4**]	Stationary Poisson	Exponential	Survival Benefit	AD Strategy	MDP
Li et al. [**5**]	Stationary Poisson	Exponential	Survival Benefit	AD Strategy	MDP

distributed. Hosseinifard et al. [**2**] simulate LOS from the Weibull distribution based on others' empirical studies.

Different design strategy will lead to different cost metrics and their related value functions. For Chan et al. [**1**], the value function is to find the optimal policy to minimize the expected cost, which is -readmission load, over all periods. They define their cost metric, -readmission load, by considering two kinds of cost. The first kind of cost is the patient health-related cost, i.e., due to premature discharge, inpatients will face a higher risk of readmission. The other one is a system-related cost from the unexpected readmitted patient. The readmission will directly increase this cost on the accommodation for a new ICU patient. -readmission load depends on inpatients' types, and it is the difference between the ratio of the probability of readmission and expected readmission LOS of a patient given he/she is naturally discharged and the ratio of the same two metrics but given he/she is premature discharged. Similarly, Hosseinifard et al. [**2**] also considered the readmission probability and expected readmission LOS in their model. However, their objective is to minimize the total expected readmission LOS, instead of defining their cost metric.

However, to policy designers who take the AD Strategy, the readmission related variables become less critical. Instead, they select other kinds of metrics, such as the remaining LOS and the survival benefits. In Li et al. [**4**] and Li et al.[**5**], their objectives are both to maximize the expected total survival benefits of patients over all periods. Before formally introducing the survival benefits, a new concept, survival probability, should be added. A bivariate probit model will be used to predict the survival probability. This model uses crowding in the ICU as the identifying variable and controls observable patients' medical

characteristics. This quantity will later be used to calculate the survival benefits, which is defined as the probability difference, i.e., the difference between the survival probability of a patient given the patient is admitted to the ICU and the same probability but given the patient is deferred to a normal ward. Then, Li et al.[**4**] and Li et al.[**5**] vary their value functions based on the beds' allocation for different patient's classes. Li et al.[**4**] divide all situations into the following: cases with all beds full of patients from a single class, cases with some empty beds, and cases with all beds full of patients from both classes. Due to the relaxed assumption on the rejection of patients, the model has a more complex structure in Li et al.[**5**], but their ideas are similar. Unlike the other selected papers, Dobson et al.[**3**] uses a discrete time Markov Chain (DTMC) to model the daily patient flow in ICU. They define patients' remaining LOS to be the difference between patients' natural LOS and patients' accrued LOS. In their DTMC model, the ICU will rank the inpatients' and incoming patients' priority based on the length of the remaining LOS. Then the ICU will keep the first C patients by rejecting the other incoming patients and discharging the rest of the inpatients, where C is the total number of beds. Therefore, the transition probability will depend on the severities of the incoming patient's health condition and the inpatients' recovery. In the end, they want to find the stationary probability after proving the stationary probability exists and is unique.

2.2 Burn-Specific Literature

The project team has determined that a simpler version of the models described above is a specialized ICU, where every patient is of the same health class. An example of a specialized ICU is a burn unit. A burn unit operates the same as an ICU in terms of having random demand and a set capacity of beds, but the difference is that all the patients in the unit are burn victims. Therefore, this will simplify the model, and will allow the research team to focus on other areas of the project instead of defining multiple health classes, and multiple health states within each health class. While this model can be applied to any ICU that specializes in some disease or injury, the project team decided to apply this model to burn units, and researched burn units in order to get a greater understanding of the parameters that will be used.

Burn units have been studied intently in many research publications [**6**, **7**, **8**, **10**, **11**, **12**, **14**]. A very helpful aspect of these papers is that some of the papers studied key metrics associated with burn units [**6**, **7**, **10**, **11**, **12**, **14**]. These papers reported average length of stay of patients, mortality rates, QALY,and readmission rates. This will allow the project team to evaluate the realism of the simulation, along with calibrating the parameters to match these results. It was also found that burn patients are more expensive than regular ICU patients [**8**]. This shows the contrast between financial incentives and healthcare incentives a burn unit might have. The burn unit has to carefully consider the excess cost of these patients while also determining health risk when having to discharge a patient. Additionally, there is a certified burn center in Minneapolis, Hennepin Healthcare's Burn Center [**9**]. This unit has 17 beds, and works with an average

of 10 patients per day, and sometimes up to 20 [**9**]. This is a good example of a burn center that the model can be based on.

3 Model

In this section, the paper will characterize the ICU admission control system combined with demand-driven discharge feature, and an MDP model. In an ICU of a specialized hospital, every patient is associated with a health state. Those health states represent how severe the patient's health condition is. More specifically, in this case, the patient's health states will solely depend on his/her approximate ICU length stay. This is due to the fact that the average ICU LOS of patients with different degrees of burn is very different. Usually, the higher the burn degree is, the longer the patient will stay in ICU. Therefore, it is reasonable to expect doctors know the new patients' approximate LOS after knowing their degree of burns. Notice that the degree of burns will be mainly considered when to assign a new patient to a health state. Later, the patients' health status can change during the stay based on his/her recovery. Overall, those health states will decide the patients' chance of death or recovering. In other words, those states will determine the cost of discharging and diverting patients to other medical units. Also, if the patient is in the state 0, it means this patient will leave from the ICU. It can be the patient is naturally discharged from the ICU, or dead. In other words, the number of patients of state 0 can represent the number of available beds in ICU.

Suppose when the ICU capacity is running out and a new patient requests to get into the ICU, ICU managers will decide to discharge the patient with less severity or to divert the new patient to other medical units. If they decide to admit the current patient, they will also need to decide who should be discharged. The notations used in the model can be found in Table 2.

3.1 Assumptions

The following assumptions are made to ensure the previous decision process can be mathematically demonstrated:

1. In this model, the time will be discrete and indexed by $t \in [0, T]$.
2. In each time period t, at most one new patient will be considered to accommodate into the current ICU, or to be rejected and transferred to another hospital's ICU based on his/her own health state and other patients' health states. If the new patient will be accommodated and the ICU is full, the discharge will happen to a current inpatient.
3. This project will only focus on one specific health class, such as burn victims. The patients will be classified into M health states. The patient is allowed to transfer between health states while in the ICU. The higher the state, the worse the patient's health is. Figure 1 gives an example on the Markov Chain.

Table 2: Table of notations

Symbol	Definition
T	Number of time periods
M	Number of health statuses
λ	Arrival rate
$q_{t,m}$	The probability that a newly arriving patient at time t is health state m
B	Number of beds
$\mathcal{S}$	State space
s_t	State at t period
$x_{m,t}$	The number of patients of status m in the ICU during the period t
$y_{m,t}$	A binary indicator for the arrival of a patient with status m during the period t
$\mathcal{A}(s_t)$	Set of actions that can be taken during the period t
$a(s_t)_m$	A variable for a decision on discharging a patient with status m in period t or not
P	The probability matrix
$\mathcal{N}$	The nature discharge matrix
$\mathcal{M}$	The mortality matrix
$\mathcal{T}$	The transition matrix
δ_i	The random component for health status i
R_t	A variable indicates the health status of an arriving patient during the period t.

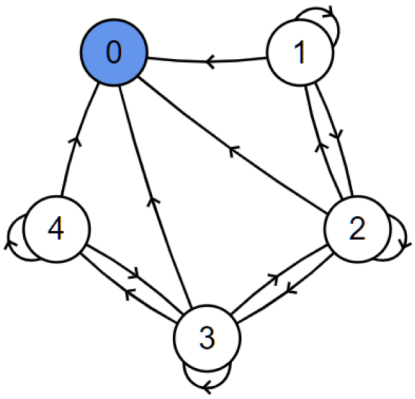

Figure 1: An example of health states being reachable from other states for a 5 state system.

4. The transition probability will not change during throughout time. This means the change on the patient's health status is uncorrelated with the patient's accrued LOS. The change in patients' health status will only depend on their recovery.

5. In each period, the arriving patient is assumed to have a nonhomogeneous poisson arrival rate λ. Let $q_{t,m}$ denote the probability that a newly arriving patient at time t is health state m. It will not change over all T periods.

