Table of Contents

PART A: Monte Carlo Simulation

Executive Summary

In the retail industry, maintaining an adequate stock level for the products sold is essential for satisfying customer demand and maximizing sales. Accurate inventory management requires understanding the probability of running out of stock for various products and the estimated demand in a given period. In such scenarios, the Monte Carlo (MC) simulation is a powerful tool to assess the likelihood of stockouts based on estimated demand. This report outlines the results of a Monte Carlo simulation analysis of the demand for our company's products. The objective is to estimate the probability of selling all products.

The analysis is based on MC simulation, a computational method that uses a random sampling method to model and analyze complex systems and processes. It involves generating multiple simulation runs with random inputs and parameters to estimate the probability of various outcomes or events. This technique is widely used in finance, engineering, physics, and other fields where obtaining precise analytical solutions is difficult or impossible.

1. Methodology

We followed a step-by-step approach to implement the Monte Carlo simulation in Excel.

1. *Expected demand*: We generated random demand values for each product, considering the respective probability distributions described in Appendix I.
2. *Unmet demand*: The unmet demand was calculated for each product by subtracting the corresponding expected demand from the stock available. If the available stock exceeds the available stock, this value was set to zero to indicate that all demand is met.
3. *Kvikk Lunsj availability*: to find the amount of available Kvikk Lunsj, we subtracted the stock on hand for Kvikk Lunsj with the expected demand to get the remaining stock on hand. We then calculated 50% of the unmet demand for Smash. Finally, we took this 50% and subtracted the amount which the remaining Kvikk Lunsj could cover.

4. *Total Sales*: This was calculated by adding the total sales for Smash and Kvikk Lunsj (including the customers who originally wanted to purchase Smash) and a column with the combined total sales.

2. Results

Considering this scenario, the probability of selling all products is approximately 34%. The total missed demand over all replications was calculated by dividing each product's unmet demand by the total expected demand. The total missed demand for Kvikk Lunsj was 3.46%, and for Smash, 9.65%.

Depicted below are summary statistic results obtained from the analysis:

Smash	220
Kvikk Lunsj	85.1506

Figure 1: Mean Demand

Smash	17.8023
Kvikk Lunsj	9.2343

Figure 2: Standard Deviation of Demand

Smash	0.7793
Kvikk Lunsj	0.4042

Figure 3: 95% Confidence Intervals

As seen in the summary statistics tables above, the average demand is higher for Smash than for Kvikk Lunsj, but the summary statistics also imply more significant variability in the demand for Smash and suggest that the demand can fluctuate a bit from day to day. Kvikk Lunsj also has some fluctuation in demand but to a lesser extent than Smash.

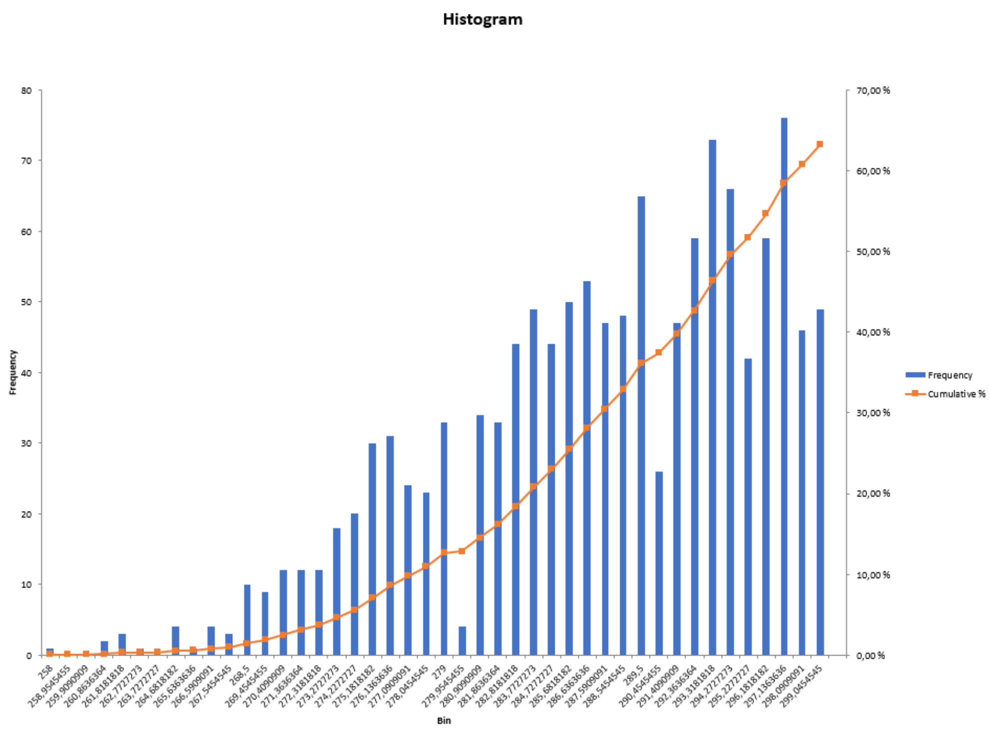


Figure 4: Probability distribution of total sales

The histogram displays the likelihood of different sales outcomes. There is about a 63.19% chance we'll sell anywhere up to 299 units. If we want to sell all 300 units, the chances drop to about 36%.

3. Recommendations

The summary statistics obtained from the analysis show that the demand for Smash shows a higher degree of variability than Kvikk Lunsj. Given the unpredictability of demand, we advise a proactive approach of maintaining a higher safety stock level for Smash. This would serve as a buffer to compensate for the fluctuations in demand and lead time, thereby preventing stock out of the product and maintaining customer satisfaction.

Where the procurement of more Smash is not possible, the supermarket should increase the inventory of Kvikk Lunsj. By doing this, the supermarket can absorb some unmet demand for Smash, avoiding sales loss. Further pricing analysis could give valuable information about which product brings more profit when sold. By doing so, one can make informed decisions on whether it is more financially sensible to procure an increased quantity of Kvikk Lunsj to satisfy the unmet demand for Smash or aim to recover sales from the lost demand by focusing on Smash. The goal of this strategy would be to attain a higher profit margin through optimal inventory management.

PART B: Restaurant Simulation

Executive Summary

In the competitive world of hospitality, strategic planning and efficient resource allocation are key determinants of a successful business. With the prospect of The Queen Vic's expansion into a restaurant, a need arises to assess the potential impacts of strategic decisions on performance. This report presents the results of our simulation model designed to aid bar owners with valuable insight, thereby guiding them in their decision-making process.

This report aims to provide bar owners with valuable insights gained through modeling the case of turning their bar into a restaurant and conducting various experiments and optimizations. Uncertainties such as customer demand, staffing requirements, and restaurant capacity make it essential to develop a reliable model that can aid in understanding the transformation process and identifying areas that require attention.

The simulation model, developed in AnyLogic, is a virtual representation of the proposed bar-restaurant setup is a simplified yet representative replication of a real-life scenario. The model showcases a restaurant with bartenders and waiters and how they interact with different types of customers, drinkers, and diners. The restaurant has an entrance where customers can queue and enter. Some of the customers buy drinks only, others may dine in.

