Project 1

BAN403: Simulation of Business Processes

Candidate numbers: 21, 33, 108

Handed out: February 24, 2025
Deadline: March 3, 2025

Contents

List of Figures

List of Tables

1 Montecarlo Simulation

1.1 Maestranza Hotel

A Monte Carlo simulation is conducted to determine the optimal overbooking limit for maximizing the Maestranza Hotel's average daily profit.

Booking	95	96	97	98	99	100	101
Avg Profit (€)	10,435.24	10,530.56	10,560.50	10,549.99	10,515.16	10,421.39	10,235.15

Table 1: Average Profit By Booking Size

Based on the Monte Carlo simulation results, the optimal overbooking limit for maximizing the average daily profit is between 97 and 98 bookings. The average profit peaks around this range, with minor variations due to the randomness in Excel's simulation. This range balances the increased revenue from additional bookings while minimizing compensation costs for overbooked guests.

1.2 Winter Park

Quantity (Q)	Expected Profit (€)	P($\geq 12,000$)
10	1,590.00	0.00%
20	3,180.00	0.00%
30	4,770.00	0.00%
40	6,360.00	0.00%
50	7,950.00	0.00%
60	9,534.87	0.00%
70	11,026.45	0.00%
80	12,047.78	64.10%
90	12,295.05	60.10%
100	12,100.19	51.20%
110	11,801.59	42.60%
120	11,493.49	38.80%
130	11,183.49	29.50%
140	10,873.49	26.10%
150	10,563.49	20.60%
160	10,253.49	14.90%
170	9,943.49	12.50%
180	9,633.49	9.10%
190	9,323.49	6.00%
200	9,013.49	4.40%

Table 2: Expected Profit and Probability of Profit

(a) Optimal Order Quantity

To maximize expected profit, the optimal order quantity should be selected based on the highest expected profit value from the given table above. The highest expected profit occurs at:

$$Q = 90, \quad \text{Expected Profit} = 12,295.05$$

Thus, the store should order 90 jackets for the upcoming season.

(b) Probability of Achieving at Least €12,000

The probability of achieving at least €12,000 in profit for different order quantities is given in the table. For $Q = 90$, this probability is:

$$P(\text{Profit} \geq 12,000) = 60.10\%$$

The decision on order quantity is influenced by the trade-off between maximizing expected profit and increasing the likelihood of reaching a specific profit threshold. While ordering 80 jackets provides a slightly higher probability of achieving €12,000 (64.1%), ordering 90 maximizes expected profit. This reflects a fundamental inventory management principle, balancing revenue potential with risk exposure.

2 Queuing theory

2.1 Analytical Model for Employment Policy Evaluation

The manufacturing system can be modeled as a M/M/c/∞/N queuing system. In this case, N represents the calling population, which constists of N=5 machines. Each machine experiences random breakdowns following an exponential distribution with a failure rate of $\lambda = \frac{1}{8}$ per hour. A fixed number of repairmen are available to repair these machines, with each repair following an exponential distribution with a service rate of $\mu = \frac{1}{2}$ per hour. The system incurs costs due to machine downtime and repairmen wages, and our objective is to determine the optimal number of repairmen that minimizes the expected average cost per hour.

We start by determining the steady-state probability of having zero broken machines, P_0.

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

Once P_0 is established, we can calculate the probabilities of having exactly n broken machines, P_n:

$$P_n = \begin{cases} \frac{N!}{(N-n)!\,n!} \left(\frac{\lambda}{\mu}\right)^n P_0 & \text{for } n < c, \\ \frac{N!}{(N-n)!\,c!\,c^{n-c}} \left(\frac{\lambda}{\mu}\right)^n P_0 & \text{for } n \geq c. \end{cases}$$

From these values, we compute the average number of broken machines waiting to be repaired, L_q, as well as the average number of broken machines, L:

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

Finally, we determine the total expected average cost per hour, TC:

$$TC = \$50L + \$10c$$

where \$50 is the cost for each hour a machine is broken down, and \$10 is the hourly cost for employing each repairmen.

The table below presents the results obtained from the analytical queuing model, which was implemented and solved using Python, assuming a system of five machiens.

