An investigation into hospital triage

ICA 2 Stat0009

2024-02-20

Contents

1 Introduction to Hospital Triage

Hospital triage is a critical process used in emergency medicine to manage patient flow efficiently and prioritise care based on the urgency of medical conditions. This system originated from military practices, where it was essential to classify soldiers' injuries to decide who needed immediate attention [citation_Mitchell_2008]. The principle behind triage is to do the greatest good for the greatest number of people, which is especially vital during mass casualty incidents or when healthcare resources are limited.

The current healthcare scenario emphasises the critical need for efficient triage due to escalating pressures on emergency departments. Recent data from [citation_NuffieldTrust2024] reveals that the median waiting time for A&E patients in England reached 1 hour and 4 minutes in November 2023, indicating significant challenges in delivering timely care. Furthermore, projections from [citation_HealthFoundation2024] suggest that the NHS waiting list could surpass 8 million by summer 2024, pointing to increasing healthcare demands.

These statistics highlight the importance of triage in managing patient flow and ensuring that urgent cases receive prompt attention. Triage ensures that patients with life-threatening conditions receive immediate care, thus maximizing the chances of survival and recovery. By assessing and categorising patients based on the severity of their conditions, triage helps in reducing waiting times for critical cases and makes the best use of available medical resources. This system also helps in preventing the overcrowding of emergency departments, thereby maintaining a manageable workflow and reducing stress on healthcare professionals [citation_Love2012].

1.1 Overview of existing triage systems

There are several triage systems used worldwide, each with its methodologies and criteria for prioritising patients. Some of the most common systems include [citation_Hinson2019]:

Manchester Triage System (MTS): Originating in the UK, the MTS uses a flowchart system to categorise patients into five urgency levels, ranging from immediate attention to non-urgent.

Emergency Severity Index (ESI): Widely used in the United States, the ESI is a five-level triage algorithm that categorises patients based on the severity of their conditions and the number of resources their care is anticipated to require.

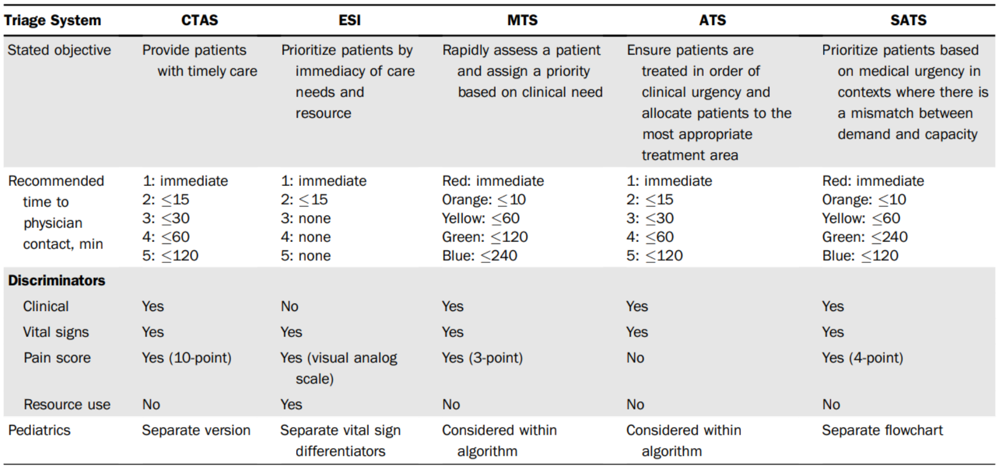
Australian Triage Scale (ATS): Similar to other systems, the ATS classifies patients into five categories, from those needing immediate attention to those who can afford to wait longer.

Canadian Triage and Acuity Scale (CTAS): In Canada, the CTAS is utilized, which similarly assigns patients to one of five categories, ranging from resuscitation to non-urgent, based on their health condition's severity and complexity.

South African Triage Scale (SATS): Used in South Africa, SATS prioritises emergency department patients with a scoring system based on vital signs and presenting complaints to determine urgency levels for treatment.

These systems differ mainly in their assessment criteria, algorithms, and the specific terms used to categorise the levels of urgency. However, all aim to quickly identify patients who need immediate life-saving interventions and manage resources effectively. See Figure 1 for a detailed overview of the triage systems.

The evolution of hospital triage has been marked by a continuous effort to improve accuracy, efficiency, and fairness in patient assessment. From simple, subjective decision-making based on visible injuries or symptoms, triage has evolved into a more systematic and objective process. Innovations in triage include the incorporation of technology, such as computer-based algorithms and digital health monitoring systems, which help in standardizing the process, reducing human error and increasing real-time communication [citation_Gao2007].



Triage System	CTAS	ESI	MTS	ATS	SATS
Stated objective	Provide patients with timely care	Prioritize patients by immediacy of care needs and resource	Rapidly assess a patient and assign a priority based on clinical need	Ensure patients are treated in order of clinical urgency and allocate patients to the most appropriate treatment area	Prioritize patients based on medical urgency in contexts where there is a mismatch between demand and capacity
Recommended time to physician contact, min	1: immediate 2: ≤15 3: ≤30 4: ≤60 5: ≤120	1: immediate 2: ≤15 3: none 4: none 5: none	Red: immediate Orange: ≤10 Yellow: ≤60 Green: ≤120 Blue: ≤240	1: immediate 2: ≤15 3: ≤30 4: ≤60 5: ≤120	Red: immediate Orange: ≤10 Yellow: ≤60 Green: ≤240 Blue: ≤120
Discriminators					
Clinical	Yes	No	Yes	Yes	Yes
Vital signs	Yes	Yes	Yes	Yes	Yes
Pain score	Yes (10-point)	Yes (visual analog scale)	Yes (3-point)	No	Yes (4-point)
Resource use	No	Yes	No	No	No
Pediatrics	Separate version	Separate vital sign differentiators	Considered within algorithm	Considered within algorithm	Separate flowchart

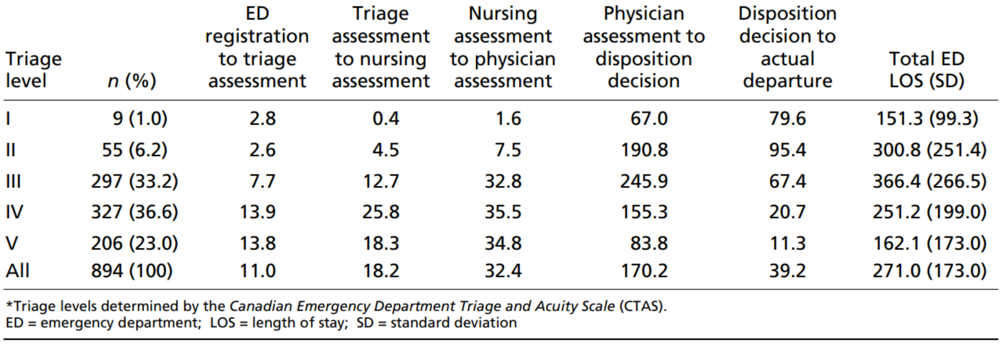
Figure 1: Triage Overview. Source: Hinson et al. (2019)

1.2 Canadian Triage and Acuity Scale (CTAS)

In this project, we will use the CTAS framework due to its comprehensive and clear directives for assigning treatment times based on patient severity, along with the availability of detailed data that supports modeling decisions. Below is a breakdown of the different triage levels [citation_CTAS]:

- **Level 1: Resuscitation**: E.g. Cardiac arrest or severe respiratory distress.
- **Level 2: Emergent**: E.g. Chest pain with sweating suggestive of a heart attack.
- **Level 3: Urgent**: E.g. Severe abdominal pain that could indicate a serious condition.
- **Level 4: Less Urgent**: E.g. A sprained ankle with swelling and moderate pain.
- **Level 5: Non-Urgent**: E.g. A patient with a minor chronic issue seeking a prescription refill.

citation_Yoon_Steiner_Reinhardt_2003 provides detailed data on the distribution of these CTAS levels by through studying $n = 894$ patients in a time frame of 7 days at the University of Alberta Hospital:



Triage level	*n* (%)	ED registration to triage assessment	Triage assessment to nursing assessment	Nursing assessment to physician assessment	Physician assessment to disposition decision	Disposition decision to actual departure	Total ED LOS (SD)
I	9 (1.0)	2.8	0.4	1.6	67.0	79.6	151.3 (99.3)
II	55 (6.2)	2.6	4.5	7.5	190.8	95.4	300.8 (251.4)
III	297 (33.2)	7.7	12.7	32.8	245.9	67.4	366.4 (266.5)
IV	327 (36.6)	13.9	25.8	35.5	155.3	20.7	251.2 (199.0)
V	206 (23.0)	13.8	18.3	34.8	83.8	11.3	162.1 (173.0)
All	894 (100)	11.0	18.2	32.4	170.2	39.2	271.0 (173.0)

*Triage levels determined by the *Canadian Emergency Department Triage and Acuity Scale* (CTAS).
ED = emergency department; LOS = length of stay; SD = standard deviation

Figure 2: Emergency Room times conditional on triage level. Source: Yoon et al. (2003).

The distribution of patients across the Canadian Triage and Acuity Scale (CTAS) levels in the study, where 93% of patients fell into levels 3, 4, and 5, reflects a common trend in emergency departments (EDs) where the majority of cases are of less severe nature. Counter-intuitively, patients of CTAS Level 3 spent the longest time in the emergency department. citation_Yoon_Steiner_Reinhardt_2003 state this is because patients often arrive with unclear symptoms, making it difficult to immediately decide on admitting or discharging them, leading to longer stays in the ED for further observation and testing.

