Prediction of queue waiting times

Oriol Prat Font

June 18th, 2023

Abstract—Queuing theory is a branch of mathematics that studies, analyses and predicts the behaviour of queues using queuing models. Even though queuing models were originated in the early 1900s, they are not widely used in modern times. This project uses Poisson and exponential distributions together with other probability techniques to implement a queuing model that can predict the behaviour of queues and estimate the expected customer waiting times and their lengths.
The project also involves the generation of plots for representing and analysing the data. The output data is then exported and uploaded to a dashboard in order to provide visualizations and get crucial insights. Furthermore, an alarm is integrated to the dashboard so it sets off when the queues exceed a previously defined limit. Finally, different alternatives to the current queuing strategy are proposed to the park's management in order to reduce the waiting times or find other ways to enhance the visitor's satisfaction.

Python is used to implement the queuing model and the simulations are made in the discrete-event simulation framework SimPy, that is based on Python. The plots are also generated in Python, using the library Matplotlib. On the other hand, the dashboard used to visualize the data is created with Power BI and the alarm is connected to this dashboard.

Index Terms — behaviour of a queue, discrete event, exponential distribution, Matplotlib, monitorization, M/M/1, Poisson distribution, Power BI, Python, SimPy

1 Introduction

AMUSEMENT parks are popular destinations for people seeking entertainment and adventure. These parks offer a wide range of attractions, from breath-taking roller-coasters to family rides. However, there's something that frustrates everyone: long queues. Queues can lead to a negative experience of the park and can cause discomfort to the visitors.

This project focuses on a big amusement park that has this exact problem. The understanding of the behaviour of its queues is needed in order to find alternatives that can provide the fast services visitors demand. Understanding how queues work and being able to predict them can optimize processes and increase the visitor's satisfaction.

The main objective of the project is to reduce the wait times of the park's queues by making accurate predictions in simulations using a queuing model and different probability techniques. Additionally, plots and charts are designed and implemented to easily analyse the queues. Furthermore, by monitoring the system using dashboards, the behaviour of the queue can be understood and, therefore, different options for improving the queuing system can be contemplated.

- *Contact e-mail: Oriol.PratF@autonoma.cat*
- *Project tutored by Antonio Espinosa Morales (Departament d'Arquitectura de Computadors i Sistemes Operatius)*
- *Course 2022/23*

2 State of the art

At present, companies from all kinds of sectors have already designed queuing models and queue management systems. Their objective is to optimize their businesses, and queuing theory can contribute to it because it can provide shorter waiting times, which lead to a higher customer satisfaction.

Queuing theory has been studied throughout the last few years, which has contributed to the creation of diverse technologies used to optimize queues. These first technologies date back to the 1970s, where physical and electronic methods such as barriers and tickets were first developed. Later on, big industries like healthcare and retail began using these technologies, which lead to an improvement in queuing management systems by implementing real-time data and customer feedback tools.

Today, queue management systems are used in mobile apps, virtual queues, and even contactless payments. Additionally, some companies are also starting to implement machine learning algorithms to predict customer behaviour and long waiting times in the near future.

In recent years, there has been significant research on queueing models in amusement parks, with the aim of improving queue management and enhancing visitor experience. This specially includes models such as M/M/1 and M/M/k, to predict queue performance based on different factors, like ride capacity, arrival rates, and service times. Research has also focused on the use of simulation and

analytical models to evaluate the effectiveness of different strategies for improving queue performance, such as single-rider lines, virtual queues [1], and express passes. Machine learning and artificial intelligence are also emerging as a promising approach in this industry.

However, there are still challenges in applying queuing models to amusement parks, such as the dynamic nature of the park environment, the complexity of its queue behaviour, and the variability of demand. Therefore, more research in this field is required to further develop and optimise queueing models in this sector, with the ultimate goal of giving visitors a better experience and increasing the park's income.

As new technologies emerge, queuing theory will become more and more valuable. A technology that can benefit from queuing models and simulations is self-driving vehicles. With autonomous cars becoming more and more common, queuing models can be used to optimize traffic flow and minimize waits during congestions. Additionally, going further, in a future world where all vehicles are autonomous, queuing models could completely get rid of traffic congestions.

3 Objectives

After defining and analysing the problem to be solved, a series of objectives are outlined to successfully conclude this project:

- Accurately predict queue waiting times.
- Understand the queue's behaviour using plots.
- Monitor the system with dashboards.
- Offer support for decision making to reduce waiting times or make queues more enjoyable.

The proposal to achieve them begins with the implementation of a queuing model programmed with Python [2], which is an easy-to-use programming language that offers a wide range of libraries such as Pandas [3] for data manipulation and NumPy for numerical computing.

The simulation library used to create the model is SimPy [4]. This discrete-event simulation library is really useful for simulating real-life events in Python. It can easily model active components such as customers and servers, which are required in this project. There are a lot of projects that use this library, some of them being related to queueing theory and queueing systems [5].

The model created with SimPy is a variation of the M/M/1 queuing model [6], which is widely used to analyse the behaviour of single-server queues. This model assumes that customer arrivals follow a Poisson distribution, which means users arrive randomly and independently over time with an average rate λ. The service times for each customer are distributed exponentially, with an average rate μ, and there is a single server in the system, which means that only one customer can be served at a time.