	$c = 1$	$c = 2$	$c = 3$	$c = 4$	$c = 5$
P_0 (probability of 0 broken machines)	0.1991	0.3149	0.3269	0.3277	0.3277
P_1 (probability of 1 broken machine)	0.2488	0.3937	0.4086	0.4096	0.4096
P_2 (probability of 2 broken machines)	0.2488	0.1968	0.2043	0.2048	0.2048
P_3 (probability of 3 broken machines)	0.1866	0.0738	0.0511	0.0512	0.0512
P_4 (probability of 4 broken machines)	0.0933	0.0185	0.0085	0.0064	0.0064
P_5 (probability of 5 broken machines)	0.0233	0.0023	0.0007	0.0004	0.0003
L (Avg. no. of broken machines)	1.7963	1.0941	1.0079	1.0003	1.0000
L_q (Avg. no. of broken machines in queue)	0.9953	0.1176	0.0099	0.0004	0.0000
W (Expected downtime)	4.4854 h	2.2409 h	2.0199 h	2.0008 h	2.0000 h
W_q (Expected time in queue)	2.4854 h	0.2409 h	0.0199 h	0.0008 h	0.0000 h
TC (Expected avg. cost per hour)	\$99.81	\$74.71	\$80.40	\$90.02	\$100.00

Table 3: Analytical Model Results

More repairmen reduce downtime, but also increase labor costs, creating a trade-off. For instance, with only one repairman (c=1), the system experiences an average of 1.7963 broken machines at any given time, leading to an expected downtime of 4.4854 hours and a relatively high expected cost of \$99.81 per hour. Increasing the number of repairmen to two (c=2) substantially improves efficiency, reducing the average number of broken machines to 1.0941, reducing downtime nearly in half to 2.2409 hours, and achieving the lowest expected cost of \$74.71 per hour.

However, beyond $c = 2$, the cost of adding more repairmen outweighs the marginal improvements in downtime. By $c = 4$ or 5, the system is nearly saturated with repair capacity, leading to minimal further reductions in downtime while increasing costs to \$90.02 and \$100 per hour, respectively.

Thus, for a system with five machines, the optimal employment policy is to hire two repairmen, as this achieves the best balance between minimizing downtime and controlling labor costs. Expanding the workforce beyond this point results in unnecessary expenses with minor operational benefit, making it an inefficient choice from a cost-effectiveness standpoint.

2.2 Simulation-Based Analysis Using JaamSim

The manufacturing system is implemented in JaamSim to validate its accuracy through simulation and comparison. The table below presents the simulation results for each policy ($c = 1$, 2, … 5):

