Parking Lot Simulation and Analysis

Ryan Stevens
810318529
San Diego University Student

Haval Ahmed
815013661
San Diego University Student

Lauren Peppard
813614016
San Diego University Student

Geoff Crews
816165149
San Diego University Student

Sami Badra
815012738
San Diego University Student

Abstract
This paper provides documentation on a parking lot simulation used to determine the optimal number of parking spaces needed for a parking lot as a function of the arrival rate and exit rate. Before beginning this project, we hypothesized that the number of parking spots needed would be greater than or equal to the arrival rate of the cars multiplied by the average time a car spends parked plus the average time spent searching for a spot. We then constructed a simulation to test this hypothesis, which is documented in this report.

Keywords
M/M/1 Queue, Normal Distribution, Mass Density Function, Simulation, Parking Lot

1. Introduction
Our parking lot simulation is set up so a car enters into the entrance gate, waits in the queue if the server is busy, is let in after 60 seconds (or whatever the user decides it to be), and finds a parking spot at random. If the parking spot is already taken the car will then look for a different random parking spot. Even though in the real world most people park closer to their desired location we decided to make it look for a random spot because we do not know where outside the parking lot is each customer's desired location. after spending a random time in the parking lot, which is defined by a random number in between a high and low value that is obtained by the user or the default settings, the car will then go to the exit queue and when they are able to be helped from the exit server they will leave the system.

For the parking lot simulation we used a single server model approach. This is because we have two single servers in our system, the entrance gate and the exit gate. Our program is based off the work of a single server queue by Averill M. Law and his book Simulation modeling and analysis fifth edition.

2. Results of Our Study
In order to test our hypothesis, we ran our simulation multiple times, while experimenting with various different values for our input parameters such as the arrival rate, the park interval, and the service rate, while taking into account the fact that each car needs time to get from the entrance gate to its parking space and to get from its parking space to the exit gate. We found that decreasing the number of parking spaces would increase the average time a car spends searching for a spot because the parking lot would be more crowded, which would increase the chance of a car randomly selecting a spot that is already occupied.
This would also increase the average number of cars waiting in the entrance queue because the parking lot would fill up faster, which would make

it so that the cars trying to get in would have to wait in the entrance queue until a spot became available. Also, we found that decreasing the service rate or increasing the arrival rate would increase the average time spent in the queue because the server would become overloaded with customers waiting to receive service. In addition, we observed that increasing the average time a customer spends parked would also increase the average queue length at the entrance due to the fact that the lot would fill up faster, causing more customers to have to wait outside the lot in the entrance queue until a spot became available. Another important observation was that when setting the number of spots equal to a value that was much larger than what was needed, the majority of the spots remained empty throughout the simulation run, which would be very wasteful in a real life system.

2.2 Math Equations

(1) minimum number of spots needed = arrival rate*(average search time + average park time)

(2) parkTime = parkIntervalLow - (std::rand() % (static_cast<int>(parkIntervalHigh) - static_cast<int>(parkIntervalLow) + 1));
This equation is used to get a random park time greater than parkIntervalLow and less than parkIntervalHigh.

(3) 72 * (1 - randomNumber)
This is the mass density function used to get a random number between 0 and 72, not including 0.

(4) (spot index [+1 if left side]) * 1.6 + 10
This is the function used to add an approximate time that a car would spend getting to a parking spot and the time getting to the exit gate.

2.3 Drawbacks and Limitation of Simulation

Because we did not have real life data from parking lots, we based our numbers on what we thought would most accurately correlate to a real life parking lot. It would be better if we had real data on the type of parking lot and real data we could compare to. Each individual parking lot would have different arrival rates and park intervals based upon the lot's location. One example is a parking lot near the san Diego Convention center during the San Diego Comic Con which would have a long park time and the arrival rate would be very busy for a hour or two then level out. This would be much different than a parking lot for a grocery store or an apartment complex. This would also change the distribution function. Instead of cars coming evenly throughout the day they would instead come at different distributions throughout the day.
Due to this our study has a somewhat limited range on how accurate it is depending on the needs of who is running the simulation. It can be repurposed for different scenarios relatively easy though.

3. Conclusion

After conducting a careful analysis of our results, we concluded that in order to keep the average time in queue within an acceptable range, we must make sure that our service rate is greater than or equal to the arrival rate of cars into the system so that the cars will not be arriving faster than the server is able to serve them. We also concluded that, in order to minimize the average time spent in the queue, the number of parking spots in the system must be greater than or equal to the arrival rate multiplied by the average time a car spends parked plus the average time a car spends searching for a spot, so that the lot does not fill up too quickly and cause customers to have to wait outside in the queue. We also determined that although we want our number of spots to be slightly greater than the minimum value we calculated above, we do not want to have the number of spots to be too much larger than the minimum required because it would be wasteful.

4. Further Research

Test what happens if entry or exit gate has down time. For example, if the attendant has to take a bathroom break, or someone can't find their ticket. Evaluate how long it takes the system to recover and come back to equilibrium (if at all).

- Simulate a parking lot with more than 2 gates. Find out if the optimal lot size is dependent on the number of gates the lot has.A lot that has high turnover might be better suited to having an extra gate in lieu of a few extra spaces that that gate may occupy on the location available for a lot.

