

CSc446 2023 Spring Term Project Report

Junyi Wei V00979016

Siheng Chen V00935153

Zihan Zhang V00974121

Application of Queuing Theory in Automatic Fare Collection System for Subway

Authors: Zhao Huawei, Wei Ziyue, Zhang Bingsen, Bina Yi

Disclaimer and Reference:

This project is referenced from the Urban Rail Transit and hence not an original project from the students, Junyi Wei, Siheng Chen, Zihan Zhang.

赵华伟, 魏子越, 张炳森, 边 毅. 排队论在地铁自动售检票系统中的应用[J]. 铁路通信信号工程技术, 2022, 19(1): 78-81,95.

Zhao Huawei, Wei Ziyue, Zhang Bingsen, Bian Yi. Application of Queuing Theory in Automatic Fare Collection System for Subway[J]. Railway Signaling & Communication Engineering, 2022, 19(1): 78-81,95.

Table of Contents

Disclaimer and Reference:	2
Table of Contents	3
Problem Definition	4
Goal for the Simulation	5
Methodology	5
How is the process simulated?	5
Running of the simulation	7
Data Collection	9
Results and Data Analysis	10
4-queue(gate) simulation	11
5-queue(gate) simulation	11
6-queue(gate) simulation	11
Conclusion	11

Table of Figures

Figure-1 General model of queueing system	4
Figure 2 - Initial setup	6
Figure 3 - Random selecting of the 4 gates	7
Figure 4- PyCharm execution	8
Figure 5 - Simulation results at size = 500	9
Figure 6,7,8,9 - Line charts	15
Line chart with Mean = 2.7&0.64; mu = 2.4;size = 5000 & 500	15
Line chart with Mean = 2.7&0.64; mu = 2.4;size = 5000 & 500	15
Line chart with Mean = 2.7&0.64; mu = 2.4;size = 5000 & 500	16

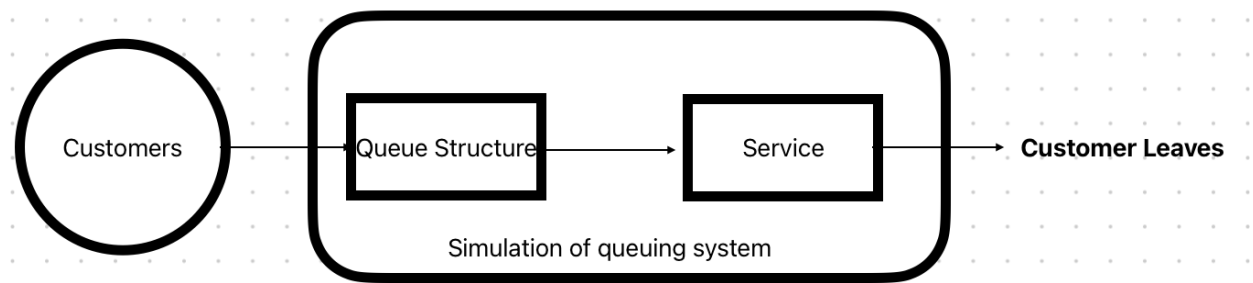


Figure-1 general model of queueing system

Problem Definition

The efficient and safe operation of transportation stations depends on the rationality of Automatic Fare Collection (AFC) terminal equipment configuration. Currently, the calculation of AFC terminal equipment is based on relevant specifications to meet predicted passenger flow during short-term and long-term peak hours in stages, without considering the characteristics of passenger arrival and the random nature of facility services. This may lead to inadequate equipment configuration, causing congestion and reducing service quality. Therefore, there is a need to develop a method that considers the characteristics of passenger arrival and facility services to optimize AFC equipment configuration and layout for better service quality. This study aims to address this problem by introducing the concept of queuing theory and establishing a queuing model to analyze queuing time, service time, and other indicators under different

conditions. The results obtained from the model will provide a scientific basis for AFC equipment configuration and layout optimization.

Goal for the Simulation

Our goal for the simulation is to demonstrate the differences in efficiency regarding the number of queues(gates) utilized in the system(subway station), particularly during rush hour of the subway system. The project source code: sim.py is implemented using built-in Python modules such as random, numpy and statistics and displays the results of each 4-gate, 5-gate and 6-gate systems for comparison purposes.

Methodology

As briefly mentioned above the simulation is written in Python with built-in modules, using arrays to store various data such as arrival time..etc.

