Project 1

BAN403 – Simulation of Business Processes

Candidates: 11, 27, 44

Contents

1.1. Montecarlo Simulation – Maestranza Hotel

We built a Monte Carlo simulation model in Excel to calculate the hotel's profit based on guest arrivals and room occupancy. If arrivals exceeded capacity, all rooms were filled; otherwise, the number of occupied rooms matched the number of guests who showed up.

Testing different values for the overbooking limit quickly showed that the best option was either three or four. However, with only 4,000 trials in the original simulation, it was difficult to determine which one was better. To improve accuracy, we applied the Law of Large Numbers, stating more stable results as the number of trials increases and raised the number of trials to 70,000 (Laguna & Marklund, 2018). This consistently resulted in slightly higher profits when the overbooking limit was set to four. Therefore, we concluded that the optimal overbooking limit for Maestranza Hotel is four.

Limit	Run # 1	Run # 2	Run # 3	Run # 4	Run # 5	Average
3	€10,570.87	€10,572.69	€10,570.81	€10,571.76	€10,572.00	€10,571.63
4	€10,573.85	€10,574.06	€10,573.70	€10,573.37	€10,573.68	**€10,573.73**

Table 1 – Simulated Profits with the Overbooking Limits Set to 3 and 4.

1.2. Montecarlo Simulation – Winter Park

a) To simulate the expected profit, we ran a Monte Carlo simulation with 1,000 demand values, assuming demand followed a Poisson distribution with a mean of 80. For each simulation and order amount, we calculated the total profit using the following formula (additional formulas can be found in Appendix A):

$$Total\ profit = Revenue\ in\ Ski\ Season + Revenue\ at\ the\ End\ of\ the\ Season - Costs\ of\ Ski\ Jackets$$

We applied these calculations to all 1,000 simulated demand values, testing order amounts between 40 and 120 units. This range was chosen because ordering less than 40 would always meet demand but result in lower revenue, while ordering more than 120 would increase losses due to excess inventory sold at a lower price after the ski season. To determine the expected profit, we averaged the profits across all 1,000 simulations for each order amount. The results showed that ordering **90** jackets yields the highest expected profit, which in this simulation amounts to **$12,286**.

	Order Amount								
	40	**50**	**60**	**70**	**80**	**90**	**100**	**110**	**120**
Avg Profit	6360	7950	9536	11036	12030	12286	12104	11810	11500

Table 2 - Expected Profit (in USD) for Different Ski Jacket Order Amounts

b) To estimate the probability of earning at least $12,000 in profit, we used Excel's "COUNTIF" function to count how many times the simulation resulted in a profit of $12,000 or more. We then divided this count by the total number of simulation trials (1,000). Based on this method, the probability of reaching at least $12,000 in profit with the optimal order quantity of 90 jackets was found to be **58.8%**.

2.1. Queuing Theory – Analytical Model

The manufacturing system follows an $M/M/c/\infty/N$ model with a single queue, c independent, parallel, and identical repairmen (servers), exponentially distributed interarrival (machine failures) and service times (repair duration), a finite queue length (at most $N - c$), a FIFO queuing discipline (machines wait in a FIFO queue if all repairmen are busy), and a finite calling population of N machines. The value of N can be adjusted in the "att-task2_analytical" Excel file. (Laguna & Marklund, 2018).

Calculating P_0 for the different values $c = 1,\ c = 2,\ c = 3,\ c = 4$ and $c = 5$, yields:

c	1	2	3	4	5
P_0	0.199067	0.314932	0.326855	0.327654	0.32768

Table 3 - Probability of Zero Jobs ($P0$) for Different Values of c

We further calculated the different P_n values for each c, the mean number of jobs L_q in the queue and the mean number of jobs in the system L (see Appendix B1).

To calculate the expected average cost per hour after obtaining L for all c, we sum downtime costs $(50\ USD/h)$ and repairmen employment costs $(c * 10\ USD)$ for all c:

$$Total\ Cost\ (in\ USD) = (L * 50) + (c * 10)$$

c	1	2	3	4	5
$Avg.Cost$	99.81337	**74.70552**	80.39722	90.016	100

Table 4 – Expected Average Cost per Hour for Different Values of c

Thus, hiring two repairmen yields the smallest expected average cost of $\approx \mathbf{74.71\ USD}$ per hour. We further calculated the average arrival rate $\bar{\lambda}$ for each c, the mean time a job spends in the system W, and the mean time a job spends in the queue W_q (see Appendix B1).