It focuses on two primary objectives that are crucial for the business: optimizing resource utilization, service level, and customer waiting times. Efficient staff management improves productivity and minimizes operational costs, while shorter waiting times elevate customer satisfaction and increase potential revenues.

To assess the best strategy for adding the restaurant, we have added vital KPI measurements, such as resource utilization, average waiting times, and service level. We can use these results to validate how resource changes and seating capacity affect performance expectations.

For the baseline scenario, we used the suggested guidelines from the owners. The baseline model investigates how the restaurant's resources can handle the inflow of customers and draw attention to whether the bar owners should consider hiring more or less staff or increasing seating capacity. We used two bartenders, one waiter, and 12 seats. We ran the experiment multiple times to test this suggestion to get the resulting averages. The outcome of this experiment on the baseline model revealed shortcomings with this combination. Only 55% of customers could order food in less than 10 minutes. The number of seats was also insufficient; on average, diners had to wait 13 minutes to get a seat, with only 53% getting a seat in less than 10 minutes. This baseline scenario is not sufficient for the amount of demand that the restaurant is expected to receive.

For the following scenario, we increased the number of bartenders. So the new numbers we tried were two waiters, two bartenders, and 12 seats to improve the KPIs. The waiting time for a seat was still high, with an average of 18 minutes. All numbers were very similar to the baseline except for waiting time for ordering food and utilization of the waiters. Although the other numbers differed, this is likely just due to the random chance of the experiment.

We used an optimization experiment to avoid trial and error to find the best solution. The optimization experiment allowed the number of bartenders, waiters, and seats to vary. The optimization attempted to find the minimum number of wait staff and seats needed while meeting the wait time requirements. The optimization experiment gave us three as the optimal number for bartenders, two for waiters, and 20 seats.

Below is a summary of statistics explaining the base case results, scenario 1, and optimization experiment.

KPI	Baseline Scenario	Scenario 1	Optimized Scenario
% Drinkers wait < 5 minutes	84%	82%	99%
% Seat Wait < 10 minutes	53%	52%	95%
% Order Wait < 10 minutes	55%	99%	100%
Average Seat Wait Time	13.59	18.18	0.976

Average Order Wait Time	11.52	0.97	0.92
Average Drink Wait Time	2.18	2.4	0.363
Average Bartender Utilization	61%	62%	41.2%
Average Waiter Utilization	55%	29%	27.6%
Average Seat Utilization	58%	58%	36.4%

Figure 1.1: Summary results

As evident in the summary statistic table above, we can see that the performance of the simulation gives various results. In terms of keeping less than 95% of customers waiting for a seat and keeping average waiting times low, the optimization experiment provided the best results. However, it also showcases lower utilization of resources. Consequently, one might think that lower utilization implies operational inefficiencies. In contrast, in this case, utilization is rather well distributed to handle the varying demand.

As shown in the base case, seating capacity is one of the main bottlenecks of Queen Vic. We, therefore, recommend updating the seating capacity to be able to serve 20 customers simultaneously. Employing an additional two bartenders and one more waiter would significantly speed up the serving time. In comparison, the baseline scenario and scenario 1 were insufficient to handle the desired performance outcomes of the restaurant and bar. Perhaps there are alternative solutions, although we firmly believe that the optimized results are “correct” solutions and satisfy the needs of QueenVic.

Technical Report

1. Introduction to the System

1.1 Purpose of the model

We have constructed a simulation model in AnyLogic using the collected data and information gathered from the pub's owners, employing a discrete-event simulation (DES) approach. The purpose of the simulation model is to provide the owners of The Queen Vic Pub with valuable insights on staff hiring and the number of seats that should be available in the restaurant to optimize the system's efficiency by minimizing the waiting time and maximizing staff utilization.

1.2 Data and assumptions

The created model has four model input parameters, which are the variables that allow us to specify values or settings that influence the model's behavior. We're informed that two bartenders are working behind the bar, one taking food orders, and they are considering hiring another waiter to take food orders and 12 seats. Restaurants do not have unlimited space for customers in queues for the bar and the restaurant. As this information was not provided, we assumed there could not be more than 100 people at each block in the system. In addition, all customers go through the whole system before leaving, and there is no set capacity for the number of seats available in the restaurant. All additional parameters in the following model are provided based on the information given about bar owners. The model is set to run a specific date within the opening hours and with all unique runs, therefore designed to be independent of the date chosen.

2. Logic

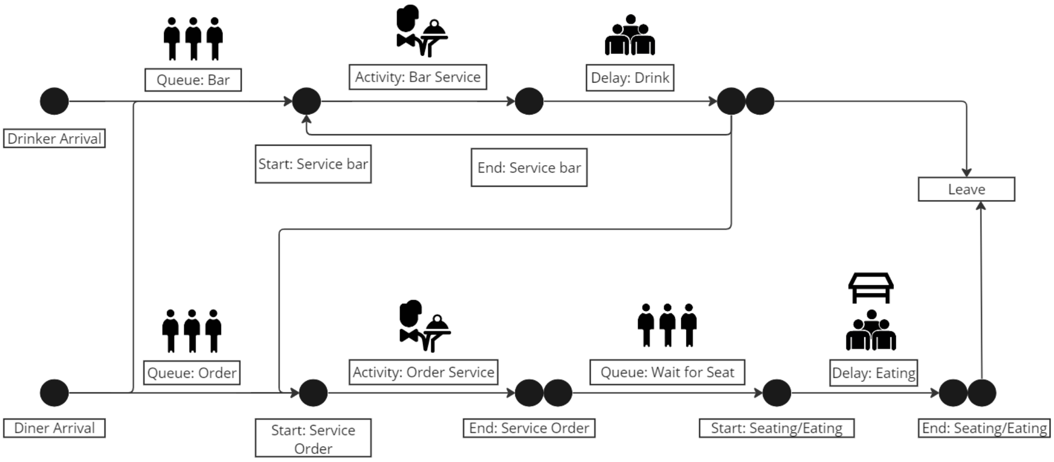


Figure 2.1: Model Overview

2.1 Model logic

The baseline model is set to run for a specific date from 17:00 until 02:30 with the model time unit in minutes. The model is set to a random seed, making each run unique. The model consists of a population of customers and three resources.