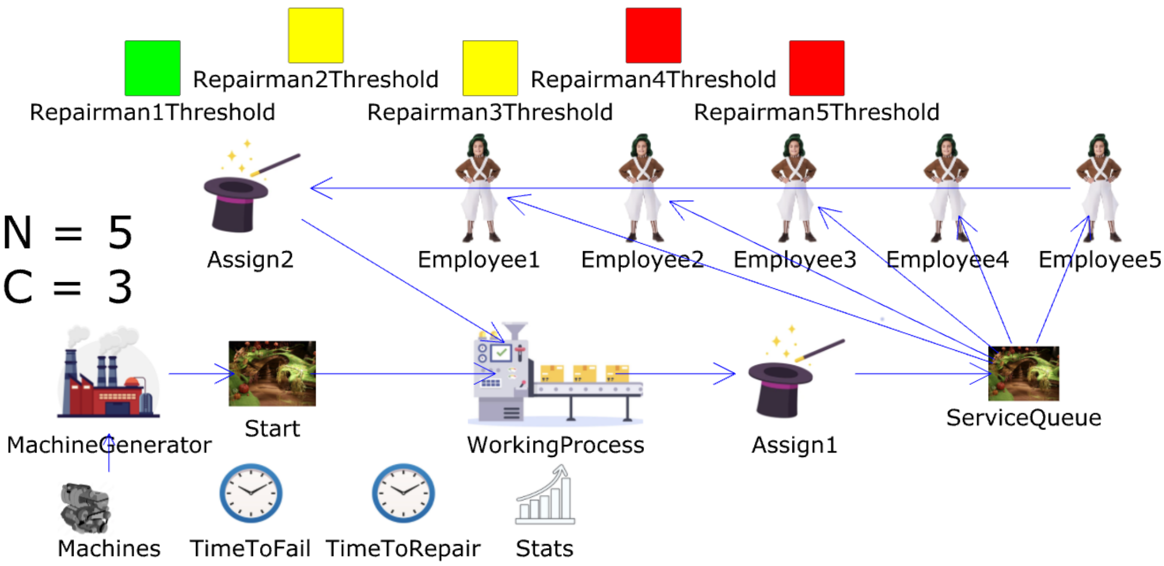

Figure 1: Jaamsim Simulation Broken Machines

	$c = 1$	$c = 2$	$c = 3$	$c = 4$	$c = 5$
P_0 (probability of 0 broken machines)	0.1957	0.3136	0.3254	0.3257	0.3258
P_1 (probability of 1 broken machine)	0.2448	0.3894	0.4054	0.4068	0.4065
P_2 (probability of 2 broken machines)	0.2429	0.2023	0.2089	0.2095	0.2099
P_3 (probability of 3 broken machines)	0.1944	0.0747	0.0510	0.0516	0.0515
P_4 (probability of 4 broken machines)	0.1014	0.0184	0.0088	0.0061	0.0060
P_5 (probability of 5 broken machines)	0.0208	0.0016	0.0005	0.0003	0.0003
L (Avg. no. of broken machines)	1.8234	1.0998	1.0137	1.0067	1.0065
L_q (Avg. no. of broken machines in queue)	1.0191	0.1164	0.0097	0.0003	0.0000
W (Expected downtime)	4.5358 h	2.2377 h	2.0194 h	2.0006 h	2.0000 h
W_q (Expected time in queue)	2.5358 h	0.2377 h	0.0194 h	0.0006 h	0.0000 h
TC (Expected avg. cost per hour)	\$101.17	\$74.99	\$80.69	\$90.34	\$100.32

Table 4: Simulation Results

As expected, we observe slight differences between the results of the analytical model and the JaamSim simulation. These differences arise naturally from the inherent variability in the simulation, where machine breakdowns and repairs are sampled from exponential distributions. In contrast, the analytical model calculates steady-state probabilities using strict mathematical equations. Despite these differences, both models reached steady state, as evidenced by stable outputs over time in the simulation and alignment with theoretical predictions. The consistency between the two methods reinforces the validity of the results and highlights that employing two repairmen (c=2) is the most effective policy for minimizing total system costs.

2.3 Impact of Machine Failure Distribution on Cost Optimization

	$c = 1$	$c = 2$	$c = 3$	$c = 4$	$c = 5$
P_0 (probability of 0 broken machines)	0.1967	0.3108	0.3219	0.3227	0.3231
P_1 (probability of 1 broken machine)	0.2610	0.4028	0.4157	0.4160	0.4156
P_2 (probability of 2 broken machines)	0.2532	0.1972	0.2072	0.2086	0.2085
P_3 (probability of 3 broken machines)	0.1793	0.0681	0.0471	0.0469	0.0474
P_4 (probability of 4 broken machines)	0.0852	0.0180	0.0071	0.0050	0.0047
P_5 (probability of 5 broken machines)	0.0246	0.0032	0.0010	0.0007	0.0006
L (Avg. no. of broken machines)	1.7690	1.0892	1.0046	0.9975	0.9968
L_q (Avg. no. of broken machines in queue)	0.9657	0.1136	0.0090	0.0007	0.0000
W (Expected downtime)	4.4065 h	2.2339 h	2.0181 h	2.0014 h	2.0000 h
W_q (Expected time in queue)	2.4065 h	0.2339 h	0.0181 h	0.0014 h	0.0000 h
TC (Expected avg. cost per hour)	\$98.45	\$74.46	\$80.23	\$89.87	\$99.84

Table 5: Simulation Results Normal Distribution

When modifying the JaamSim simulation to use a normal distribution for machine failures, the results show a slight reduction in expected queue length and waiting time, compared to the exponential model. The optimal policy remains c=2, with the lowest expected average cost per hour at \$74.46, slightly lower than in the analytical model (\$74.71) and the original JaamSim simulation (\$74.99). This difference likely arises because the normal failure distribution shifts the nature of variability, reducing the likelihood of frequent early failures that can cause temporary congestion in the service queue. Unlike the exponential distribution, which has a constant failure rate and memoryless property, the normal distribution clusters failures symmetrically around the mean, leading to a more stable and predictable repair demand.