6. The patient length of stay distribution is assumed to be memoryless.

7. The cost for demand-driven discharges and for diversions is associated with patients' state. Therefore, patients in the same class will have the same cost for discharging. Therefore, doctors will randomly choose the patient with the same health status to discharge if it is needed.

8. Updating the patient's health states will occur after the action is taken.

9. The ICU has a fixed number of beds B.

3.2 State Space & Action Set

The state of system can be characterized as following. Let x_{mt} be the number of patients of health state m in the ICU in the period t. let y_{mt} be a binary indicator for the arrival of a patient with status m in period t. The system state at period t will be a vector $s_t = \{(x_{mt}, y_{mt}) : \forall m \in [0, M], \sum_m x_{mt} = B, \sum_m y_{mt} \leq 1\}$.Therefore, the state space is defined as follows: $\mathcal{S} = \{s_t, \forall t \in [0, T]\}$

For each state, $\mathcal{A}(s_t)$ is the set of actions that can be taken in period t. let $a(s_t)_m$ be a variable for a decision on discharging a patient with status m in period t or not. If m=0, it can be understood as a decision on admitting a new patient without discharging. The action can only be made when there are some patients with health state m in period t in ICU. In other words, it can be expressed as

$$a(s_t)_m = \begin{cases} \text{-1 or 0} & \text{if } x_{mt} > 0 \\ 0 & \text{Otherwise} \end{cases} \tag{1}$$

Hence, an action $A \in \mathcal{A}(s_t)$ will be $\{a(s_t)_m : \forall m \in [0, M], \sum_m |a(s_t)_m| \leq 1\}$

3.3 Probability Characteristics

This system also has a random attribute associated with every time step. During the time step, a patient's health state is allowed to change according to a probability matrix P. The probability matrix P has three components, i.e., the natural discharge matrix ($\mathcal{N}$), the mortality matrix ($\mathcal{M}$) and the transition matrix ($\mathcal{T}$). Those three metrics record the following probabilities respectively:

the probability for patients in health status 1 to be naturally discharged; the probability for non-zero health status patients to die, and the transition probability between all non-zero health statuses. Those three matrices all have size m*m. Let P be the sum of $\mathcal{N}$, $\mathcal{M}$ and $\mathcal{T}$. Therefore, P_{ij} would refer to the probability of a patient being in health state i in time t, and then transitioning to state j in time t+1. By considering expected change in the number of patients in a state, this random component, δ_i, can be described as the following.

$$\delta_i = \sum_{j \neq i} x_{jt} * P_{ji} - \sum_{i \neq j} x_{it} * P_{ij}, \forall i, j \in [0, M] \tag{2}$$

3.4 Dynamic Components

Let $s_{t+1} = S(s_t, A)$ denote the random next state by taking action A in state s_t. Since the policy will contain the decision on admission, if the ICU decides not to admit the patient, i.e., $\sum_m a(s_t)_m = 0$, the y_{mt} needs to be cancelled out. Also, for the case if no one arrives, there will be no action to take. Define R_t to be independent random variables, which indicate the health status of an arriving patient in each period. Therefore, R_t take values in 0, 1,2,. . . ,m with probability $\lambda q_{t,m} \forall m \in [0, M]$. Thus,$s_{t+1} = S(s_t, A)$ is defined as

$$x_{mt+1} = x_{mt} + y_{mt} + (a(s_t)_m - y_{mt} * \mathbf{1}_{\{\sum_{\mathbf{m}} \mathbf{a(s_t)_m = 0}\}}) * \mathbf{1}_{\{\sum_{\mathbf{m}} \mathbf{y_{mt} = 1}\}} + \delta_{\mathbf{m}} \tag{3}$$

$$y_{mt+1} = \mathbf{1}_{\{\mathbf{R_{t+1} = m}\}} \tag{4}$$

3.5 Cost Function & Objective Function

$C(s_t, A)$ is the function calculating the total cost based on the chosen action A, and the current state s_t. The cost function can capture different quality metrics. The cost for discharging and rejecting is specific to each patients' health statuses, and $C(s_t, 0) = 0$.

Let Π be the set of feasible discharge policies, and let π map the state space $\mathcal{S}$ to the set of feasible actions $\mathcal{A}$. Define the expected total cost-to-go with policy π as

$$J^{\pi}(s) = E[\sum_{t'=t_1}^{T-1} C(s_{t'}, \pi(s_{t'}))|s_{t_1} = s] \tag{5}$$

Let $J^*(s) = min_{\pi \in \Pi} J^{\pi}(s)$ be the minimum expected total cost-to-go under any policy. Hence, π^* will be a responding optimal policy, i.e., $\pi^*(s) \in argmin_{\pi \in \Pi} J^{\pi}(s)$. Then, the paper define the operator $\mathcal{H}$ according to

$$(\mathcal{H}J)(s_t) = min_{A \in \mathcal{A}} E[C(s_t, A) + J(S(s_t, A)] \tag{6}$$

for all $s_t \in \mathcal{S}$. The optimal value function J^* may be found as the solution to the Bellman equation $\mathcal{H}J = J$.

4 Cost Metrics

The goal of this paper is to demonstrate the trade-offs between the patients' welfare and the ICU's operation efficiency. From both sides of patients and hospitals, they all expect that patients can have a low mortality risk during their stay of ICU. Also, in the health care field, researchers usually use QALY to measure the associated quality-of-life losses of patients. Therefore, the QALY for discharge can indicate the cost of discharging from the patient perspective. On the other hand, from the perspective of ICU operation managers, the readmission rate can have a significant impact on the ICU operational efficiency, since a high readmission risk can cause ICU congestion as a consequence of higher demand and longer LOS of patients. Hence, this paper's proposed cost function needs to capture those costs carried by those measures. While those three have different units, a common unit, such as dollars, can be associated with them, provided an estimate of cost is associated per unit. The costs related to mortality risk and readmission risk will vary between hospitals. Hence, the preferred cost function will be in the unit of dollars and has the followed format

$$\text{The mixed cost function} = Q_i * C_{QALY} + \mathcal{M}[0,i] * C_{Mortality} + R_i * C_{Readmission}, \forall i \in [1, M]$$

Here, Q_i is the QALY for discharging a health state i patient, $\mathcal{M}[0,i]$ is the probability for health state i patient to be dead, R_i is the probability for health class i patient to be readmitted into the hospital. $C_{QALY}, C_{Mortality}$, and $C_{Readmission}$ are the cost associated with the QALY, the mortality in a burn center, and receiving a readmission patient on hospitals' side, respectively. Ideally, this function can capture the loss from both patient' welfare and hospitals' operational efficiency, and convert it into money, which is easy to to interpret. Therefore, the $C(s_t, A)$ is defined as following

$$C(s_t, A) = \begin{cases} Q_i * C_{QALY} + \mathcal{M}[0,i] * C_{Mortality} + R_i * C_{Readmission} & \text{if } a(s_t)_i \neq 0, \forall i \in [1, M] \\ C_{transfer} & \text{Otherwise} \end{cases} \tag{7}$$

If the ICU decides not to admit the new coming patient, a fixed cost will occur for transferring this patient to other hospitals, which is described as $C_{transfer}$. Since the specific costs associated with QALYs, mortality, and readmission will vary between hospitals, the paper will focus on each specific component. To find the total cost for a specific hospital, simply multiply each cost by the predetermined cost for the hospital to find the total cost.

5 ICU's Policy

Based on this model setting, the optimal policy will depend on the current state s_t, the information on the distribution of future patient arrivals, $\lambda q_{t,m} \forall m \in [0, M]$., and the influence of the present action decision on the future state.

However, the size of $\mathcal{S}$ causes the straight-forward dynamic programming approach to be nearly impossible under a limited time. Furthermore, it is questionable that if this model can be widely implemented; because it's required detailed information on future patients' arrivals can be nonstationary and hard to obtain.

Therefore, this paper will use a simulation optimization approach to find a relatively optimal discharge policy from proposed policy candidates related to the concerned cost metrics. All of the policies will be implemented when the ICU run out of its capacity:

1. Arbitrary (ARB): The ICU will always admit the new incoming patient, and one of the inpatients is selected randomly for demand-driven discharge.