By using the random module, we are able to create a simulation that is more likely to be the case in real-world scenarios.

How is the process simulated?

```
import random
import numpy as np
import statistics

random.seed(12)

size = 5000

mean = 2.7

service_mean = 2.4

#Start of 4-Queue
inter_arrival_time1 = np.random.exponential(mean, size - 1)
inter_arrival_time = [0 for i in range(size)]
for i in range(len(inter_arrival_time1)):
    inter_arrival_time[i + 1] = inter_arrival_time1[i]
```

Figure 2 - Initial setup

As we can see from the code snippet, this is where we set the seed, size, μ , and mean. As given by the paper, $\mu = 25$ people per minute (0.42 people/sec) and hence here we have service mean = 2.4 sec per person to pass through a gate.

$$\text{service mean} = 1/\mu$$

$$\text{mean} = 1/\lambda$$

Peak Hours	People On	People Off
Morning Peak	5561	5790
Evening Peak	5346	4911

Table 1- morning and evening peak passenger flow multiplied by peak hour coefficient K=1.2

According to the paper the given $\lambda = 93 \text{ people/min} \rightarrow 1.55 \text{ people/second}$.

Take this 4-gate simulation, we randomly generated exponential inter-arrival time for the transit takers. See figure 3 about how we simulate which of the four gates or queues each transit taker chooses using a randomized approach. This approach is used similarly on 5 and 6-gate systems.

```

arrival_time_1 = []
arrival_time_2 = []
arrival_time_3 = []
arrival_time_4 = []

service_time_1 = []
service_time_2 = []
service_time_3 = []
service_time_4 = []

last_arrival_time = 0

for i in range(0, len(inter_arrival_time)):
    service_time = np.random.exponential(service_mean, size=1)[0]

    q = random.randint(1, 4)
    if q == 1:
        arrival_time_1.append(last_arrival_time + inter_arrival_time[i])
        service_time_1.append(service_time)
        last_arrival_time = arrival_time_1[-1]
    elif q == 2:
        arrival_time_2.append(last_arrival_time + inter_arrival_time[i])
        service_time_2.append(service_time)
        last_arrival_time = arrival_time_2[-1]
    elif q == 3:
        arrival_time_3.append(last_arrival_time + inter_arrival_time[i])
        service_time_3.append(service_time)
        last_arrival_time = arrival_time_3[-1]
    else:
        arrival_time_4.append(last_arrival_time + inter_arrival_time[i])
        service_time_4.append(service_time)
        last_arrival_time = arrival_time_4[-1]

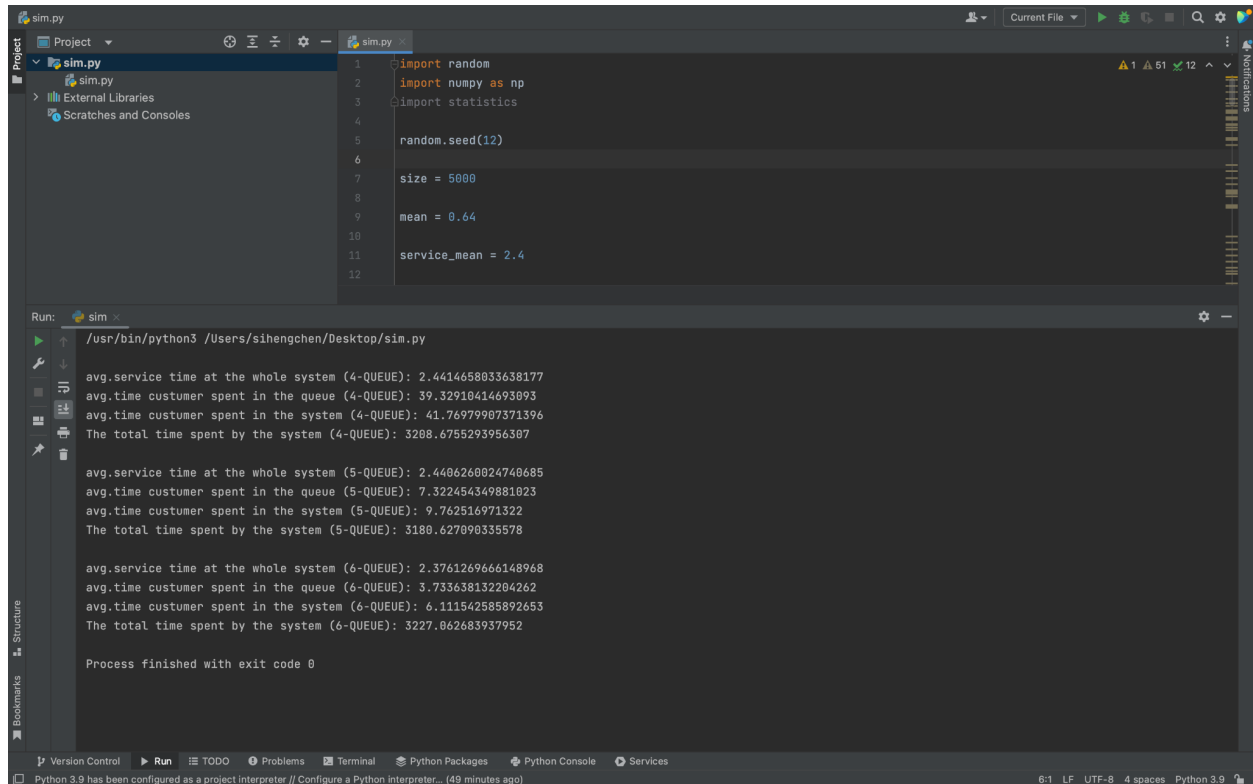
```