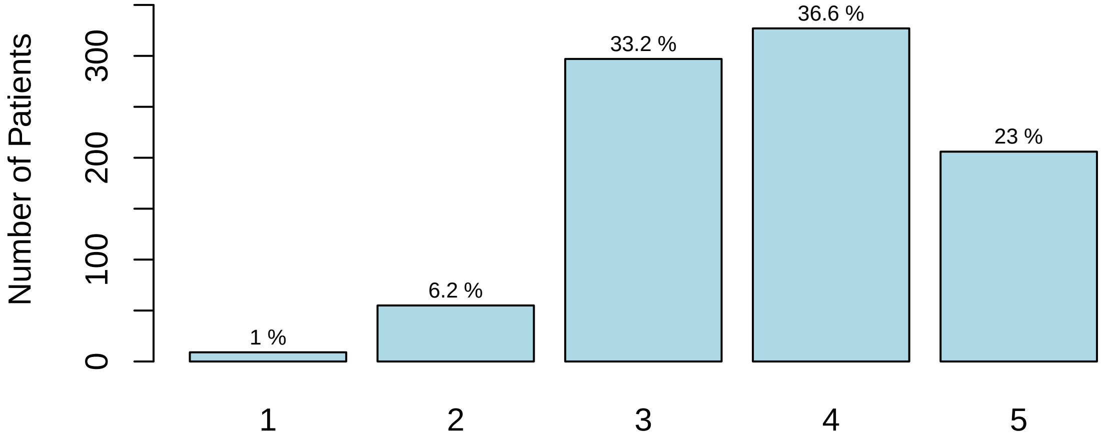


Figure 3: Number of Patients Per Triage Level

We will make informed decisions based on the data from citation_Yoon_Steiner_Reinhardt_2003 within our paper, as we will have reference values to benchmark and evaluate our models.

2 Aim of project

The aim of this project is to analyse hospital triage systems through the lens of queueing theory and simulations, thereby uncovering factors that influence their efficiency and effectiveness. By simulating various queueing scenarios, this project seeks to bridge the gap between theoretical models and the complex realities of emergency department operations. Our objective is to assess the performance of current triage practices, particularly those akin to the Canadian Triage and Acuity Scale (CTAS), under different conditions and constraints. This investigation will enable us to evaluate the robustness of existing triage systems based on empirical evidence and theoretical insights.

2.1 Overview of simulations

The project is structured around five key simulation scenarios, each designed to iteratively simulate hospital triage dynamics with increasing accuracy:

1. **MG1 Queue Simulation**: This foundational model will simulate a basic queueing system with a single server, representing a simplified version of patient flow in an emergency department. This scenario will serve as a benchmark, helping us understand the baseline performance of a triage system under the simplest conditions.

2. **Priority Queue with Severity (1-5 mimicking the Canadian scale)**: Building on the MM1 model, this simulation will introduce a priority queueing mechanism based on a 5-level severity scale similar to the CTAS. This scenario aims to explore how prioritization based on severity affects patient wait times and overall system efficiency.

3. **MMC Queue with Priority**: This model will extend the priority queue concept to a more complex MMC system, where multiple servers are available, representing multiple healthcare providers in an emergency department. This scenario will help us understand the impact of resource availability on the performance of a priority-based triage system.

4. **MMC with Conditional Patient Departure**: This simulation will investigate scenarios where patients decide to leave the queue based on the severity of their condition. The likelihood of departure will vary, with those having less severe conditions potentially leaving sooner due to prolonged wait times. This approach will allow us to analyze the impact of patient abandonment on system efficiency and explore strategies to retain patients until they receive care.

5. **MMC with Markov Chain for Condition Deterioration**: This simulation will use a Markov chain to model the progression of patients' conditions over time, including the possibilities of deterioration, departure due to frustration or perceived neglect, and death. This model will provide insights into how the triage system adapts to patients' evolving needs and the effectiveness of the system in preventing adverse outcomes through timely intervention.

6. **Incorporation of Real-World Data**: The final simulation will integrate real-world data into a sophisticated model to create a more accurate representation of an emergency department's triage system. This scenario aims to validate the theoretical models and simulations against actual patient flow data, offering a grounded assessment of triage system performance.

By examining these diverse scenarios, the project aims to identify key factors that influence the efficacy of triage systems, such as resource allocation, patient prioritization strategies, and the handling of dynamic patient states.

2.2 Measures of performance

In our project, we will employ key metrics to evaluate the effectiveness and shortcomings of hospital triage systems, focusing on quantifying the system's "loss" or "badness." These metrics will help in understanding the areas where triage systems may not perform optimally under various simulated conditions.

Total Waiting Time: This metric will assess the cumulative waiting period for patients before receiving care. Extended waiting times can indicate inefficiencies and can significantly impact patient outcomes, especially for those with urgent needs.

Violations of Triage System Guidelines: We will measure instances where the simulated triage processes fail to meet the established guidelines for maximum waiting times, based on the severity of patients' conditions. This will help identify how often and by how much current triage practices deviate from their intended standards.

Number of Patients Left Unseen (Indicator of Satisfaction): This metric counts patients who leave without being seen, serving as an indirect measure of patient satisfaction. High numbers may indicate dissatisfaction due to long waits, highlighting areas for improvement in the triage system to enhance patient experience and reduce departures. We aim to minimize the number of patients left unseen, ensuring everyone receives timely care, especially those in urgent need.

By integrating these metrics into our simulation analysis, we aim to create a holistic assessment of triage systems, identifying key areas for improvement. This approach will enable us to propose targeted recommendations for enhancing triage efficiency, reducing waiting times, adhering more closely to triage guidelines, and ultimately, minimizing preventable patient deaths.

3 Simulations

3.1 M/G/1 FIFO Queue

For modeling an emergency department as a simple starter case with one doctor, we employ the MG1 queueing model. In this scenario, "M" indicates that patient arrivals follow a Poisson process, reflecting the randomness of emergency visits. "G" signifies that the service times, or the time each patient spends with the doctor, follow a general distribution—specifically, a Gamma distribution with shape 6 and rate 0.1, as recommended by citation_Oladimeji2020 for healthcare settings. The "1" in MG1 highlights that there is a single server, in this case, one doctor, available to attend to patients. Additionally, FIFO means the queue uses a First In First Out System, where patients are prioritized by their arrival time.

We know that the median time spent waiting in the emergency department was 64 minutes in November 2023 [citation_NuffieldTrust2024]. However, we could not find a reasonable parameter value for the exponential arrivals (λ) and will thus calculate it theoretically.

To calculate the value of λ, we will use the Pollaczek–Khinchine formula [citation_Pollaczek1930] [citation_Khi32], which gives the relationship between the Laplace transform of the service time distribution and the queue length. However, it can be recast to show the relationship between the mean length of the queue, and the mean waiting time [citation_APAQ2003, p. 192]:

$$L = \rho + \frac{\rho^2 + \lambda^2 \operatorname{Var}(S)}{2(1-\rho)} \tag{1}$$

In the above, L denotes the mean length of the queue, ρ renotes the capacity utilization of the queue, and S is the distribution of the service time. Furthermore, ρ is defined as:

$$\rho = \frac{\lambda}{\mu}$$

where:

$$\mu = \frac{1}{\mathbb{E}(S)}$$

thus:

$$\rho = \lambda \mathbb{E}(S)$$

For a stable system, we desire $\rho \leq 1$ [citation_VirtamoPriorityQueues]. If $\rho \geq 1$, the queue length would tend to infinity, as more people arrive than can be served. In our case, we want to find a value of λ such that $S \sim \text{Gamma}(6, 0, 1)$, and ρ. In order to find a value of λ such that the expected waiting time is 64 minutes, we turn to Little's Law, which gives:

$$L = \lambda W \tag{2}$$

where W is the mean total time spent in a system. We will define W_1 as the time spent in the waiting room, and thus:

$$W = W_1 + \mathbb{E}(S) \tag{3}$$

Substituting (1) and (3) into (2) and re-arranging:

$$\frac{L}{\lambda} = W_1 + \mathbb{E}(S)$$

$$\frac{\rho}{\lambda} + \frac{\rho^2 + \lambda^2 \operatorname{Var}(S)}{2\lambda(1-\rho)} = W_1 + \mathbb{E}(S)$$

As $\frac{\rho}{\lambda} = \mathbb{E}(S)$:

$$\frac{\rho^2 + \lambda^2 \operatorname{Var}(S)}{2\lambda(1-\rho)} = W_1$$

Using $\rho = \lambda \mathbb{E}(S)$:

$$\frac{(\lambda \mathbb{E}(S))^2 + \lambda^2 \operatorname{Var}(S)}{2\lambda(1-\lambda \mathbb{E}(S))} = W_1$$

Simplifying:

$$\lambda \frac{\mathbb{E}(S)^2 + \operatorname{Var}(S)}{2(1-\lambda \mathbb{E}(S))} = W_1$$

The above could be further simplified by noticing $\mathbb{E}(S)^2 + \operatorname{Var}(S) = \mathbb{E}(S^2)$, but we choose to leave the equation in the above form for ease of calculation.

Recall the objective is to calculate λ such that $w_1 = 64$, as motivated by [citation_NuffieldTrust2024]. We therefore substitute parameters α and β into our above equation:

$$\lambda \frac{\left(\frac{\alpha}{\beta}\right)^2 + \frac{\alpha}{\beta^2}}{2\left(1 - \lambda \frac{\alpha}{\beta}\right)} = W_1$$