2. First in first out (FIFO): The ICU will always admit the new patient. When the demand-driven discharge is needed, the ICU will discharge the patient who has stayed for the longest time in the ICU. The reason for doing FIFO is because the longer length for patients to stay in the ICU, the higher chance they may get infected [2].

3. Lowest QALY then FIFO (QSA): The ICU will rank current inpatients and the new patient based on their QALY for discharging or rejecting. If the new coming patient has the lowest QALY, the ICU will not admit the patient. Otherwise, the ICU will pick the current inpatient with the lowest QALY to discharge. The FIFO is used to select the patient if the lowest QALY gives several discharging choices. This policy is the one often used in practice, it will be treated as status quo.

4. Lowest Mortality risk then FIFO (MSA): The mortality risk is defined as the probability of dying. Therefore, like the previous one, the ICU will rank current inpatients and the new coming patient based on their mortality risk to make decisions on admissions and discharging

5. Lowest readmission risk then FIFO (RSA): The readmission risk is defined as the probability of readmission. Hence, like the previous two, the ICU will rank current inpatients and the new coming patient based on their readmission risk to make decisions.

6. Lowest QALY then FIFO, always admit (QAA): The ICU will not have a policy of always admitting new patients. Therefore, there is no possibility of rejection, and a current patient will need to be discharged if all the beds are occupied. That patient will be chosen based on QALY cost and then FIFO, similar to policy QSA.

7. Lowest Mortality risk then FIFO, always admit (MAA): This policy is similar to MSA, except the incoming patient cannot be rejected.

8. Lowest readmission risk then FIFO, always admit (RAA): This policy is similar to RSA, except the incoming patient cannot be rejected.

Table 3: Summary table of the parameters and their realizations used in the simulation

Characteristic	Patient Type		Publication
The readmission rate			
	First Degree	11.30%	
	Second Degree	61.50%	Eldelson et al. [**10**]
	Third Degree	28.60%	
Mean Quality-of-life loss by months since burn discharge			
	Any degree(Treated as the QALY for rejection)	0.237 QALY	
	First Degree	0.227 QALY	
	Second Degree	0.269 QALY	Miller et al. [**11**]
	Third Degree	0.269 QALY	
The mortality rate			
	First Degree	0.88%	
	Second Degree	7.43%	Ryan et al. [**12**]
	Third Degree	32.33%	
Number of beds		17 beds	"Burn Center"[**9**]
Arrival rate		0.9 person per period	
Arrival probability			
	First Degree	93.4 %	
	Second Degree	5.2 %	S.L.Taylors et.al. [**14**]
	Third Degree	1.4 %	
Mean LOS for different degree of burn			
	First Degree	4 days	
	Second Degree	32 days	S.L.Taylors et.al. [**14**]
	Third Degree	58 days	

6 Simulation

The project team has made a simulation based on the models described above. All of data used in the simulation are summarized in Table 3. The mortality and readmission risk of rejection will be the same as the ones for discharge. The QALY discharge costs are shown above. The QALY rejection cost for any state will be 0.237, as stated in literature. The probability matrix $\mathcal{P}$ is chosen to make the mean LOS of each health status match with the results seen in literature. You can see the value of the probability matrix $\mathcal{P}$ in Appendix C. This simulation is written in Java, and the code can be found in Appendix A. The code has been annotated with notes to help readability. A sample output demonstrating how the model works can be found in Appendix B. All numbers in the code, such as the number of beds, the arrival rate, or the number of patient classes, can be altered and were simply used to see if the simulation was working correctly. These numbers could be updated for further research make the simulation as accurate as possible.

The simulation creates 4 different classes: health classes, patients, beds, and an ICU. These classes all have characteristics and functions that are associated with them, such as patients have a numerical ID and beds can discharge and assign patients to them. While it would not be worthwhile to go through every characteristic and function in the code, the full code is available in Appendix A, and any feedback on confusing functions or lines is appreciated.

The simulation works as a discrete time simulation. At every time step, there are a series of actions that can take place. First, every patient currently in the ICU is updated. For every health class, there is a probability that a patient will increase their health state, decrease their health state, or stay in the same health state. This is a Markov model, so these probabilities do not depend on previous states the patient was in, only the current state. If a patient ever reaches state 0, the bottom state, this is a mortality. In this instance, the

mortality counter increases by one, and the patient is discharged from the bed. If the patient ever improves enough to reach the final health state, call it state n, then the person is healthy enough to be discharged. This is a natural discharge, and incurs no cost to the ICU. Once every patient has been updated for the time step, the model determines if there is a patient arrival. Consistent with some of the previous models, it is assumed that either 0 patients arrive in a time step or 1 patient arrives. This has some limitations, such as an instance of a car accident or fire, where multiple people are injured at once and will all arrive at the same time. However, if the time step is small enough, this can be a reasonable assumption. In the situation where no patients arrive, then the simulation simply increases the time by one time step, and the process starts again. The situation where a patient does arrive is as follows.

If there is a patient arrival, then the health class of the patient needs to be determined. The likelihood of the patient's initial health state being 1 is 0.934, as determined by previous studies. The other 2 health states have probabilities determined by previous studies as well. Next, the simulation will look through all the ICU beds to see if there is an open one. If there is an open bed, it will simply assign the patient to the open bed and that is the end of the procedure. If there is no open bed, then a demand driven discharge needs to take place. Which patient is discharge depends on the policy being implemented. The model will look through all the patients in every bed, and determine the cost associated with them, should they be the patient chosen to discharge. Every state for every health class has a discharge cost associated with it. These costs are increasing in state, so a discharge in state 1, which is a poor health state, has a higher cost associated with it compared to a discharge in state n-1. This is intuitive, since given a choice between discharging two patients of the same health class, the healthier patient would be the easiest to discharge. Each health class has different costs of discharge, so it is not enough to simply look at health states. The simulation chooses the minimum cost of all patients, discharges them, incurs a forced discharge cost, and then assigns the new patient to the bed. The flowchart showing this procedure can be seen in Figure 2.

7 Results

The initial simulation study found the FIFO and arbitrary policies to have the lowest scores for QALY cost. For mortality cost, MSA had the lowest cost, while QSA had the second lowest. Figure 3 shows an inverse relationship between QALY cost and mortality cost, as policies with high QALY cost had low mortality cost, and vice versa. Using the metrics provided in literature, it shows that, in those eight policies tested, there is not a policy that is superior in both categories.

Figure 4 shows the results of the readmission costs versus QALY costs. The policy that provided the lowest readmission cost was FIFO, followed by RAA, then arbitrary, then RSA. These policies were displayed against QALY cost, and the results showed that the there is a positive correlation between the two.

