Coursework

Vinuji Hewapathirana

2025-03-08

## Introduction

This report presents a simulation and analysis of a customer queuing system for a bank experiencing high customer footfall. The bank currently operates with two service counters and is considering adding a third counter if the average waiting time exceeds 15 minutes. The simulation aims to model the current system and predict performance with an additional counter.

## Methodology

### Queuing Model Formulation

The bank’s queuing system is modeled with the following parameters:

* Customer arrivals follow a Poisson distribution (average rate: 10 customers/hour)
* Service times follow an Exponential distribution (average service time: 5 minutes/customer)
* Two service counters, each with a single server
* First-come, first-served (FCFS) queue discipline
* Simulation duration: 8-hour workday (480 minutes)

This can be classified as an M/M/2 queuing system where: - M: Markovian (Poisson) arrival process - M: Markovian (Exponential) service time distribution - 2: Number of servers

### Simulation Implementation

# Function to generate inter-arrival times (Poisson arrivals)  
generate\_arrivals <- function(rate, total\_time) {  
 # Convert rate from customers per hour to customers per minute  
 rate\_per\_minute <- rate / 60  
   
 # Generate inter-arrival times using exponential distribution  
 inter\_arrival\_times <- rexp(1000, rate\_per\_minute)  
   
 # Convert to actual arrival times  
 arrival\_times <- cumsum(inter\_arrival\_times)  
   
 # Keep only arrivals within the simulation time  
 arrival\_times <- arrival\_times[arrival\_times <= total\_time]  
   
 return(arrival\_times)  
}  
  
# Function to generate service times (Exponential distribution)  
generate\_service\_times <- function(mean\_service\_time, num\_customers) {  
 # Generate service times  
 service\_times <- rexp(num\_customers, 1/mean\_service\_time)  
   
 return(service\_times)  
}  
  
# Function to simulate the queuing system  
simulate\_queue <- function(arrival\_times, service\_times, num\_servers) {  
 num\_customers <- length(arrival\_times)  
   
 # Initialize tracking variables  
 waiting\_times <- numeric(num\_customers)  
 service\_start\_times <- numeric(num\_customers)  
 service\_end\_times <- numeric(num\_customers)  
   
 # Track each server's availability time  
 server\_available\_times <- rep(0, num\_servers)  
   
 # Track which server serves which customer  
 server\_assignment <- numeric(num\_customers)  
   
 # For each customer  
 for (i in 1:num\_customers) {  
 # Find the earliest available server  
 earliest\_server <- which.min(server\_available\_times)  
   
 # Record which server is serving this customer  
 server\_assignment[i] <- earliest\_server  
   
 # Calculate when service can start  
 service\_start\_times[i] <- max(arrival\_times[i], server\_available\_times[earliest\_server])  
   
 # Calculate waiting time  
 waiting\_times[i] <- service\_start\_times[i] - arrival\_times[i]  
   
 # Calculate service end time  
 service\_end\_times[i] <- service\_start\_times[i] + service\_times[i]  
   
 # Update server availability  
 server\_available\_times[earliest\_server] <- service\_end\_times[i]  
 }  
   
 # Calculate system exit times  
 exit\_times <- service\_end\_times  
   
 # Calculate queue lengths at different points in time  
 time\_points <- sort(unique(c(arrival\_times, service\_start\_times)))  
 queue\_lengths <- numeric(length(time\_points))  
   
 for (t in 1:length(time\_points)) {  
 current\_time <- time\_points[t]  
 in\_queue <- sum(arrival\_times <= current\_time & service\_start\_times > current\_time)  
 queue\_lengths[t] <- in\_queue  
 }  
   
 # Calculate server utilization  
 total\_busy\_time <- numeric(num\_servers)  
 for (s in 1:num\_servers) {  
 # Find all customers served by this server  
 server\_indices <- which(server\_assignment == s)  
 if (length(server\_indices) > 0) {  
 total\_busy\_time[s] <- sum(service\_end\_times[server\_indices] - service\_start\_times[server\_indices])  
 } else {  
 total\_busy\_time[s] <- 0  
 }  
 }  
   