2.2. Queuing Theory – JaamSim I

Our *JaamSim* model represents a manufacturing system with $N = 5$ machines, where each machine operates until it randomly breaks down based on an exponential distribution with a mean uptime of 8 hours. When a machine fails, it enters a FIFO repair queue labeled as "broken" and waits for an available repairman, which it gets assigned from the resource pool. The repair process happens at a service station with capacity c, where each repair follows an exponential distribution with a mean service time of 2 hours. After being repaired at the service station, machines are assigned the state "operating" in *Operational* and return to normal function. The initial state of all machines is "operating", and both N (machines) and c (repairmen) can be manually adjusted to explore the different scenarios.

In our simulation, the estimated lowest cost per hour is $\mathbf{\$74.56}$ for $\boldsymbol{c = 2}$, which is very close to the $\mathbf{\$74.71}$ calculated in Task 2.1., where $c = 2$ also gave the lowest expected cost. Other values for the model can be found in Appendix B2. The slight difference is expected because the simulation is based on randomized events, while the analytical model gives an exact theoretical value. If we simulated for even longer

than 10,000 hours, the results would likely converge even closer to the analytical solution due to the law of large numbers (Laguna & Marklund, 2018).

c	1	2	3	4	5
$Avg.Cost$	98.896	**74.555**	80.516	90.076	100.074

Table 5 – Simulated Average Cost per Hour for Different c

2.3. Queuing Theory – JaamSim II

When adjusting the machine breakdown time to follow a normal distribution with a mean of eight hours and a standard deviation of three hours, we observe only small changes in the expected costs. The most cost-efficient policy remains employing two workers. The expected cost for $c = 2$ increases slightly from $\$74.555/h$ to $\$74.820/h$, which is due to a slight rise in L from 1.091 to 1.096 (see Appendix B3). Our results align with the remark that the $M/M/c/\infty/N$ model can be extended to cases where the time spent outside the system follows a different probability distribution if the mean remains the same (Bunday & Scraton, 1980; Laguna & Marklund, 2018). While switching from an exponential to a normal distribution model and keeping the mean at eight hours, we observe only minor differences in costs.

c	1	2	3	4	5
$Avg.Cost$	99.466	**74.820**	80.477	90.049	100.015

Table 6 – Simulated Average Cost per Hour with Normal Distribution

3.1. A Simple Network

The system consists of two $M/M/1$ queues – one for the microphone and one for the window. Together, they form an $M/M/1$ tandem queue. Queues and interarrival times are independent and identically distributed, following an exponential distribution. Further details can be found in Appendix C1.

Our JaamSim model includes an entity generator that creates customers following an exponential distribution with a mean of 15 minutes. The model includes two servers with their queues, representing the microphone and the window, with exponential service times of seven and nine minutes, respectively. Further details on variable measurements can be found in Appendix C1.

The delay in the window queue is longer than in the microphone queue. Additionally, the average number of customers is higher in the window queue, and the window station has a higher utilization rate. This indicates that the window acts as the main bottleneck of the system, which is expected due to its longer average service time. The total average waiting time per customer in both queues, W_q, is $6.16 + 13.40 = 19.56\,minutes$, while the total average number of customers in the queues, L_q, is $0.41 + 0.90 = 1.31$. Furthermore, cycle time (CT) and work in process (WIP) were included for comparison with future models, where CT represents the total time for a job to complete the process, and WIP includes all jobs within the process boundaries (Laguna & Marklund, 2018).

	Delay in Queue	Customers in Queue	Utilization	Cycle Time	WIP
Microphone	6.16	0.41	46.85 %	35.574	2.377
Window	13.40	0.90	60.19 %		

Table 7 – Queue Performance Metrics for the Simple Network Model – Task 1

3.2. A Simple Network

Removing the queue at the window transforms the system into an $M/M/1/1$ system, where $a4$ represents the maximum number of jobs allowed in the system (Laguna & Marklund, 2018). Since there is no queue, the only job permitted is the one currently being processed by the server. The microphone system remains unchanged as an $M/M/1$ system, allowing an unlimited number of jobs. We adjusted the model from Task 1 accordingly to implement blocking by adding an expression threshold. Further descriptions of the simulation can be found in Appendix C2.

