NHH

BAN403 - Project 1

Spring Semester 2024

Candidates: 69, 86, 134

March 4, 2024 - March 11, 2024

Table of Contents

1 Monte Carlo Simulation **3**

2 Queuing Theory **3**

3 General Analytics Model **4**

3.1 Steady-state probabilities 4

3.2 Average number of customers 5

3.3 Is the system stable? 5

3.4 Simulation model with JaamSim 5

3.5 Normal distribution 6

4 Airline Ticket Counter **6**

4.1 Multiple lines 6

4.2 Single line queue 6

4.3 Compare the systems (a) and (b) 7

4.4 Own queue for ticket buying costumers 7

1 Monte Carlo Simulation

The objective is to maximize the return relative to risk, and we will use the Sharpe Ratio as our target indicator: $\frac{R_p - R_f}{\sigma_p}$, where R_p is the expected return of the portfolio, R_f is the risk-free rate, and σ_p is the standard deviation of the portfolio (Curry, 2024). We decided to collect data for the following tickers: KO, GME, AAPL, JNJ, & JPM. For data collection, we downloaded data from Yahoo Finance through Python (see *task1.ipynb*). After downloading and calculating return and variance per asset, we used a Monte Carlo approach to simulate different portfolio strategies. In our case, the Monte Carlo method involves generating a large number of random portfolio compositions, allowing us to do a number of trials and assess different compositions. Based on each portfolio composition, we calculated the expected return and standard deviation for calculating the Sharpe Ratio. For the R_f, we assumed 2,5%, which is close to the 5 year average of the US 10 year treasury (CNBC, 2024). Based on our simulations, we got the following best portfolio:

Ticker	KO	GME	AAPL	JNJ	JPM
w_i	15.2%	4.3%	59.3%	10.6%	10.6%

Table 1: Optimal portfolio weights based on the Sharpe Ratio maximization.

with an expected return of 220%. It's worth mentioning that this is based on our last run, and the optimal portfolio will change a bit if the model is re-ran. This is due to randomness in the way the different weight gets generated.

2 Queuing Theory

This case is a $M/M/2/\infty/3$ – system, with three machines = N, and two service teams = c. Since the calling population is always $N - n$, where N is the calling population outside the system and n the number numbers of machines in the system, we designed a loop-like simulation process, where three machine entities are introduced to the system, and then move along the loop path, depending on their state. A machine can have three states; "working" = the machine is functional, "ServiceQue" = machine is broken-down and is in the service queue, and "Faulty" = machine is broken-down and being serviced by one of the service teams. Both service team servers use the same exponential distribution with a mean service time of 4 hours. The mean time a machine spends in the entity processor as a functioning machine is 8 hours, before it is sent to the service que and assigned a faulty state. To keep track of the time the whole system has n number of faulty machines in the service pipeline, we use an assign object to track the number of faulty machines over time, which can be seen as probabilities of state (N-n). When running the simulation 8760 hours (one year), we achieve the following results shown in table 2. The results of the simulation are somewhat diverging from the analytical approach. However, if the simulation period is set to longer periods i.e., 8 760 000 hours (1000 years), we see that the numbers converge to the values of the analytical model. This indicates that our initial difference is due to variance in the distributions. In our model setup we have to manually calculate the mean downtime of a broken machine in the system. We did this by calculating the average time it took to complete a service in both teams. We then took the average of the values from both teams and added the average time

P_0	$P_1 1$	P_2	P_3
29.17%	44.87%	20.94%	5.00%

Statistics	Value	Unit
Mean jobs in the queue	0.05	faulty machines
Mean service jobs in the system	1.02	faulty machines
Mean time spent in service queue	0.22	hours
Mean downtime	4.10	hours

Table 2: base case simulation

a machine spent in the service que, giving us an approximation for mean downtime for broken machine. This ignores partly finished jobs at the end of the simulation and might be a suboptimal approach introducing some bias to our statistics. However, we found it to be a close enough approximation given our simulation design. We then revised our model using different distributions for our interarrival times, to verify the findings presented by Bunday and Scraton (1980) that all distributions would have the same results given the same mean interarrival rate stayed $\lambda = 0.125$.