Substituting our desired values:

$$\lambda \frac{\left(\frac{6}{0.1}\right)^2 + \frac{6}{0.1^2}}{2\left(1 - \lambda \frac{6}{0.1}\right)} = 64$$

$$\lambda \frac{60^2 + 600}{2 - 120\lambda} = 64$$

This can be solved numerically (e.g. via. Newton-Raphson) to find:

$$\lambda \approx 0.0108$$

We can verify this using simulations to find that the mean waiting time in our M/G/1 queue is indeed 64 minutes, using the law of large numbers.

```r
# Define a function to create severities with probabilities from Yoon et al. (2003)
severity_gen <- function(n) {
    # Generates a sample of severity levels (1 to 5) for 'n' patients
    sample(1:5, n, replace = TRUE, prob = c(0.01, 0.062, 0.332, 0.366, 0.23))
}

# Simulates a triage system using a gamma distribution for service times
# from Oladimeji & Ibidoja (2020)
triage_MG1 <- function(n, arrival_rate) {
    # Generate cumulative arrival times based on an exponential distribution
    arrivals <- cumsum(rexp(n, arrival_rate))
    # Generate a list of severities for 'n' patients
    severity_list <- severity_gen(n)
    # Combine arrivals and severities to represent patients in the waiting room
    waiting_room <- cbind(arrivals, severity_list)

    # Initialize a matrix to store wait times and priorities
    wait_times <- matrix(numeric(0), nrow = 0, ncol = 2)
    # Initialize service time
    services <- 0

    # Process patients until all have been served
    while(services < max(arrivals) && nrow(waiting_room) > 0) {

        # Identify patients available for service
        available_patients <- waiting_room[waiting_room[, 1]<=services,,drop = FALSE]
        # Assume the next patient is the first in line (FIFO)
        next_patient_index <- 1
        next_patient <- waiting_room[next_patient_index, , drop = FALSE]
        # Calculate wait time for the patient
        wait <- max(0, services - next_patient[1])
        priority <- next_patient[2]

        # Add patient's wait time and priority to the record
        wait_times <- rbind(wait_times, c(wait, priority))
        # Remove the served patient from the waiting room
        waiting_room <- waiting_room[-next_patient_index, , drop = FALSE]
        # Update the chosen server's next available service time
        services <- max(services, next_patient[1]) + rgamma(1, 6, 0.1)
    }

    # Assign column names to the wait times matrix
    colnames(wait_times) <- c("Wait", "Priority")
    return(wait_times)
}
```

```
# Using arrival rate = 0.0108 as calculated
queue = triage_MG1(n, arrival_rate = 0.0108)
mean(queue[,"Wait"])
```

```
## [1] 69.09165
```

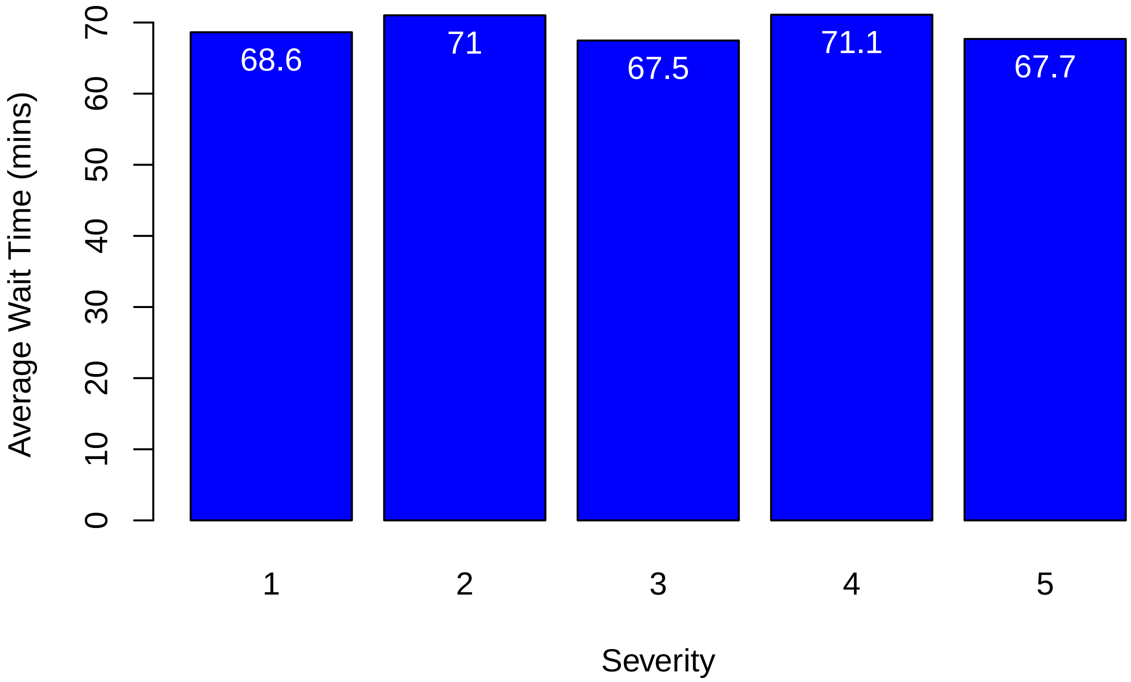


Figure 4: Average Wait Times For MG1 Queue

In comparing the average waiting times per severity level within this service system, it becomes evident that despite variations in patient severity, the average waiting times remain remarkably uniform. This uniformity raises concerns regarding the efficiency of the FIFO service process in adequately addressing the needs of patients with varying degrees of severity. Particularly troubling is the potential impact on patients with lower severity levels, who may face disproportionately longer waits in spite of higher urgency. Such delays in accessing medical attention pose significant risks, as prolonged wait times could exacerbate their conditions and lead to detrimental outcomes. This highlights the critical importance of reevaluating the FIFO service approach within the broader context of patient care and service system. We know that hospitals typically employ more nuanced triage systems that take other factors such as severity and total time waited, into account. Moreover, the inclusion of the FIFO system in our analysis serves as a foundation benchmark for our coming exploration from which we will evaluate and refine more sophisticated triage models.

3.2 Generic Function

The following function serves as a versatile tool for investigating the impact of different service systems in a triage scenario. By incorporating the priority system as a parameter, we can isolate the variable being changed to solely the service process and the method of patient prioritization. This flexibility enables us to simulate a wide range of theoretical queuing systems that utilise prioritisation based on severity and/or time, all within a single function.

Moreover, the function is designed to be scalable, accommodating various simulations by allowing the number of patients, number of doctors, and arrival rate to be specified as parameters. Therefore, this function enhances the modularity of our simulations. This modularity empowers us to easily conduct diverse scenarios using simple command codes, streamlining the evaluation process and facilitating comprehensive analysis.

```r
triage_sim <- function(n, drs, arrival_rate, priority_system){
  severity_gen <- function(n) {
    sample(1:5, n, replace = TRUE, prob = c(0.01, 0.062, 0.332, 0.366, 0.23))
  }
  arrivals <- cumsum(rexp(n, arrival_rate))
  severity_list <- severity_gen(n)
  waiting_room <- cbind(arrivals, severity_list)

  wait_times <- matrix(numeric(0), nrow = 0, ncol = 3)
  services <- rep(0, drs)

  while(min(services) < max(arrivals) && nrow(waiting_room) > 0) {


    chosen_server <- which.min(services)


    time_of_service <- services[chosen_server]


    available_patients <- waiting_room[waiting_room[, 1] <= time_of_service, , drop = FALSE]



    if (nrow(available_patients) != 0){
    severity_scores <- available_patients[, 2]
    arrival_times <- available_patients[, 1]
    holding_times <- available_patients[,ncol(available_patients)]
    current_wait_times <- time_of_service - unlist(arrival_times)

    if (nrow(available_patients) == 1 ) {
      next_patient_index <- 1
    }

    else {
      next_patient_index <- priority_system(severity_scores, current_wait_times)
    }
    }
    else {
      next_patient_index <- 1
      }
    next_patient <- waiting_room[next_patient_index, , drop = FALSE]


    wait <- max(0, time_of_service - unlist(next_patient[1]))
    priority <- unlist(next_patient[2])

    service_time <- rgamma(1,6,priority/35)

    los <- wait + service_time
    wait_times <- rbind(wait_times, c(wait, priority,los))
    waiting_room <- waiting_room[-next_patient_index, , drop = FALSE]
    services[chosen_server] <- max(time_of_service, unlist(next_patient[1])) + service_time
  }
  return(wait_times)
}
```

```
#priority_system inputs for the triage_sim function
FIFO_system <- function(severity_scores, current_wait_times){1}
severity_system <- function(severity_scores, current_wait_times){which.min(severity_scores)}
mixed_system <- function(severity_scores, current_wait_times){
  severity_weight = 1
  wait_time_weight = 1.5
  scores <- mapply(function(severity, wait_time){
    if (severity == 1){
      priority_score <- Inf
    }
    else {
      severity_score <- 6 - severity
      wait_time_score <- log(wait_time + 1)
      priority_score <- severity_weight * severity_score + wait_time_weight * wait_time_score
    }
    return(priority_score)
  }, severity_scores, current_wait_times)

  return(which.max(scores))
}
```

3.3 M/G/K Queue

Expanding our analysis to a M/G/K queue, which involves multiple servers, brings us closer to real-world situations by adding complexity to the model. One significant factor that has not yet been incorporated is the consideration that patients with higher severity levels require longer treatment times. To simulate this, we have chosen to employ a service time distribution modeled by a $Gamma(6, \frac{severity}{35})$ distribution, with the parameters being conditional on the patient's severity level. This decision is grounded in the work of citation_Oladimeji2020, which indicates that the mean service time in a hospital setting is approximately 60 minutes. Consequently, the service time distribution for a patient X_i, with a severity level $i \in 1, 2, 3, 4, 5$, is given by $X_i \sim Gamma\left(6, \frac{i}{35}\right)$. The weighted average of the service time across our severity-based system can be represented as:

$$\sum_{i=1}^{5} \mathbb{E}[X_i] P(X = i).$$

In our specific case, this equation simplifies to:

$$210 \left(0.01 + \frac{1}{2} \cdot 0.062 + \frac{1}{3} \cdot 0.332 + \frac{1}{4} \cdot 0.366 + \frac{1}{5} \cdot 0.23 \right) = 60.725.$$

The proportions of severities follows from citation_Yoon_Steiner_Reinhardt_2003. This approach allows us to more accurately model the service dynamics within a hospital environment, accounting for the variability in treatment times as a function of patient severity. Additionally, our figure of 60.7 is close to our value of a 60 minute service time. According to [citation_Millington2018], there are typically 12 specialist ER doctors available, which is the figure we will use henceforth.