The resources are managed and allocated dynamically based on customer needs and availability. The model simulation consists of three resource pools:

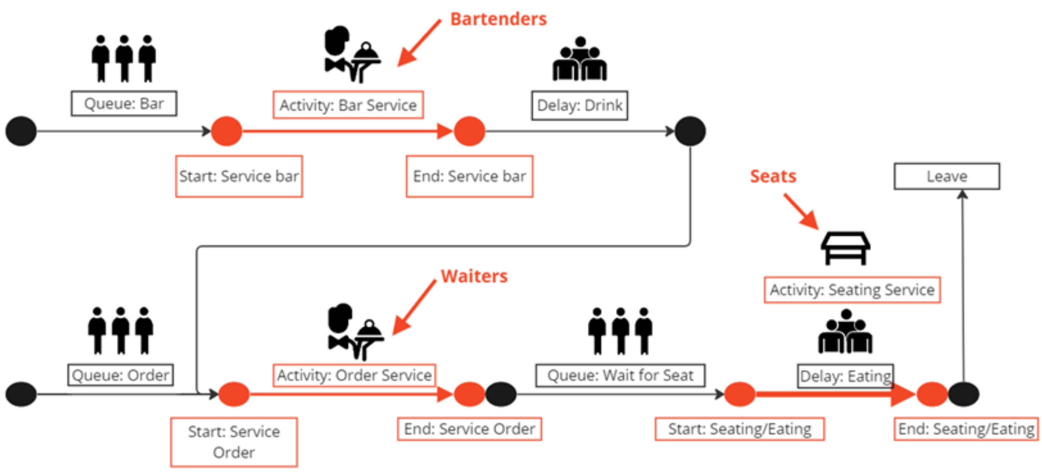


Figure 2.2: Resources

- *Bartenders* represent the static staff members dedicated to serving the drinks at the bar area. The bartenders are seized when the bar service event starts (when the bartender takes a drink order) and released once the event ends (when the customer has gotten their drink).
- *Waiters* represent the static staff members responsible for taking food orders at the dedicated order area of the restaurant. The waiters are seized

when the Service order event starts (when a diner order is placed) and released once the event ends (when the order is delivered).

- *Seats* represent the physical seating arrangements in the restaurant area. A seat is seized once the Seating/Eating event starts (when a diner has gotten their food order) and is released after the event ends (when a diner is done eating and the table has been cleaned).

The availability and utilization of bartenders, waiters, and seats affect customer waiting times and the overall capacity of the restaurant. The availability and utilization of these resource pools impact the efficiency of customer service and the overall flow of customers through the system.

Customers are the objects of interest moving through the system and driving the dynamics of the model. These entities are subject to events and processes within the simulation, they may arrive at the pub or restaurant, join a queue, place an order for a drink or food, consume their food/drink, queue for a seat, and then eventually leave. The customers are divided into diners and drinkers; while they have similar attributes overall, their process of moving through the system varies. A parameter defining customer type is set to separate whether the customer is a diner or a drinker, where a drinker has a customer type of 1 and a diner has a customer type of 0. Each customer has a unique ID, which increments each new customer entering the corresponding source. Each customer has an arrival and waiting time, which tracks their time in the system, and ultimately influences their experience. The customer is also given a variable set to count the number of drinks they have. Each customer has a limit of drinks they may have at the bar, which is uniformly distributed between 1 and 10.

Within the model, the entities engage in several activities that capture the system's dynamics. These activities are centered around the events within the pub and the restaurant and customers' interactions with the resources and processes. The drinkers and diners have two different routes.

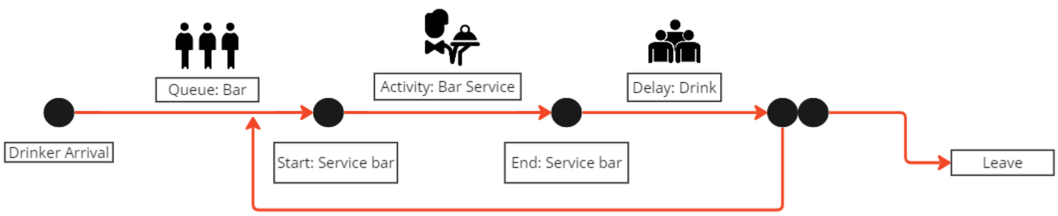


Figure 2.3: Drinker route

A drinker arrives between 17:00 and 22:30, which is modeled using an allowDrinkerArrivals-variable that is initially set to True, a drinkerArrivalEvent, which allows drinkers to arrive when the model time starts, and a stopDrinkerArrivalEvent which stops the drinker from arriving from 22:30. When allowDrinkerArrivals is set to True it injects one drinker into the drinker arrival source at a rate of, on average, 1 every 7 minutes. Once the drinker has arrived, they join a queue for the bar, where they eventually get served by a bartender. This activity does not start until a bartender is available. The delay to get their drink is uniformly distributed between 0.5 and 5 minutes. Subsequently, they drink their beer with a triangular distribution of a minimum of 10 minutes, a maximum of 30 minutes, and 20 minutes on average. They are then moved to a select output block that checks if the customer is a drinker; if that is true, the customer is moved to another select output block where a condition checks whether the drinker has reached their drink limit; if no, then 85% of the customers will go back for another drink. Otherwise, 15% of the customers and those that have reached their drink limit will leave.

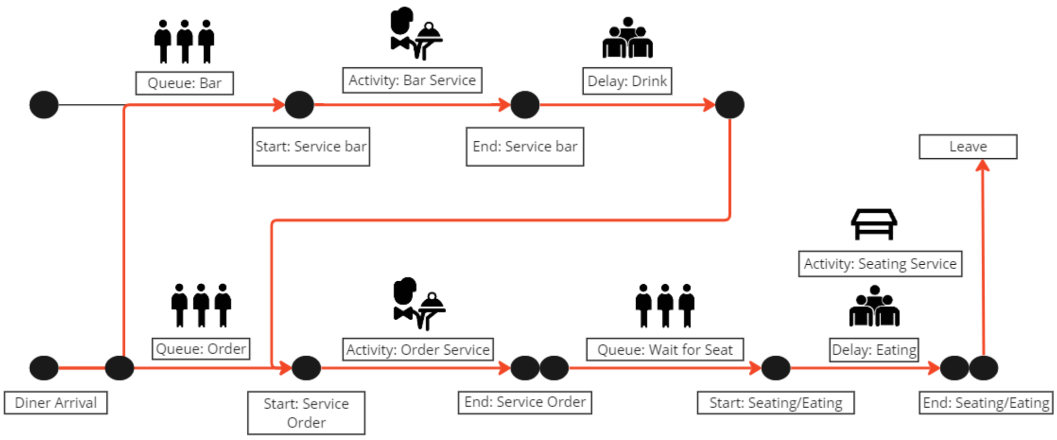


Figure 2.4: Diner route

The activities and routes of a diner are depicted in *Figure 3.2.2*. Diners arrive in groups of one to four between 17:00 and 21:00, modeled by an allowDinerArrivals variable initially set to True and a dinerArrivalEvent, which injects these agents using a custom distribution called groupDistribution at a random rate of 1 group every 12 minutes. This event is stopped when stopDinnerEvent activates at 21:00. Once the diners have arrived, they are seen as separate agents and are given the choice of either going straight to order or having one drink with a probability of 50%, with the scheme displayed in the appendix. The diner that goes for a drink goes through the same process as a drinker, except after having one drink, they pass through the same select output block and check if the customer is a diner; if yes, they return to the queue to order. In the order queue, they get served by a waiter who takes their food order, which is uniformly distributed within 1 to 6 minutes. Once they have received their order, they wait for a seat. Once a seat is available, they eat their food which, on average, they spend a minimum of 25 minutes, a maximum of 90 minutes, and a mean of 55 minutes. After they are done eating, they leave the restaurant.