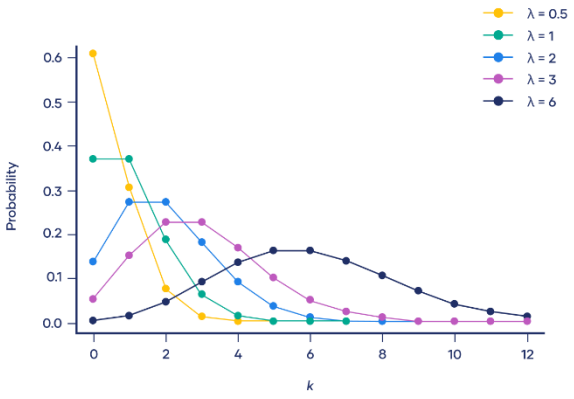

Figure 1: Poisson distribution

As previously mentioned, the model used in this project shares similarities with the M/M/1 queuing model. However, it incorporates a different service time distribution. As the service times in an amusement park represent the time in which the attraction is operational and running, I have modified the model to make them constant through all the attractions, instead of having an exponential distribution. This adjustment enhances the accuracy of the model in simulating amusement park scenarios and accurately represents real life attraction queues.

The next step, once the model is implemented, is to create plots and charts using the output data from the simulations. This is also made in Python using the library Matplotlib, which provides a wide range of visualization options and customizations while being easy to use and supporting other libraries [7]. Most of these plots use queuing and service times, but they are also useful to visualize the number of people in queue at any time. All these visualizations are fundamental in order to better comprehend and easily analyse the queues.

Additionally, the data from the simulations can be visualized through a dashboard created with Power BI [8], which is widely recognized as one of the best tools for data visualization and analysis, making it the perfect choice for the most visual part of the project. This tool stands out due to its easily connection to Excel spreadsheets, the user-friendly and interactive dashboards it provides and the availability to use real-time data. This last feature is not applicable to this project at the moment but the park is looking forward to implementing other ways of obtaining data from the queues in order to be able to get real-time

information about the number of people waiting in each ride.

The dashboard designed in Power BI provides crucial information from the waiting times throughout the day and from all the attractions in the park. The total number of people waiting in all the attraction queues is also displayed next to a limit that will be defined by the park's managers. This last chart will be really useful in the future, once the park implements a way of obtaining real-time data of the queues, because it will provide real-time information using the Power BI dashboard. Furthermore, alarms will also be sent to the park's staff when the queues exceed this limit.

Finally, different alternatives to the current queue management are provided to the park in order to try to reduce the waiting times, manage them more efficiently or find other ways to make the visitor's experience better.

4 Methodology and Planning

The methodology used to complete this project is based in the Scrum framework. As only one person is actively involved in the project, this framework has been adapted to suit the needs [9].

In the project, the Scrum responsibilities of product owner, scrum master and development team are slightly modified so that they can all be assigned to a single person.

The agile methodology used is based in sprints. The longer tasks have been divided in more than one sprint by separating the task in smaller parts. The sprints have lasted between one and two weeks each, depending on the complexity of the task. In addition, there have been weekly meetings with the project tutor. The tasks planning is shown in figure 2.

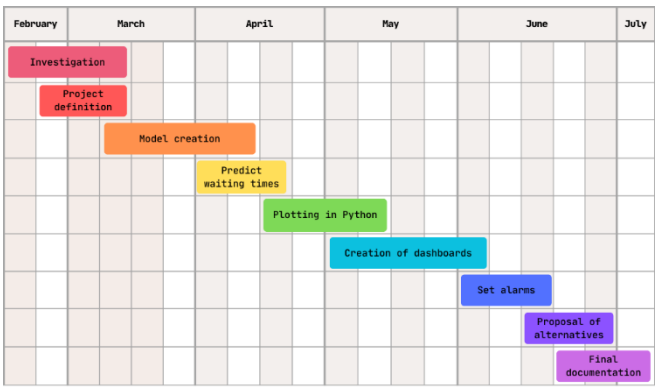

Figure 2: Planning of the project

As there's only one actively involved person in this project, no planning, tracking or control tools have been used. On the other hand, Microsoft Teams [10] and Outlook have been used as communication tools with the project tutor, although there have also been in-person meetings.

5 Development

In order to effectively achieve the objectives, a systematic and well-defined approach has been designed. This approach consists of a series of steps that must be executed following a specific order. This section presents each of these steps in detail, providing a comprehensive understanding of the proposed methodology.

5.1 Creation of the model

The first thing needed to start implementing the model was the data, so a dataset containing the data of 50 attractions from the amusement park was created. This data contains the number of people who join the queue of each attraction every five minutes, being the columns the time of the day and each row an attraction in the park. The columns begin at 10:00 and end at 19:55 as this is the park's opening schedule.

The model has been created in Python using the simulation framework SimPy. This model consists of a Queue Simulation class that receives two variables: the number of users and the arrival rate. This class has a function called *run()* that loops over each of the users and processes them in the simulated queue.

It's important to acknowledge that, in order to make the simulations as similar as possible to a real attraction, the service time of each attraction isn't computed using the exponential distribution as in the M/M/1 queuing model because the time of service in a park is the running time of each ride, which is constant. Therefore, the service time in the queuing model is a constant value.

The simulated queue follows the same process as a real amusement park's queue:

1. First of all, the arrival time of the user is generated using Poisson distribution and the arrival rate defined in the class.
2. The user then waits for the simulation to get to this *Arrival time*.
3. When this instance arrives, the user is placed in the last position of the queue and it waits until the server is available and it is the next in line, with the server being unavailable referring to it attending other users in that moment.
4. When the server is available and the user is the next in queue, it will be attended. This instance is defined as the *Queue end time*, which will also be the *Service start time*.