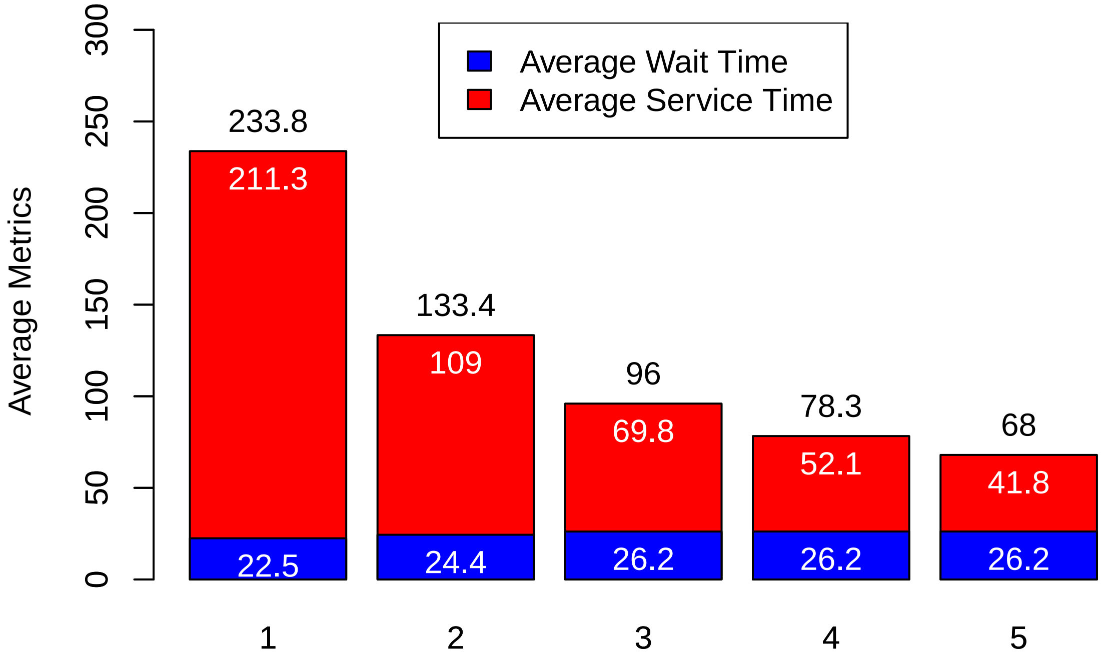


Figure 5: Wait and Service Times For FIFO MGK

Figure 5 indicates that waiting times remain indistinguishable across different severities, mirroring the findings observed in the M/G/1 FIFO system. This uniformity in waiting times raises the same concerns regarding its implications for patient outcomes and satisfaction as before.

This suggests that, irrespective of the complexity of the service distribution, a FIFO service approach fails to adequately address the varying urgency of patient needs, thus motivating the development of a dynamic service process that incorporates several factors other than the arrival time of patients when prioritising the queue.

Our choice of including multiple servers in the simulation aligns with the resource allocation typically observed in triage settings found in average hospitals in Europe and North America. However, we anticipate that the results from our recommendations will be effective in a variety of triage scenarios regardless of their respective resources.

3.4 M/G/K Severity Based Priority Queue

In our revised queuing system, we have implemented a prioritisation process aimed at optimising the time until a patient is attend to. Whereas the previous approach where patients were ordered by the time of their arrival, our system now prioritises patients according to their severity scores, with higher urgency (lower severity score) patients receiving immediate attention. In cases where multiple patients share the same severity score, priority is given to the patient who arrived first at the hospital.

Our analysis of the resulting waiting times, as depicted in Figure 6, demonstrates the efficacy of this new process in minimizing wait times for patients requiring urgent care. The waiting times across different severity levels exhibit an observable pattern, with patients of higher urgency experiencing shorter wait times compared to those with lower severity scores. This reduction in wait times for high-urgency patients underscores the success of our system in aligning service delivery with patient needs and priorities.

While it is evident that the total time spent in the system remains relatively high for patients with the highest urgency (severity of 1), it is important to emphasize that our primary objective is to minimize the average time until a patient is seen, particularly for those in urgent need of care. By focusing on this key metric, this system could be considered as significantly more effective than the FIFO system.

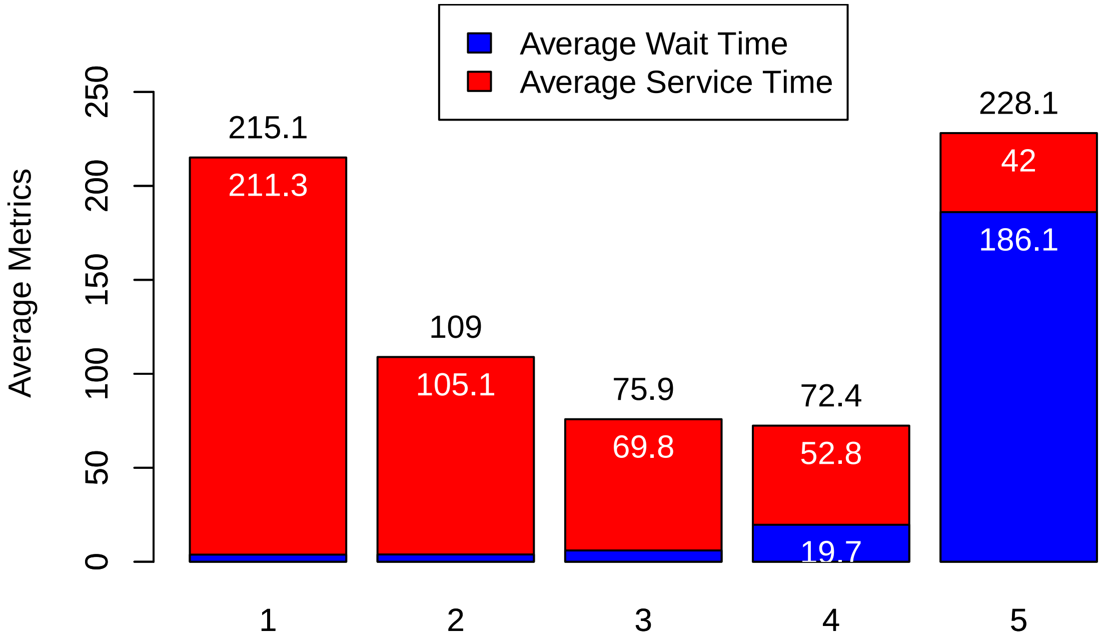


Figure 6: Wait and Service Times For Severity-Based MGK

However, although the urgency of the patients with a CTAS score of 5 is low, in accordance with patient care thresholds regarding the maximum amount of time patients should have to wait to be seen, the waiting time for these patients could be seen as too high. Therefore we will attempt to lower this time, on average, by introducing other factors into our service process prioritisation.

3.5 M/G/K (Severity + Time) Based Priority Queue

In our final iteration of the service process, we have devised an approach that integrates both patient severity and wait time to calculate a comprehensive severity score. By prioritising patients based on this score, which is derived from a carefully calibrated weighting of severity and wait time, we aim to balance between urgency and fairness in service delivery. Through extensive simulations and numerical optimization using grid search techniques, we have fine-tuned the two key parameters governing the severity score calculation to optimise the performance of the service process.

The results from Figure 7 demonstrate the success of this mixed service process in reducing the average wait time for patients with lower urgency (CTAS 5), compared to a system that prioritises solely based on severity. While it is true that the average waiting times for more urgent patients may have increased as a consequence, it is important to note that all waiting times remain comfortably below the maximum thresholds established in the guidelines in Figure 1 , Section 1.1. This indicates that despite the adjustments made to increase the priority severity 5 patients, the system remains responsive to the needs of urgent cases.

It is worth highlighting that patients with the highest severity (CTAS 1) are assigned a severity score of infinity in our system, reflecting the real-life urgency associated with such cases. This ensures that patients in critical condition are promptly attended to thus remaining consistent with established medical guidelines and protocols.

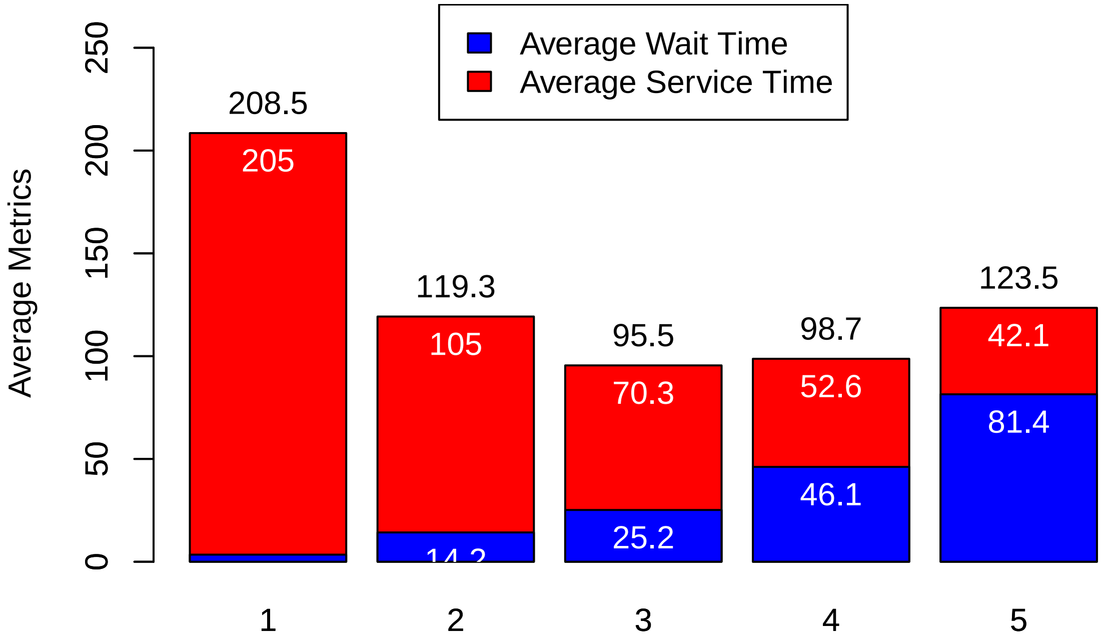


Figure 7: Wait and Service Times For Mixed System MGK

4 Patient depature

Another key metric that is used by triage systems to identify successful service prioritisation and processing is the number of patients that Left Without Being Seen (LWBS). To model this, we employ the Weibull distribution, as proposed by citation_Wiler2013, to determine their tolerance levels. If the waiting period surpasses a patient's calculated tolerance threshold, the patient will opt to leave. Specifically, we utilise the Weibull distribution parameters with a shape factor of 1.3 and a scale factor of 300, denoted as $\text{Weibull}(1.3, 300)$, to compute these tolerance times.