Queues are essentially waiting times in a system. Our model simulation includes a queue to the bar, a queue to order food, and a queue to get an available seat. The queues in our system incorporate the FIFO queuing discipline, which governs how the entities are prioritized for service, which means that the first entity at the service (the first one to arrive) is selected first for service. The queues are constrained to a people limit, set to a capacity of 100 people at the queue. Numerous assumptions can be made here; however, we believe this is a reasonable top limit for this simulation. In real-life scenarios, some people leave the queue if the waiting time is too long, called reneging (Shastrakar et al.), but in our model, we assume they do not.

The queues are also affected by the resources at the service, either a bartender, waiter or an available seat, whose utilization affects operational costs. "Queuing theory aims to discover a balance between waiting times and resource utilization. Ideally, waiting times should be minimal, while the components in the queue system should have high utilization" ("Queueing Theory for Simulation"). People do not enjoy long waiting times, so an optimal balance between the utilization of resources and waiting times should be considered

2.2 Model Validation

To validate the model, we used both the visuals from the model and histograms. We created histograms for the drinker arrivals by getting the time in the source, subtracting the time from the last arrival, and storing this in a dataset. And histograms for all delays using TimeMeasureStarts and TimeMeasureEnds around the delay blocks. These histograms can be found in the appendix. Although the numbers did not match the guidelines precisely, this was expected due to stochasticity.

3. Model Output KPIs

3.1 Service Time Levels

Measuring an individual's service delivery duration is crucial. Customer satisfaction will increase if the customer is given quick and efficient service in the system. In our assessment, the service level includes the duration for individuals to place their orders for drinks or food and secure an available seat. To calculate the service level, we used measures that record the time the customer queues at the bar before receiving service from a bartender, how long a diner waits to order their food, and how long they wait to get seated.

Four TimeMeasureStart blocks and three TimeMeasureEnd were implemented to capture the time in which the customer passes certain parts of the system; one to track the time it takes from a drinker/diner arrives at the bar until they get to place an order at the bar (timeBarStart/timeDinerStart → timeGetServed), one to track the time it takes to order food (timeDinerStart/timeAfterDrink → timeOrderEnd) and to get a seat (timeSeatingStart → timeSeatingEnd). We then added histograms to get the time-measure end distributions for each block. These histograms were only included in the main simulation as they relate strongly to the waiting time.

3.2 Waiting Times

Prolonged waits can give the impression that the establishment is understaffed or inefficient and reduce the impression of value. As stated by the pub owners, 95% of the customers should be served within 5 minutes or less at the bar, wait less than 10 minutes to order their food, and wait less than 10 minutes to get a seat. A condition was added in the TimeMeasureStart blocks to track the time each agent entered through that time measure and a condition in the respective TimeMeasureEnd to calculate the waiting time for the agent. The logic is described in the appendix. If the waiting time were above 5 minutes for the bar, 10 minutes to order, or 10 minutes to get a seat, a variable would count the number of these instances and add this to a pie chart. This logic can be seen in a code snippet in the appendix. The average mean waiting time for the customer was then stored in a histogram.

3.3 Utilization of resources

Another model output measure is the utilization of bartenders, waiters, and the utilization of seats. This represents the number of resources used out of the total number of resources available in the corresponding resource pool. While maximizing customer satisfaction is a key objective for a business, it is also essential to minimize costs. Having too few resources would lead to long waiting times and decrease customer satisfaction, but having too many would lead to unnecessary costs.

To measure the utilization of the three different resources, the utilization of each resource is recorded using utilization() and stored in a dataset for each resource at different indices. It adds the mean utilization value from a specific point in time to a histogram.

4. Experiments

4.1 Experimentation Aims

The experiments within our model are to help provide detailed insight into the impact of varying input parameters and how changing these parameters would affect the system's state and potentially make room for improvements within the current base model. As the owners of The Queen Vic pub have initial thoughts on how many staffers they want to hire and how many seats they intend to have, the experiments can guide potential decision-making towards a good trade-off between system efficiency and customer satisfaction.

The baseline simulation considers parameters evident from the bar owners, who currently have two bartenders and one waiter taking food orders and considering 12 seats. This baseline simulation examines how the system will affect set KPIs on these original input parameters. The business owner tends to have KPIs that they want to reach. However, these often conflict with their initial thought of their intended allocation of resources.

The parameter variation experiment aims to analyze the sensitivity and impact of different input parameter values on the model outputs and the overall performance of the simulation model. In this case, input parameters can be systematically varied, and one can observe how the changes affect the state of the system and the model outputs. The optimization experiment aims to identify the best possible parameters that minimize the waiting time for the customer by their expectations of KPI results. Ultimately, the goal is to optimize system efficiency, enhance customer satisfaction, and minimize overall operational costs by having an optimal utilization of the staff that they hire.

4.2 Optimization Experiment

Optimization experiments allow us to find the optimal solution based on an objective and limited by some set constraints. Using only parameter variation or the original simulation, we would manually need to try many different solutions, while the optimization experiment tries many different combinations for us. Optimization is now used in many real-world DES simulations, including manufacturing and health care. Although optimization is a helpful tool, it is not perfect. Finding the optimal solution is often timely and computationally expensive. Our model was a very simplified version. A real-life example would be much more complex. In addition, there are many different optimization algorithms, and each one can lead to a different result (Raska & Ulrych, 2014).

For this experiment, the maximum number of people was set at 100, the same as before. The numbers for the bartenders, waiters, and seats were set to vary. The minimums were set to one waiter, one bartender, and twelve seats. The maximum number of waiters and bartenders was set to six, and the maximum number of seats was set to 20. All three parameters were set to a step of one.

The objective of the optimization was set to minimize the number of bartenders + the number of waiters, + the number of seats. This was chosen because we wanted the minimum number of staff and seats to meet demand. Requirements were added so that 95% of people would wait less than their respective wait times. This was done by using the variables to track waiting times created earlier and finding the ratio of people waiting less than the required wait time and those who had to wait for more, these ratios were stored in variables called barRatio, seatRatio, and orderRatio in the main simulation. This ratio was then placed in the requirement section of the optimization and set to be greater than or equal to 0.95.

The results of the optimization experiment suggest three bartenders, two waiters, and 20 seats. This is the minimum number of bartenders, waiters, and seats required, to meet the condition of 95% of people waiting less than 10 minutes for a seat and food service and 5 minutes to be served a drink. Multiple runs of the optimization experiment lead to the same objective function. For example, the output we chose led to an objective of 25, but having 19 seats, three waiters, and three bartenders was also a feasible result that led to the same objective. We chose this outcome as we felt it was better to have a higher number of seats and to get the number of staff as low as possible, as seats are a fixed cost.