5. Once the user starts being attended, it will stay there until the service time is completed. That instance will be the *Departure time*.
6. At that moment the user will depart and the next user in line will start the service.

A flowchart of this process is shown in figure 3.

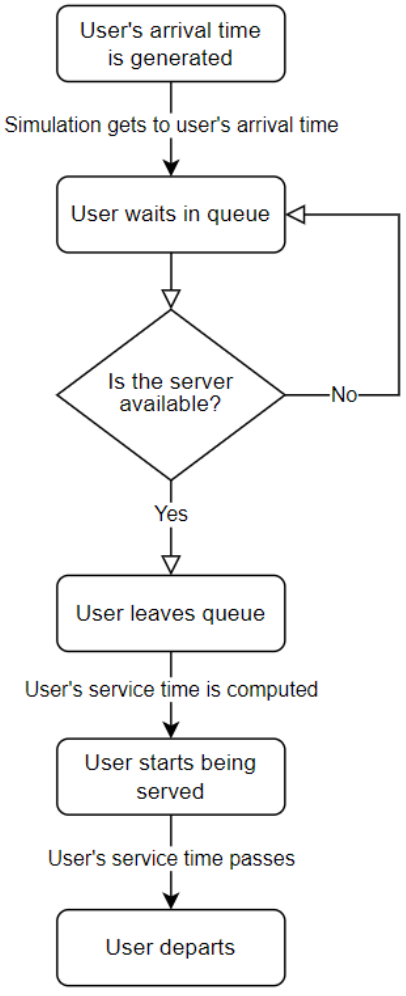

Figure 3: Simulated queue's flowchart

Once a user departs, its simulation times are saved in a Pandas dataframe where the rows refer to each of the users in the simulation. Later on, the *Time in queue* and *Service time* variables are computed using the following formulas:

$$Time\ in\ queue = Queue\ end\ time - Arrival\ time$$

$$Service\ time = Departure\ time - Service\ start\ time$$

These two variables are also saved in the dataframe with the rest of the simulation times. When all the users in the queue have been served, the simulation ends and the Queue Simulation class returns this dataframe.

Four different input spreadsheets have been designed, each one with a different range of people at every attraction. The queuing model is able to plot all of them to get a better understanding of how the queues react to more or less people waiting. These four models represent different situations that can happen in the park depending on the number of people who visit the park. The models are, from more to less people: special events, weekends, weekdays, and days with bad weather like rainy days. The purpose of having four different models is to give the park's management the possibility to use each one of them depending on their needs and which one they expect to be more similar to the reality in their park.

5.2 Prediction of wait times

Once the queuing model was implemented, the data was extracted from the dataset and loaded in a dataframe using Python's library Pandas. Each of the values in the dataframe refer to the number of people who get in a specific attraction in a range of 5 minutes from the park's opening schedule. These values are used in the Queue Simulation class as the number of users.

When all the simulation are completed and the *run()* function returns the dataframe of simulation times, the average of all the *Time in queue* times is computed. This value will be the average waiting time for a specific attraction in a specific hour and it is stored in a new dataframe. The average waiting times for all attractions and times are computed and stored in this dataframe as soon as each simulation is completed.

As a result, when all the simulations are done, the script exports the average waiting times for every hour and for each attraction. Additionally, the system prints the average wait time of all the values in the exported dataframe in order to get a better understanding of how the queues have performed in the simulations.

5.3 Plotting in Python

When all the simulations are completed and the wait times have been exported, the plotting part of the code begins. In this section a Gantt chart and a histogram of wait times are generated using Python's Matplotlib library [11].

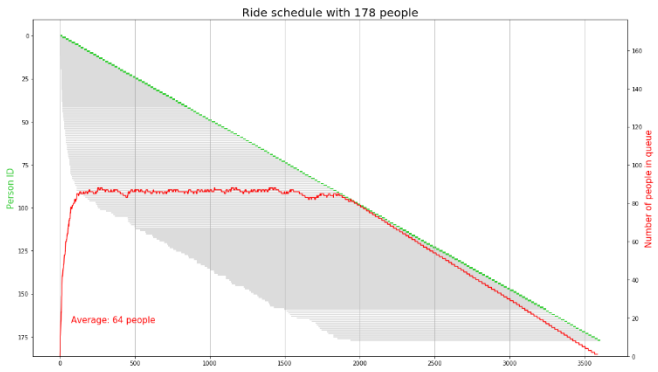

Figure 4: Gantt chart of a simulation

This chart represents the instances when every user in one of the simulations joins and waits in queue, starts being served and departs. The part where a user is waiting in the queue is shown with a grey bar, while the part where it is being served is painted in green. It can be seen that the arrival times, which are the beginning of the grey bars, show a Poisson distribution. On the other hand, the service times, which are the lengths of the green bars, are almost the same in all the instances because they represent the ride times, which are constant in the attraction.

The number of people waiting in the queue at any moment can also be seen in colour red. We can see that as the attraction opens its gates the number of people in queue starts rising until it reaches approximately 80 people waiting. At that point the number of people in queue maintains until every user in that simulation has joined the queue. Then it starts going down as the remaining rides are completed.