The most noticeable effect of implementing blocking is that the number of customers and delay in the window queue drop to zero, confirming that the mechanism works as expected. However, removing the window queue leads to a significant increase in both delay and queue length at the microphone. As a result, the expected waiting time in the queue (W_q) increases to $36.14\ minutes$ per order, compared to the previous $19.56\ minutes$. Similarly, the average number of customers in the queue (L_q) rises to $2.42\ per\ hour$. We notice a 20-minute increase in cycle time when the window queue is removed. Further, the WIP increases, showing that more customers are in the drive-through simultaneously.

The microphone and window utilization remains largely unchanged, as removing the queue at the window doesn't impact the total number of customers served. However, blocking increases queue lengths since customers must wait at the microphone for space at the window. If the arrival rate increases, blocking occurs more, leading to idle time at the microphone while the window's utilization stays constant since it serves the same number of customers per hour.

	Delay in Queue	Customers in Queue	Utilization	Cycle Time	WIP
Microphone	36.14	2.42	46.85 %	56.433	3.485
Window	0.00	0.00	60.19 %		

Table 8 – Queue Performance Metrics for the Simple Network Model – Task 2

3.3. A Simple Network

In a stable process, the average inflow rate equals the average outflow rate (Laguna & Marklund, 2018, p.152/153). To calculate the average inflow rate, we divided the customers added in the entity generator with the simulation time. The average outflow rate was calculated similarly by dividing the number of customers entering the entity sink by the simulation time. In both systems above, $R_i = R_o = 4.01/h$. We can, therefore, conclude that the systems in tasks 3.1 and 3.2 are stable.

3.4. A Simple Network

In the baseline scenario, with four clients arriving per hour, the system demonstrated stability, with an average cycle time of 56m29s. However, the system showed significant instability as the client arrival rate increased to five per hour. The cycle time surged to more than 57 hours. A further increase of arrival rate to six clients per hour resulted in a complete failure of the system as a runtime error occurred because the microphone queue exceeded its maximum capacity.

The bottleneck in the system is determined by whichever of the microphone and window that has the longest service time at that instance (further explanation in Appendix C4), with the window being the constraint most often since its mean service time is longer (9min vs. 7 min).

To improve capacity at the window, two approaches can be taken: (1) increase resource availability by adding a new window, and (2) reduce workload by shifting tasks to non-bottleneck resources (Laguna & Marklund, 2018). Therefore, the owner should expand capacity at the window to handle higher customer volumes, maintain stable operations, and prevent long cycle times or failures under increased demand.

3.5. A Simple Network

In this part, we assume customers balk rather than renege, meaning they leave immediately if the microphone queue reaches its three-person limit instead of joining and later abandoning it (Laguna & Marklund, 2018). With a maximum queue length of three before the microphone, the system is now represented as $M/M/1/4$, where 4 includes the queue capacity plus the customer currently being served.

To handle the maximum queue length before the microphone in the simulation, we added a branch between the entity generator and the queue. Customers join the queue if it has space; otherwise, they exit through an entity sink, which allows tracking the number of balking customers.

In comparison to the previous simulations in task 3.1. and 3.2, both W_q and L_q has decreased when implementing the restricted queue length to 12.1 and 0.74, respectively. As a result of shorter queues, the cycle time and WIP have also decreased. In addition, both servers have a decreased utilization rate. Fewer served customers because of balking will naturally lead to lower utilization rates.

	Delay in Queue	Customers in Queue	Utilization	Cycle Time	WIP
Microphone	12.10	0.74	42.82 %	32.042	1.71
Window	0.00	0.00	54.91 %		

Table 9 – Performance Metrics with Restricted Microphone and Window Queue Length

The system appears more efficient in theory, but over a 10,000-hour simulation, the drive-through lost 3,531 customers – about one in ten. This results in lost revenue and potential dissatisfaction among balking customers. However, those served may be more satisfied due to the shorter cycle time. The system is also stable, as the microphone queue cannot exceed three customers.

List of References

Bunday, B. D., & Scraton, R. E. (1980). The G/M/r machine interference model. European Journal of Operational Research, 4(6), 399-402.