4.3 Scenarios

4.3.1 Base Case Scenario

To conclude different what-if scenarios, first, we must understand QueenVic restaurant's baseline with the assumption of one waiter, two bartenders, and 12 seats at the restaurant. To achieve a solid result of the base case, we ran the model with the above-mentioned parameters a hundred times. Multiple runs help capture the system's variability, which helps identify patterns and characteristics.
What If Scenario Analysis (WISA) with the combination of DES is a robust framework for understanding and exploring the possibilities that might happen in the future. We can gain insights into potential outcomes by considering various scenarios by altering parameters. Moreover, this approach allows us to visualize the impact of different decisions, providing valuable information for strategic planning and decision-making processes (BigBear.ai., 2021).

In the base case scenario, the utilization of bartenders is relatively high, 61%, indicating that the bartenders are occupied most of the time. The KPI of Drink Wait Time shows how long it takes to get a drink on average. With the utilization of one bartender over the simulation of a hundred runs, the mean Drink Wait Time is approximately 2.18 minutes, as referenced in the appendix, which is considered optimal. According to the article by Seattle Weekly. (2016): *"at a hopping restaurant at 7:30 on a Saturday, a 10-minute wait is probably a good starting point."* However, the order wait time is slightly above 11 minutes. Moreover, 45% of the customers need to wait more than 10 minutes to order. Therefore, we believe it is a sub-optimal scenario and would lead to unsatisfied customers for QueenVic.

4.3.2 Scenario 1: Two waiters, two bartenders, and 12 seats.

Compared to the base case, the average utilization of the bartenders is similar to the base case, with a mean of 0.62, as referenced in the appendix. It indicates that usually, the bartenders are occupied but not working at their peak, meaning they can handle high variances in utilization. The Drink Wait time has a minor increase to 2.4 minutes which is considered negligible. However, a significant improvement in Average Order Wait Time, almost all customers (99%) receive their orders within 10 minutes.

The fact entails that, in this case, most of the customers get their service within a reasonable time; hence we would suggest having a minimum of two Full Time Employees (FTE) as bartenders in the restaurant. Nevertheless, the Average Seat Wait Time of 18 minutes is sub-optimal therefore, we recommend further optimization techniques.

4.3.3 Scenario 2: Optimized parameters: Two waiters, three bartenders, and 20 seats.

In our third experiment, we ran the model with the above-mentioned optimized parameters to compare with the II. scenario and conclude. With the optimized parameters, the Drink Wait Time stays reduced to 0.36 minutes (21 seconds). Consequently, compared with Scenario II, 99% of customers must wait less than 5 minutes to get a drink. Compared to baseline results of 84%, this entails an absolute increase of 15% improvement. The mean utilization of the bartenders from 62% decreased to 41.2%. Looking at the graph in the appendix, one can see that FTE bartenders aren't working at their peak performance level. Although we believe that when it comes to resource utilization, similar to how machines tear and wear down quickly on their peak utilization level, the same happens to human resources. According to Business Insider, continuous peak utilization levels normally lead to employees making mistakes, experiencing burnout, and eventually quitting their jobs. Business Insider. (2021, November 10). Entertainment and restaurant workers are leading the Great Resignation. Therefore, optimization is key to finding the right balance and can contribute to customer satisfaction while optimally taxing FTEs.

Regarding seating, one can see that eight additional seats would decrease the time to about 1 minute to get seated from 13.59 minutes. Assuming that seats are fixed and cannot be moved around, it would result in a 66% space increase in restaurant size (12*1.66 = 20). According to the article published by Toast, a restaurant service solution consultancy and service provider, different seating strategies can enhance and improve space utilization at the restaurant. Therefore we would strongly recommend for QueenVic, that sufficient enough space is available before converting the pub to a restaurant.

5. Results and Analysis

The results of the experiments indicate that using the parameters from the optimization experiment would improve most KPIs. The number of drinkers waiting less than five minutes is 99%, and the number of diners waiting less than 10 minutes to order and get a seat is also above 95%. Although the utilization KPIs are not particularly high, it seems a fair trade-off to get the other KPIs higher. This is especially true considering that the number of seats is a one-time fixed cost and that the utilization of waiters did not change from the base case to the optimized case. In addition to the KPIs, we also wanted to view the average number of drinks and customers coming in. These numbers do not change from the base case to the optimized case, therefore, are not considered KPIs. But we thought it might be useful to see these statistics. In addition, if we wanted to allow customers to leave before the exit if unsatisfied, these numbers would be more useful and could potentially be KPIs.

6. Conclusion

As Queen Vic expands, the simulation model offers a strategic compass, steering decision-making with valuable insights. Whether it's adjusting the number of staff or seating arrangements, the model's findings enable informed choices that enhance customer experience and profitability. This model can also be expanded to include more parameters and more KPIs as the need for this arises. Based on our thorough analysis, the recommendations for Queen Vic are:

1. Expand/organize the seating area to fit 20 rather than the given 12 seats. This would ensure the restaurants give adequate service in a reasonable time to the arriving customers.

2. To cope with the utilization and demand, the number of bartenders must be at least three.
3. To handle the orders, the optimal number of waiters would be two.

PART C: Critical essay

1. Research Paper I: “Discrete event simulation of an automated warehouse inspection system with drones”

The paper “Discrete event simulation of an automated warehouse inspection system with drones” by Al-Mahdifrom et al. from 2021 explores how to solve the problem of optimizing the performance of an automated warehouse inspection system using drones. The authors use DES to evaluate system performance and optimize two crucial control parameters: the minimum battery level for deploying drones and the number of drones employed in the system. The objective is to reduce task completion times and the percentage of failed tasks due to drone crashes and other variables.

The authors manipulate control parameters to optimize system efficiency and reduce operational expenses. These complex systems typically involve discrete events occurring at specific times with many components. Such components include interactions, drones and server queues, warehouse environments, and inspection tasks. These tasks are initiated and terminated at specific times, making it reasonable to model using DES, and this is one of the main reasons behind the author’s choice of technique. DES is more applicable for deterministic modeling, while ABS tends to be used for stochastic approaches (Mallipeddi, R., & Altiok, T. (2016)). Another reason is that the events are well-defined in the environment, and the administrator unit that sends out the coordinates is centralized. Therefore, this particular problem naturally calls for a DES system approach to modeling (Siebers et al., 2010).

To conclude whether ABS could have been used as a different approach, we must understand the key differences between ABS and DES. Siebers et al. (2010) discuss the applicability of DES and ABS. According to the study, the main difference between discrete-event simulation (DES) and agent-based simulation (ABS) is their modeling approaches. DES represents systems as a series of distinct events, whereas ABS systems model a collection of autonomous agents interacting with one another and their environment. At the same time, excels at modeling systems with clearly defined events and processes, while ABS proves more suitable for modeling many interacting components of complex systems with numerous interconnected components Siebers also concludes that the choice between simulation systems should be based on the problem requirements rather than the application domain.

If the drones could make decisions autonomously (in a decentralized approach), it would reflect an individualistic behavior and decision-making process. In such a case, ABS could be a better approach, and the granularity of the system could be increased by extending the level of interactions between drones and the environment. In DES, queues are the main key elements of the systems, whereas ABS normally does not have queues.