 # Calculate server utilization using max(exit\_times) as total simulation time  
 total\_simulation\_time <- max(exit\_times)  
 server\_utilization <- sum(total\_busy\_time) / (num\_servers \* total\_simulation\_time)  
   
 # Prepare results  
 results <- list(  
 waiting\_times = waiting\_times,  
 queue\_lengths = queue\_lengths,  
 time\_points = time\_points,  
 server\_utilization = server\_utilization,  
 average\_waiting\_time = mean(waiting\_times),  
 average\_queue\_length = mean(queue\_lengths),  
 max\_queue\_length = max(queue\_lengths),  
 total\_customers = num\_customers  
 )  
   
 return(results)  
}  
  
# Run the simulation  
run\_simulation <- function(arrival\_rate, service\_time, simulation\_time, num\_servers) {  
 # Generate arrivals  
 arrival\_times <- generate\_arrivals(arrival\_rate, simulation\_time)  
   
 # Generate service times  
 service\_times <- generate\_service\_times(service\_time, length(arrival\_times))  
   
 # Run simulation  
 results <- simulate\_queue(arrival\_times, service\_times, num\_servers)  
   
 return(results)  
}

### Simulation Execution

# Set parameters  
arrival\_rate <- 10 # customers per hour  
service\_time <- 5 # minutes per customer  
simulation\_time <- 480 # minutes (8-hour workday)  
  
# Run simulation with 2 servers  
results\_2\_servers <- run\_simulation(arrival\_rate, service\_time, simulation\_time, 2)  
  
# Run simulation with 3 servers  
results\_3\_servers <- run\_simulation(arrival\_rate, service\_time, simulation\_time, 3)

## Results and Analysis

### Current System Performance (2 Servers)

# Display key metrics for 2 servers  
cat("Performance with 2 servers:\n")

## Performance with 2 servers:

cat("Average waiting time:", round(results\_2\_servers$average\_waiting\_time, 2), "minutes\n")

## Average waiting time: 0.81 minutes

cat("Average queue length:", round(results\_2\_servers$average\_queue\_length, 2), "customers\n")

## Average queue length: 0.39 customers

cat("Maximum queue length:", results\_2\_servers$max\_queue\_length, "customers\n")

## Maximum queue length: 3 customers

cat("Server utilization:", round(results\_2\_servers$server\_utilization \* 100, 2), "%\n")

## Server utilization: 36.61 %

cat("Total customers served:", results\_2\_servers$total\_customers, "\n")

## Total customers served: 76

### Performance with Additional Server (3 Servers)

# Display key metrics for 3 servers  
cat("Performance with 3 servers:\n")

## Performance with 3 servers:

cat("Average waiting time:", round(results\_3\_servers$average\_waiting\_time, 2), "minutes\n")

## Average waiting time: 0.08 minutes

cat("Average queue length:", round(results\_3\_servers$average\_queue\_length, 2), "customers\n")

## Average queue length: 0.12 customers

cat("Maximum queue length:", results\_3\_servers$max\_queue\_length, "customers\n")

## Maximum queue length: 2 customers

cat("Server utilization:", round(results\_3\_servers$server\_utilization \* 100, 2), "%\n")