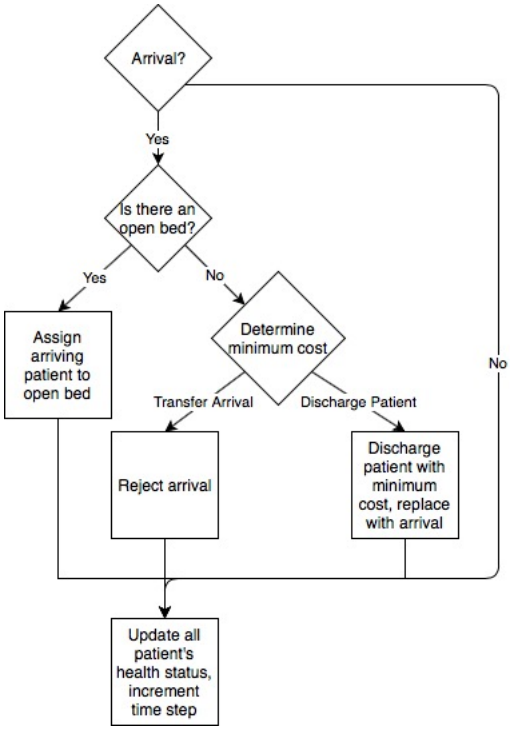

Figure 2: A flowchart of the simulation

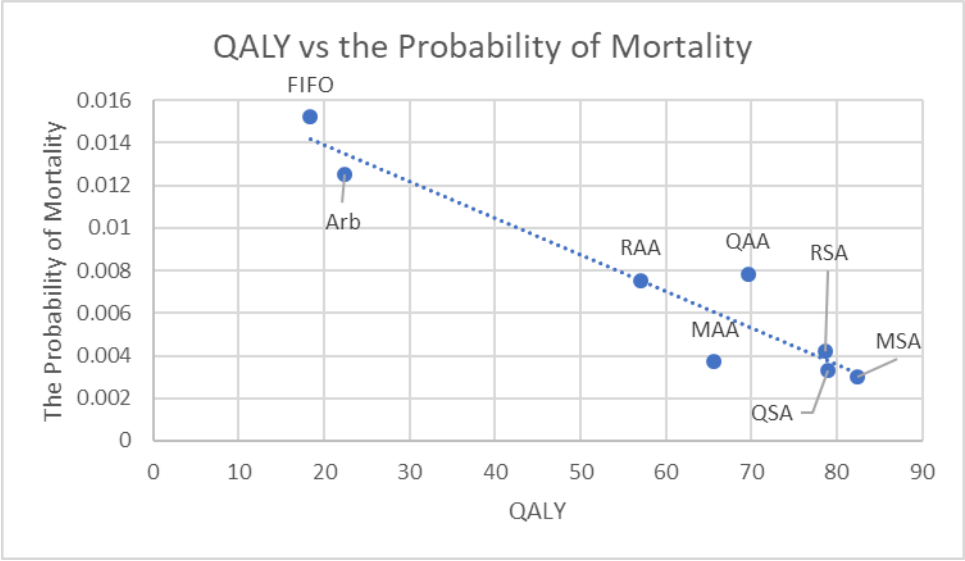

Figure 3: A graph showing average QALY and mortality costs for each policy.

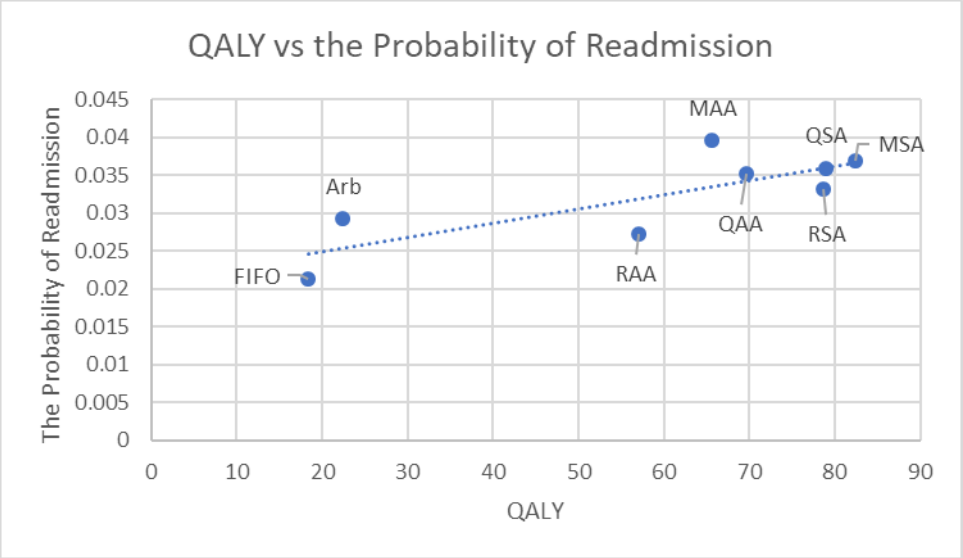

Figure 4: A graph showing average QALY and readmission costs for each policy.

Therefore, the policies with low QALY cost also tended to have lower readmission cost, and vice versa. Figure 5 is the final graph showing the initial results, which demonstrates the readmission costs displayed against the mortality costs. This showed a negative correlation between the two, meaning the policies with lower readmission costs generally had higher mortality costs, and vice versa.

A sensitivity analysis was then performed to determine the impact changing the cost parameters had on the results. Figure 6 shows the impact changing the QALY discharge cost parameter had on the total QALY cost for two policies, FIFO and QAA. These two policies were chosen because displaying all eight would result in an unreadable graph. This sensitivity analysis kept the state 1 discharge cost constant, but increased the state 2 and state 3 discharge costs to see if the results changed. The graph displays that both policies experience higher costs, but the cost for FIFO grows more rapidly than the cost for QAA. For higher costs, QAA has a lower QALY cost than FIFO.

A similar sensitivity analysis was performed for the mortality discharge cost parameters, and is shown in figure 7. MAA and FIFO were chosen to study, and the mortality cost for state 2 and 3 were increased, while it was constant for state 1. This allowed for the studying of how each policy's total cost changed as the difference between states became for drastic. Similarly as before, the FIFO policy grew more quickly than the MAA. A final sensitivity analysis was performed for the readmission discharge cost parameters. The results can be seen in figure 8. The two policies tested were FIFO and RAA. The readmission cost parameters for state 2 and 3 were increased, while the parameter for state 1 was constant. Similarly as before, the FIFO policy grew more quickly than

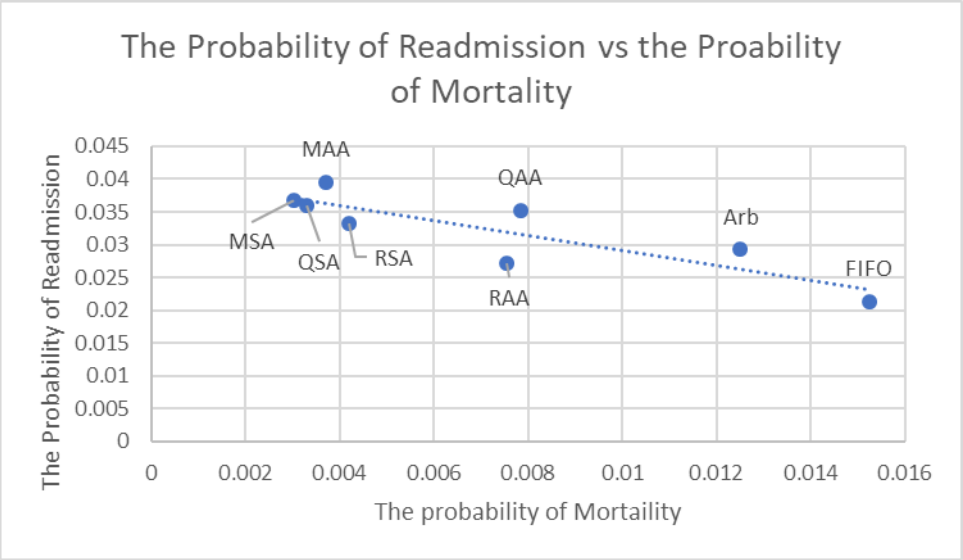

Figure 5: A graph showing average mortality and readmission costs for each policy.

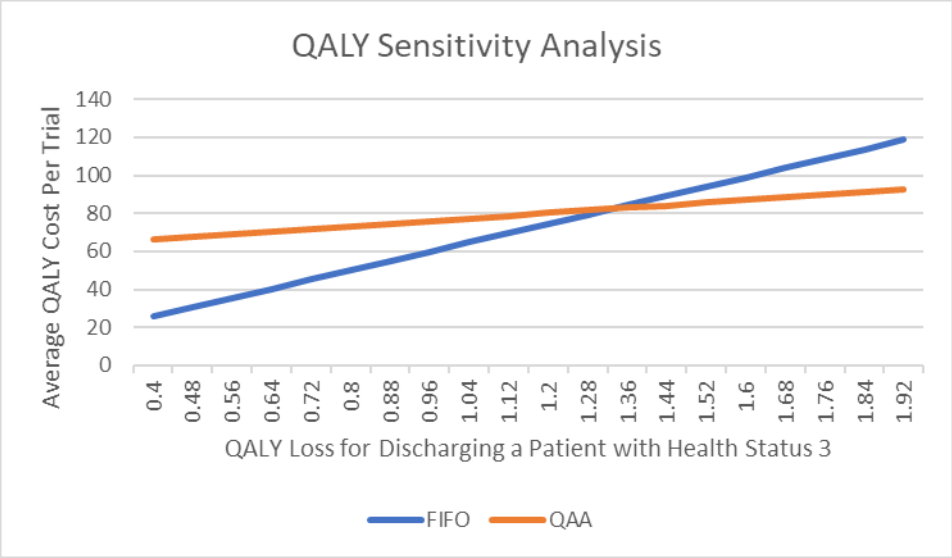

Figure 6: A graph showing the results of the sensitivity analysis of the discharging QALY costs for the FIFO and QAA policies.