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

Appendix A – Montecarlo Simulation

Revenue in Ski Season:

$$\text{If Demand} \geq \text{Order Amount:}$$
$$\text{Demand} \cdot \$245 \quad \text{else} \quad \text{Order Amount} \cdot \$245$$

Revenue at the End of the Season:

$$\text{If Demand} < \text{Order Amount:}$$
$$\text{Order Amount} - \text{Demand} \cdot \$55 \quad \text{else} \quad 0$$

Costs of Ski Jackets:

$$\text{Order Amount} \cdot \$86$$

Appendix B – Queuing Theory

1)

We calculated the steady-state probability $P_n = 0$ of this $M/M/c/\infty/N -$ system by using the formula:

$$P_0 = \frac{1}{\sum_{n=0}^{c-1} \frac{N!}{(N-n)!\,n!}\left(\frac{\lambda}{\mu}\right)^n + \sum_{n=c}^{N} \frac{N!}{(N-n)!\,c!\,c^{n-c}}\left(\frac{\lambda}{\mu}\right)^n}$$

We further calculated the different P_n values for each c using:

$$P_n = \begin{cases} \frac{N!}{(N-n)!\,n!}\left(\frac{\lambda}{\mu}\right)^n * P_0 & for\ n = 0,1,2,\dots c \\ \frac{N!}{(N-n)!\,c!\,c^{n-c}}\left(\frac{\lambda}{\mu}\right)^n * P_0 & for\ n = c, c+1, \dots, K \\ 0 & for\ n > K \end{cases}$$

We calculated the mean number of jobs L_q in the queue by using the formula:

$$L_q = \sum_{n=c}^{N} (n-c)P_n$$

c	1	2	3	4	5
L_q	0.995334	0.117638	0.00993	0.0004	0

Table 10 – Mean Number of Jobs in the Queue (L_q) for Different Values of c

Eventually, we were able to calculate the mean number of jobs in the system L based on the formula:

$$L = L_q + \sum_{n=0}^{c-1} nP_n + c\left(1 - \sum_{n=0}^{c-1} P_n\right)$$

c	1	2	3	4	5
L	1.796267	1.09411	1.007944	1.00032	1

Table 11 – Mean Number of Jobs in the System (L) for Different Values of c

c	1	2	3	4	5
P_1	0.248834	0.393664	0.408569	0.409567	0.4096
P_2	0.248834	0.196832	0.204284	0.204784	0.2048
P_3	0.186625	0.073812	0.051071	0.051196	0.0512
P_4	0.093313	0.018453	0.008512	0.006399	0.0064
P_5	0.023328	0.002307	0.000709	0.0004	0.00032

Table 12 – Probability Distribution (P_n) for Different Values of c

We further calculate the average arrival rate $\bar{\lambda}$ for each c, using:

$$\bar{\lambda} = \lambda(N - L)$$

c	1	2	3	4	5
$\bar{\lambda}$	0.400467	0.488236	0.499007	0.49996	0.5

Table 13 – Average Arrival Rate ($\bar{\lambda}$) for Different Values of cc

Having obtained the $\bar{\lambda}$-values, we calculate the mean time a job spends in the system W, given by $W = \frac{L}{\bar{\lambda}}$ and the mean time a job spends in the queue W_q, given by $W_q = \frac{L_q}{\bar{\lambda}}$.

c	1	2	3	4	5
W	4.485437	2.240945	2.0199	2.0008	2
W_q	2.485437	0.240945	0.0199	0.0008	0

Table 14 – Mean Time in System (W) and Queue (W_q) for Different c

2)

c	1	2	3	4	5
P_0	0.1971	0.3048	0.3152	0.3158	0.3158
P_1	0.2532	0.4076	0.4255	0.4275	0.4275
P_2	0.2492	0.1995	0.2032	0.2031	0.2032
P_3	0.1951	0.0702	0.0465	0.0466	0.0465
P_4	0.0862	0.0155	0.0092	0.0069	0.0070
P_5	0.0192	0.0024	0.0004	0.0001	0
L_q	0.975	0.108	0.010	0	0
L	1.778	1.091	1.010	1.002	1.001
$\bar{\lambda}$	0.4011	0.4890	0.4983	0.4992	0.4992
W	4.4337	2.2313	2.0279	2.0063	2.0062
W_q	2.4308	0.2216	0.0203	0.0001	0