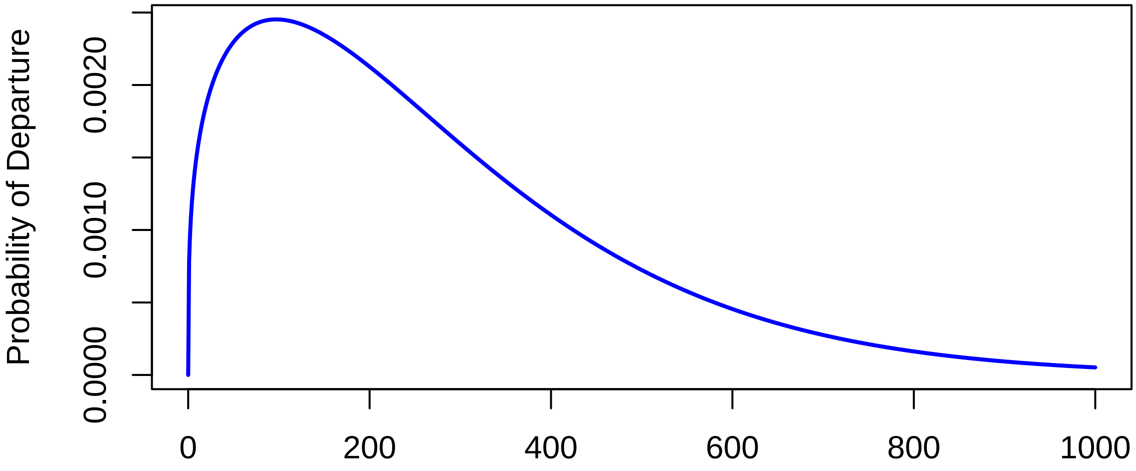


Figure 8: Weibull distribution of patient waiting time tolerance.

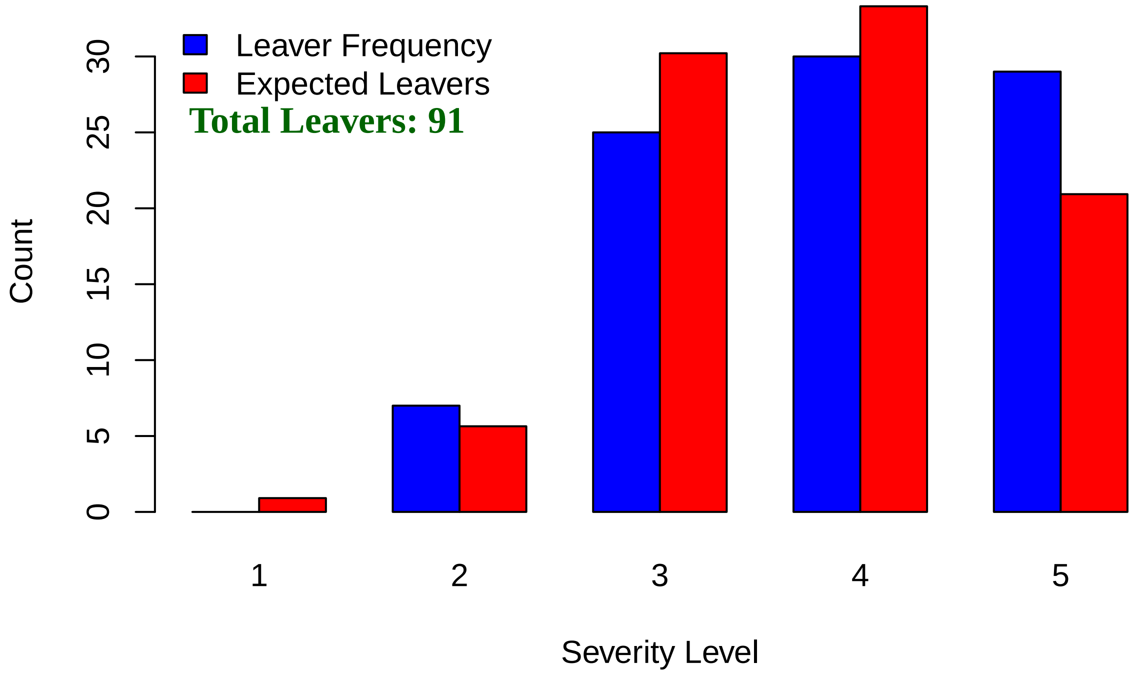


Figure 9: Number Of Leavers vs Proportion Of Arrivals For FIFO MGK

4.1 LWBS M/G/K FIFO Queue

Figure 9 depicts the number of leavers in a FIFO priority system, where the blue bars are the number of leavers per severity from a simulation of 20,000 patients. The red bar indicates the proportion of leavers as per the proportion of arrivals. The comparison is key as it highlights that the severity of the patients is not being taken into account in the service. The leaving of the patients with high severity (low CTAS) is detrimental as these are patients who could have potentially life threatening conditions and are not being seen to.

4.2 LWBS M/G/K Severity Based Priority Queue

In this system, while prioritising patients based on severity has its merits, there are noticeable drawbacks, particularly for patients with low severity (CTAS 5). As illustrated in Figure 10 , there has been a significant increase, in comparison to the results in Figure 9, in the number of low-severity patients who leave without being seen. This can be attributed to the fact that the threshold for these patients was reached before they could receive attention from a healthcare professional. Consequently, this trend has contributed to a notable uptick in the overall number of LWBS cases.

Although the decrease in LWBS cases among patients with life-threatening conditions may initially appear favorable, the stark contrast in the total number of LWBS cases between our current system and the FIFO prioritisation method raises concerns regarding the overall effectiveness of this approach. While it is encouraging to see a reduction in LWBS cases for patients in critical condition, the disproportionate increase in LWBS cases among low-severity patients suggests that the success of our system may be limited in this regard.

4.3 LWBS M/G/K (Severity + Time) Based Priority Queue

The priority system implemented above shows a significant improvement in mitigating the number of patients who leave without being seen (LWBS) compared to the system solely based on patient severity. In essence,

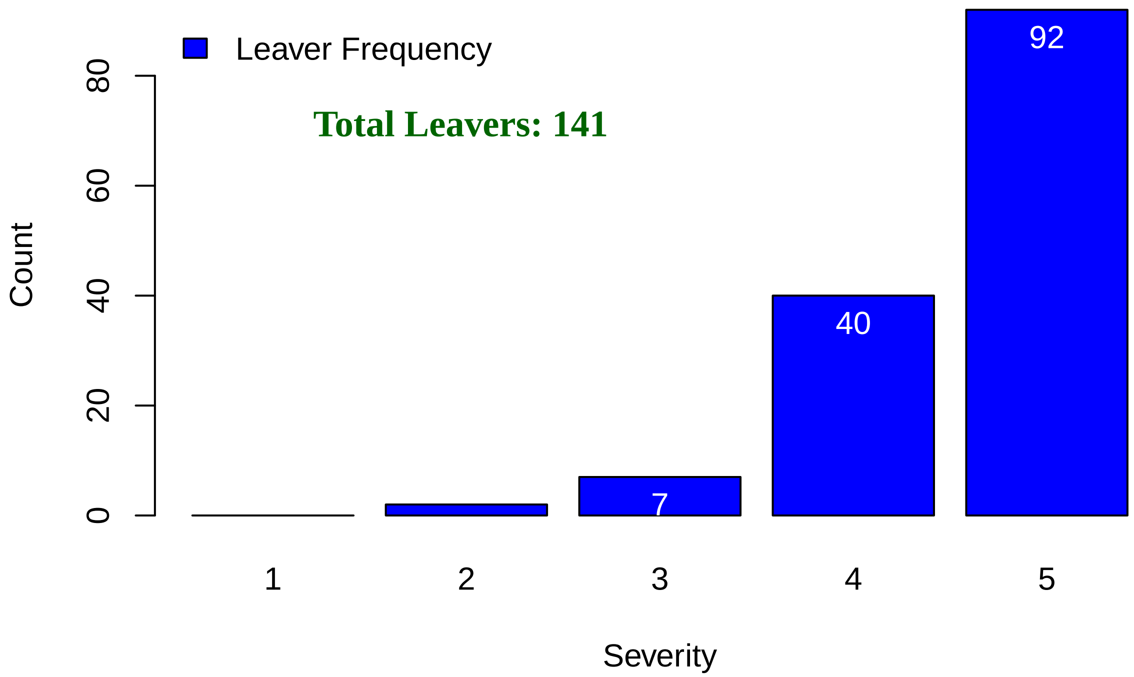


Figure 10: Number Of Leavers For Severity-Based MGK

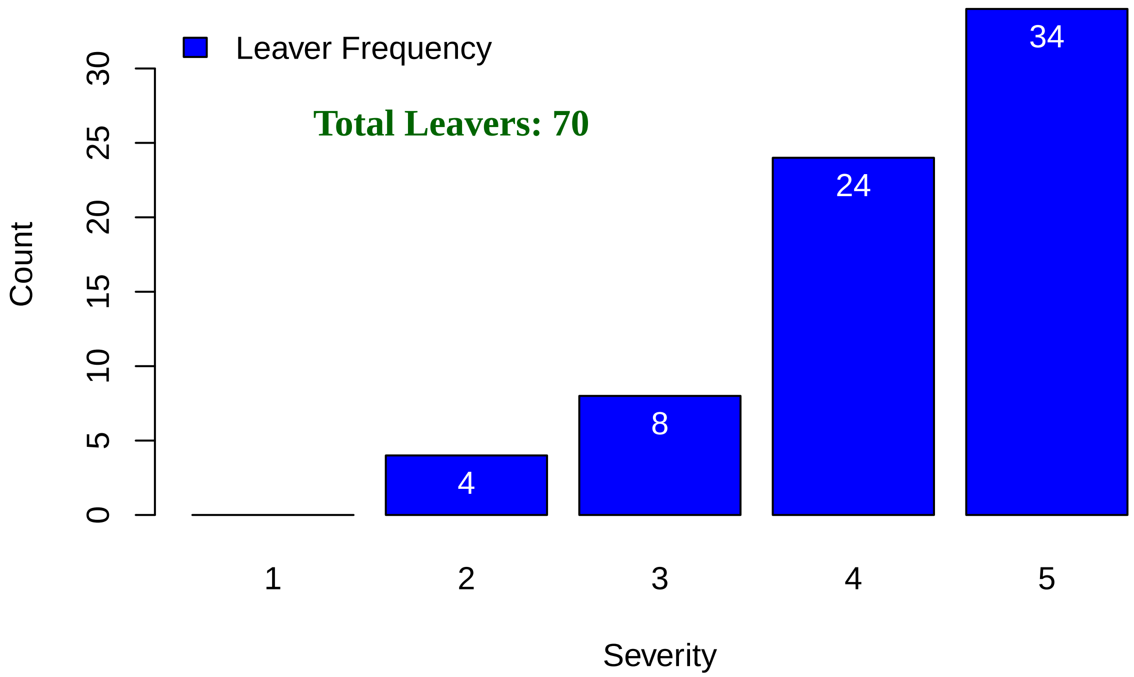


Figure 11: Number Of Leavers For Mixed System MGK

the severity-based prioritisation approach can be seen as the mixture model with a zero weighting on the total time waited by patients. By increasing the weighting of the time factor in our model, we have observed (in Figure 11) a notable reduction in the overall number of LWBS cases, as well as a decrease in the number of low-severity patients leaving prematurely.