Figure 3 - random selecting of the 4 gates

Running of the simulation

The execution of our simulation simply requires entering the command:

python3 sim.py in the terminal or by using PyCharm simply click the “run” button.



```
1 import random
2 import numpy as np
3 import statistics
4
5 random.seed(12)
6
7 size = 5000
8
9 mean = 0.64
10
11 service_mean = 2.4
12
```

```
Run: /usr/bin/python3 /Users/sihengchen/Desktop/sim.py
avg.service time at the whole system (4-QUEUE): 2.4414658033638177
avg.time customer spent in the queue (4-QUEUE): 39.32910414693093
avg.time customer spent in the system (4-QUEUE): 41.76979907371396
The total time spent by the system (4-QUEUE): 3208.6755293956307

avg.service time at the whole system (5-QUEUE): 2.4406260024740685
avg.time customer spent in the queue (5-QUEUE): 7.322454349881023
avg.time customer spent in the system (5-QUEUE): 9.762516971322
The total time spent by the system (5-QUEUE): 3180.627090335578

avg.service time at the whole system (6-QUEUE): 2.3761269666148968
avg.time customer spent in the queue (6-QUEUE): 3.733638132204262
avg.time customer spent in the system (6-QUEUE): 6.111542585892653
The total time spent by the system (6-QUEUE): 3227.062683937952

Process finished with exit code 0
```

Figure 4- PyCharm execution

In the source code there are adjustable parameters which decide the results of the simulation. During our simulation process we adjust the seed number every time from 8-12 respectively for 4, 5 and 6- gate simulation. We could also adjust the mean value of our simulation as the mean value determines how busy or the passenger flow of the system; lower the mean the busier the system.

Data Collection

The first part of the data collection is done by the simulation results using sim.py.

The simulation results provide an intuitive and explicit comparison between our 4-queue, 5-queue and 6-queue simulations.

Take figure 5 for example, we can see that the period of average time a customer spends in the system/queue(in seconds) decreases drastically as we add more gates to the system.

```
/usr/bin/python3 /Users/sihengchen/Desktop/sim.py

avg.service time at the whole system (4-QUEUE): 2.3929558584200556
avg.time customer spent in the queue (4-QUEUE): 10.164009996736487
avg.time customer spent in the system (4-QUEUE): 12.557357216826437
The total time spent by the system (4-QUEUE): 356.4297270248117

avg.service time at the whole system (5-QUEUE): 2.3720244697696824
avg.time customer spent in the queue (5-QUEUE): 6.473646621266086
avg.time customer spent in the system (5-QUEUE): 8.846598843168302
The total time spent by the system (5-QUEUE): 332.84220051374615

avg.service time at the whole system (6-QUEUE): 2.3834193666128867
avg.time customer spent in the queue (6-QUEUE): 3.5620845229734397
avg.time customer spent in the system (6-QUEUE): 5.926853778276699
The total time spent by the system (6-QUEUE): 322.4701100077393

Process finished with exit code 0
```

Figure 5 - simulation results at size = 500

The second stage of data handling requires the use of Excel. In the initial phase of the simulation, we represent passenger arrivals as approximate Poisson arrivals, as detailed in the paper, using an $M/M/C/\infty$ queueing model. The paper discusses setting the number of queues (C) to 4, 5, or 6.