We ran three scenarios with additional distributions; normal, gamma, and uniform, with the interarrival-distributions being the only thing changing from the base-case. The normal distribution was truncated, which will have an effect on the variability of the interarrival times. The uniform distribution used a min value of 0, and a max value of 8. We gave the gamma distribution the parameter 3, giving it a more symmetric shape than the exponential distribution. The results from the revised runs are presented in Table 2, with all results rounded off with two decimal places, as done in the book. Again, we also did long runs, h = 8 760 000(1000 years), verifying that the numbers converge. All results can be found in the Excel-file "Simulation results task 2.xlsx".

Distribution	P_0	P_1	P_2	P_3
Exponential	29.18%	44.87%	20.95%	5.01%
Normal	29.64%	43.94%	21.27%	5.15%
Gamma	30.11%	42.85%	22.11%	4.94%
Uniform	29.16%	44.50%	21.77%	4.57%
Statistic	**Exponential**	**Normal**	**Gamma**	**Uniform**
Mean jobs in queue	0.05	0.05	0.05	0.05
Mean service jobs	1.02	1.02	1.02	1.02
Mean service time	0.20	0.21	0.20	0.18
Mean downtime	4.11	4.13	4.12	4.10

Table 3: Combined Simulation Results

Our results imply that the statement regarding the distributions for interarrival times in the task description is correct. If the simulation sample is sufficiently large, the observed values for P_0, P_n, L, L_q, W and W_q converges towards the analytical model values in the M/M/2/∞/3-model. We see that the analytical values are dependant on the λ value; as long as this, i.e. the mean of the interarrival times stay consistent, the assumption is correct.

3 General Analytics Model

3.1 Steady-state probabilities

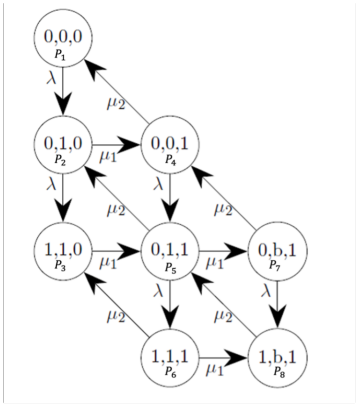

Figure 1: Takeaway states

Based on the transition diagram of the drive-through we can set up the following equations:

$$\lambda P_1 = \mu_2 P_4 \tag{1}$$

$$\lambda P_2 + \mu_1 P_2 = \lambda P_1 + \mu_2 P_5 \tag{2}$$

$$\mu_1 P_3 = \lambda P_2 + \mu_2 P_6 \tag{3}$$

$$\lambda P_4 + \mu_2 P_4 = \mu_1 P_2 + \mu_2 P_7 \tag{4}$$

$$\lambda P_5 + \mu_2 P_5 + \mu_1 P_5 = \mu_1 P_5 + \lambda P_1 + \mu_2 P_8 \tag{5}$$

$$\mu_1 P_6 + \mu_2 P_6 = \lambda P_5 \tag{6}$$

$$\lambda P_7 + \mu_2 P_7 = \mu_1 P_5 \tag{7}$$

$$\mu_2 P_8 = \mu_1 P_6 + \lambda P_7 \tag{8}$$

$$P_1 + P_2 + P_3 + P_4 + P_5 + P_6 + P_7 + P_8 = 1 \tag{9}$$

This gives us the following probabilities (solved with Python task3.ipynb):

P1	P2	P3	P4	P5	P6	P7	P8
4.67%	10.65%	32.47%	5.84%	15.45%	11.04%	5.15%	14.72%

Table 4: Probabilities solved with Python task3.ipynb

3.2 Average number of customers

To answer this question, we needed to extract some numbers from the tranisation diagram above. We needed the number of clients in each state, and the average number of clients arriving in each state The probabilities, number of clients (n), and arrival rates (λ) are given in the table below:

P_i	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8
$p(P_i)$	4.67%	10.65%	32.47%	5.84%	15.45%	11.04%	5.15%	14.72%
n	0	1	2	1	2	3	2	3
λ	50	50	0	50	50	0	50	0

