CS166 First Project

Feedback and Grading

There’s several things I want to get feedback on.

1. Writing the report: 1500 words sound like a lot of words for something that is mostly coding and for me, self-explanatory. I would like to know how clearly I structure this assignment, is there anything I can improve in terms of structure/ clarity, or any other things that would be helpful to include in the assignment which I didn’t.
2. Data visualization. I’m just using a basic visualization because I’m mainly interested in the structuring of the code. I do think the visualization can be improved/ made interactive but don’t know how to do that given the limits of time. Any feedback on potential plots/ simulations/ resources etc. would be immensely helpful.

Overview

Queueing theory models after queues in the real world. From adjusting certain parameters, we can predict queue lengths and waiting time. This has important consequences especially on business decisions in order to provide adequate resources to lead to highest customer sastisfaction/ business efficiency while being frugal with the amount of resources needed.

In this project, the model simulates a bus route, and passengers’ average waiting time as they embark and disembark along the route. The bus route operates 24 hours a day, along a circular route with 15 different stops before going back to the

original. The task is to model different number of bus given the rate of arrival for each passengers to find the optimal number of buses that should be on this route.

Model Assumptions

There are a few given assumptions in the model.

1. Passengers arrival rate is constant and follow a Poisson distribution, and the time between consecutive passengers joining the queue follows an exponential distribution with a rate parameter of $\lambda = 1$.
2. Each passenger chooses randomly a destination at most 7 stops a way.
3. The bus has a maximum capacity of 130 passengers.
4. Time taken for the bus to travel between each stop follows a normal distribution where mean = 2 minutes and standard deviation = 0.5 minutes.
5. The time taken for passengers to disembark is distributed with mean = 0.03n and standard deviation = 0.01 * sqrt(n) where n is the number of passengers.
6. The time taken for passengers to embark is distributed with mean = 0.05n and standard deviation = 0.01 * sqrt(n) where n is the number of passengers.

Given the above information, there's several more things I assumed for my simulation.

7. I decided that the bus would start at different stops and this would be divided equally among different buses. For instance, if there's 2 buses, the first bus would start at stop 0 and the second bus would starts at stop 7. This way the buses are as spaced out as possible and would be more efficient with picking up customers.
8. The bus would start at 0 and start embarking and leaving the bus stop immediately. This means that if there's no passengers at 0, the bus would not pick up anyone. Since in the simulation, passenger arrives after a certain amount of time, the bus always leave the first stop at 0 and start travelling to the next.
9. There's no buffer time between disembarking, embarking and traveling to the next stop. So customer would disembark first, then immediately after start embarking, then immediately after bus would start traveling.

In addition, I made a few modifications, specifically in relation to point 5 and 6. Because it is a normal distribution dependent on n, we can write the normal distribution for the time to disembark 1 passenger = 0.03, and standard deviation =

0.0001. This is because $E(nX) = nE(X)$ and $\sigma(\sqrt{nX}) = 0.01\sqrt{n} \Rightarrow \sigma(\sqrt{X}) = 0.01$. This is reflected in the code structure later. I also use truncated normal distribution instead of normal distribution as all the values cannot be negative.