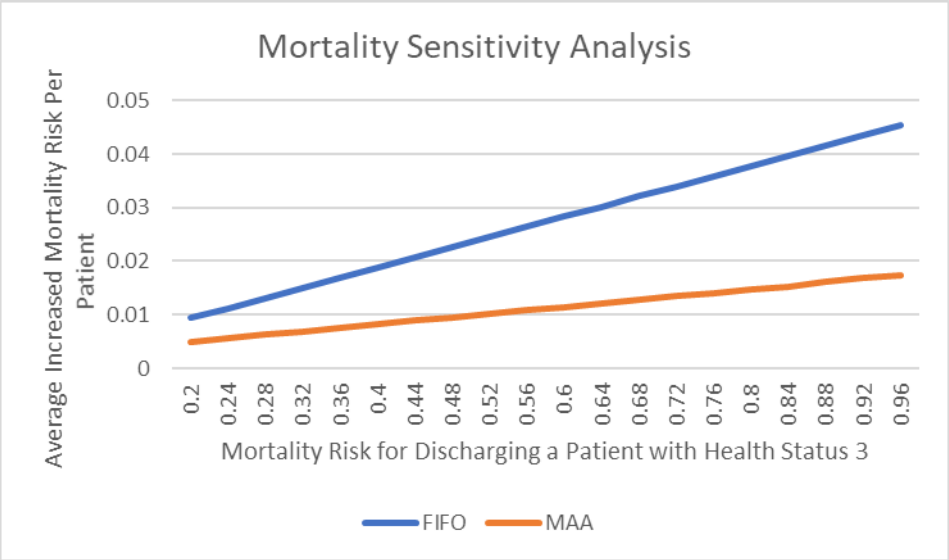

Figure 7: A graph showing the results of the sensitivity analysis of the discharging mortality costs for the FIFO and MAA policies.

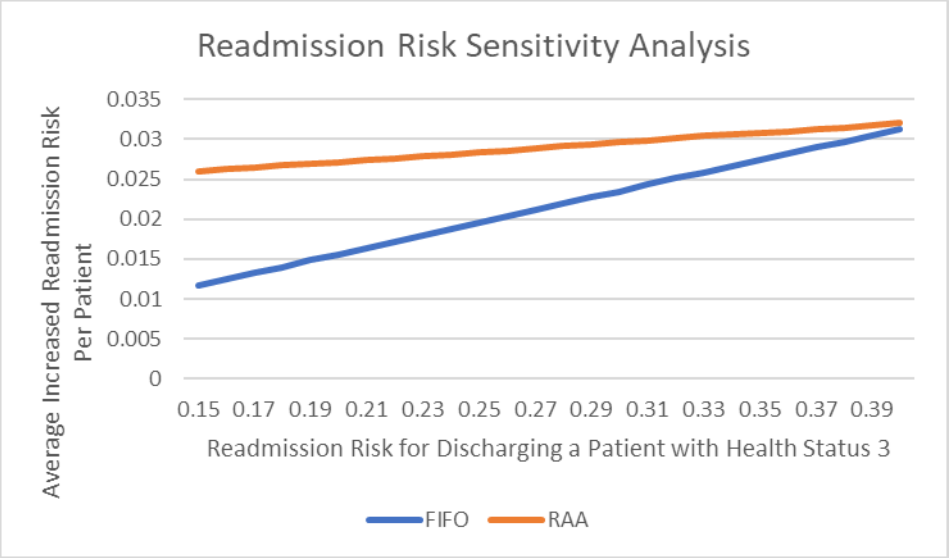

Figure 8: A graph showing the results of the sensitivity analysis of the discharging readmission costs for the FIFO and RAA policies.

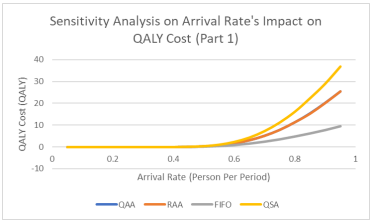

Figure 9: Arrival rate sensitivity analysis of QALY cost for QAA, RAA, FIFO, and QSA

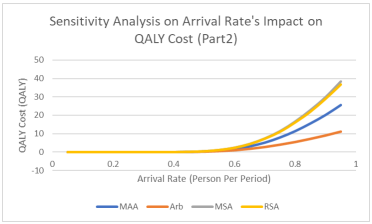

Figure 10: Arrival rate sensitivity analysis of QALY cost for MAA, Arb, MSA, and RSA

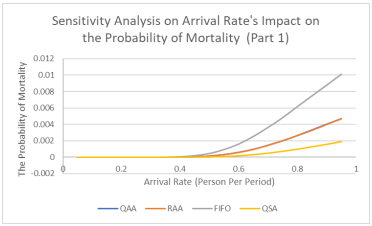

Figure 11: Arrival rate sensitivity analysis of mortality cost for QAA, RAA, FIFO, and QSA

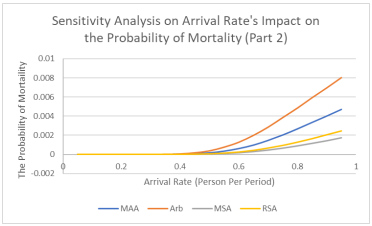

Figure 12: Arrival rate sensitivity analysis of mortality cost for MAA, Arb, MSA, and RSA

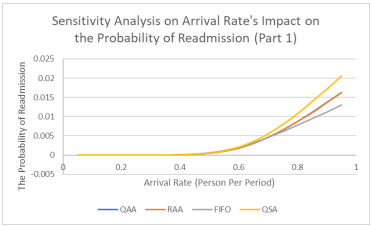

Figure 13: Arrival rate sensitivity analysis of readmission cost for QAA, RAA, FIFO, and QSA

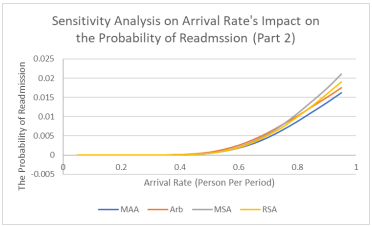

Figure 14: Arrival rate sensitivity analysis of readmission cost for MAA, Arb, MSA, and RSA

RAA. Appendix C shows all simulations' results on previous sensitivity analysis.

Finally, an analysis was done researching the impact arrival rate had on the policies. The original discharge costs were used. The probability of arrival was varied from 0.05 to 0.95 in increments of 0.05. The first two graphs show total QALY cost per policy per arrival rate. These were split into two graphs to help readability. The policies QAA and RAA are almost indistinguishable in total cost in figure 9, but were calculated separately and are both on the first graph. Additionally, MSA and RSA have similar results, but are both in figure 10.

The impact of arrival rate was tested on mortality and readmission as well, and experienced similar results.

8 Discussion

The simulation results provided some fascinating insights into what the optimal discharge policy should be. Naively, it would be expected that in order to minimize the total cost for a metric, such as QALYs, mortality rate, or readmission rate, it would be optimal to discharge the patient that has the lowest discharge cost for that metric. However, the results showed that, in some situations, this will not lead to an optimal policy. The insight this simulation study provided is that it is sometimes better to incur a higher cost and discharge a higher state patient if future benefits outweigh that cost. The results showed this since FIFO performed better under the initial conditions. The FIFO policy, by discharging the patients who have been in the longest, naturally discharges higher state patients, since they have the longest LOS. Since this policy performed better for some metrics initially, namely QALY and readmission, this shows that discharging higher state, and higher cost, patients can lead to lower overall costs. This idea was derived from the sensitivity analysis of each cost metric. For the QALY example, if the discharge costs are very similar, then FIFO has a lower cost than QAA. However, as the difference between discharge costs becomes greater, the FIFO policy does worse. The reason behind this result is that discharging higher state patients leads to discharging fewer patients in the future.