To gather statistics during the simulation, we've optimized the program to efficiently store results for different queue counts directly in a unified Excel file, with each queue represented by a dictionary. Throughout the program's execution, we organized the data into separate, named lists and stored them in dictionaries. Once the calculations were finished, we employed Python 3's "pandas" library to transfer data from each list to the Excel file. Each simulation run produces three .xlsx files containing all data, while the average time calculation results are displayed in the terminal.

Results and Data Analysis

For our data analysis we have divided into three cases as we have mentioned above 4-queue, 5-queue and 6-queue respectively. In the following 3 subsections we will compare our simulation results with size = 500 and size = 5000 and outline the key points in our data comparison.

4-queue(gate) simulation

Mean=2.7 Mu = 2.4	Size = 5000				Size = 500			
Seed	avg.service time	avg.time customer spent in the queue	avg.time customer spent in the system	The total time spent by the system	avg.service time	avg.time customer spent in the queue	avg.time customer spent in the system	The total time spent by the system
8	2.38214	0.67124	3.05294	13516.758	2.32503	0.54226	2.86930	1423.4758
9	2.39668	0.71886	3.11558	13231.357	2.46150	0.54467	3.01672	1350.5563
10	2.42866	0.75199	3.17960	13157.773	2.42315	0.67680	3.09711	1157.5253
11	2.41865	0.73318	3.15239	13260.702	2.38087	0.68805	3.05042	1324.7710
12	2.42761	0.77177	3.20067	13689.666	2.29530	0.64005	2.91116	1310.2438

Mean=0.64 Mu = 2.4	Size = 5000				Size = 500			
Seed	avg.service time	avg.time customer spent in the queue	avg.time customer spent in the system	The total time spent by the system	avg.service time	avg.time customer spent in the queue	avg.time customer spent in the system	The total time spent by the system
8	2.44972	21.5564	24.0057	3413.4622	2.37252	14.2365	16.6100	340.59983
9	2.38354	27.5656	29.9480	3326.8860	2.51549	15.4774	18.0106	354.53014
10	2.42325	74.9423	77.3665	3226.3357	2.33767	13.7609	16.1015	362.42726
11	2.44429	30.1425	32.5864	3249.0148	2.40010	12.4834	14.8827	367.94489
12	2.46627	35.3228	37.7895	3295.2395	2.48283	15.3323	17.7926	428.94210

Table 1 - Calculated Time Data of 4-Queue

5-queue(gate) simulation

Mean=2 .7 Mu = 2.4	Size = 5000				Size = 500			
Seed	avg.servic e time	avg.time customer spent in the queue	avg.time customer spent in the system	The total time spent by the system	avg.servic e time	avg.time customer spent in the queue	avg.time customer spent in the system	The total time spent by the system
8	2.39186	0.53667	2.92832	13675.043	2.36962	0.57350	2.93934	1371.1173
9	2.39312	0.49324	2.88681	13908.330	2.21417	0.36691	2.62385	1332.7941
10	2.45285	0.57531	3.02527	13226.803	2.31034	0.50436	2.80895	1323.0839
11	2.45276	0.57233	3.02318	13413.745	2.29600	0.44154	2.72450	1369.1935
12	2.39298	0.57358	2.96591	13280.508	2.39952	0.43589	2.83062	1403.5774

Mean=0 .64 Mu = 2.4	Size = 5000				Size = 500			
Seed	avg.servic e time	avg.time customer spent in the queue	avg.time customer spent in the system	The total time spent by the system	avg.servic e time	avg.time customer spent in the queue	avg.time customer spent in the system	The total time spent by the system
8	2.42532	7.49217	9.91786	3179.8122	2.20742	4.18342	6.39714	326.73150
9	2.41124	7.11696	9.52862	3193.1879	2.48403	5.96743	8.44959	364.90077
10	2.34730	6.53597	8.88230	3209.3314	2.50774	8.12844	10.6322	344.37755
11	2.39292	6.20743	8.60140	3287.3749	2.43321	3.69234	6.11769	374.91100
12	2.38156	7.05085	9.43196	3213.4141	2.43206	9.11655	11.5519	321.2827