3 A simple network

3.1 Drive-Through System

The drive-through system consists of two interconnected service stations, the microphone and the food window. Customers first place an order at the microphone before proceeding to the food window to receive their food. This forms a tandem queuing network, where congestion at one service station influences the performance of the entire system. Each station follows a FIFO discipline, and both interarrival and service times are exponentially distributed, making the system a network of M/M/1 queues. To analyze system performance, a discrete-event simulation was conducted in JaamSim to estimate key operational metrics over a 10.000 hour runtime.

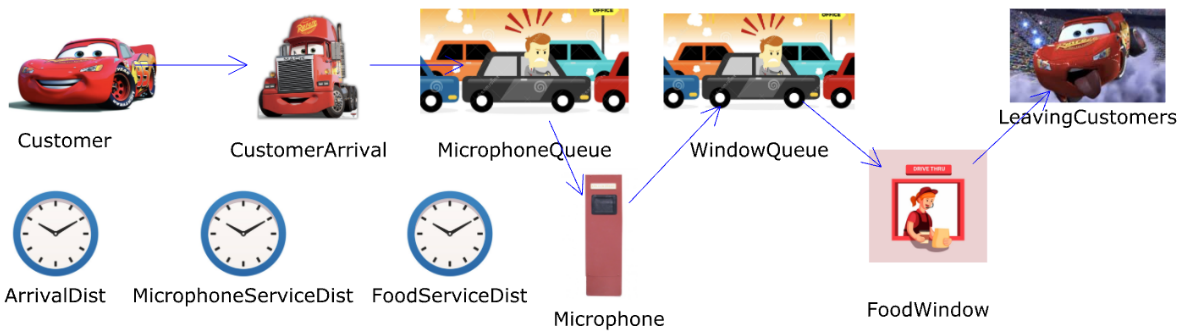

Figure 2: Jaamsim Simulation Drive-Through System

Metric	Microphone Queue	Food Window Queue
Expected Average Delay in Queue (min)	6.178	13.416
Expected Number of Customers in Queue	0.413	0.897
Utilization	0.469	0.602

Table 6: Queue Performance Metrics

The food window is the bottleneck, with the highest utilization at 0.602, leading to longer waiting times. Since the microphone utilization is lower at 0.469, congestion is concentrated at the food window. The system remains stable, but if arrivals exceed 4 per hour, waiting times at the food window will increase nonlinearly. Expanding food window capacity or reducing service time variability would improve performance.

3.2 Drive-Through System with blocking

Blocking occurs when customers at the microphone cannot proceed to the food window due to service unavailability. This dependency prevents new arrivals from entering service, leading to longer waiting times and increased queue lengths at the microphone.

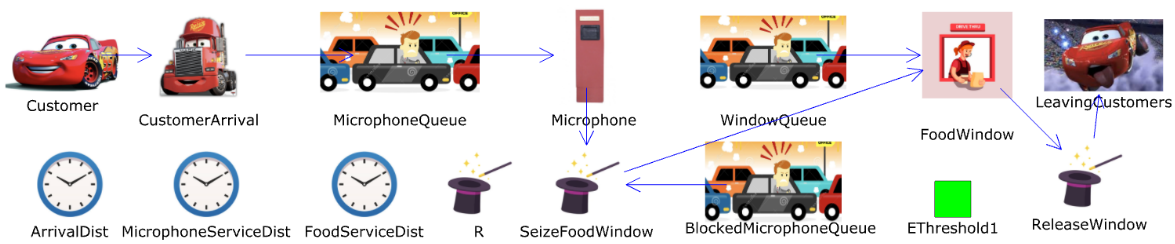

Figure 3: Jaamsim Simulation Drive-Through System with Blocking

Metric	Microphone Queue	Food Window Queue
Expected Average Delay in Queue (min)	36.114	-
Expected Number of Customers in Queue	2.413	-
Utilization	0.469	0.602

Table 7: Queue Performance Metrics under Blocking

The JaamSim simulation results show that under blocking conditions, the expected average delay in the microphone queue increases from 6.19 to 36.114 minutes, while the expected number of customers in the queue rises from 0.414 to 2.413. Despite this significant increase in congestion, the utilization of the microphone remains at 0.469. This is because utilization is defined as the fraction of time the server is actively processing customers, which remains unchanged in both scenarios due to the assumption of infinite queue capacity. Blocking does not alter total server utilization but affects queue dynamics by increasing waiting times and redistributing idle time, reinforcing the impact of service dependencies in queuing networks.