This improvement is due to a marginal increase in the number of patients with higher severity scores leaving the system. Importantly, the structure of our priority system ensures that patients with the highest priority (CTAS 1) do not leave without being seen — a critical requirement for the success of any triage department.

The results from Figure 11, compared with plots depicting LWBS cases in other systems, serves as a compelling illustration of the success of the mixture service model. The identifiable decrease in LWBS cases, particularly among low-severity patients, highlights the tangible benefits of integrating both severity and time factors into our prioritisation process, ensuring an equitable level of patient care.

By striking a balance between severity-based prioritisation and FIFO service, our model will optimises successful outcomes and patient satisfaction.

5 Warzone Hospital Triage Simulation

We will now consider a different setting to conventional hospitals, in which the proportion of high severity patients is significantly greater: warzone military hospitals. Due to the high proportion of critical patients, the CTAS will not be able to distinguish between patients effectively, and we therefore propose a relative scale inspired by Red Cross methodologies [citation_REFERENCE]. Here severities 1-5 are relative, and deteriorate rapidly In practice, this might mean that a 5 is someone with a moderate shrapnel wound, whose condition may deteriorate in a matter of hours. We assume that this deterioration process can be modeled as a continuous-time, discrete-state space Markov Chain. We have included the generator matrix Q below and a diagram of the deterioration Markov Chain in Figure 12

$$Q = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ \frac{1}{20} & -\frac{1}{20} & 0 & 0 & 0 \\ 0 & \frac{1}{40} & -\frac{1}{40} & 0 & 0 \\ 0 & 0 & \frac{1}{60} & -\frac{1}{60} & 0 \\ 0 & 0 & 0 & \frac{1}{80} & -\frac{1}{80} \end{bmatrix}$$

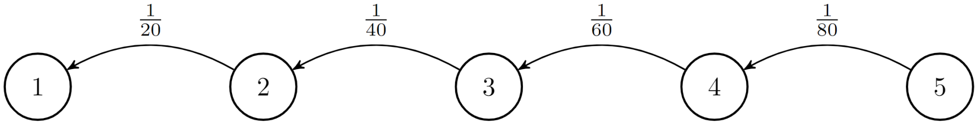


Figure 12: Diagram of Markov Chain Representing Severity Deterioration.

While we have formulated parameters that we consider reasonable, we acknowledge that our model may appear removed from the chaotic reality of warzone hospitals. We have crafted our methods and code to be modular, allowing for adjustment and calibration by practitioners who may have access to empirical data or wish to apply their mathematical estimations. This adaptability is crucial for the model's applicability across various conflict scenarios.

A notable modification from our previous environment is the adjustment of service times. Initially, severity-based service times resulted in a 'snowball' effect—minor delays led to increasing wait times and an upsurge in high-urgency cases. To counter this, we have revised the service time criteria to remove the condition on severity.