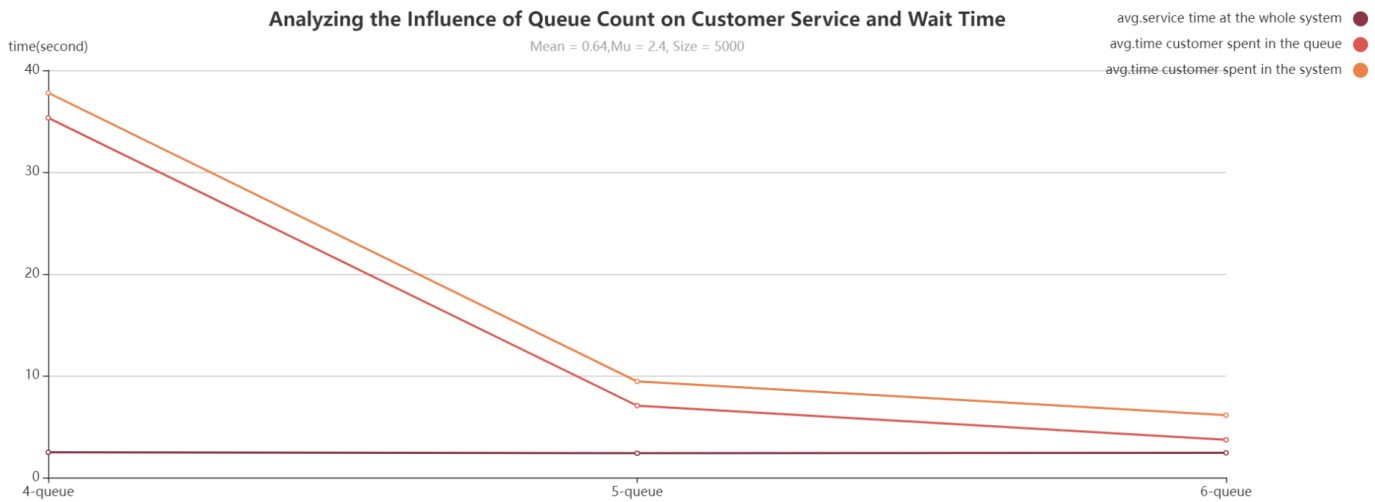
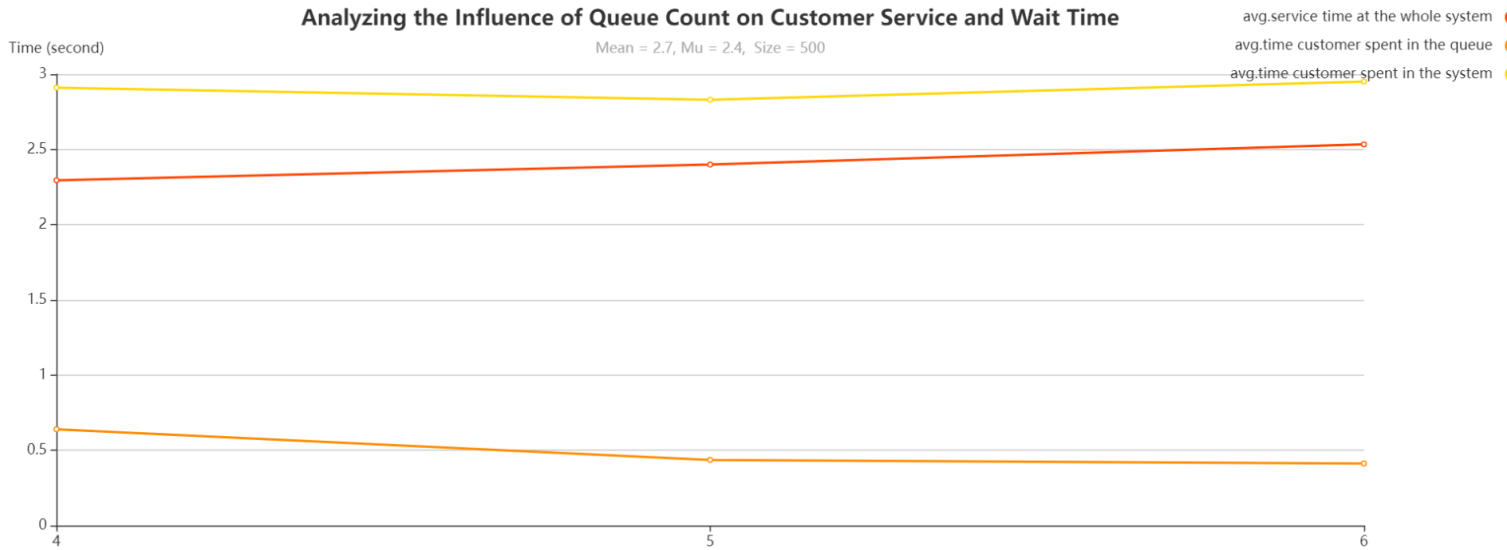