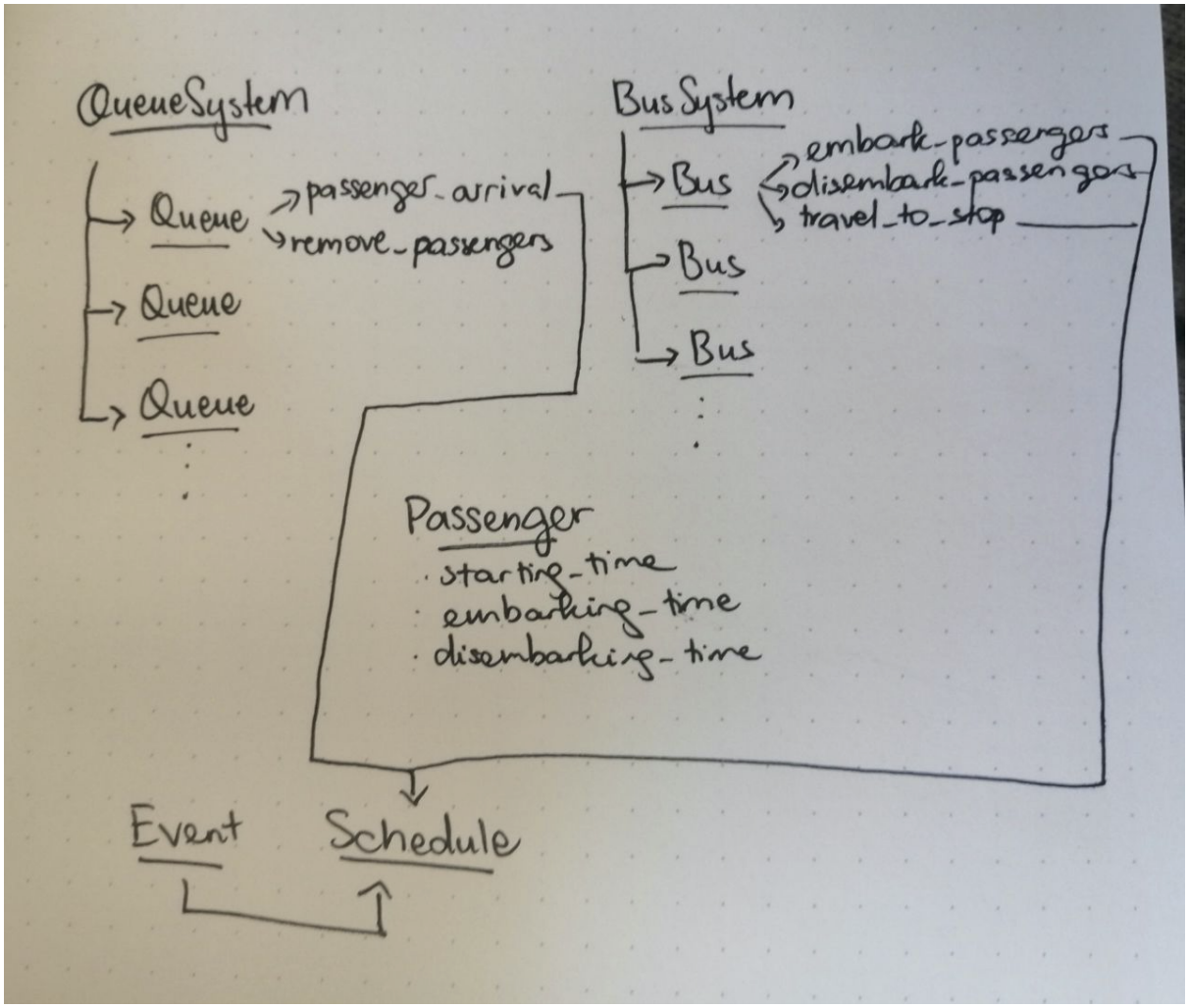
Theoretical Analysis

The queue follows the M/G/k structure. M represents the Poisson process for arrival of passengers, G represents the general distribution (with mean and variance) of the service time, and k represents the number of servers. k is our variable of interests because we want to change the number of queues (in this case, number of buses) and see how it influences waiting time and queue length.

There's no formula for M/G/k queueing structure, and exact numerical values remain unknown. Further, even if there was, it would be difficult to calculate the distribution of service time, in this case, it is the travelling time of every passenger, because it is further dependent on how many stops the passenger travels, whether there are many people embarking and disembarking, etc. Thus, we need to rely on simulation to get the results of this simulation. The simulation structure and its results is discussed below.

Simulation Structure

To simulate the bus route, there's 4 main classes interact with each other. The code is available here.



1. (Code cell 2) The `QueueSystem` class contains multiple `Queue` objects. Each queue is responsible for scheduling when a passenger arrives and when a passenger leaves a queue. It also stores the waiting time after each passenger leaves, maximum length, etc. which is useful for analysis further on.
2. (Code cell 3) The `BusSystem` class contains multiple `Bus` objects. Each bus is responsible for embarking passengers, disembarking passengers and travelling to next stop. It interacts with the queue when embarking passengers (as this is when the passenger leaves a queue).
3. (Code cell 4) The `Passenger` class represents a single passenger as they go through the system. This is mainly to store their starting time, embarking time and disembarking time, as well as starting and ending stops.
4. (Code cell 1) The `Schedule` class interact with both the `Bus` and `Queue` class by creating an `Event` each time a new action is created (arrival, embarking, etc.) and arranging the time in a priority queue such that the earliest event always happens first.