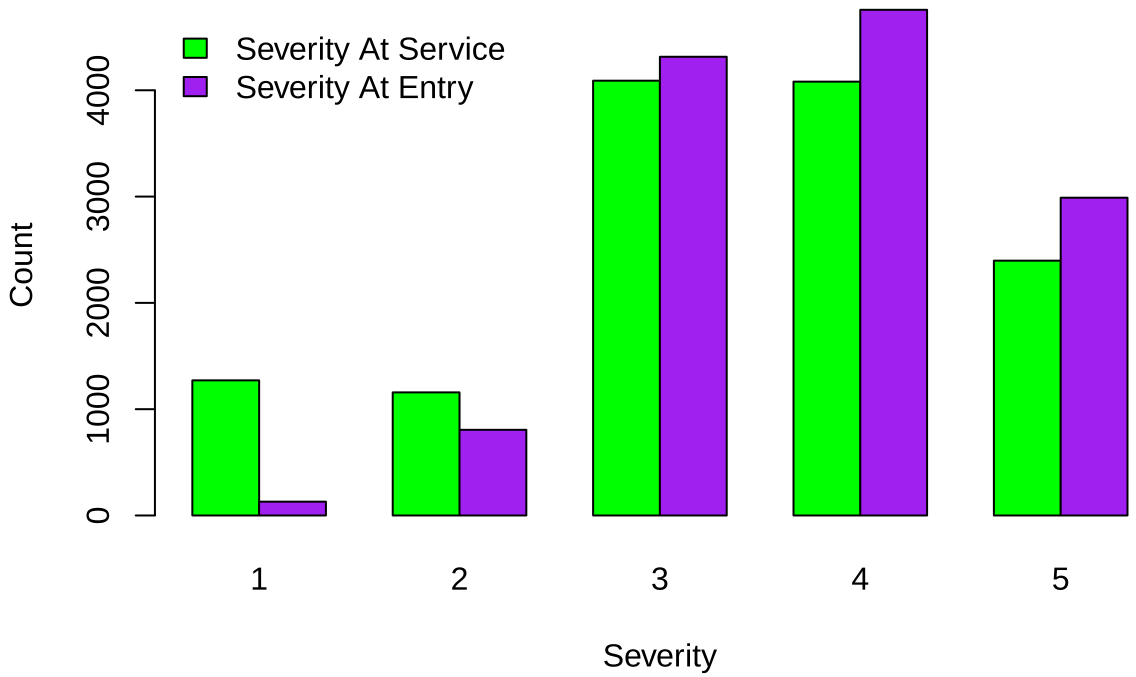


Figure 13: Comparison Of Severities At Entry vs Service. - Warzone Hospital

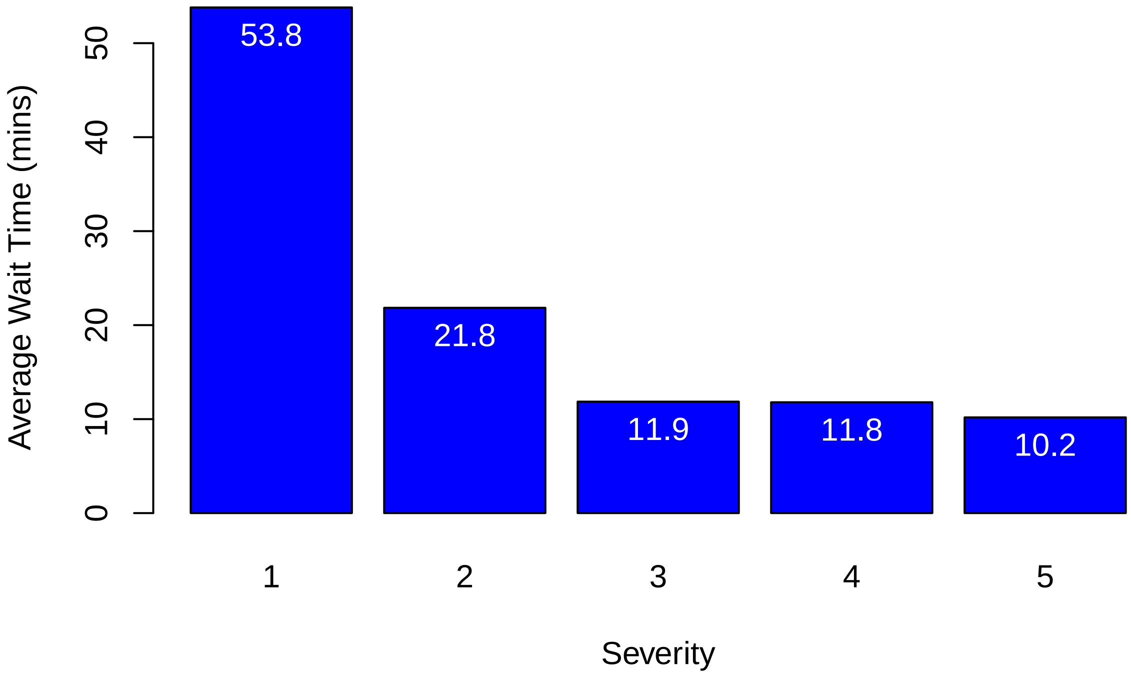


Figure 14: Comparison Of Wait Times For Mixed System - Warzone Hospital

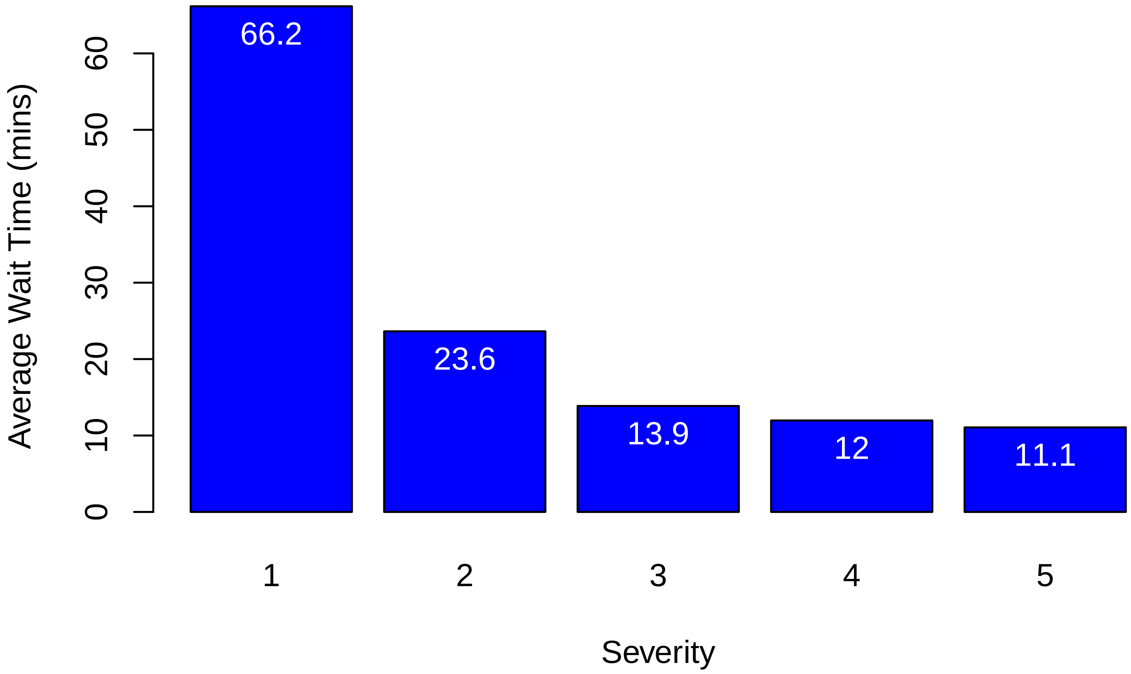


Figure 15: Comparison Of Wait Times For FIFO System - Warzone Hospital

In our empirical analysis, as depicted in Figure 13, we observe a marked increase in high-urgency cases at the point of service compared to their entry condition. This highlights the progressive nature of patient condition decay and underscores the importance of an adaptive triage system in warzone scenarios.

Our comparative studies, illustrated in Figures 14 and 15 reveal minimal differences between our specialised triage system and a traditional First-In-First-Out (FIFO) approach in a warzone context. This outcome suggests that while our system excels in conventional emergency departments, its efficacy in other environments is not conclusively superior, thereby necessitating further research.

6 Conclusion remarks

6.1 Limitations

In our study, we simulated the triage process within a hospital setting under the assumption that all individuals undergo correct classification according to their immediate medical needs. However, this assumption presents a significant limitation to the realism and applicability of our findings to actual healthcare environments. In real-life scenarios, the triage process is subject to human error and variability in judgment, leading to potential misclassification of patients (Zachariasse et al., 2019). This discrepancy can affect patient outcomes, resource allocation, and overall efficiency of emergency care services. Consequently, while our simulation provides valuable insights into the potential functioning of an ideal triage system, it may not fully capture the complexities and challenges inherent in real-world triage practices, including the implications of misclassification on patient care and system performance.

The process which we use in the above experiments to assign a severity to patients does not take into account the time of day. This is something which may not align with reality as a patient with a low severity may not choose to attend a triage at an unsociable hour. We have uniformly applied the proportions of patient severity to every patient that arrives to determine or model their severity, this is a limitation of our model. Contrary to this, a patient with a high severity will visit a hospital regardless of the time of day due to their condition - this has been accurately reflected in our simulations.

Furthermore, we assume a constant rate of patient arrivals throughout each day and year, which simplifies real-world dynamics where patient flow varies by time of day and season. This approach overlooks important fluctuations, such as increased visits during holidays or flu season, noted by Zhang et al. (2022). Ignoring these variations can limit the accuracy of our findings in predicting hospital operations and resource needs, as actual patient arrivals often dictate staffing and resource allocation. Additionally, our assumption of the arrival rate and process is not aligned with reality with regards to the fact that patients may also choose to enter a waiting room depending on the number of patients currently awaiting treatment. If a patient with a low severity observes that there are a large number of patients in a waiting room, it is unlikely that, given their severity, they will also choose to join the waiting room.

When evaluating the success of a triage system based on the number of patients who leave without being seen, we use the Weibull distribution to estimate the threshold beyond which a patient will no longer choose to wait. This random variable that is assigned to patients does not take into account the severity of the patient. As a result, this does not accurately reflect a real-world scenario as patients with worse conditions are likely to be willing to stay longer in a queue to be seen than patients with a lower severity. This implies that this distribution should, perhaps, be conditioned on the patient severity.

We have also assumed that as severity increased for patients so would the average service times, however it could be argued that patients with the lowest severity often have a very accurate diagnosis of their condition and that patients with a medium severity (CTAS 3 or 4) may not be easy to diagnose. Therefore, this suggests that the patients of medium severity would take longer, on average, to complete their servicing. However this has been overlooked in our simulations by assuming a perfect triage model, where each patient is perfectly diagnosed upon entry to the triage.

6.2 Further Research and Final Remarks

Based on our empirical study, we have determined that the mixture priority system, one which weights the importance of the severity as well as total time waited of a patient, emerged as the most effective approach for optimising the triage process. This conclusion is supported by the analysis of average wait times per severity level and the incidence of patients leaving without being seen. The mixture priority system demonstrated superior performance in mitigating wait times across different severity levels compared to alternative approaches and aligned with the existing thresholds that are currently in place (Figure 1). Furthermore, our analysis revealed a notable reduction in the number of patients leaving without being seen under the mixture priority system; this occurred across severity levels - a result that was not achieved in either of the other service systems. This outcome highlights the system's ability to prioritise and allocate resources efficiently, ensuring that patients receive timely and appropriate care based on their clinical needs.

Moving forward, we recommend further research and implementation of the mixture priority system, especially in extreme scenarios. While our code to simulate extreme scenarios is included in the appendix, the lack of real-world data limited our ability to fully stress-test the mixture model without making unsubstantiated assumptions.

Overall, our findings emphasise the significance of adopting an adaptable prioritisation system that will minimise waiting time and number of patients that leave without being seen, thus successfully being able to service patients with a variety of different severities. We believe that by leveraging empirical evidence and insights gained from our study, healthcare institutions can use a mixture model to enhance their triage processes, ultimately improving patient care and satisfaction.

References

7 Appendix

7.1 Code

Find below the code for the LWBS function:

```
triage_sim <- function(n, drs, arrival_rate, priority_system, LWBS = FALSE){
  #Severity generator
  severity_gen <- function(n) {
    sample(1:5, n, replace = TRUE, prob = c(0.01, 0.062, 0.332, 0.366, 0.23))
  }

  #Patient arrivals
  arrivals <- cumsum(rexp(n, arrival_rate))
  severity_list <- severity_gen(n)
  #Attach a severity to each patient, this is the "waiting room". This is every patient that will arriv
  waiting_room <- cbind(arrivals, severity_list)

  #Generates tolerance for each patient (currently not dependent on severity but prob should be) and in
  if (LWBS){
    tolerance <- rweibull(n,1.3,300)
    leaver_matrix <- matrix(numeric(0), nrow = 0, ncol = 3)
    waiting_room <- cbind(waiting_room, tolerance)
  }

  #Initialise the function output (wait_times) and services. services[i] is the time when doctor i is a
  wait_times <- matrix(numeric(0), nrow = 0, ncol = 3)
  services <- rep(0, drs)


  #First while condition states that the minimum of the doctor's availability time has to be less than
  #If this isn't upheld then we would be assuming that after the latest arrival time no more patients e
  #Second condition just checks that the waiting room isn't empty
  while(min(services) < max(arrivals) && nrow(waiting_room) > 0) {

    #Choose the earliest available doctor
    chosen_server <- which.min(services)

    #time_of_service is the time at which the patient is taken out of the room.
    time_of_service <- services[chosen_server]

    #available_patients is the patients we can actually observe, i.e. those not in the future. If a doc
    available_patients <- waiting_room[waiting_room[, 1] <= time_of_service, , drop = FALSE]

    #Checks if anyone's tolerance has been violated. If so kick them out of the system and store them i
    if (LWBS){
      leavers <- which(available_patients[, 3] < (time_of_service - available_patients[, 1]))
      if (length(leavers) !=0 ){
        leaver_matrix <- rbind(leaver_matrix,available_patients[leavers, ])
        available_patients <- available_patients[-leavers, , drop = FALSE]
        waiting_room <- waiting_room[-leavers, , drop = FALSE]
      } }

    #If there's no patients available, automatically take the "first" patient, i.e. the one who will be
    #Equally if there's only one patient, just take them don't worry about a priority system.
    if (nrow(available_patients) == 0 | nrow(available_patients) == 1 ) {
      next_patient_index <- 1
    }
    #If there are >1 patients to make a decision between.
    else {
      severity_scores <- available_patients[, 2]
      arrival_times <- available_patients[, 1]
      current_wait_times <- time_of_service - arrival_times
      #Our inputted system will decide who the next patient should be.
      next_patient_index <- priority_system(severity_scores, current_wait_times)
    }
    #Choose the patient
    next_patient <- waiting_room[next_patient_index, , drop = FALSE]
    #Calculate how long the patient had to wait
    #Add the 0 argument in case we're jumping to the next patient in the future, in this case the secon
    wait <- max(0, time_of_service - next_patient[1])
    priority <- next_patient[2]
    #How long the patient is seen for, service time.
    service_time <- rgamma(1,6,(priority/30))
    #Time in queue, length of stay
    los <- wait + service_time
    wait_times <- rbind(wait_times, c(wait, priority,los))
    waiting_room <- waiting_room[-next_patient_index, , drop = FALSE]
    #Change the time the doctor who has served the patient to when they're next available. Just add ser
    services[chosen_server] <- max(time_of_service, next_patient[1]) + service_time

  }
  if (LWBS){
    return(list(wait_times,leaver_matrix))
  }
  return(wait_times)
}
```