The histogram generated compares the difference on waiting times from an underloaded, a standard and an overloaded attraction respectively. The waiting times are plotted using a histogram in order to bring into focus how the waiting times vary depending on the flow of the queue and the number of users. There's also a red vertical line that indicates the average waiting time in each of the three attractions.

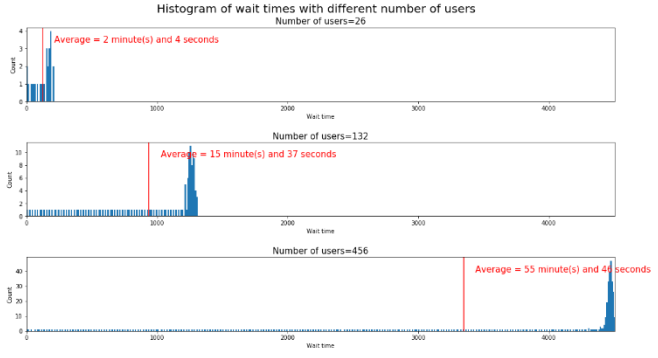

Figure 5: Histogram of waiting times

5.4 Power BI dashboard

Once all the simulations are done and all the charts are plotted using Matplotlib, the wait times of all the simulations are saved in a dataframe and exported to an Excel spreadsheet. Excel provides a versatile platform and is really useful and easy-to-use for manipulating and organizing data, which makes it the perfect tool for exporting the outputs of the model. The Excel book that is generated contains the average waiting times of each attraction for every hour it is operating, so it uses the same format as the original spreadsheet that is input in the model but with the waiting times for every hour instead of the number of people for every five minutes.

To make the most of the data obtained in the simulations, a Power BI dashboard has been designed. Power BI can provide interactive but at the same time intuitive visualizations and can be easily connect to Excel spreadsheets. These visualizations can help the company visualize the data obtained and get crucial insights about the queues' behaviour and their waiting times throughout the day.

As it can be seen in figure 6, the dashboard has been designed to have just one page where information from all the attractions is displayed. There are charts related to both waiting times and the number of people in the queues. The visualizations also show data throughout the day and between the two parks that conform the whole amusement park. Additionally, there's also a filter on the top right corner of the figure that can filter all the charts on the dashboard by this two parks.

Power BI also has the feature of filtering the charts depending on what you click in the dashboard. For example, it is possible to filter by one of the attractions or by a specific hour of the day by simply clicking on what you want to filter by. Once you click what you want to filter by, all the charts in the dashboard will automatically and instantly update, and by clicking again the filtering will be removed.

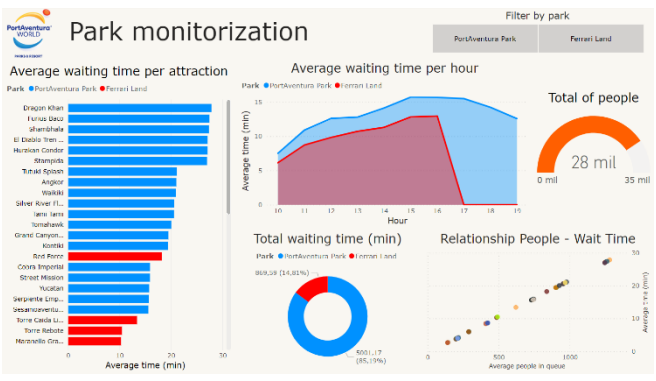

Figure 6: Power BI dashboard

5.5 Alarms

As mentioned previously, the implementation of alarms isn't a critical part of this project as these alarms are not useful to the park's management at the moment, but they are looking forward to work with real-time data in the near future using cameras and different recognition techniques in order to obtain the number of people waiting in each queue at every moment. Once this technology is implemented in the park the alarms will be really useful as they will be able to alert the park's management when a queue is exceeding a previously defined limit in real-time. This system will give the park's personnel the opportunity to take immediate action when queues become excessively long.

The use of real-time data in the park will allow the management to implement a lot of new technology that couldn't be used before and that will provide a whole new world of possibilities. Additionally, if the park exploits these possibilities, it will definitely improve the visitors experience and allow them to manage a lot better the queues.

The alarm can be accessed through Power BI service [12], a cloud-based platform that allows users to publish, share, collaborate and view dashboards. This platform provides multiple functionalities such as apps, datasets, automatic reports, and alarms. While not all types of visualizations support alarms, certain plots such as Gauge Charts, Line Charts, and Scatter Charts do offer alarm integration capabilities.

The alarm has been integrated in the dashboard's Gauge Chart, which can be seen in figure 14, as it represents the total of people in queue through all the attractions of the park. This value is compared to the maximum people expected to be in the park at the same time, which is currently set at 35.000 but can be modified later by the park's management. Keeping in mind the current dashboard doesn't work with real-time data, this is the best visualization for integrating an alarm as it allows the park's management to receive an alert at the end of the day if the sum of all the queues has exceeded a limit they had previously defined. This limit can be modified and even multiple alarms can be integrated in the same visualization with a different limit each. These features give the management more flexibility and allows them to have a better control of the park.

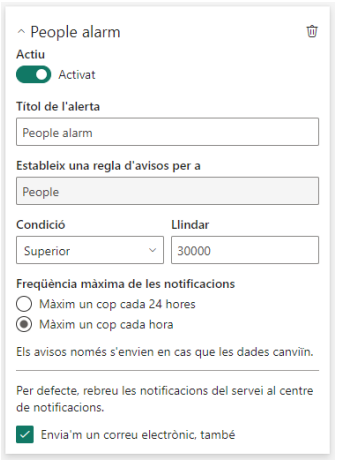

Figure 7: Creation of the alarm

As mentioned before, the alarms can send email notifications when the queues exceed the defined limits. These emails are automated and contain relevant information about the current state of the dashboard. They also contain a link that directly sends you to the dashboard.