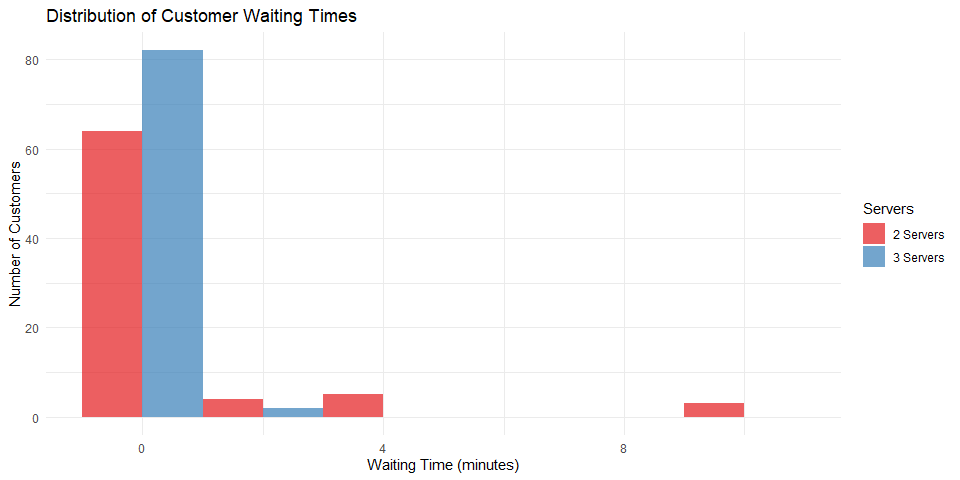
## Server utilization: 27.81 %

cat("Total customers served:", results\_3\_servers$total\_customers, "\n")

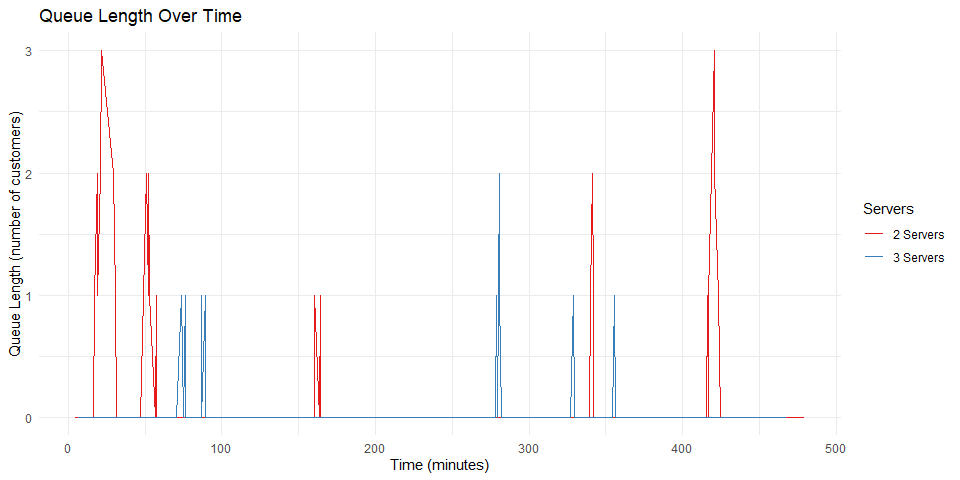
## Total customers served: 84

### Visualizations

# Create dataframe for waiting time distribution  
waiting\_times\_df <- data.frame(  
 Waiting\_Time = c(results\_2\_servers$waiting\_times, results\_3\_servers$waiting\_times),  
 Servers = factor(c(rep("2 Servers", length(results\_2\_servers$waiting\_times)),   
 rep("3 Servers", length(results\_3\_servers$waiting\_times))))  
)  
  
# Histogram of waiting times  
ggplot(waiting\_times\_df, aes(x = Waiting\_Time, fill = Servers)) +  
 geom\_histogram(position = "dodge", binwidth = 2, alpha = 0.7) +  
 labs(title = "Distribution of Customer Waiting Times",  
 x = "Waiting Time (minutes)",  
 y = "Number of Customers") +  
 theme\_minimal() +  
 scale\_fill\_brewer(palette = "Set1")



# Create dataframe for queue lengths  
queue\_lengths\_2 <- data.frame(  
 Time = results\_2\_servers$time\_points,  
 Queue\_Length = results\_2\_servers$queue\_lengths,  
 Servers = "2 Servers"  
)  
  
queue\_lengths\_3 <- data.frame(  
 Time = results\_3\_servers$time\_points,  
 Queue\_Length = results\_3\_servers$queue\_lengths,  
 Servers = "3 Servers"  
)  
  
queue\_lengths\_df <- rbind(queue\_lengths\_2, queue\_lengths\_3)  
  