There's also a `run_simulation` function which creates all the object above and starting running a simulation. For instance, with 5 buses and letting the time run for 5 minutes, we get this result:

```
(Stop 0: 7 passengers
 Stop 1: 1 passengers
 Stop 2: 3 passengers
 Stop 3: 1 passengers
 Stop 4: 2 passengers
 Stop 5: 4 passengers
 Stop 6: 7 passengers
 Stop 7: 1 passengers
 Stop 8: 0 passengers
 Stop 9: 4 passengers
 Stop 10: 0 passengers
 Stop 11: 0 passengers
 Stop 12: 2 passengers
 Stop 13: 1 passengers
 Stop 14: 1 passengers,
 ------------------
 Bus system with 5 buses
 Bus id 0 currently at stop no. 2 with 2 passengers
 Bus id 1 currently at stop no. 5 with 0 passengers
 Bus id 2 currently at stop no. 9 with 4 passengers
 Bus id 3 currently at stop no. 11 with 6 passengers
 Bus id 4 currently at stop no. 14 with 1 passengers
 ------------------)
```

Simulation Results

I run each simulation for 100 times with an ending time of 2 hours (120 minutes). Note that even though this is insufficent (we should ideally run 60 * 24 minutes to represent a full day), the code cells already run for long that it would require much more computation to finish running a full day simulation. 100 minutes suffice because even in the worst scenario, when there's 1 bus, it takes 15 * 2 = 30 minutes on average for the bus to finish a round. My 3 variables of interests are average waiting time, maximum waiting time and maximum queue length. I then calculate the 95% confidence interval for all these 3 values for the number of buses ranging from 1 to 10. The result is shown below.

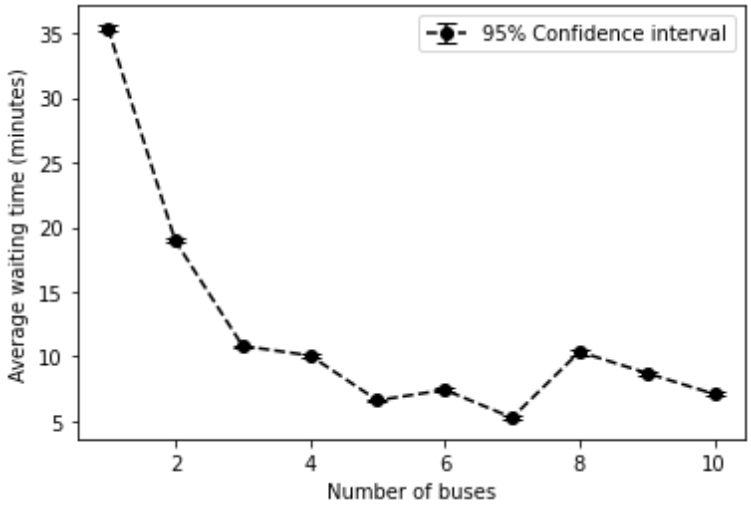


Figure 1: 95% confidence interval for average waiting time versus number of buses in 120 minutes after 100 simulations.