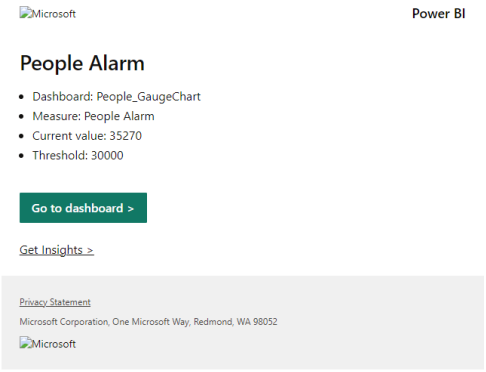

Figure 8: Automated email

5.6 Proposal of alternatives

After creating the queuing model, designing the Power BI dashboard, and analysing all the results obtained I was able to get a better understanding of the queues' behaviour in the parks. The insights obtained allowed me to identify the weak spots of the current queuing strategy and propose improvements as well as alternative queue management approaches. The following three strategies are suggested:

1. Enhance queue entertainment and engagement: the principal problem of queues in amusement parks is that visitors go there to have fun but spend hours in boring and tiring queues. My first proposal, and the simplest one, is to incorporate entertainment and engagement activities in the queues in order to make the wait more enjoyable for visitors. These may include interactive displays, games, and

themed elements. These activities not only reduce boredom but also make the perception of a shorter wait time and enhance visitor satisfaction.

To go a step further, once the park is able to work with real-time information about queues, a group of entertainers could go to the attractions with the longest queues by accessing real-time data. This way, the busiest attractions' queues would be a lot more enjoyable.

2. Implementation of a virtual queue system: this strategy allows visitors to reserve a spot in the most popular attractions through their phones. This way they don't have to stay physically in the queue for a long time. Instead, they can explore the park while they are virtually waiting and will receive a notification once their turn is approaching. This system improves the visitor experience as it allows them to do more activities throughout the day instead of spending hours just waiting.

   A month ago, the park announced this system will actually be implemented in one of its most popular attractions [13]. The app will be free and will feature a countdown to precisely know when it will be your turn.

3. Real-time recommendation for alternative queues: this system consists of implementing digital signs and screens in the most popular attractions. These screens will recommend to the visitors waiting in the busiest attractions some nearby and similar attractions with shorter queues. This way, people who don't want to wait for that long will be given alternatives which are close and similar to their current attraction. This system will help reduce the longest queues while relocating visitors to less popular attractions.

   In order to identify which are the best attractions to recommend, a matrix of similarity will be designed so that all attractions have a value that represents the similarity and proximity towards other attractions. The most similar attractions with the shortest queues will be the ones displayed in the digital screens for that ride.

   Ideally, this system should rely on real-time data from all attraction queues. However, in the absence of such data, recommendations can be based on information provided by employees working at nearby attractions with shorter queues. In this scenario, the attractions displayed on the digital screens would constantly need to be manually updated by the employees at the respective attraction.

6 Results

This section focuses on presenting and analysing the outcomes obtained throughout this project, going from the queuing model to the Power BI dashboard.

6.1 Results of the predictions and plots

The results of the model are really realistic as the waiting times are higher in the most popular attractions and lower in the least. The hour of the day is also an important variable, as in the first and last hours there's less people in the queues and, therefore, the waiting times are shorter. This can be seen in figure 9, which is the output for one of the executions of the queuing model coloured depending on the waiting times.