Nevertheless, it is possible to model drones with their own decisions about where to fly next, such as cells in the study of Grimm, V., & Railsback, S. F. (2005). According to Grimm's model, "Rules define the nature of interactions between cells and how cells release chemicals or other entities." Consequently, considering the warehouse inspection systems with drones implies that key parameters such as battery discharge, charging time, and so on could also be modeled using ABS. An ABS model could effectively mimic the behavior of drones in reaction to distinct scenarios and control parameters. For instance, it could replicate how drones maneuver through the warehouse setting to accomplish inspection tasks, all while avoiding hitting obstacles and other agents. At the same time, it also simulates how drones communicate with one another and the server queue to optimize task completion times. Therefore we believe that with the right modification, ABS could have achieved similar results as DES.

2. Research Paper II: A Heuristic-Based Airport Shopping Behavior Model with Agent-Based Simulation

The study explores probability theory and eye-tracker data, which models participants' glaze in the environment to model different shopping behaviors at the airport based on the time remaining until departure. Modeling how humans interact with the environment and what purchases they make at the airport is a complex problem focusing on individuals and their interaction with the environment. Such a problem combined with probability theory naturally gravitates to an ABS approach since individuals are decentralized and have their own decisions (which shops to go to) and environmental interactions. Hence we believe that it is one of the main reasons that influenced the author's choice of technique.

Referring to the previously mentioned paper helps distinguish between DES and ABS and highlights the main differences between modeling systems. It is clear that due to the granularity of the problem, it would have been quite difficult to model it in DES and might be inapplicable due to the decentralized nature of the system. In contrast to deterministic modeling approaches, MCS focuses on random sampling from input variables. (IBM. (n.d.). Monte Carlo Simulation.) These input variables are used for random sampling within predefined ranges to predict various outcomes for a given system. Aggregating the results of MCS would also give a range of possible outcomes that are highly beneficial to assess uncertainty or variability. In that sense, uncertainty is normally a part of ABS.

According to Watson, M.D., Mesmer, B.L., & Farrington, P.A. (2020), published by NASA, "ABS drives the use of MC simulation to determine a probabilistic distribution of the outcome." MCS is ideal to use in tandem with ABS, although MCS per se would have been insufficient to address the entire problem. The research revolves around modeling customer behavior from crossing security checks and estimates the cumulative probability of passengers staying at the shopping area along with shop preferences and crowdedness level. ABS is perfectly achieved to simulate and visualize the agents' interactions with the environment which MCS could not have done.

Emphasizing the procedure of the model that captures the behavior, decision-making processes, and interactions of agents over time also indicates why MCS would not be possible. We also see a tendency in the operations research that customer behavior problems tend to be addressed by ABS, whereas queuing problems such as airports commonly are modeled with DES.

References

Grimm, V., & Railsback, S. F. (2005). Immunity through swarms: agent-based simulations of the human immune system. In Multi-Agent-Based Simulation V (pp. 79-88). Springer.

IBM. (n.d.). Monte Carlo Simulation. IBM. Retrieved from https://www.ibm.com/topics/monte-carlo-simulation

König, M., & Schultmann, F. (2021). Beaming market simulation to the future by combining agent-based modeling and scenario analysis. Management Review Quarterly, 71(3), 383-408. https://doi.org/10.1007/s11573-021-01046-9

M. A. Al-Mahdi and A. M. Al-Saedi, "Discrete event simulation of an automated warehouse inspection system with drones," 2021 International Conference on Computer and Information Sciences (ICCIS), 2021, pp. 1-6, doi: 10.1109/ICCIS52698.2021.00005.

Maidstone, R. (2016). Discrete Event Simulation, System Dynamics and Agent Based Simulation: Discussion and Comparison. ResearchGate. Retrieved May 16, 2023 from https://www.researchgate.net/publication/306157680_Discrete_Event_Simulation_System_Dynamics_and_Agent_Based_Simulation_Discussion_and_Comparison

Toast - How to Improve Your Restaurant's Seating Strategy
Retrieved May 20, 2023 from
11 Ways to Get More Out of Your Restaurant Seating Strategy - On the Line | Toast POS (toasttab.com)

Mallipeddi, R., & Altiok, T. (2016). Discrete-Event Simulation, System Dynamics, and Agent-Based Simulation: Discussion and Comparison. Simulation, 92(1), 5-21. doi: 10.1177/0037549715606076

BigBear.ai. (2021, September 14). What is discrete-event simulation and how does it help in hospital process optimization and capacity planning? https://bigbear.ai/blog/what-is-discrete-event-simulation-and-how-does-it-help-in-hospital-process-optimization-and-capacity-planning/

Seattle Weekly. (2016, June 1). The Bar Code: How Long Should You Wait for Your Drink? https://www.seattleweekly.com/food/the-bar-code-how-long-should-you-wait-for-your-drink/

Business Insider. (2021, November 10). Entertainment and restaurant workers are leading the Great Resignation. Retrieved May 20, 2023, from https://www.businessinsider.com/entertainment-and-restaurant-workers-leading-great-resignation-jolts-2021-11?r=US&IR=T

"Queueing Theory for Simulation." *Queueing Theory for Simulation | Software Solutions Studio*, 11 Sept. 2021, softwaresim.com/blog/queueing-theory-for-simulation.

Watson, M.D., Mesmer, B.L., & Farrington, P.A. (2020). Engineering Elegant Systems: Theory of Systems Engineering. NASA/TP–20205003644.

Raska, P., & Ulrych, Z. (2014). Testing Optimization Methods on Discrete Event Simulation Models and Testing Functions. Science Direct. https://reader.elsevier.com/reader/sd/pii/S1877705814002999?token=AE6D935F91AD100BAD8168ED45F0F553ADE731526D24A84D14E0029CD6BB90997A0EDA9F6D02A58AB231C34F9C4BDAAC&originRegion=eu-west-1&originCreation=20230519080109

Shastrakar , Damodhar, et al. "Literature Review of Waiting Lines Theory and Its Applications in Queuing Model." *IJERT*, no. 2278–0181, 2016.

Siebers, P. O., Aickelin, U., & Menachof, D. A. (2010). Discrete-event simulation is dead, long live agent-based simulation! Journal of Simulation, 4(3), 204-210.