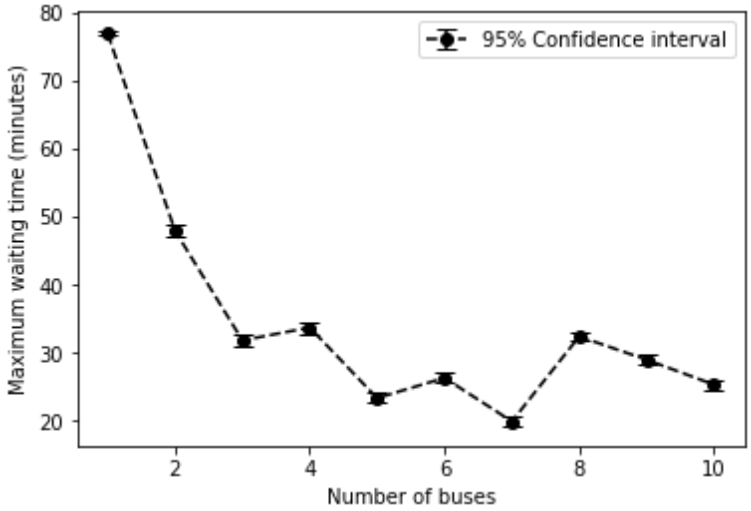


Figure 2: 95% confidence interval for maximum waiting time versus number of buses in 120 minutes after 100 simulations.

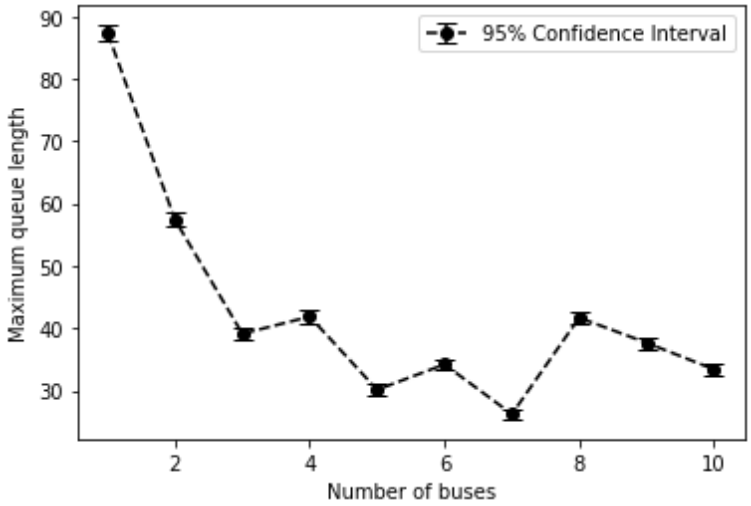
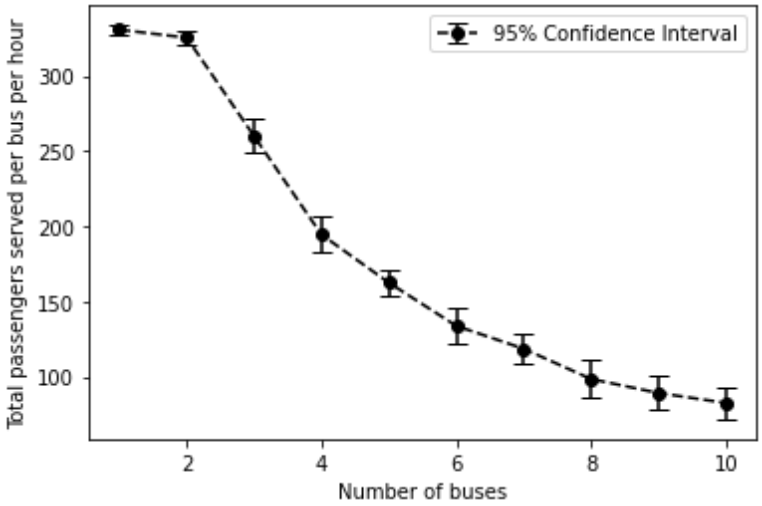


Figure 3: 95% confidence interval for maximum queue length versus number of bus in 120 minutes after 100 simulations.

We can see in all 3 graphs, there's a dramatic drop as the number of buses increase from 1 to 3, and the values start to plateau at around 5, and increase at around 8 buses. This is expected because we can imagine that the marginal benefit for each bus added gets smaller and smaller. The increase when there's 8 bus can be explained as each bus only differs one another by 1 stop (calculated by `len(no_of_stops)//no_of_buses`), so the bus are not spread out but start at stop 0, 1, 2, ..., 8 altogether, leaving passengers at stop 9 and below to accumulate.