AttractionName	Park	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00	19:00
Angkor	PortAventura Park	17,45	21,66	28,45	30,1	31,46	37,66	31,6	39,07	32,04	31,33
Armadillos	PortAventura Park	3,37	4,91	6	6,2	6,28	5,78	7,8	8,56	5,9	5,01
Buffalo Rodeo	PortAventura Park	3,5	3,99	5,5	6,14	5,84	7,31	7,81	8,06	5,9	5,29
Canoes	PortAventura Park	7,36	9,95	11,65	10,34	13,38	12,42	15,28	13,43	13,24	11,07
Carousel	PortAventura Park	2,86	5,53	5,09	5,02	6,99	6,02	7,44	6,54	5,28	4,63
Cobra Imperial	PortAventura Park	11,58	18,53	20,97	20,85	26,28	31,57	29,88	27,15	23,81	20,91
Coco Piloto	PortAventura Park	3,32	3,88	5,78	5,66	5,85	7,01	5,33	5,98	5,93	6,16
Crazy Barrels	PortAventura Park	3,23	4,7	5,11	5,08	5,99	7,61	6,99	6,19	7	5,31
Dragon Khan	PortAventura Park	21,43	36,91	37,77	37,66	40,39	38,65	46,36	43,98	39,01	37,73
El Diablo Tren de la Mina	PortAventura Park	23,73	30,5	37,22	41,75	40,48	43,81	40,81	45,87	45,94	34,68
El Salto de Blas	PortAventura Park	8,42	10,9	11,12	11,37	12,19	14,84	13,88	13,79	12,68	12,19
Estacio del Nord	PortAventura Park	2,72	4,21	5,73	5,45	6,03	6,8	6,55	6,56	5,04	6,55
Furius Baco	PortAventura Park	23,37	34,24	37,43	31,34	38,19	50,88	50,36	42,15	46,7	35,39
Grand Canyon Rapids	PortAventura Park	16,09	22,29	25,85	24,69	29,07	31,96	33,03	34,55	26,54	27,83
Hurakan Condor	PortAventura Park	19,63	30,22	34,59	34,65	45,36	55,79	42,42	43,04	41,82	32,54
Kiddie Dragons	PortAventura Park	2,97	4,37	4,73	5,33	5,91	7,77	6,81	6,81	6,03	6,32
Kontiki	PortAventura Park	16,39	22,51	29,45	27,01	28,61	30,9	29,53	38,74	28,02	23,51
La Granja de Elmo	PortAventura Park	2,08	5,38	5,46	5,52	6,72	6,77	6,72	6,83	5,96	5,99
Los Potrillos	PortAventura Park	2,87	5,41	4,86	5,19	6,03	6,96	6,13	7,38	6,15	4,82
Magic Fish	PortAventura Park	3,63	3,95	3,94	4,61	6,35	7,35	6,19	6,4	6,12	4,75
Mariposas Saltarinas	PortAventura Park	2,93	4,53	5,32	5,29	5,06	7,17	6,68	7,03	5,43	4,62
Penitence Station	PortAventura Park	3,21	4,13	6,17	6,23	5,83	7,6	7,26	7,22	5,12	4,91
Port de la Drassana	PortAventura Park	3,28	4,69	5,33	6,25	6,49	6,45	6,93	6,85	6,7	5,26
Serpiente Emplumada	PortAventura Park	11,07	17,86	22,34	19,27	21,39	27,24	23,98	24,32	21,31	21,37
Sesamoaventura Station	PortAventura Park	12,69	18,14	21,18	21,28	20,73	26,12	26,15	25,64	26,17	20,68
Shambhala	PortAventura Park	21,72	34,17	38,3	39,47	40,95	48,36	41,7	43,05	35,91	41,38
Silver River Flume	PortAventura Park	15,62	25,5	27,62	29,35	32,62	33,5	35,19	36,58	26,32	27,33
Stampida	PortAventura Park	23,36	35,65	37,64	39,94	38,58	40,5	46,29	43,17	39,39	35,65
Street Mission	PortAventura Park	11,09	19,11	22,93	20,02	25,26	22,92	30,47	26,14	23,8	22,02
Tami Tami	PortAventura Park	15,44	25,16	28,17	28,89	33,67	38,74	32,07	33,95	31,59	26,23
Tea Cups	PortAventura Park	3,05	4,82	6,06	5,14	6,8	7,04	6,96	6,14	6,72	5,22
Tomahawk	PortAventura Park	16,34	20,5	28,41	29,27	30,73	33,78	33,53	32,98	33,4	24,8
Tutuki Splash	PortAventura Park	16,26	24,6	28,11	29,7	30,51	37,67	33,44	34,7	32,64	27,6
Volpaiute	PortAventura Park	3,05	5,31	5,65	4,27	6,61	6,08	7,39	7,13	5,43	4,86
Waikiki	PortAventura Park	18,09	23,62	24,66	28,13	29,7	38,87	38,76	31,19	32,05	28,66
Waitan Port	PortAventura Park	5,9	9,67	11,47	10,37	12,57	14,68	14,45	15,01	13,07	12,88
Wild Buffalos	PortAventura Park	7,33	9,49	10,63	14,15	14,53	16,06	14,18	14,24	13,59	11,49
Yucatan	PortAventura Park	14,48	18,32	20,85	24,1	23,89	29,21	26,15	21,44	22,63	21,28

Figure 9: Coloured output spreadsheet of the model

As the actual data is not real and it had to be created synthetically, it is evident that the waiting times have a big dependency on the attraction, as the best rides always have the longest queues. On the other hand, this is actually what happens in real parks, as there are a few attractions that everybody wants to ride and are always full. If we take a closer look at the numbers, we can see that they are all between approximately 2 and 50 (minutes). According to these results, we can imagine this would be a quiet busy day at the park.

Now, let's take a look at one of these simulations. We can analyse each simulation by plotting its Gantt chart. For example, in figure 4 we have a simulation with a total of 178 people in the queue. In this case the number of people waiting maintains in approximately 80 users along most of the simulation, until all the people have joined the queue

and the remaining rides are completed, where we can see that the number of people waiting is constantly getting lower.

This could represent that there's a limit where people would consider that the queue is too long and it's not worth it to wait for that much time, so they go to other attractions or spend their time somewhere else along the park. These insights will be really useful later on for the proposal of alternatives and final conclusions.

If we move on to figure 5, which is the last plot made with Python's Matplotlib, we can see the comparisons between the most and least popular attractions in the simulations. The first plot represents the one with least people, the last one is the one with the most people, and the plot in the middle represents the attraction that is closer to the average waiting time of all the simulations made.

We can see in colour red that the waiting times go from an average of almost two minutes all the way to almost an hour in this case, having the average in about 11 minutes. These values come from the output file shown in figure 9 and can give us some ideas of if that day was really busy for the employees or if it was a quiet day.

As these plots are histograms, the bars represent the number of times each value in the x axis has appeared in the data. So, it is really obvious that the most repeated waiting times in these simulations are the higher ones. This can also be seen in figure 4 where, as mentioned before, the maximum number of people waiting at the same time is repeated through most of the simulation. This means that the arrival rate is really similar to the departure rate, or the rate in which people leave the queue to get in the rides. Once again, this demonstrates that it is possible to define a limit where people no longer feel like the attraction is worth all the wait time.

