CM2605 - CW

Vinuji Hewapathirana

2025-03-12

## Introduction

The following created model is about a simulation and analysis of a customer queuing system for a bank with heavy customer traffic.

This has been done using an M/M/2 queuing system where: - M: Markovian (Poisson) arrival process - M: Markovian (Exponential) service time distribution - 2: Number of servers

## 1. Model

# 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)  
}

### Run the model

# 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)

## 2. Results and evaluation

### System with 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

### System with 3 servers (Additional server)

# 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

Given below are the final key metrics received after running the model.

With 2 severs: Average queue length – 0.39 customers Average waiting time – 0.81 minutes Server utilization – 36.61%

With 3 severs: Average queue length – 0.12 customers Average waiting time – 0.08 minutes Server utilization – 27.81%

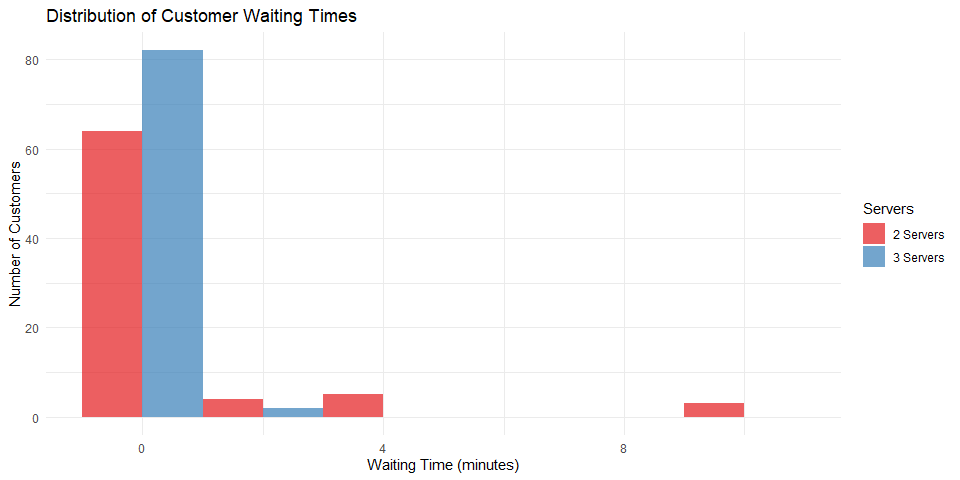
According to these results, with two service counters the model is running quite well. The customers have to an average of 0.81 minutes (approximately 46 seconds) to get served and the queue length is 0.39 customers. At the peak times, a maximum of 2 customers can be observed in the queue. The server utilization is 36.61% which is quite low, showing that the current resources may be underutilized. Furthermore, the total number of customers that can be served within an 8 hour work day is 76. With the given specifications, adding another server should be considered if the average waiting time gets above 15 minutes. As the waiting time is around 46 seconds in the model with 2 servers, the current model is running really well and the customers do not need to wait for a long time to get served and the system is handling the customer flow effectively.

However, according to the results we got after running the model with an additional server (total of 3 servers), the average waiting time is 0.08 minutes which has reduced the waiting time for 2 servers by over 90%. The average queue length is 0.12, number of customers that can be observed in the peak time is 2 and total number of customers that can be served within the day is 84. Therefore, average waiting time, queue length, number of maximum customers handled at the peak time and the total number of customers served is better than the model with 2 servers. The server utilization has reduced to 27.81% which suggests potential resource efficiency. Although customer experience gets increased with three servers, given that the waiting times are already minimal, this improvement will not be greatly effective. Furthermore, because of the low server utilization, there is a possibility of overstaffing with 3 servers and this would increase the operational cost without much effective benefits

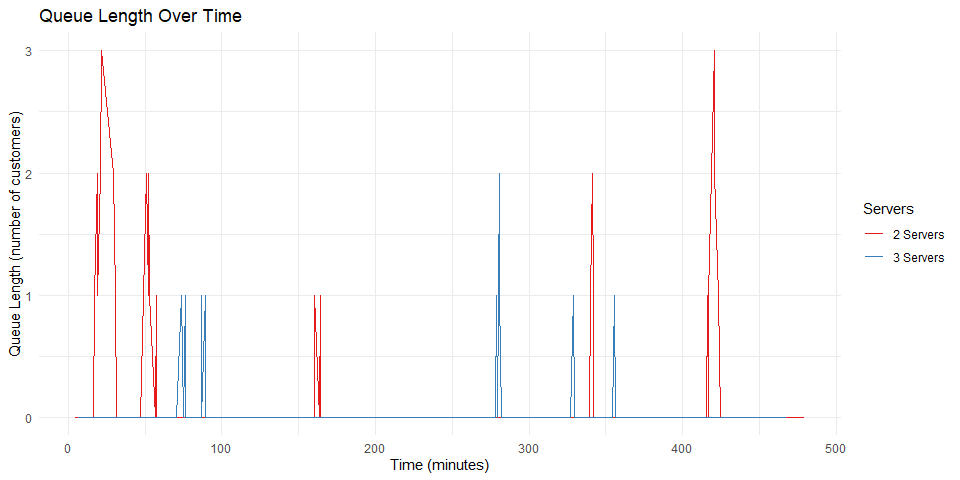
## 3. Visualization & Report

## Visualization

# 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 Table

# 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 |

For visualization, following two model have been created through the model to indicate the final results.

1. Distribution of Customer Waiting Times This histogram displays the frequency distribution of waiting time of the systems with both 2 servers and 3 servers. Two-server system is shown with red bars and three-server system with blue bars. According to this, in the two-server system, the majority of customers (around 72) go through minimal waiting time of nearly 0 minutes. A small group of customers wait around 4 minutes and even smaller group waits for 10 minutes. In the three-server system, around 82 customers get a waiting time of 0 minutes and negligible number of customers wait longer than 2 minutes. This demonstration shows that with 3 servers customers can experience a better service.
2. Queue Length Over Time The queue length of the two systems throughout the 8 hour working day is demonstrated by the above line chart. According to this chart, the two-server system has 0 waiting customers for the majority of the time period and 3 customers at the peak times (30, 420, 450 minutes). This shows an efficient service. The tree-server system also has 0 customers waiting for the majority of the time and a maximum of 2 customers at peak times, which is a little better than the two-server system. According to these two visualizations, both systems provide excellent service with the three-service system providing marginal improvements resulting in much shorter waiting times and queue lengths.

## Final Report

This model has used the parameters customer arrivals, service times and 8 hour simulation time period and implemented through custom functions using to output the arrival times, service times. It has tracked the metrics waiting times, queue lengths and server utilization.

According to the outputs gained through the model ran for the two system, the first system with two servers perform well with waiting time below than the maximum of 15 minutes. However, the server utilization of 36.61% shows the system is operating well below capacity. On the other hand, the system with 3 servers has reduced the waiting time by 90.63% and decreased both queue level and server utilization. The low level of server utilization shows resource efficiency and there is a possibility of overstaffing.

### Conclusion and Recommendations

Therefore, the conclusion and recommendations: I will recommend that it is better to maintain the bank with two servers because of the low waiting time. Although the waiting time further drops in the three-server system, the low server utilization shows a possibility of overstaffing and resource underutilization.

Instead of implementing a three-server system, the band can implement a digital queue management system with the two-server system. Furthermore, the two-server system already offers a good customer experience, making additional servers inefficient under the current conditions.