We can also further see the optimal number of buses hover in between 3, 5 and 7. At these values, the average waiting time are 9.28, 5.30 and 4.16 minutes, maximum waiting times are 24.27, 15.54 and 14.76 minutes, and maximum queue lengths are 30.9 people, 21.4 people and 20.44 people respectively. We can see that while with 7 buses, all 3 values decrease pretty marginally compared to when there are 5 buses, thus it is probably not worth the increase cost of operating 2 other buses.

However, which is the optiomal value also depends on the cost of operating the bus versus the profit the bus has for each passengers. We can estimate that the cost of operating a bus to be $200/ hour in San Francisco (MacKennie, 2019), and each passenger is charged $3/ ride. Thus it would need to have roughly 67 passengers/ hour to break even. We then run the simulation fto find the total passengers served per bus per hour with different number of bus. Thus, we can calculate the number of passengers being served per bus to find what's the maximum number of passengers that each bus can serve.



As expected, we can see that the total number of passengers being served per hour per bus decrease the more buses there are, because that means the passengers are spread out between more buses. Looking at our 3 values of interest above (3 buses, 5 buses and 7 buses), the number of passengers being served decrease from 259.83 people to 162.41 people and 118.51 people respectively. This means a profit margin of 280%, 140% and 77% respectively. Given that the above cost of $200/ hour is only the cost of operations, without capital cost, we need more buffer to account for this extra cost, uncertainty, and so on.

The result for 3 values with standard deviation is printed here:

```
Number of buses: 3
Average waiting time: 11.43 minutes, standard deviation: 0.15 minutes
Max waiting time: 35.30 minutes, standard deviation: 1.07 minutes
Max queue length: 43.18 people, standard deviation: 1.43 people
Number passengers served/bus/hour: 259.83 people, standard deviation: 11.43 people

Number of buses: 5
Average waiting time: 7.29 minutes, standard deviation: 0.16 minutes
Max waiting time: 26.63 minutes, standard deviation: 0.82 minutes
Max queue length: 34.12 people, standard deviation: 1.04 people
Number passengers served/bus/hour: 162.41 people, standard deviation: 8.85 people

Number of buses: 7
Average waiting time: 5.94 minutes, standard deviation: 0.15 minutes
Max waiting time: 22.92 minutes, standard deviation: 0.84 minutes
Max queue length: 30.32 people, standard deviation: 0.99 people
Number passengers served/bus/hour: 118.51 people, standard deviation: 9.89 people
```

Closing Thoughts

In closing, it seems like 3 to 5 buses are a much better range, and the exact value would further need to depend on other considerations such as cost of operations, profit margins, uncertainty, etc. We also note in the assignment that constant rate of passenger arrival is probably not as accurate, and would need to modify accordingly. Nonetheless, this is a primitive attempt to find the number of buses that would be most optimal given the constraints above.

Acknowledgement

Thanks to Minh for helping me brainstorm and structure the code. Instruction taken from CS166 Assignment 1. Part of the code is taken from CS166 Session 3.

Learning Outcomes

- #Modeling: I have listed the variables, parameters and rules of the model, and explain why my simulation is accurate and interpret the results clearly.
- #PythonImplementation: I have used object-oriented programming to structure a working simulation with clear visualization of state and objects to make sure that the code functions.
- #CodeReadability: I have used accurate and clear variable names, include docstrings and comments in my code block, and followed the pep8 convention throughout the assignment.
- #TheoreticalAnalysis: I have explained the theoretical model and why it is not possible to calculate the mathematical results.
- #EmpiricalAnalysis: I have explained the results obtained from the simulation clearly both visually and in prose, with clear interpretations and good choice of variable measurements. Each simulation is run 100 times and the average value and confidence interval is displayed.
- #Professionalism: I have written a clear report accompanied by a well-organized Python notebook with clear variables and units of measurements, good captions, axis labels, etc.

References

MacKennie, C. (2019, January 31). *How Much Does It Cost to Purchase and Operate a Bus?* LiveAbout. https://www.liveabout.com/bus-cost-to-purchase-and-operate-2798845

M/G/k queue. (2021). In *Wikipedia*. https://en.wikipedia.org/w/index.php?title=M/G/k_queue&oldid=1061273789