6.2 Results of the dashboard

Lastly, we will examine the results and visualizations of the Power BI dashboard. Starting with the column chart, it is a crucial visualization as it illustrates the average waiting times across all simulations for each attraction. By analysing this data, the company can get a better understanding of the current popularity of attractions. Moreover, all the charts in the dashboard use a colour scheme that categorizes attractions based on the specific park they belong to.

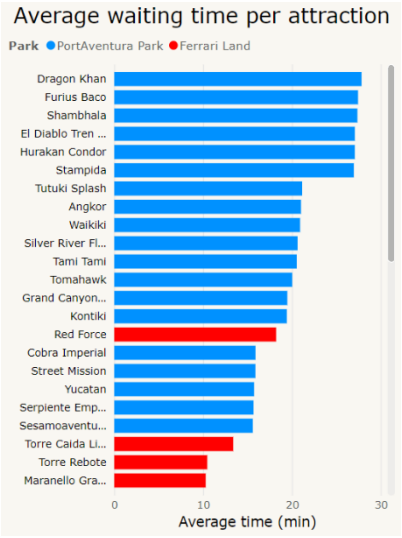

Figure 10: Dashboard's column chart

If we take a closer look at figure 10 it is obvious that most of the attractions with the longest queues belong to the same park, while only a few rides in the other park have that many people in their queues. Therefore, there is a big difference between either the management of both parks or the number of people who go to each park. We will need more information to identify which is the cause of this difference in waiting times.

The dashboard's area chart also visualizes the average waiting times, but this time in relation to the hour of the day. It is noticeable that, as well as in the previous chart, one of the parks has higher waiting times on average through the day. We can also appreciate that the waiting times drop to zero for the red park once it's 17:00. The reason to this is that this hour is the park's closing time and, therefore, all the data goes to zero at that moment. There's not a lot more to see in this figure except for the detail that the longest queues tend to form between 15:00 and 17:00. This is really valuable information for the park as the management can plan its resources and personnel's schedule based on this data.

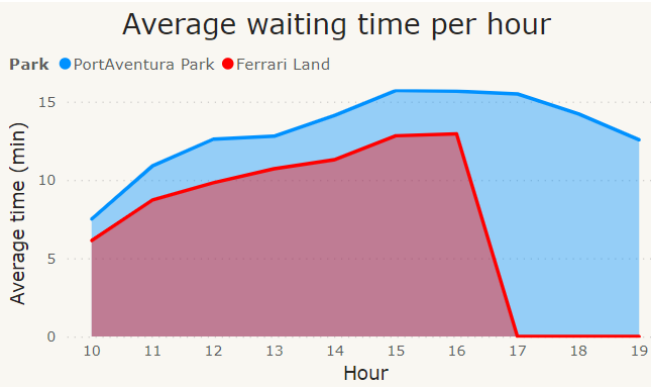

Figure 11: Dashboard's area chart

Moving on to the doughnut chart, this one is really simple while giving interesting information, as it shows the total number of minutes that people have spent their time waiting in a queue for all the simulations. This visualization gives information about the amount of people who go to each of the parks and it can also be useful for managing the number of employees in each park.

There is a big difference between the waiting times in each of the parks. This information helps us understand the cause of the difference in waiting times identified in figure 10. By seeing that more than 85% of the waiting time is accumulated in just one park it is obvious that this park is the most visited and has the most popular rides, while the park coloured in red has less attractions and, therefore, the total waiting time is lower.

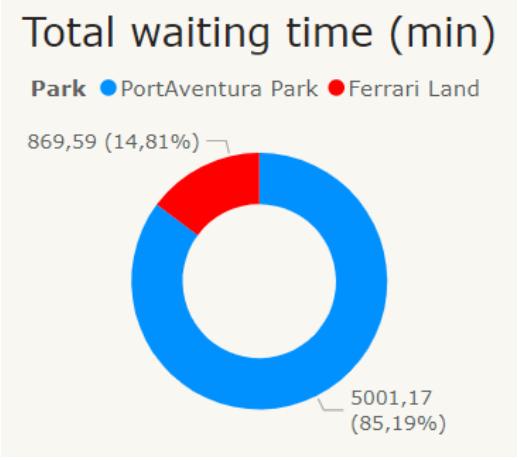

Figure 12: Dashboard's doughnut chart

The figure 13 shows the relation between the number of people who join a queue and the waiting time for all the attractions. The figure displays a really straight line. The cause to this is that the data used in the simulations is synthetic data. If the data implemented in the dashboard was real, the dots in this plot may still be in a straight-line-like distribution but it wouldn't be that obvious, as some rides would have more waiting time for less people and vice versa.

By looking at the gridlines in the figure we can say that, approximately, for every 500 people who get in a queue there are 10 minutes of waiting time. Identifying this ratio is really important for the park as it allows the management to predict the queue length of their rides with just an approximate value of how many people will visit the park.