Find below the code for the delay function:

```
decay_triage_sim <- function(n, drs, arrival_rate, priority_system, decay = TRUE){
  severity_gen <- function(n) {
    sample(1:5, n, replace = TRUE, prob = c(0.01, 0.062, 0.332, 0.366, 0.23))
  }

  arrivals <- cumsum(rexp(n, arrival_rate))
  severity_list <- severity_gen(n)
  waiting_room <- cbind(arrivals, severity_list)

  if(decay){
    holding_time_gen <- function(severity){
      if (severity != 1){
        decays <- seq(from = 1, to = (severity -1))
        num_decays <- length(decays)
        decays <- 0.05 / decays
        holding_times <- cumsum(rev(rexp(num_decays,decays)))

      }
      else{holding_times<-Inf}
      return(holding_times)
    }

    holding_time_list <- mapply(holding_time_gen,waiting_room[,2])
    decay_times <- mapply(function(n, m) n + m, holding_time_list, waiting_room[,1], SIMPLIFY = FALSE)
    waiting_room <- cbind(waiting_room, decay_times)
    decay_func <- function(t, severity, holding_times){

      if (length(which(holding_times < t)) != 0) { #if there is at least one decay
        num_decay <- length(which(holding_times < t))
        if (num_decay == length(holding_times)){
          holding_times = Inf
        }
        else{
          holding_times <- holding_times[(num_decay+1):length(holding_times)]
        }
        severity <- severity - num_decay
      }
      return(list(holding_times,severity))
    }

  }

  wait_times <- matrix(numeric(0), nrow = 0, ncol = 3)
  services <- rep(0, drs)

  while(min(services) < max(arrivals) && nrow(waiting_room) > 0) {


    chosen_server <- which.min(services)


    time_of_service <- services[chosen_server]


    available_patients <- waiting_room[waiting_room[, 1] <= time_of_service, , drop = FALSE]

    if (nrow(available_patients) != 0){
    severity_scores <- available_patients[, 2]
    arrival_times <- available_patients[, 1]
    holding_times <- available_patients[,ncol(available_patients)]
    current_wait_times <- time_of_service - unlist(arrival_times)
    if (decay){
    decay_output <- mapply(decay_func, rep(time_of_service,length(severity_scores)), severity_scores, h
    holding_times <- decay_output[1,]
    severity <- unlist(decay_output[2,])
    available_patients[,2] <- severity
    waiting_room[1:nrow(available_patients),2] <- severity
```

```
    waiting_room[1:nrow(available_patients),ncol(waiting_room)] <- holding_times
    }
    if (nrow(available_patients) == 1 ) {
      next_patient_index <- 1
    }

    else {
      next_patient_index <- priority_system(severity_scores, current_wait_times)
    }
    }
    else {
      next_patient_index <- 1
      }
    next_patient <- waiting_room[next_patient_index, , drop = FALSE]


    wait <- max(0, time_of_service - unlist(next_patient[1]))
    priority <- unlist(next_patient[2])

    service_time <- rgamma(1,6,0.1)

    los <- wait + service_time
    wait_times <- rbind(wait_times, c(wait, priority,los))
    waiting_room <- waiting_room[-next_patient_index, , drop = FALSE]
    services[chosen_server] <- max(time_of_service, unlist(next_patient[1])) + service_time
  }

  if(decay){
    return(list(wait_times,severity_list))
  }
  return(wait_times)
}
```

Find below the code for the plot generating functions:

```
#All simulation plots we will need for our report
plot_wait_times <- function(b, system) {
  priority_levels = 1:5
  averages = sapply(priority_levels, function(p) {
      mean(b[b[, 2] == p, 1]) }
    )
  names(averages) = priority_levels
  bp = barplot(averages, names.arg = priority_levels, xlab = "Severity", ylab = "Average Wait Time (mins
  text(bp, averages, labels = round(averages, 1), pos = 1, cex = 1, col = "white")
}
plot_wait_times_and_LOS <- function(b, system) {
  priority_levels = 1:5
  averages = sapply(priority_levels, function(p) {
      mean(b[b[, 2] == p, 1]) }
    )
  names(averages) = priority_levels

  averages_LOS = sapply(priority_levels, function(p) {
    mean(b[b[, 2] == p, 3])
  })
  par(mar = c(3, 5, 5 ,2))
  barplot(averages_LOS, col = "red", names.arg = priority_levels, xlab = "Severity", ylab = "Average Me

  bp = barplot(averages, names.arg = priority_levels, col = "blue", add = TRUE, )
  text(bp, averages, labels = round(averages, 1), pos = 1, cex = 1, col = "white")
  text(bp, averages_LOS, labels = round(averages_LOS-averages, 1), pos = 1, cex = 1, col = "white")
  text(bp, averages_LOS, labels = round(averages_LOS, 1), pos = 3, cex = 1, col = "black")

  legend("top", legend = c("Average Wait Time", "Average Service Time"), fill = c("blue", "red"))
}
plot_leavers <- function(c, system, RED = T) {
  c <- c[[2]]
  observed_freq <- sapply(1:5, function(x) sum(c[, 2] == x))
  theoretical_props <- c(0.01, 0.062, 0.332, 0.366, 0.23)
  total_observations <- length(c[,2])
  expected_counts <- theoretical_props * total_observations
  combined_counts <- rbind(observed_freq, expected_counts)
  if(RED){
  barplot(combined_counts, beside = TRUE,
          names.arg = c(1:5),
          col = c("blue", "red"),
          xlab = "Severity Level",
          ylab = "Count",
          main = paste("Leaver Frequency For", system))
  legend("topleft",
         legend = c("Leaver Frequency", "Expected Leavers"),
         fill = c("blue", "red"),
         bty = "n")
  }
  else{
    bp = barplot(observed_freq,
          names.arg = c(1:5),
          col = "blue",
          xlab = "Severity",
          ylab = "Count",
          main = paste("Leaver Frequency For", system))
      text(bp, observed_freq, labels = observed_freq, pos = 1, cex = 1, col = "white")
  legend("topleft",
         legend = c("Leaver Frequency"),
         fill = c("blue"),
         bty = "n")
  }
  text(x = 0.95, y = max(combined_counts) * 0.75,
       labels = paste("Total Leavers:", total_observations),
       adj = c(0, 0), cex = 1.2, font = 2, col = "darkgreen", family = "serif")
}
plot_decay_severities <- function(c) {
  c <- c[[1]]
  c <- c[,2]
  observed_freq <- sapply(1:5, function(x) sum(c == x))
  theoretical_props <- c(0.01, 0.062, 0.332, 0.366, 0.23)
```

```r
  total_observations <- length(c)
  expected_counts <- theoretical_props * total_observations
  combined_counts <- rbind(observed_freq, expected_counts)
  barplot(combined_counts, beside = TRUE,
          names.arg = c(1:5),
          col = c("green", "purple"),
          xlab = "Severity",
          ylab = "Count",
          main = paste("Severity Frequencies"))
  legend("topleft",
         legend = c("Severity At Service", "Severity At Entry"),
         fill = c("green", "purple"),
         bty = "n")
}
```

Unused in the final report, as we felt it was unnecessary and departed too far from reality, is the addition of an "extreme" argument to the generic function, which simulates a rare and extreme scenario happening within normal patient arrivals, with a flood of high urgency patients coming in at once:

```r
triage_sim <- function(n, drs, arrival_rate, extreme = TRUE) {
  severity_gen <- function(n) {
    sample(1:5, n, replace = TRUE, prob = c(0.01, 0.062, 0.332, 0.366, 0.23))
  }
  extreme_severity_gen <- function(n) {
    sample(1:5, n, replace = TRUE, prob = c(0.15, 0.35, 0.3, 0.16, 0.04))
  }
  extreme_events = 0
  if (extreme){extreme_events <- sum(runif(n/100 ) < 0.01) }

  priority_function <- function(severity, wait_time) {
    if (severity == 1){
      priority_score <- Inf
    }

    else{
      severity_weight <- 5
      wait_time_weight <- 10
      severity_score <- 6 - severity
      wait_time_score <- log(wait_time + 1)
      priority_score <- severity_weight * severity_score + wait_time_weight * wait_time_score
    }
    if (severity == 2){
      priority_score <- 2 * priority_score
    }
    return(priority_score)
  }


  arrivals <- cumsum(rexp(n, arrival_rate))
  severity_list <- severity_gen(n)
  waiting_room <- cbind(arrivals, severity_list)

  wait_times <- matrix(numeric(0), nrow = 0, ncol = 2)
  services <- rep(0, drs)

  if (extreme_events != 0){
    for (i in 1:extreme_events){
    event_time <- runif(1,1,n)
    print(event_time)
    extreme_arrivals <- rep(waiting_room[event_time,1], 60)
    extreme_severity_list <- extreme_severity_gen(60)
    extreme_patients <-  cbind(extreme_arrivals, extreme_severity_list)
    waiting_room_1 <- waiting_room[1:event_time,]
    waiting_room_2 <- waiting_room[(event_time+1):length(waiting_room[,1]),]
    waiting_room <- rbind (waiting_room_1, extreme_patients, waiting_room_2)
    }
  }
  while(min(services) < max(arrivals) && nrow(waiting_room) > 0) {
    chosen_server <- which.min(services)
    service_time <- services[chosen_server]
    available_patients <- waiting_room[waiting_room[, 1] <= service_time, , drop = FALSE]




    if (nrow(available_patients) == 0 ) {
      next_patient_index <- 1
    }
    else {
      severity_scores <- available_patients[, 2]
      arrival_times <- available_patients[, 1]
      current_wait_times <- service_time - arrival_times
      priority_scores <- mapply(priority_function, severity_scores, current_wait_times)
      next_patient_index <- which.max(priority_scores)
    }

    next_patient <- waiting_room[next_patient_index, , drop = FALSE]
    wait <- max(0, service_time - next_patient[1])
    priority <- next_patient[2]
    wait_times <- rbind(wait_times, c(wait, priority))
    waiting_room <- waiting_room[-next_patient_index, , drop = FALSE]
    services[chosen_server] <- max(service_time, next_patient[1]) + rgamma(1,6,(priority/30))

    if (nrow(available_patients) > 50){
      cat("Time: ", wait, "Priority: ", priority ,"\n")
    }
  }

  colnames(wait_times) <- c("Wait", "Priority")
  return(wait_times)
}
```