3.3 System Stability

A queuing system is stable if the arrival rate λ is lower than the service rate μ, ensuring utilization remains below one. The process flow rate is calculated as $p = \frac{\lambda}{\mu}$. If $p > 1$, the system is unstable. In our context, the food window queue is the bottleneck, meaning its service rate μ_2 determines stability. Substituting its value, we get $p_w = \frac{4}{6.67} = 0.602$. Since $p_W < 1$, the system is stable.

3.4 Impact of Increased Arrival Rate

To analyze the robustness of the system, two scenarios were tested where the customer arrival rate increased from 4 to 5 and 6 customers per hour. However, both scenarios present unrealistic operating conditions when evaluated using queuing theory.

Metric	5 Customers/hour	6 Customers/hour
Microphone Utilization	0.579	0.580
Microphone Queue Delay (h)	57.34	874.42
Microphone Queue Length	286.63	5,256.38
Food Window Utilization	0.745	0.746

Table 8: Performance Metrics for Increased Arrival Rates

Table 8 shows that increasing the arrival rate from 4 to 5 or 6 customers per hour results in a substantial rise in queue delay, reaching 57.86 hours and 891.42 hours, respectively. Similarly, the expected number of customers in the queue increases to 289 and over 5,360. In practice, such extreme delays are unrealistic, as customers would exhibit balking by choosing not to enter the system or reneging by abandoning the queue after joining. Little's Law confirms that excessive wait times result from constrained service rates unable to accommodate higher arrivals. Rather than experiencing infinite queue growth, the system would face declining demand as customers seek alternatives, making such congestion infeasible in real-world operations.

To maintain a feasible system, the owner must consider increasing order processing capacity, introducing parallel service stations, or optimizing service time. The current configuration cannot handle demand beyond 4 customers per hour without severe inefficiencies.

3.5 Impact of Finite Queue Capacities

Limiting the microphone queue to $K_M = 3$ and setting the food window queue to $K_W = 0$ introduces blocking at both stages. The key performance metrics are summarized in Table 9.

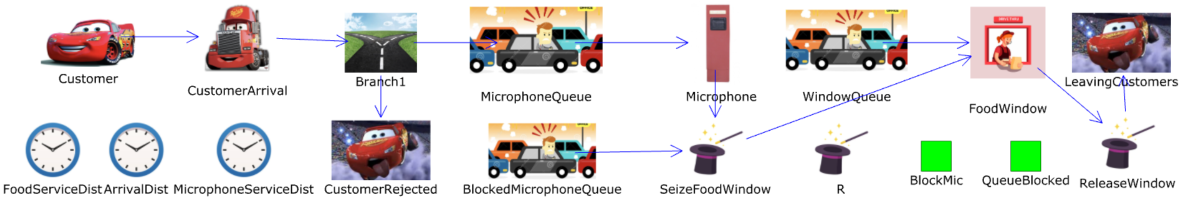

Figure 4: Jaamsim Simulation Drive-Through System with Finite Queue Capacities

Metric	Infinite Q	Blocking FW	Blocking Both Q
Expected Delay in Microphone Queue (min)	6.178	36.114	12.097
Expected Number in Microphone Queue	0.413	2.413	0.737
Microphone Utilization	0.469	0.469	0.428
Expected Delay in Food Window Queue (min)	13.416	-	-
Expected Number in Food Window Queue	0.897	-	-
Food Window Utilization	0.602	0.602	0.549

Table 9: Performance Comparison: Infinite Queue, Blocking at Food Window, and Blocking at Both Queues

Introducing capacity constraints at both the microphone and food window alters system performance. The finite queue at the microphone prevents indefinite queue growth, stabilizing the system by limiting the number of customers in service. However, blocking reduces the effective arrival rate, leading to a decrease in microphone utilization from 0.469 to 0.428 as some customers are turned away. Similarly, food window utilization declines from 0.602 to 0.549, reflecting the throughput losses caused by restricted capacity.

Compared to blocking only at the food window, congestion at the microphone is lower, and waiting times are significantly reduced (12.097 vs. 36.114 min), demonstrating that restricting queue capacity redistributes rather than eliminates delays. Unlike in previous designs, where all arrivals were eventually served, customers now face service denial, emphasizing the trade-off between queue stability and service accessibility. These results highlight the impact of capacity limitations on both system efficiency and customer experience, underscoring the importance of balancing queue capacity to minimize congestion while maintaining throughput.