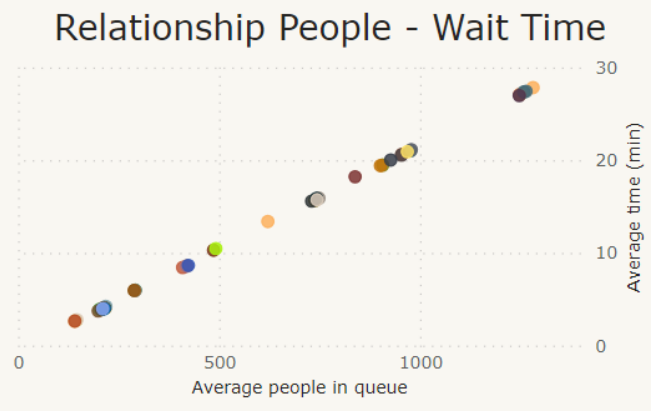

Figure 13: Relationship scatter plot

The last chart represents the average number of people waiting in the queues compared to a previously specified limit. The limit defined at the moment is just provisional and can be modified in the future as this value should be defined by the park's management.

If the number of people in the queues is higher than the limit defined by the park, it means that there is too many people waiting and the park's management needs to do something to prevent that number from rising even more. For this reason, an alarm has been created in this project. The alarm is activated when the limit previously defined is surpassed. This way, if in the future the park has the technology to work with real-time data, the employees won't have to constantly be looking at this chart. Instead, an email will be sent to them when the number of people in queue is higher than the limit.

Figure 14: Dashboard's gauge chart

7 Conclusions

The queuing model visibly works and the simulations generate realistic waiting times. By using the four models created, within just a single click and a few seconds the system generates representative plots of the simulations and an

output file with all the waiting times. Then, this file can be inserted into the Power BI dashboard to get more information and easily analyse the outputs. On the other hand, real data from the park can be used in the dashboard instead of simulation outputs. This will make the insights obtained from the visualizations a lot more useful and applicable to reality.

However, there's a few pain points about the systems developed during this project. The queuing model has been tested with synthetic data. As I didn't have real data to work with at any moment of this project the outputs obtained may not be as precise and similar to reality as they could have been. Additionally, if the structure of the real data's file isn't the same as the synthetic data the model may not work correctly.

As I don't have real waiting times of attractions in the park I don't have data to compare my results to, but considering the number of people in the synthetic data and the length of the queues in the simulations I believe the results obtained are close to reality, or at least as close to reality as I could with the few real information about the park and its rides I could obtain during this whole project.

Moving forward to the dashboard, I think it is really useful in order to better understand the data obtained from the simulations, but even better to analyse real data from the park. Again, if the park's management want to implement real data to the dashboard, it will only work correctly if this data has the same structure as the output file from the queuing model I have been using during these months.

This project moves the park's queues a step closer to technology as its current queue management system is really rudimentary and almost inexistent. I feel like the three proposals presented to reduce waiting times or make queues more enjoyable should really be considered, even if they currently can't be implemented, because they will help the park's management realize the potential they have in queuing systems and the importance of using real-time data. If I am able to make them notice the potential in the park I am more than satisfied with the outcome of this project.

Finally, I have realized that by using real-time data and designing a good queuing system with a well-thought queuing strategy behind, the park would gain a lot of visitors and the customer satisfaction would increase astronomically. It would be a relatively small investment that would totally pay off and definitely give more popularity and prestige to the park.

References

[1] Virtual queuing in amusement parks (2021). Attractions.io. [consulted: March 8, 2023]. Available on the Internet: https://attractions.io/learn/virtual-queuing-the-future-of-theme-park-experiences

[2] Welcome to Python (2014). Python Organization. [consulted: March 8, 2023]. Available on the Internet: https://www.python.org/

[3] Pandas documentation (2023). Pandas team. [consulted: April 7, 2023]. Available on the Internet: https://pandas.pydata.org/docs/

[4] SimPy overview (2016). Team SimPy. [consulted: March 5, 2023]. Available on the Internet: https://simpy.readthedocs.io/en/latest/

[5] Introduction to simulation with SimPy (2022). Darío Weitz. [consulted: March 12, 2023]. Available on the Internet: https://towardsdatascience.com/introduction-to-simulation-with-simpy-322606d4ba0c

[6] M/M/1 queuing system (2021). EventHelix. [consulted: March 6, 2023]. Available on the Internet: https://www.eventhelix.com/congestion-control/m-m-1/

[7] Matplotlib (2012). The Matplotlib development team. [consulted: April 6, 2023]. Available on the Internet: https://matplotlib.org/

[8] Power BI (2015). Microsoft. [consulted: March 12, 2023]. Available on the Internet: https://powerbi.microsoft.com/en-us/

[9] Adapting Agile Scrum methodology for individuals (2017). Lucidchart. [consulted: March 8, 2023]. Available on the Internet: https://www.lucidchart.com/blog/scrum-for-one

[10] Microsoft Teams (2020). Microsoft. [consulted: March 12, 2023]. Available on the Internet: https://www.microsoft.com/en-us/microsoft-teams/group-chat-software

[11] Gantt charts with Python's Matplotlib (2021). Thiago Carvalho. [consulted: April 2, 2023]. Available on the Internet: https://towardsdatascience.com/gantt-charts-with-pythons-matplotlib-395b7af72d72

[12] Power BI Service (2014). Microsoft. [consulted: June 7, 2023]. Available on the Internet: https://app.powerbi.com/home?experience=power-bi

[13] Implementation of a virtual queue (2023). Alejandro Alcolea. [consulted: June 9, 2023]. Available on the Internet: https://www.xatakamovil.com/aplicaciones/port-aventura-quiere-acabar-colas-fisicas-su-solucion-codigo-qr