Higher state patients correspond to more critical patients who have longer LOS on average than lower state patients. If an ICU has many critical patients occupying their beds, it is intuitive that those beds will be occupied for longer, since these patients have a long LOS. Therefore, the less critical patients have fewer beds available. These less risky patients, who also arrive at a much higher frequency, will then have fewer beds to occupy, and the ICU will fill up very quickly. If a policy is to discharge the patient with the least cost, this will correspond to discharging less critical patients, who would have been discharged anyway rapidly since they have a shorter average LOS. This cycle will then repeat itself quickly since more patients keep arriving, and the ICU will have to discharge another less critical patient. In essence, policies that discharge based on least cost will result in more discharges due to the ICU being occupied mostly with patients with long LOS. In some situations, such as the initial test using cost values found in literature, it is cheaper, in the long run, to incur one more considerable discharge cost now, if that means incurring less, smaller discharge costs in the future.

Additionally, it was shown in the sensitivity analysis that costs, and the difference in costs between policies, grows with arrival rate. This is the same as saying that as the ICU's utilization increases, costs increase, and there is a more substantial difference between the costs of the policies. While this intuitively makes sense, the simulation results proved this idea. This is important because it shows that inefficiencies in lower utilization ICUs will not have as big of an impact on total cost as inefficiencies in higher utilization ICUs. The implementable heuristic derived from this is those critical patients are best suited in lower utilized ICUs. Therefore, the "inefficiency" costed by their long LOS

will not cost as much, as opposed to if they were in a highly utilized ICU.

Initially, this project wanted to produce an optimal policy for an ICU discharge policy. Because of the intractability of the model, it was not possible to solve the model to get the optimal policy. Upon this realization, a simulation study was conducted to uncover key insights and heuristics that the optimal policy should have. The beginning of this section provides those. However, the essential part of this project is determining how the results can be implemented in practice. Clearly, the simulation and model incorporate assumptions to make them implementable. The largest of these assumptions is the assumption that all patients can be categorized by health states, and all patients in a health state have the same discharge costs. In practice, this is not a realistic assumption. Every individual is unique and has their characteristics. Categorizing them into rigid health states inherently loses these unique characteristics. Additionally, a patient's health, especially in a place such as the ICU, is constantly fluctuating.

For this reason, the cost of discharging a patient can also vary considerably from day to day, or even hour to hour. This assumption does not invalidate this research, however. Instead, these heuristics mentioned early can serve as guidelines for physicians, who can still use updated and personalized data for each patient. The heuristics can be counter-intuitive, such as discharging a more critical patient, so having these as part of a policy is an excellent reminder for physicians, and might have been overlooked if not for this research.

Another difference between this research and practice is that in this simulation and model, it is never optimal to discharge a patient while there is an open bed. Until a patient arrives when the ICU is full, it is not optimal to voluntarily discharge someone and incur a cost. However, in practice, this might not be the case. As previously mentioned, more critical patients should be in lower utilized ICUs, and therefore it is sometimes optimal to incur a cost to "discharge" them and transport them to another ICU or hospital. However, as also noted above, a patient's health is always fluctuating, and therefore the cost of discharging them will also vary. This means that, if a critical person has stabilized and their health is better than expected, it would be optimal to move them while the discharge cost is lower than usual. While this is not an option in the simulation and model because the discharge costs are constant in states, this is an example of how the heuristics in the research can be used in a situation found outside of the study.

This research also looked at the ICU as an independent unit. For the simulation and model, either a patient is in the ICU, or he/she is not. However, ICUs in practice are connected to other units, such as a hospital and emergency department. Therefore, to be discharged from the ICU does not mean the patient is forced to heal independently. Instead, getting discharged from the ICU could expect getting moved to a hospital ward, or perhaps a different ICU. Therefore, it is easy to imagine a situation where getting discharged to a hospital could have different discharge costs than getting moved to another ICU. While in the model and simulation the discharge cost was constant, the previous example shows how discharge costs can vary by where the patient is getting discharged.

9 Conclusion

In this paper, the research team provides some new perspectives on designing an ICU discharge policy for a burn center, by considering and comparing different policies, which emphasized on either ICU operational efficiency or patients' welfare or both. These policies included ranking patients based on costs of QALY, mortality, and readmission and discharging the patient with the lowest cost. The team designed a mathematical model of the situation to find the optimal policy, but the model was not able to be solved. The project team then performed a simulation study to analyze the situation and discovered some heuristics the optimal policy should have. It was discovered that discharging the least cost patient is not always the optimal policy, as it is sometimes better to incur a larger cost to discharge a critical patient. When the discharge cost difference between states is low, it is better to discharge a more critical patient, while if the difference between states is drastic, it is better to discharge the less critical patient.

Additionally, it was seen that the arrival rate plays a large role in the magnitude of the difference in costs between policies. While all policies incur larger costs as arrival rate increases, the increase effects inefficient (higher cost) policies more. Therefore, busier ICUs will need to be extra vigilant regarding their policies, while less busy ICUs will not have to be.

For the future research opportunities, it will be worthwhile to relax some assumptions without ignoring some primary burn specific features, so that the research team can use some approximation algorithms to find a nearly optimal policy which can also be meaningful to the current burn centers. Besides, by combining case studies utilized with clinical data, the arrival patterns and LOS of patients with different health status can be more realistic, and the result can be more applicable to hospitals.

10 Contribution

Both team members worked hard throughout the entire semester. For the final delivery, Zitao mainly contributes to refine the model, cost metric and policy design. Nate is responsible for updating the simulation and collecting the data. In term of writing, the work is distributed equally between them.

References

[1] C. Chan, V. F. Farias, N. Bambos and D. Escobar, *Optimizing Intensive Care Unit Discharge Decisions with Patient Readmissions.* OPERATIONS RESEARCH, vol. 60, p. 1323–1341, 2012.

[2] S. Z. Hosseinifard, B. Abbasi and J. Minas *Intensive care unit discharge policies prior to treatment completion.* Operations Research for Health Care, vol. 3, pp. 168-175, 2014.

[3] G. Dobson, H.-H. Lee and E. Pinker *A Model of ICU Bumping*. OPERATIONS RESEARCH, vol. 58, p. 1564–1576, 2010.

[4] J. Li and W. Dong *Admissions optimisation and premature discharge decisions in intensive care units*.International Journal of Production Research, pp. 7329-7342, 2015.

[5] X. Li, D. Liu, N. Geng and X. Xiaolei *Optimal ICU Admission Control With Premature Discharge*.IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING, vol. 16, pp. 148-165, 2019.

[6] S. Mzezewa, K. Jonsson *A Prospective Study on the Epidemiology of Burns in Patients Admitted to the Harare Burn Units*.Burns, vol. 25, issue 6, pp. 499-504, September 1996.

[7] L. Desanti *The Journal of Burn Care Rehabilitation*.Volume 19, Issue 5, September-October 1998, Pages 414–419.

[8] V.Patil, J.Dulhunty, A.Udy, P.Thomas, G.Kucharski, J.Lipman *Do Burn Patients Cost More? The Intensive Care Unit Costs of Burn Patients Compared With Controls Matched for Length of Stay and Acuity*.Journal of Burn Care Research, Volume 31, Issue 4, July-August 2010, Pages 598–602, https://doi.org/10.1097/BCR.0b013e3181e4d6a4

[9] "Burn Center." Hennepin Healthcare, www.hennepinhealthcare.org/specialty/burn-center/

[10] T. Miller,S. Bhattacharya,W. Zamula et al. *Quality-of-life loss of people admitted to burn centers, United States*.Qual Life Res (2013) 22: 2293. https://doi.org/10.1007/s11136-012-0321-5

[11] S.Eidelson, J.Parreco, M.Mulder, A.Dharmaraja, J. Kaufman, K.Proctor, L.Pizano, C.Schulman, N.Namias, R.Rattan, *Variation in National Readmission Patterns After Burn Injury*. Journal of Burn Care Research, Volume 39, Issue 5, September/October 2018, Pages 670–675. https://doi.org/10.1093/jbcr/iry034

[12] C. Ryan,D. Schoenfeld, W.P. Thorpe, *Objective Estimates of the Probability of Death from Burn Injuries*. N Engl J Med 1998; 338:362-366 DOI: 10.1056/NEJM199802053380604

[13] S. Alan. *MHPE 441: Medical Decision Making*. araw.mede.uic.edu/ alansz/courses/mhpe441/week10.html.