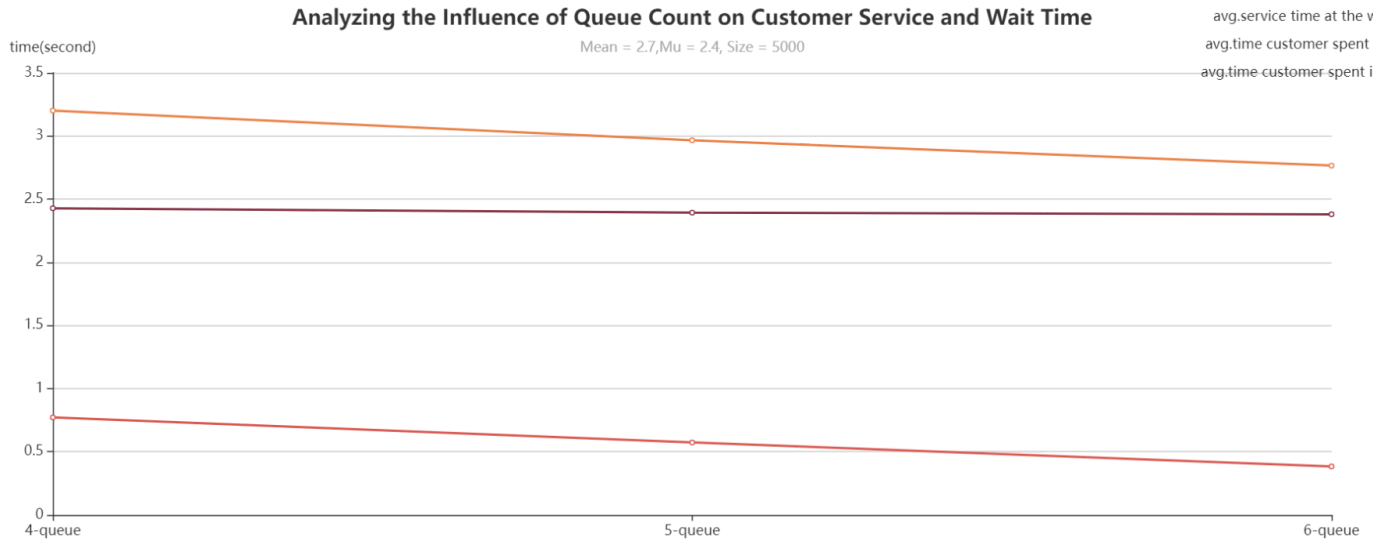
Table 2 - Calculated Time Data of 5-Queue