Appendix

Appendix I: Monte Carlo Simulation

Generation of Demand:

- In the "Expected Demand" column, we separated a column for Smash and one for Kvikk Lunsj. For the "Smash" column, we generated random numbers using the UNIFORM function between 190 and 250 to simulate the demand for Smash.
- In the "Kvikk Lunsj" column, we generated random numbers using the POISSON function with a mean of 85 to simulate the demand for Kvik Lunsj.

https://docs.google.com/spreadsheets/d/1IcC3rE_Wfs58QaOt-ZJX0XbGVKZ6VZ5C/edit?usp=sharing&ouid=113640976188927998607&rtpof=true&sd=true

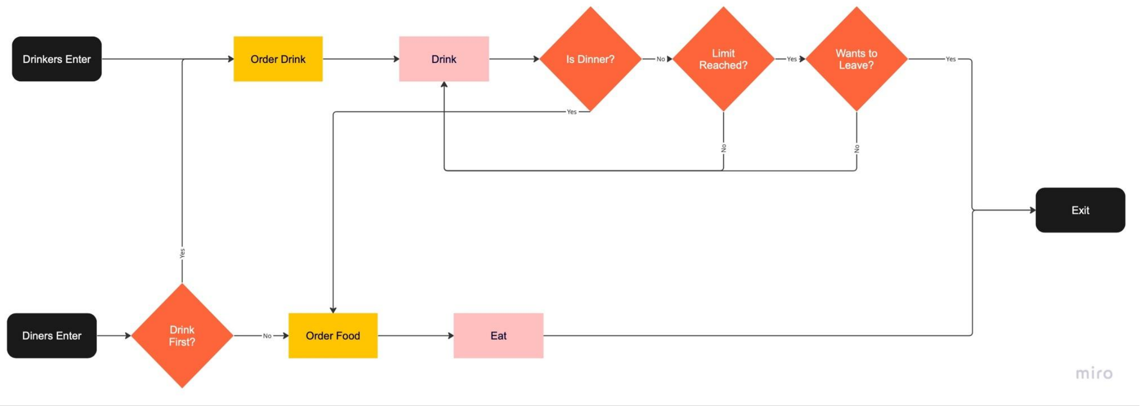
Appendix II: Queen Vic Simulation

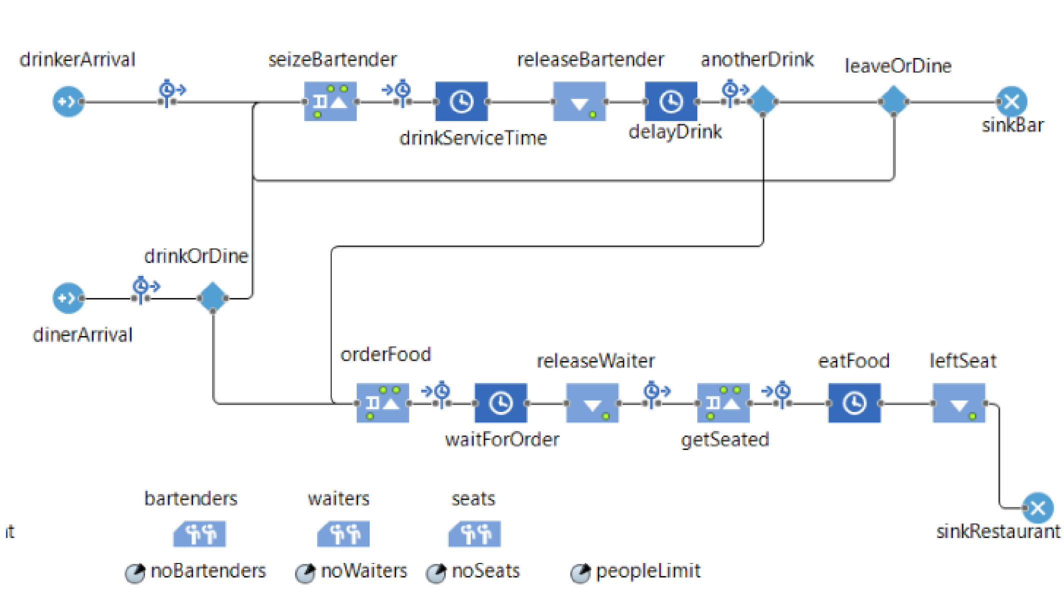
I. Figures

Optimization result

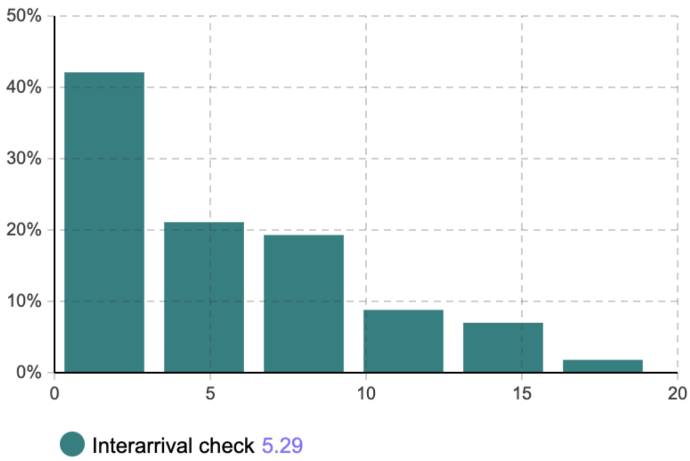
	Current	Best
Iterations completed:	270 infeasible	80
Replications:	20	20
Objective: ↓	28	25
Parameters		Copy best
noSeats	**20**	**20**
noBartenders	**2**	**3**
noWaiters	**6**	**2**
peopleLimit	100	100