[14] S.L.Taylor, S.Sen, D.G. Greenhalgh, et al. *Real-Time Prediction for Burn Length of Stay Via Median Residual Hospital Length of Stay Methodology*.Journal of Burn Care Research, Volume 37, Issue 5, September-October 2016, Pages e476–e482, https://doi.org/10.1097/BCR.0000000000000332

Appendices

A Simulation Code

Listing 1: Java example

```
\hspace*{-3cm}
import java.io.*;
import java.util.Random;
class Healthclass //Making Healthclass
{
        public int identification; //ID of healthclass, from 1..M
        public int[] states; //Different health states, can be different for
             different classes
        public double[] Qcost;
        public double[] Mcost;
        public double[] Rcost;
        public double[][] statechangingprobabilities; //probability of moving
              up or down health states, or staying the same
        public int startingtime;
        public Healthclass(int identification, int statenumber, double[]
             Qcost, double[] Mcost, double[] Rcost, double[][]
             statechangingprobabilities)
        {
                this.identification=identification;
                this.Qcost=Qcost;
                this.Mcost=Mcost;
                this.Rcost=Rcost;
                this.statechangingprobabilities=statechangingprobabilities;
                states=new int[statenumber];
        }
}
class Patient //Making a patient
{
        public int identification; //Patient ID
        public Healthclass healthclass; //Healthclass of Patient
        public int initialhealthstate;
        public int healthstate; //Current health class
        public int starttime; //When they arrived to ICU
        public int endtime; //When they left ICU
        Random generator= new Random(); //Create random number generator
             between 0 and 1
        public Patient(int identification, Healthclass healthclass, int
             starttime)
        {
                this.identification=identification;
                this.healthclass=healthclass;
                this.starttime=starttime;
                double healthstatevalue=Math.random();
                if (healthstatevalue<=0.934) //Randomly assign starting
                     health state
                {
                        healthstate=1;
                        initialhealthstate=1;
                }
                else if(healthstatevalue<=0.986)
                {
                        healthstate=2;
                        initialhealthstate=2;
                }
                else
                {
                        healthstate=3;
                        initialhealthstate=3;
```

```
                }
        }
}
class Bed //Making a bed
{
        public int bednumber; //Bed number
        public Patient icupatient; //Patient currently in bed
        public int patientnumber; //Patient number in bed
        public int state; //State of bed, 0 = open, 1 = busy
        public double BedQcost; //Sum of the discharge costs for this bed
        public double BedMcost;
        public double BedRcost;
        public int mortalities; //Number of mortalities in this bed
        public int[] initialpatients;
        public int[] initialpatientsLOS;
        public Bed(int bednumber)
        {
                this.bednumber=bednumber;
                state=0;
                BedQcost=0;
                BedMcost=0;
                BedRcost=0;
                mortalities=0;
                initialpatients=new int[3];
                initialpatientsLOS=new int[3];
                for (int i=0; i<3; i++)
                {
                        initialpatients[i]=0;
                        initialpatientsLOS[i]=0;
                }
        }
        public void addpatient(Patient icupatient) //Adds a patient to the
             bed
        {
                this.icupatient=icupatient;
                patientnumber=icupatient.identification;
                initialpatients[icupatient.initialhealthstate-1]+=1;
                state=1;
        }
        public void forceddischarge(Patient x, int time) //Force discharges
             the current patient, adds a new patient x
        {
                BedQcost+=icupatient.healthclass.Qcost[icupatient.healthstate
                     ];
                BedMcost+=icupatient.healthclass.Mcost[icupatient.healthstate
                     ];
                BedRcost+=icupatient.healthclass.Rcost[icupatient.healthstate
                     ];
                initialpatientsLOS[icupatient.initialhealthstate-1]+=(time-
                     icupatient.starttime);
                icupatient=null;
                addpatient(x);
        }
        public void naturaldischarge(int time) //Natural discharge, so state
             = 0
        {
                initialpatientsLOS[icupatient.initialhealthstate-1]+=(time-
                     icupatient.starttime);
                icupatient=null;
                state=0;
        }
        public void gonextstepbed(int time) //Next step in the process, i.e.
             moves 1 day up
        {
                if (icupatient==null)
                {
                        return; //Don't do anything if bed is empty
                }
```

```
                double probstate=Math.random(); //Decides whether the patient
                     improves, stays put, or gets worse
                if (probstate<icupatient.healthclass.
                    statechangingprobabilities[icupatient.healthstate-1][0])
                     //patient gets worse
                {
                        icupatient.healthstate+=1;
                        if (icupatient.healthstate==icupatient.healthclass.
                            states.length) //if new healthstate is the last,
                             they die, and are "naturally discharged"
                        {
                                mortalities+=1;
                                naturaldischarge(time);
                        }
                }
                else if (probstate<icupatient.healthclass.
                    statechangingprobabilities[icupatient.healthstate-1][1])
                {
                        //stays the same
                }
                else //Patient improves
                {
                        icupatient.healthstate-=1;
                        if (icupatient.healthstate==0) //if patient gets to
                             0, they are discharged
                        {
                                naturaldischarge(time);
                        }
                }
        }
}
class ICU
{
        public Bed[] Allbeds; //An ICU is a list of beds
        public double TotalQ;
        public double TotalM;
        public double TotalR;
        public ICU(int numberofbeds)
        {
                TotalQ=0;
                TotalM=0;
                TotalR=0;
                Allbeds=new Bed[numberofbeds];
                for (int Q=0; Q<numberofbeds; Q++)
                {
                        Allbeds[Q]=new Bed(Q+1);
                }
        }
        public void assignpatienttobed(Patient x, int time)//assigns a
             patient to a bed
        {
                for (int J=0; J<Allbeds.length; J++)//goes through all beds,
                     tries to find an empty one
                {
                        if (Allbeds[J].state==0)
                        {
                                Allbeds[J].addpatient(x);//adds patient to an
                                     empty bed
                                return;
                        }
                }
                forcedischargepatient(x, time);//if no beds are open, then
                     there is a forced discharge
                return;
        }
        public void forcedischargepatient(Patient x, int time)//force
             discharges a patient
        {
```

```
                double bestcost=1000000;//starts with high number
                int bestpatient=-1;//and imaginary patient
                for (int J=0; J<Allbeds.length; J++)//goes through all beds
                {
                        if (Allbeds[J].icupatient.healthclass.dischargecost[
                             Allbeds[J].icupatient.healthstate]<bestcost)//if
                              this bed has lowest cost
                        {
                                bestcost=Allbeds[J].forcedischargecost;//
                                     update best cost
                                bestpatient=J;//and this patient
                        }
                }
                Allbeds[bestpatient].forceddischarge(x, time);//after looking
                      at all beds, force discharge the patient with least
                     cost
        }
        public void gonextstep(int time)//simulates one step
        {
                for (int i=0; i<Allbeds.length; i++)//goes to next step for
                     all beds
                {
                        Allbeds[i].gonextstepbed(time);
                }
        }
}
class UpdatedICUArb
{
        public static void main(String [] args) throws IOException
        {
                File outFile = new File ("ICU.text");//Creates file
                FileWriter fWriter = new FileWriter (outFile);//Creates File
                PrintWriter pWriter = new PrintWriter (fWriter);//Creates
                     File
                int time=0;//totaltime
                int daystoobserve=1000;//total days to observe
                int mortalitycount=0;//total mortalities
                int totalbeds=17;//total number of beds
                int numberofhealthclasses=1;//number of health classes we
                     have
                int healthstates=4;//number of health states in each class
                int patientcounter=0;//total number of patients
                double[] Qcost=new double[healthstates];//discharge costs for
                      first health class
                double[] Mcost=new double[healthstates];//discharge costs for
                      first health class
                double[] Rcost=new double[healthstates];//discharge costs for
                      first health class
                Qcost[0]=2;Qcost[1]=3;Qcost[2]=4;
                Mcost[0]=2;Mcost[1]=3;Mcost[2]=4;
                Rcost[0]=2;Rcost[1]=3;Rcost[2]=4;
                double[][] statechangingprob=new double[healthstates][3];//
                     Probabilities for changing states
                int MCNum=1000;
                int trialnumber=0;
                int totalLOStypeone=0;
                int totalLOStypetwo=0;
                int totalLOStypethree=0;
                int totalpatientstypeone=0;
                int totalpatientstypetwo=0;
                int totalpatientstypethree=0;
                statechangingprob[0][0]=0.1; //10% chance they get worse in
                     state 1
                statechangingprob[0][1]=0.6; //50% chance they stay the same
                     in state 1
                statechangingprob[0][2]=1; //40% chance they improve in state
                      1
                statechangingprob[1][0]=0.02;//2.5% chance they get worse in
```

```
                state 2
            statechangingprob[1][1]=0.96;//92.5% chance they stay the
                same in state 2
            statechangingprob[1][2]=1;//4% chance they improve in state 2
            statechangingprob[2][0]=0.004;
            statechangingprob[2][1]=0.99;
            statechangingprob[2][2]=1;
            Healthclass[] healthclasses=new Healthclass[1]; //creating
                the health classes
            healthclasses[0]=new Healthclass(1, healthstates, Qcost,
                Mcost, Rcost, statechangingprob); //first health class
            while (trialnumber<MCNum)
            {
                    System.out.println(trialnumber);
                    time=0;
                    ICU MyICU=new ICU(totalbeds);//creates ICU
                    while (time<=daystoobserve)//runs for 10 days
                    {
                            MyICU.gonextstep(time);//increase the step
                            time+=1;//move up 1 day
                            /* Generate arrival or not*/
                            Random generator= new Random();
                            if (Math.random()<0.9) //% chance of arrival
                            {
                                    patientcounter+=1; //add patient
                                    Patient newpatient=new Patient(
                                        patientcounter, healthclasses
                                        [0], time);//create the new
                                        patient
                                    MyICU.assignpatienttobed(newpatient,
                                        time);//assign patient to bed
                            }
                    }
                    for (int i=0; i<totalbeds; i++)
                    {
                            totalLOStypeone+=MyICU.Allbeds[i].
                                initialpatientsLOS[0];
                            totalpatientstypeone+=MyICU.Allbeds[i].
                                initialpatients[0];
                            totalLOStypetwo+=MyICU.Allbeds[i].
                                initialpatientsLOS[1];
                            totalpatientstypetwo+=MyICU.Allbeds[i].
                                initialpatients[1];
                            totalLOStypethree+=MyICU.Allbeds[i].
                                initialpatientsLOS[2];
                            totalpatientstypethree+=MyICU.Allbeds[i].
                                initialpatients[2];
                    }
                    trialnumber+=1;
            }
            double averageLOStypeone=(double)totalLOStypeone/
                totalpatientstypeone;
            double averageLOStypetwo=(double)totalLOStypetwo/
                totalpatientstypetwo;
            double averageLOStypethree=(double)totalLOStypethree/
                totalpatientstypethree;
            pWriter.print("LOS for customers initially of state 1: " +
                averageLOStypeone + " days\n");
            pWriter.print("LOS for customers initially of state 2: " +
                averageLOStypetwo + " days\n");
            pWriter.print("LOS for customers initially of state 3: " +
                averageLOStypethree + " days\n");
            pWriter.close();
        }
}
```