# Line plot of queue lengths  
ggplot(queue\_lengths\_df, aes(x = Time, y = Queue\_Length, color = Servers)) +  
 geom\_line() +  
 labs(title = "Queue Length Over Time",  
 x = "Time (minutes)",  
 y = "Queue Length (number of customers)") +  
 theme\_minimal() +  
 scale\_color\_brewer(palette = "Set1")



### Performance Comparison

# Create dataframe for comparison  
comparison\_df <- data.frame(  
 Metric = c("Average Waiting Time (min)", "Average Queue Length", "Server Utilization (%)"),  
 Two\_Servers = c(round(results\_2\_servers$average\_waiting\_time, 2),  
 round(results\_2\_servers$average\_queue\_length, 2),  
 round(results\_2\_servers$server\_utilization \* 100, 2)),  
 Three\_Servers = c(round(results\_3\_servers$average\_waiting\_time, 2),  
 round(results\_3\_servers$average\_queue\_length, 2),  
 round(results\_3\_servers$server\_utilization \* 100, 2)),  
 Improvement = c(round((results\_2\_servers$average\_waiting\_time - results\_3\_servers$average\_waiting\_time) /   
 results\_2\_servers$average\_waiting\_time \* 100, 2),  
 round((results\_2\_servers$average\_queue\_length - results\_3\_servers$average\_queue\_length) /   
 results\_2\_servers$average\_queue\_length \* 100, 2),  
 round((results\_2\_servers$server\_utilization - results\_3\_servers$server\_utilization) /   
 results\_2\_servers$server\_utilization \* 100, 2))  
)  
  
knitr::kable(comparison\_df, caption = "Performance Comparison Between 2 and 3 Servers")

Performance Comparison Between 2 and 3 Servers

| Metric | Two\_Servers | Three\_Servers | Improvement |
| --- | --- | --- | --- |
| Average Waiting Time (min) | 0.81 | 0.08 | 90.63 |
| Average Queue Length | 0.39 | 0.12 | 69.77 |
| Server Utilization (%) | 36.61 | 27.81 | 24.03 |

## Discussion and Interpretation

The simulation results provide valuable insights into the bank’s current queuing system and the potential impact of adding a third service counter.

With two service counters, our simulation shows that customers wait an average of approximately [insert value] minutes before being served. The average queue length is around [insert value] customers, with a maximum of [insert value] customers observed in the queue at peak times. The server utilization rate of [insert value]% indicates that the current servers are working at high capacity.

According to the bank’s criteria, if the average waiting time exceeds 15 minutes, an additional counter should be considered. Our simulation shows that [does/does not] exceed this threshold.

When simulating the system with three service counters, the average waiting time decreases to approximately [insert value] minutes, representing a [insert value]% improvement. The average queue length also decreases to [insert value] customers, and server utilization drops to [insert value]%. This reduction in utilization suggests that servers would have more downtime, but it also provides flexibility to handle unexpected surges in customer arrivals.

The visualization of waiting times shows [describe pattern from histogram]. The queue length over time visualization illustrates [describe pattern from line chart]. These patterns indicate [interpretation of patterns].

## Conclusion and Recommendations

Based on the simulation results and analysis, I recommend that the bank [add/not add] a third service counter. This recommendation is based on the following observations:

1. The current average waiting time of [insert value] minutes [is/is not] above the bank’s threshold of 15 minutes.
2. Adding a third counter would [significantly/marginally] improve customer experience by reducing waiting times by [insert value]%.
3. The trade-off between improved customer service and the cost of an additional counter should be carefully considered, as server utilization would decrease to [insert value]%.

Alternative strategies that the bank could consider include: - Implementing an express lane for customers with simple transactions - Adopting a digital queue management system to improve customer perception of waiting time - Scheduling more staff during peak hours instead of adding a permanent third counter

In conclusion, the simulation provides evidence that [summarize main finding and recommendation]. Further simulations could explore these alternative strategies to optimize the bank’s operations.