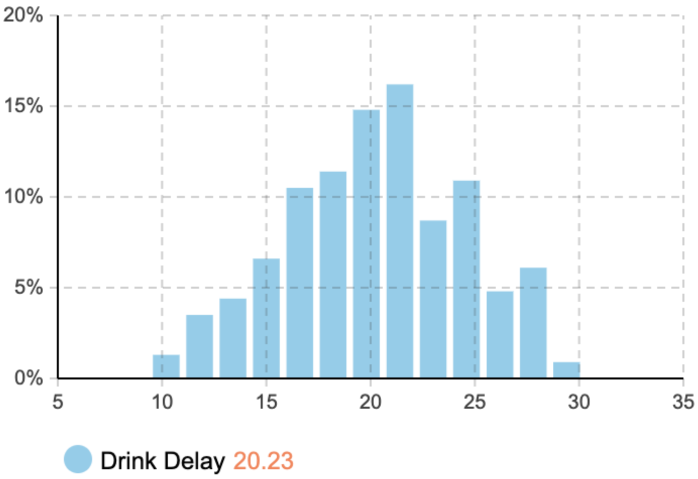
Scheme

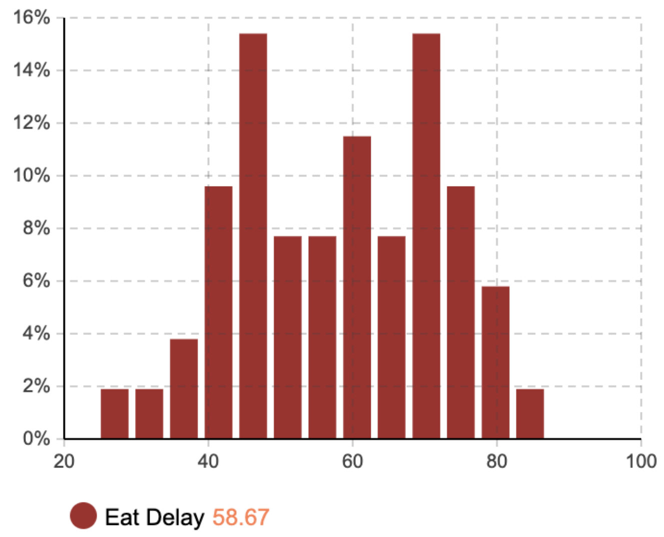


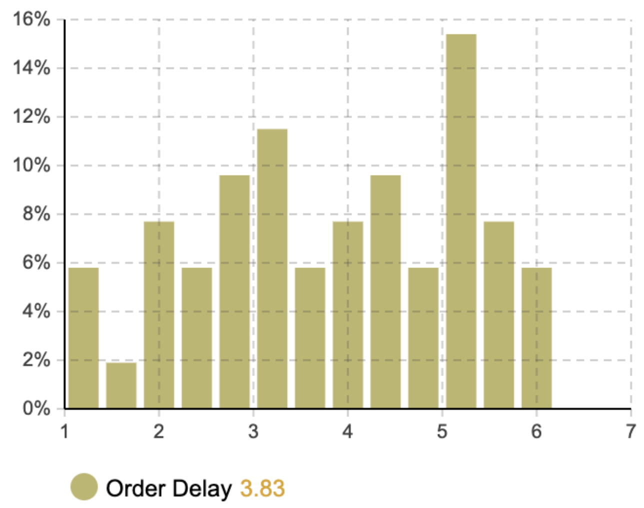

drinkerArrival
seizeBartender
releaseBartender
anotherDrink
leaveOrDine
sinkBar
drinkServiceTime
delayDrink
drinkOrDine
dinerArrival
orderFood
releaseWaiter
eatFood
leftSeat
waitForOrder
getSeated
sinkRestaurant
bartenders
waiters
seats
noBartenders
noWaiters
noSeats
peopleLimit

Validation


50%
40%
30%
20%
10%
0%
0
5
10
15
20
Interarrival check 5.29


20%
15%
10%
5%
0%
5
10
15
20
25
30
35
Drink Delay 20.23

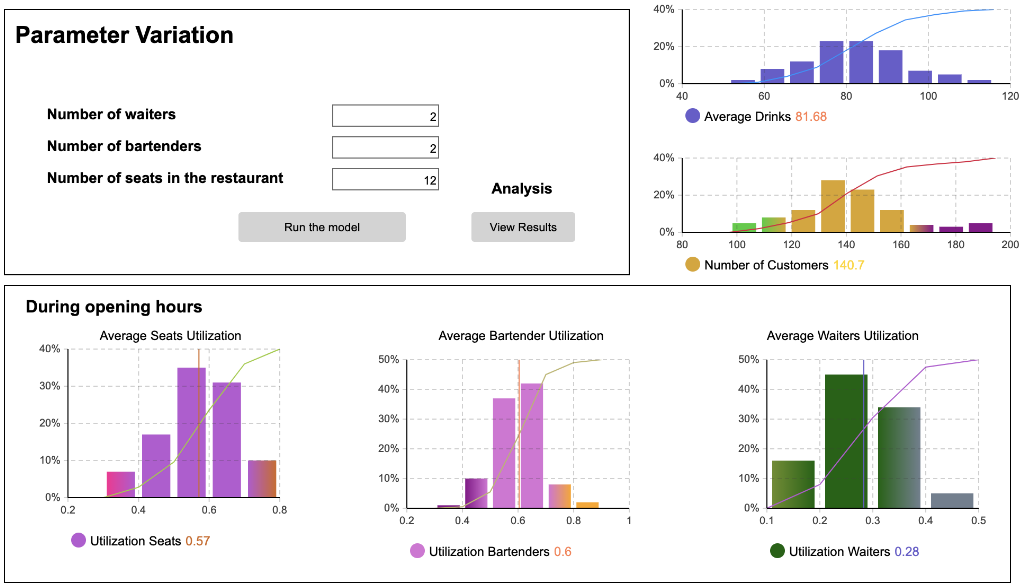

16%
14%
12%
10%
8%
6%
4%
2%
0%
20
40
60
80
100
Eat Delay 58.67


16%
14%
12%
10%
8%
6%
4%
2%
0%
1
2
3
4
5
6
7
Order Delay 3.83

Parameter variation

Parameter Variation
Number of waiters
2
Number of bartenders
3
Number of seats in the restaurant
20
Analysis
Run the model
View Results
Average Drinks 81.33
Number of Customers 136.72
During opening hours
Average Seats Utilization
Utilization Seats 0.38
Average Bartender Utilization
Utilization Bartenders 0.4
Average Waiters Utilization
Utilization Waiters 0.28

Average Seat Wait Time
Seat Wait Time 0.72
> 10 minutes 208 (4%)
< 10 minutes 5,391 (96%)
Average Drink Wait Time
Drink Wait Time 0.31
> 5 minutes 75 (0%)
< 5 minutes 19,471 (100%)
Average Order Wait Time
Order Time 1
> 10 minutes 37 (1%)
< 10 minutes 5,391 (99%)
Utilization of Seats
Utilization of Bartenders
Utilization of Waiters



II. Codes

Calculating wait times

```
agent.waitTime= time()-agent.arrivalTime;
traceln(agent.waitTime);
if (agent.waitTime >5){
morethan5_Bar++;
}
else{
```

```
lessthan5_Bar++;
}
updateRatio(1);
stat_drink_Wait.add(agent.waitTime);
```

Calculation - Counter for wait times above 5 minutes

```
agent.waitTime= time()-agent.arrivalTime;
traceln(agent.waitTime);
if (agent.waitTime >5){
morethan5_Bar++; }
else{ lessthan5_Bar++; }
updateRatio(1);
stat_drink_Wait.add(agent.waitTime);
```

DinerArrivalEvent

```
if (allowDrinkerArrivals) {
drinkerArrival.inject(1); // Inject one drinker agent
 }
double arrivalRate = 1.0 / 7.0;
double interArrivalTime = exponential(arrivalRate);
drinkerArrivalEvent.restart(interArrivalTime); // Restart event with the next
inter-arrival time
```

DrinkerArrivalEvent

```
if (allowDinerArrivals) {
  int groupDistribution = groupDistribution();
  groupSize = groupDistribution;
  int groupID = 1;
  for (int i = 0; i < groupDistribution; i++) {
        dinerArrival.inject(1); // Inject one diner agent
  }
double arrivalTime = (1.0/12.0);
double interArrivalTime = exponential(arrivalTime); //double
interArrivalTime = triangular(5,12,17); dinerArrivalEvent.restart(interArrivalTime); //
Restart event with the next inter-arrival time groupID++; }
```

III. KPI Measures

KPI	Low 95%	Optimized Average	High 95%
% Drinkers wait < 5 minutes	-	99%	-

% Seat Wait < 10 minutes	-	95%	-
% Order Wait < 10 minutes	-	100%	-
Average Seat Wait Time	0.552	0.976	1.4
Average Order Wait Time	0.805	0.92	1.035
Average Drink Wait Time	0.324	0.363	.402
Average Bartender Utilization	39.9%	41.2%	42.5%
Average Waiter Utilization	26.3%	27.6%	28.9%
Average Seat Utilization	34.9%	36.4%	37.9%