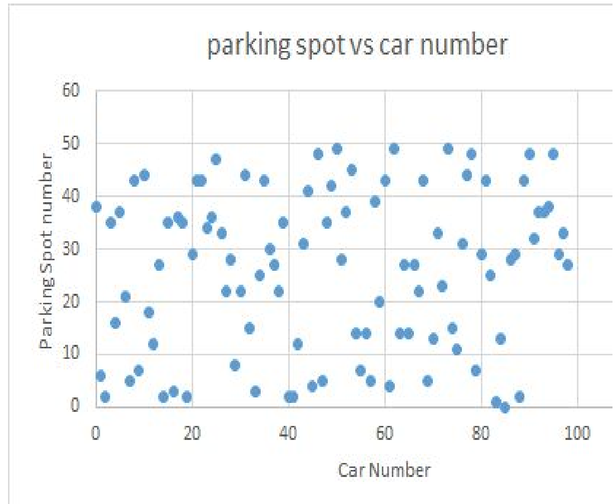
- Simulate a parking lot at a location that may have have peak periods. For example, a parking lot near SDSU may see cars trickle in, but then have peak departures of cars at times that may classes nearby get out. Find out how big the exit queue may grow during these peaks and how long it takes for them to come back to normal.

- Since the parking lot layout is pretty basic, the time searching for a spot and leaving is a simple function of the parking spot index chosen for the car (Equation 4). A better real world simulation might want to actually calculate the distance from the exit and arrival gates of the nearest empty spot to estimate a time someone is driving around in the lot getting to and from the spot.

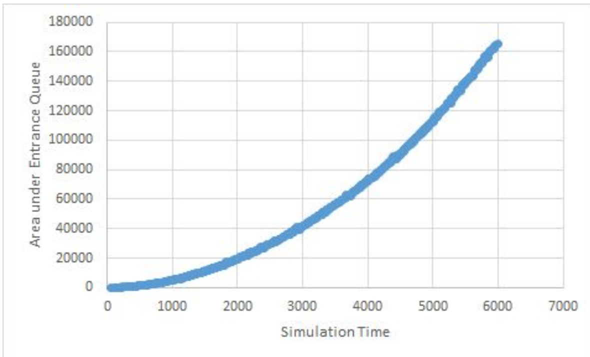
5. Assumptions

- No spots become unusable by bad parkers.
- Service rate of arrival/exit gates are constant.
- Gates are independent of each other.
- Arrival rate distribution independent of time. (No popular times of day)
- All drivers are equal in spot searching ability.

6. Graphs of Output



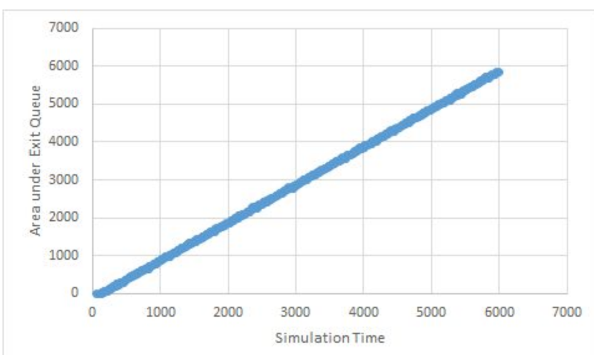
This is our parking lot scatter graph. This data plot gives us the parking spots over the duration of simulation time for the simulation. We got our parking spots by randomly assigning and see whether or not that spot was available for a car to enter. The randomization was done with a pseudo-random function that returned a value between 0 and our parking lot max capacity. As we analyzed our data, we saw a steady distribution of spots throughout the simulation. We didn't see any clusters of the same random number being assigned which means we had a wide variety of random spots which is great in a simulation when dealing with random inputs.



This graph represents the area under the entrance queue versus simulation time over the entire simulation. The area under the curve shows a steady increase over time. This is due to cars getting backed up as time goes on. It will grow exponentially since we have a normal distribution. In the real world though cars would not continously come in so it would not have this issue.

Parking Lot Layout

```
--------------------------
|****    <          <    ****|
|____        ____||____      ____|
|____        ____||____      ____|
|____   v    ____||____  ^   ____|
|____        ____||____      ____|
|____        ____||____      ____|
|____        ____||____      ____|
|____   v    ____||____  ^   ____|
|____        ____||____      ____|
|____        ____||____      ____|
|____        ____||____      ____|
|____   v    ____||____  ^   ____|
|____        ____||____      ____|
|____        ____||____      ____|
|____        ____||____      ____|
|            >        >          |
----    v    --------    ^    ----
     [Exit]         [Enter]
```



For the area under the exit queue versus the simulation time we were presented with what we expected. Because the exit queue was never backed up it grew at a linear rate. This is because the cars are already spread out due to having to wait at the entrance queue it will rarely get backed up since not all the cars will exit at the same time.

If the service time of the exit gate was significantly higher than the entrance gate, the exit gate may have more similar behavior as the entrance gate.

References

[1] Averill M. Law. Simulation Modeling and Analysis 5th Edition., 1-77, 2015.