6-queue(gate) simulation

Mean=2 .7 Mu = 2.4	Size = 5000				Size = 500			
Seed	avg.servic e time	avg.time customer spent in the queue	avg.time customer spent in the system	The total time spent by the system	avg.servic e time	avg.time customer spent in the queue	avg.time customer spent in the system	The total time spent by the system
8	2.43908	0.43667	2.87459	13951.605	2.56655	0.33874	2.88806	1445.1701
9	2.41304	0.45104	2.86285	13228.661	2.23119	0.32759	2.55371	1342.0958
10	2.49099	0.45474	2.94593	13689.774	2.47029	0.34422	2.81870	1248.5236
11	2.36287	0.44189	2.80529	13228.645	2.55678	0.52860	3.07331	1419.8576
12	2.38001	0.38384	2.76563	13087.870	2.53366	0.41231	2.95180	1329.8125

Mean=0 .64 Mu = 2.4	Size = 5000				Size = 500			
Seed	avg.servic e time	avg.time customer spent in the queue	avg.time customer spent in the system	The total time spent by the system	avg.servic e time	avg.time customer spent in the queue	avg.time customer spent in the system	The total time spent by the system
8	2.38316	3.72368	6.10731	3206.4721	2.21742	1.73582	3.95007	339.94385
9	2.37177	3.93079	6.29931	3209.7507	2.26555	2.66248	4.93603	346.03213
10	2.41074	3.83250	6.24367	3218.4202	2.42333	4.05336	6.47573	321.49831
11	2.41535	4.21212	6.62906	3228.2699	2.26949	3.26963	5.52742	325.56547
12	2.41930	3.70120	6.11988	3300.4678	2.25536	2.41174	4.67674	342.2366

Table 3 - Calculated Time Data of 6-Queue



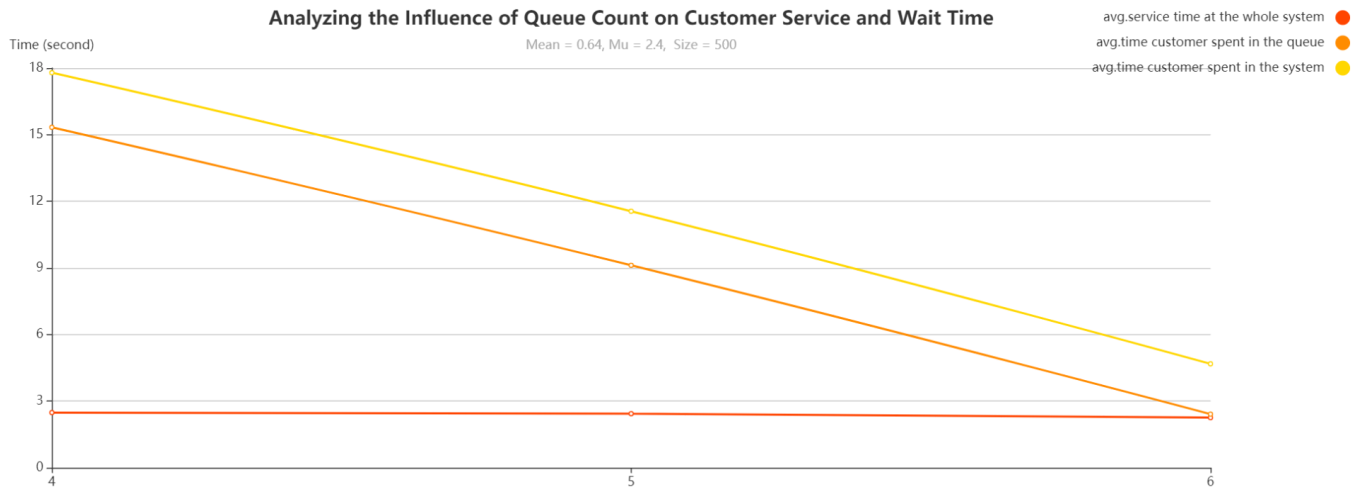


Figure 6,7,8,9 - Waiting Time Comparasion with different gates' number

To enhance the clarity of the polyline, we've used all calculation results with a seed of 12. By examining the trends in the broken lines displayed in Figures 6 and 7, we can see that when the mean is large, the system's occupancy rate is low. In this scenario, different numbers of gates exhibit minimal differences in the average time customers spend in the system when being served through the gates. Notably, when the number of customers reaches 500, the processing efficiency of the 5-Queue system even surpasses that of the 6-Queue system. This observation closely aligns with the conclusions drawn in the paper.

As the flow of people increases and the system utilization rises, it's evident from Figures 8 and 9 that the processing efficiency of the 5-Queue system is significantly higher than that of the 4-Queue system. However, when comparing the 6-Queue system's processing efficiency to the 5-Queue system's, the

difference is less pronounced when the total number of customers is large (e.g., 5000 people). This conclusion aligns with the paper's perspective as well.

Conclusion

In conclusion, this project has successfully achieved its goals through careful consideration of the project scope and implementation structure. By selecting a topic with a manageable scope and utilizing a familiar implementation structure, we were able to effectively simulate and analyze the performance of a queueing system in a station. Our simulation results have clearly demonstrated that the number of gates constructed in the station has a significant impact on the system's performance, as increasing the number of queues results in reduced wait times and improved efficiency. Overall, this project has provided valuable insights into the design and optimization of queueing systems, and can serve as a useful reference for future research in this field.