Table 5: State probabilities, number of clients, and arrival rates (transposed)

For calculating the average number of clients, we multiply the number of clients in each state with the probability of being in each state.

$$L = \sum_{i=1}^{8} n_i P_i \rightarrow 1.9991 \approx 2$$

The second number we are going to calculate, the average rate of clients entering the drive-through, is calculated by multiplying the sum of arrival rates with the probability of the system being in the state.

$$\bar{\lambda} = \sum_{i=1}^{8} P_i \lambda_i \rightarrow 20.88$$

Finally, we can use Little's Law to calculate the average time a client spends in the system.

$$W = \frac{L}{\bar{\lambda}} \rightarrow \frac{2}{20.88} = 0.0957\,\text{h} = 5.74\,\text{min}$$

3.3 Is the system stable?

To decide whether the system is stable or not, we have to calculate the process flow rate: $p = \frac{\lambda}{\mu}$. If p is greater than 1, then the system is considered unstable. In our context, it's the μ_1 that is the bottleneck, and thus its value we put inside the formula: $p = \frac{50}{30} = 1.66 \Rightarrow p > 1 =$ unstable system. But since this is a closed system due to the cars having to leave, the queue can never exceed 1, and because of this, the system is considered stable (Laguna and Marklund, 2018).

3.4 Simulation model with JaamSim

We decided to run the model for 1,000 hours, and from the CustomerServerStatistics, we can see that 20,841 customers have been through our system. This gives us a λ of $\frac{20,841}{1,000\text{ hours}} = 20.841$. Furthermore, we can see that the Sample Average is 5.747 minutes, and with Little's Law, we can calculate the average number of customers in our system: $L = W \times \bar{\lambda} = \frac{5.747\text{ min}}{60} \times 20.841 = 1.9962$.

From the above output, we can see that the calculations and JaamSim output are quite close to each other, and the small differences could be explained by natural variance in the use of the exponential distribution.

Parameter	JaamSim Output	Calculated Value
Average number of clients (L)	1.9991	1.9962
Average arrival rate ($\bar{\lambda}$)	20.88	20.841
Cycle time (W)	5.74 min	5.747 min

Table 6: Simulation of takeaway-queue

3.5 Normal distribution

There are two things to mention when changing to a normal distribution. The first thing is that we don't have any estimates of the standard deviation. This will affect the result we will get out of the model. The second thing is that the normal distribution can't return a negative number, and because of this, we have to truncate the x-axis, which will affect the variability. If we assume the following distribution: mean = 1.5 min, min = 0.001 min, max = 3 min. With these assumptions, we get the following results. The output is quite similar too the previous model, but has a slightly better Cycle time, which means that the Drive-through could server more people per hour than before. The reason for this has to do with the extreme values in a exponential distribution, who could end up blocking the system more than in a normal distribution with truncated x-axis.

Parameter	Value
Average number of clients (L)	2.01
Average arrival rate ($\bar{\lambda}$)	22.6810
Cycle time (W)	5.31 min

Table 7: Normal distribution results

4 Airline Ticket Counter

4.1 Multiple lines

The first task was to create a model of the airline ticket counter in JaamSim. To start we performed a stable system calculation for one and two servers: $\frac{0.2}{0.167} \approx 1,2$, $\frac{0.2}{2*0.167} \approx 0.6$. Since the steady-state probability exceeds 1, the system is considered unstable with one server, and thus the queue will grow indefinitely. Based on this, we implemented a model with two queues and two servers. We then modified the model with only one queue and one server, which confirmed that this would be an unstable system. After implementing the model in JaamSim, we tracked the entity statistics and the average waiting time, which is shown in table 8.

N servers	Average Queue Time
One Server	44,428.16 min
Two Server	2.2843 min

Table 8: Multiple queue system

From table 8 we can also see that a system of two servers are stable, while one server is not. Based on this we can conclude that the minimum number of agents required for an average waiting time of less than 5 minutes is two.