B Output Sample

Here is the sample output for the simulation with 3 beds.

At Day 1.0, hour 0, minute 10
Bed 1 is open.
Bed 2 is open.
Bed 3 is open.
At Day 1.0, hour 0, minute 20
Bed 1 is busy, working on patient 1 who is in health state 1 of health class 3
Bed 2 is open.
Bed 3 is open.
At Day 1.0, hour 0, minute 30
Bed 1 is busy, working on patient 1 who is in health state 1 of health class 3
Bed 2 is open.
Bed 3 is open.

C Probability Matrix

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0.4 & 0.5 & 0.1 & 0 \\ 0 & 0.04 & 0.94 & 0.02 \\ 0.004 & 0 & 0.01 & 0.9896 \end{bmatrix}$$

D The Output for the Sensitivity Analysis for Different Cost Metrics

Table 4: The sensitivity analysis on the loss of QALY for different health statuses

Mortality				
State 1	**State 2**	**State 3**	**FIFO**	**MAA**
0.0088	0.04	0.2	0.009373137	0.004968989
0.0088	0.05	0.24	0.011234773	0.005599081
0.0088	0.06	0.28	0.013141939	0.006260247
0.0088	0.07	0.32	0.015052087	0.006918394
0.0088	0.08	0.36	0.016910691	0.007568797
0.0088	0.09	0.4	0.01880927	0.008247381
0.0088	0.1	0.44	0.020667925	0.008856202
0.0088	0.11	0.48	0.02261545	0.009517932
0.0088	0.12	0.52	0.024512753	0.010123149
0.0088	0.13	0.56	0.026414951	0.010846279
0.0088	0.14	0.6	0.028267269	0.011465171
0.0088	0.15	0.64	0.030150618	0.012068868
0.0088	0.16	0.68	0.032127914	0.012751327
0.0088	0.17	0.72	0.033908572	0.013456425
0.0088	0.18	0.76	0.035788183	0.014029379
0.0088	0.19	0.8	0.037776993	0.014721036
0.0088	0.2	0.84	0.039759045	0.015307521
0.0088	0.21	0.88	0.041621518	0.016053822
0.0088	0.22	0.92	0.043439444	0.016791645
0.0088	0.23	0.96	0.045355235	0.017253223

Table 5: The sensitivity analysis on the mortality risk for different health statuses

QALY				
State 1	**State 2**	**State 3**	**FIFO**	**QAA**
0.227	0.3	0.4	25.5977368	66.4243367
0.227	0.35	0.48	30.4338803	67.9466011
0.227	0.4	0.56	35.3827377	69.1959349
0.227	0.45	0.64	40.2849058	70.4444276
0.227	0.5	0.72	45.280182	71.9143304
0.227	0.55	0.8	49.9506096	73.1814243
0.227	0.6	0.88	55.177484	74.5437559
0.227	0.65	0.96	59.7506538	75.8601295
0.227	0.7	1.04	64.7430135	77.2751438
0.227	0.75	1.12	69.5318802	78.7765697
0.227	0.8	1.2	74.523255	80.1882281
0.227	0.85	1.28	79.4326621	81.6236768
0.227	0.9	1.36	84.2824946	83.096942
0.227	0.95	1.44	89.2229236	84.0024742
0.227	1	1.52	94.3231503	85.9056929
0.227	1.05	1.6	98.6279677	87.0489696
0.227	1.1	1.68	104.191477	88.5932074
0.227	1.15	1.76	108.903907	89.7652782
0.227	1.2	1.84	113.284431	91.3126296
0.227	1.25	1.92	118.610133	92.3255587

Table 6: The sensitivity analysis on the readmission risk for different health statuses

Readmission				
State 1	**State 2**	**State 3**	**FIFO**	**RAA**
0.113	0.35	0.15	0.01165316	0.02597588
0.113	0.375	0.16	0.01241734	0.02622708
0.113	0.4	0.17	0.01320881	0.02644507
0.113	0.425	0.18	0.01399314	0.02670622
0.113	0.45	0.19	0.01482692	0.02696848
0.113	0.475	0.2	0.01558528	0.0271234
0.113	0.5	0.21	0.01636164	0.02740573
0.113	0.525	0.22	0.01718992	0.02760681
0.113	0.55	0.23	0.01794766	0.02789974
0.113	0.575	0.24	0.01876118	0.02813655
0.113	0.6	0.25	0.01956144	0.02836617
0.113	0.625	0.26	0.02035083	0.02859686
0.113	0.65	0.27	0.02110185	0.02886266
0.113	0.675	0.28	0.02195666	0.02917199
0.113	0.7	0.29	0.02269879	0.0293007
0.113	0.725	0.3	0.02338711	0.02960412
0.113	0.75	0.31	0.02433822	0.02985923
0.113	0.775	0.32	0.02509731	0.03008759
0.113	0.8	0.33	0.02580833	0.03041928
0.113	0.825	0.34	0.02667324	0.03058669