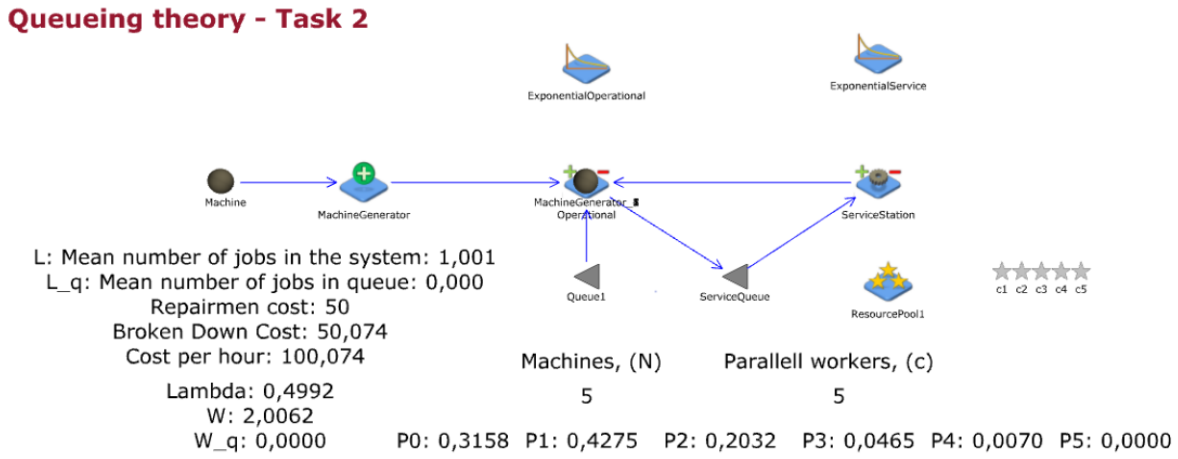
Table 15 – Simulation Results for Different Numbers of Repairmen (c) with Exponential Distribution

3)

c	1	2	3	4	5
P_0	0.1976	0.3119	0.3258	0.3282	0.3281
P_1	0.2541	0.3973	0.4081	0.4073	0.4078
P_2	0.2503	0.1986	0.2082	0.2077	0.2076
P_3	0.1853	0.0702	0.0477	0.0496	0.0496
P_4	0.0846	0.0176	0.0091	0.0065	0.0063
P_5	0.0281	0.0039	0.0011	0.0007	0.0007
L_q	0.987	0.118	0.011	0.001	0
L	1.789	1.096	1.010	1.001	1
$\bar{\lambda}$	0.4010	0.4869	0.4973	0.4984	0.4984
W	4.4655	2.2518	2.0304	2.0084	2.0070
W_q	2.4612	0.2415	0.0228	0.0014	0

Table 16 – Simulation Results for Different Numbers of Repairmen (c) with Normal Distribution

Simulation Model:



Appendix C – A Simple Network

1)

The classification of $a1$ and $a2$ as M is justified by the exponential distribution of both queues and interarrival times. Since there are no parallel servers, $a3$ is set to 1 (Laguna & Marklund, 2018).

The cycle time was recorded using a statistics object before the entities exited the model through the entity sink. The expected average delay in the queues was obtained from “AverageQueueTime,” while the expected number of customers in the queues was retrieved from “QueueLengthAverage.”

2)

In our simulation, the expression threshold opens when the window state is set to "idle," meaning no customers are currently at this station, and acts as a release condition for the microphone server.

4)

The cycle times when turning up the customer arrival rate underscore a critical issue: the system's inability to deal with increased demand. Since a customer at the microphone is not released until the window is vacant, there will always be a customer starting at the microphone as soon as one moves to the window – assuming there are customers in the queue, which there will be when the system is unstable. This means that the system's processing speed is limited by the activity with the longest service time at any given moment.

Explanation on bottleneck: The limited capacity at the window, which results in a longer service time, acts as the primary constraint on the system’s throughput: *“The*

process capacity is determined by the capacity of the bottleneck" (Laguna & Marklund, 2018, p. 167). Since no process can operate beyond its bottleneck's maximum throughput, the excessive cycle time and eventual failure at higher arrival rates indicate that the system is operating at its capacity utilization limit.

Simulation Model: