Assignment 4

In Assignment 4, the class was tasked with calculating wait times for a Jurrasic Park themed theme park ride. In this report, you will find my algorithm for dealing with asyncronously running processes and my overall findings.

Algorithm

Strategy

In the program, I used a combination of semaphores and mutex locks to make sure everything runs as intended. The mutex locks are for making sure that data is not manipulated by other threads while a thread is running. The semaphores are used to signal when a thread should continue after an iteration of a while loop.

Pseudocode

```
start lineThread;
start carThreads;
while(timeStep < 600){
    signal lineThreadSemiphore;
    for each carThread:
        signal carThreadSemiphore;

    wait(timeStepSemiphore)
}
lineThread(){
    wait(lineThreadSemiphore)
    mutex lock;
    calculate Line Data;
    mutex unlock;
    signal timeStepSemiphore;
}
carThread(){
    wait(carThreadSemiphore)
    mutex lock;
    calculate car Data;
    mutex unlock;
}
```

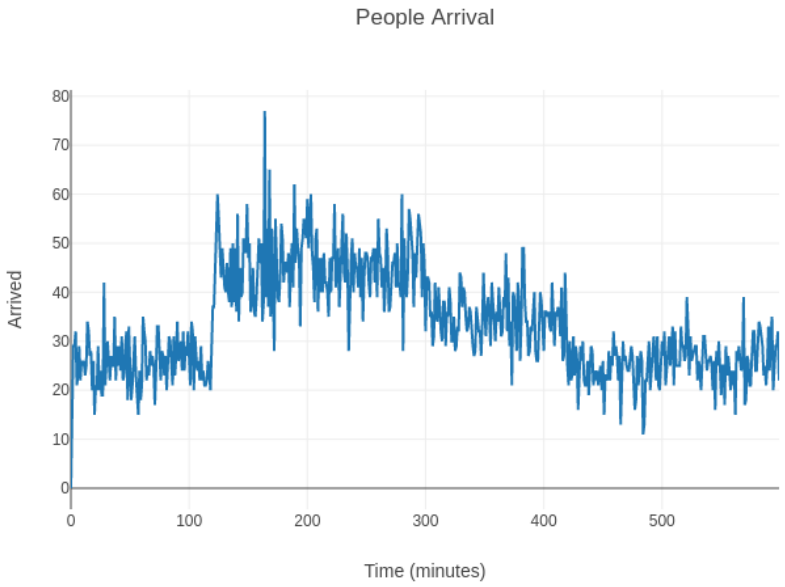
Results

Found below is a table of my findings for the project. Total GoAway represents how many people left after seeing a line that was over 800 people. The Rejection Ratio is calculated by dividing the Total GoAway by the number of total riders, so the lower the better.

(N,M)	Total Arrival	Total GoAway	Rejection Ratio	Average Wait Time (mins)	Max Waiting Line
N=2, M=7	19999	11613	1.38	53	800
N=2, M=9	19999	9205	.85	40	800
N=4, M=7	19999	3505	.21	18	800
N=4, M=9	19999	969	.05	9	800
N=6, M=7	19999	0	0	3	667
N=6, M=9	19999	0	0	0	77

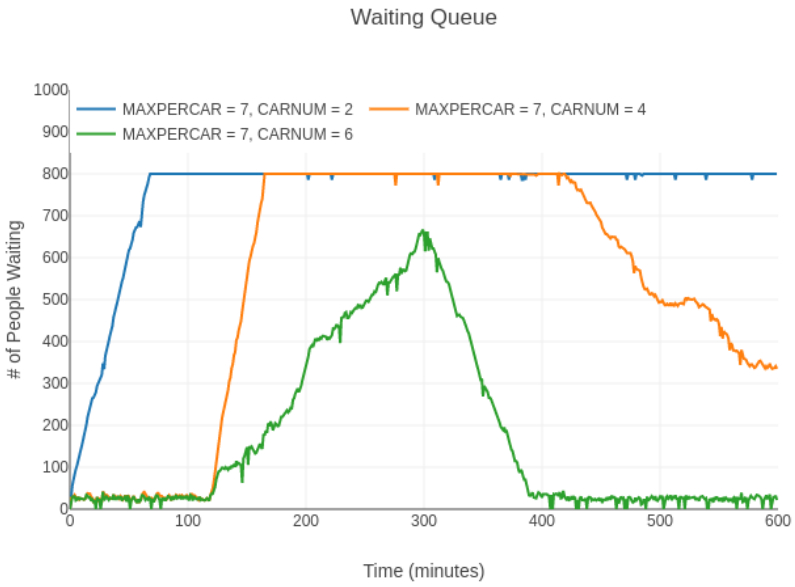
Arrivals

The below graph shows how many people arrived at the ride at each time interval.



Waiting

The below graph shows the number of people waiting in line at each time interval.



Leaving

The below graph shows the number of people leaving due to the line length at each time interval. This occurs when the line length reaches capacity (800 people).

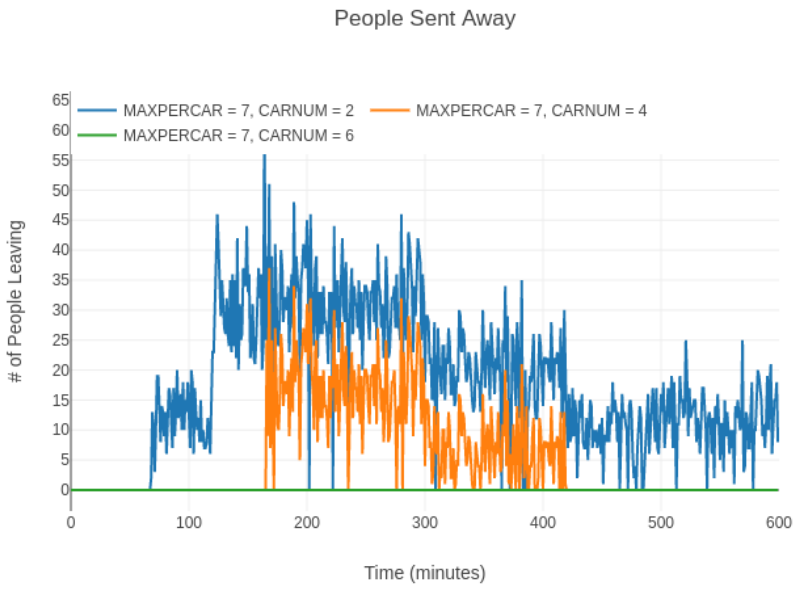


Figure 1: leaving