4.2 Single line queue

The only change in this task is that all customers are now waiting in the same queue, so that the first customer in the queue gets sent to the first available server. Since the interarrival rate and service time remain unchanged from task A, we expect the number of servers required to achieve a stable system to be the same. Implementing the model in JaamSim confirmed this, results are presented in table 9.

N servers	Average Queue Time
One Server	44 483.15 min
Two Server	2.02 min

Table 9: One queue system

4.3 Compare the systems (a) and (b)

Due to the exponentially distributed arrival rate of customers, achieving a maximum wait time is theoretically impossible. This is because an exponentially distributed arrival rate always will have a small probability of exceeding the capacity of our system, causing a queue to build up. However, in a practical application, like a simulation, the probability becomes lower and lower for each server added until the probability becomes infinitely small, thus making a maximum wait time practically applicable. For our simulation, we have have chosen a simulation period of a year, which gave us the following maximum queue times seen in table 10. It is however important to note that this is the result of the exponential distribution not achieving an inter-arrival rate high enough to create a temporary queue block during our simulation. If the simulation was ran towards ∞ the maximum wait time would always surpass 5 minutes.

n servers	One queue	Multiple Queues
5 Servers	6.56 min	9.56 min
6 Servers	3.81 min	9.25 min
7 Servers	2.94 min	7.32 min
8 Servers	2.45 min	3.19 min

Table 10: Max queue time

However, from our simulation we can see that there is a quite large difference between the two queue designs. To understand why we can imagine a simulation with 6 servers where we assume a state at time (t). At this time t we go from n = 0 number of customers in the system at time (t-1) to n = 7. The 7th person must now wait in a queue. Because the service time is uniformly distributed between 2 and 10, the probability of a server using more than 5 minutes is $\frac{5}{8} = 62,5\%$. In the one-queue system, where the customer can stand and wait for the first available ticket agent, the probability of a queue time > 5 minutes becomes $0,625^6 \approx 6\%$. However, in the multiple queue system, the customer must "gamble" on a queue and therefore has a 62,5% probability of a queue time > 5 minutes. As we can see, the difference in servers needed for maximum queue time in our simulation is caused by the more optimized structure of the "one-queue system"; In this system a customer can never choose the "wrong" queue, and resources are better utilized as a result. This is also shown when we take a look at the average queue time from task a) and b) where the one-queue system outperforms the multiple-queue system.

4.4 Own queue for ticket buying costumers

For this task we assumed the following: A client that needed a ticket, also got checked in at the same agent. The next was that the objective of the task was to determine the minimum number of agents/servers in both counters, such that both had a mean queue time of less than 5 min, and not that the average queue time for the whole system was less than 5 min. From task4_4.ipynb we have calculated the expected queue time for both the ticket counter and the check-in counter with only one staff at each counter. From the calculation we can see that the check-in counter is expecting a average queue time of 1.6, while ticket counter is expecting a average queue time of 6.2. This indicates that the ticket counter would need two staffs. After implementing a model with only one staff at each counter, and a model with 1 staff at the check-in and 2 staffs at the ticker, we got the following results, who supported our initial calculations. The average queue time is shorter compared to tasks A and B, improving customer experience but requiring an additional staff member, increasing the total to three.

n server	Average Queue time Check In	Average Queue time Ticket
1 Check-in- and 1 ticket agent	1.62 min	6.33 min
1 Check-in- and 2 ticket agents	1.6 min	0.51 min

Table 11: Split ticket an non-ticket queue

References

Bunday, B., & Scraton, R. (1980). The g/m/r machine interference model. *European Journal of Operational Research*, *4*(6), 399–402. https://doi.org/https://doi.org/10.1016/0377-2217(80)90192-7

CNBC. (2024). *U.s. 10 year treasury*. 2024-03-05

Curry, R. B. B. (2024). *Understanding the sharpe ratio*. https://www.forbes.com/advisor/investing/sharpe-ratio/?fbclid=IwAR3_hZk38Wbmp2rNEJzZTdFGd0911MB2gSzd6ns5fbcQRJL62iv-6pjyIm0

Laguna, M., & Marklund, J. (2018). *Business process modeling, simulation and design*. Chapman; Hall/CRC.