ECE 6730 Project 1 Final Report

Haoting Li hli617 903270224

Project Description

The topic of this project is the simulation of a bank queue system. Assuming that there is a bank providing two kinds of services (information consulting and opening new account). The information consulting service takes shorter time than opening new account. The banks will open for eight hours everyday and there will be some customers coming in and served by the bank. Due to the limit of the availability of the staffs in the bank, there will be a queue in the system for customers to wait. There are six counters in the bank but not all of them are opened. If a customer come but there is no available counter, the customer will be waiting in a queue. If there is any counter available, the customer will be served immediately. If the customer waits for too long, he may leave. The patience (longest time for the customer to wait) is different from person to person. What's more, there would be some VIP customer coming to the bank. If the customer is a VIP, he will go directly to the head of the queue. If there are still some customers in the queue when the bank closes, one counter will remain opened until all customers are served.

Code Explanation and structure:

In the code, each unit of time is one minute in reality. The bank opens at 9 a.m. (corresponding timestamp is 0) and closes at 5 p.m. (corresponding timestamp is 480).

The files in this project are listed here:

- heap.py: The class of the queue of customers waiting in the bank. The queue is implemented with a priority queue using heap. The items in the queue is customers with different timestamp. The timestamp of customers is the time this customer entering the bank.
- heapFEL.py: The class of the future event list. The FEL is implemented with a priority queue using heap. Each event has three attributes: event type, event timestamp and event carrier (which customer or which counter)
- customer.py: The class of customers with attributes including VIP, coming time and patience initialized. Each attribute of a customer is randomly generated.
- counter.py: The class of counter to serve the customer. The attribute when the counter is available is here. If the counter becomes available, a customer from the

queue will come and served by the counter. If there is no customer in the queue, the counter will wait until that there is a customer to serve.

- banksim.py: The main function of the simulation. The handler function is in this file.

Event and Priority Queue Explanation

As the customers coming to the bank, two kinds of events will be added to the FEL. The first kind of event is “arrival” denoting the arrival of a customer. The second kind of event is “free” denoting that a counter has finished serving a customer and available to a customer in the queue. There are three attributes as an event. The first is the event type, an arrival type or a free type. The second attribute is the timestamp. The third is the carrier of the event (which customer or which counter). For arrival type, it means what time the customer arrives at the bank. For free type, it means when the counter is available to a new customer.

Some customer may quit when waiting in the queue. In reality, a customer is given a service number. When it is that customer’s turn, that service number will still be called but nobody answers. It is the same case in this simulation. When a customer is popped from the queue and found by the counter that the maximum waiting time is less than the time he has been waiting in the queue, this customer will be passed and printed on the output file “a customer quit from the queue”.

The implementation of priority queue here is heap. Once an event is inserted, the event will go to the right position in the heap with O(logn) directly. If the event with the smallest timestamp is popped, it takes O(logn) to re-heap the priority queue. Since the number of customer entering the queue and number of pop is generally the same and the simulation scale is neither too big nor too small, heap will be a good data structure to implement.

Pseudo-code of the simulation

```
while current time < close time:
    event = pop(FEL)
    if event is arrival:
        if there is a counter available:
            handle the customer
            next available time = current time + time needed for service
            insert free event to FEL
        else:
            if customer is VIP:
                insert the customer to the head of queue
            else:
```

```
                insert the customer to the queue with timestamp
        else if event is free:
            if there is some customer in the queue:
                pop(queue) and put the customer to counter
                insert free event to FEL
        insert new customer to the FEL
while the queue is not empty
        pop(queue) and handle the customer
```

Validation of the Simulation

Since this model is nearly the same as the case in the real world, the model itself should be valid. Generally speaking, the verification of the model can be done by face verification. The key part is to validate the model. Tuning the parameter is one key method to validate the model. Although some standard parameters are provided in references, the service cases can be different in different time and place.
To validate the simulation, all the parameters are tuned to the value based on the references. To make the simulation flexible, all the parameters can be tuned to some other values to simulate a different case. Here are the default values of the simulation based on papers:

- Serve time for customer type one: uniform(3.32,7.71)
- Serve time for customer type two: uniform(15.34,25.98)
- Waiting patience: uniform(31.41,55.20)
- New customer arrival at different time: exponential(scale = 8.42, 6.94, 12.63)

To make sure that the simulation works well, a txt output file of the result of the simulation is created. Here is part of the output file:

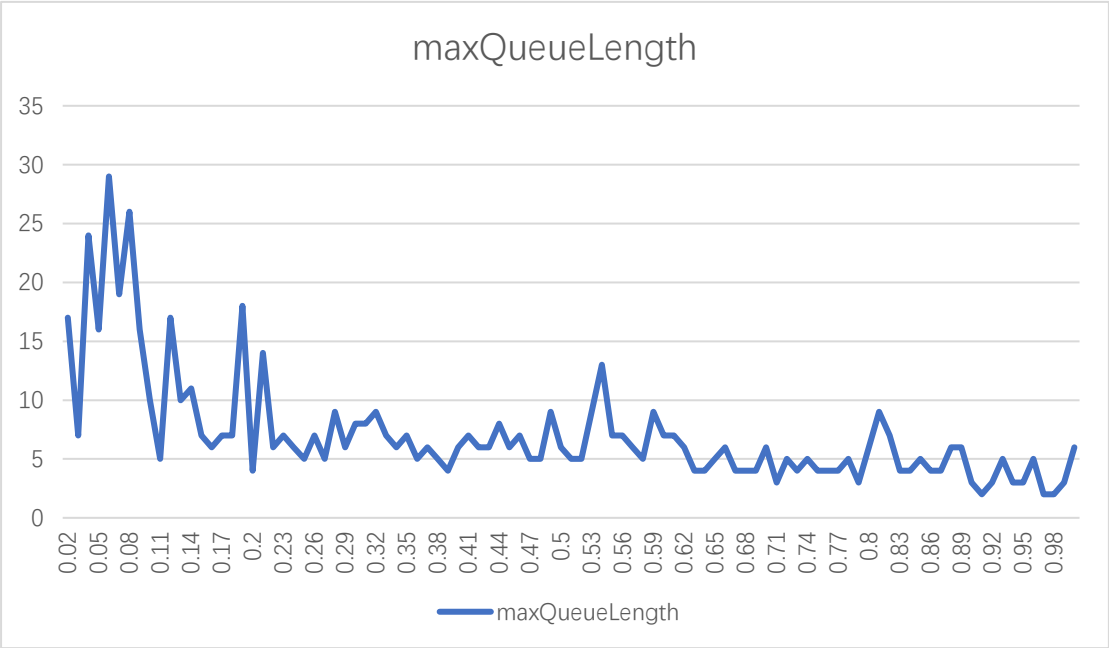
```
This is the 44 free
A customer served by the counter, queue length is3
This is the 45 free
A customer served by the counter, queue length is2
A customer quited from the queue
This is the 46 free
A customer served by the counter, queue length is1
A customer quited from the queue
This is the 47 free
A customer served by the counter, queue length is0
A customer quited from the queue
This is the 48 free
This is the 50 arrival
This is the 49 free
```

This is the 51 arrival
This is the 52 arrival
This customer went to the queue, queue length is 1
This is the 50 free
A customer served by the counter, queue length is0
A customer quited from the queue
This is the 51 free
This is the 53 arrival
This is the 54 arrival
This customer went to the queue, queue length is 1
This is the 52 free
A customer served by the counter, queue length is0
A customer quited from the queue
This is the 53 free
This is the 55 arrival
This is the 54 free
This is the 55 free
This is the 56 arrival
This is the 57 arrival
This is the 56 free
This is the 57 free
This is the 58 arrival
This is the 58 free

However, this case is only a statistic average number. The real case can be different by weekday/weekend, the number of counters opened, ratio of different services, etc. to make sure that each factor is taken into consideration, different test based on different parameter will be carried out in the next part.

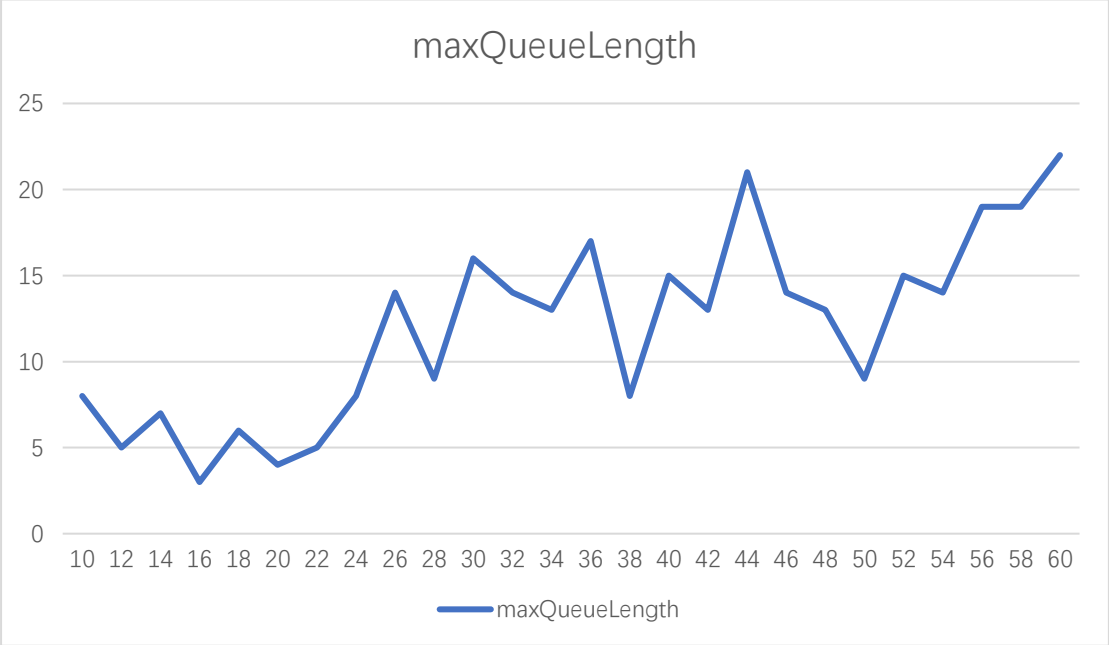
Test of simulation and Model Validation

The first parameter to test is the ratio of different kinds of services. The time required for opening a new account is much more than that requesting information consulting. Therefore, the different probability of a customer being a new account opener is tested. To measure the performance of each test, the maximum of the queue is recorded. The longer the maximum queue length, the more time for the customer to wait. Here, the different probability for each customer to be an information consulting customer is tested.



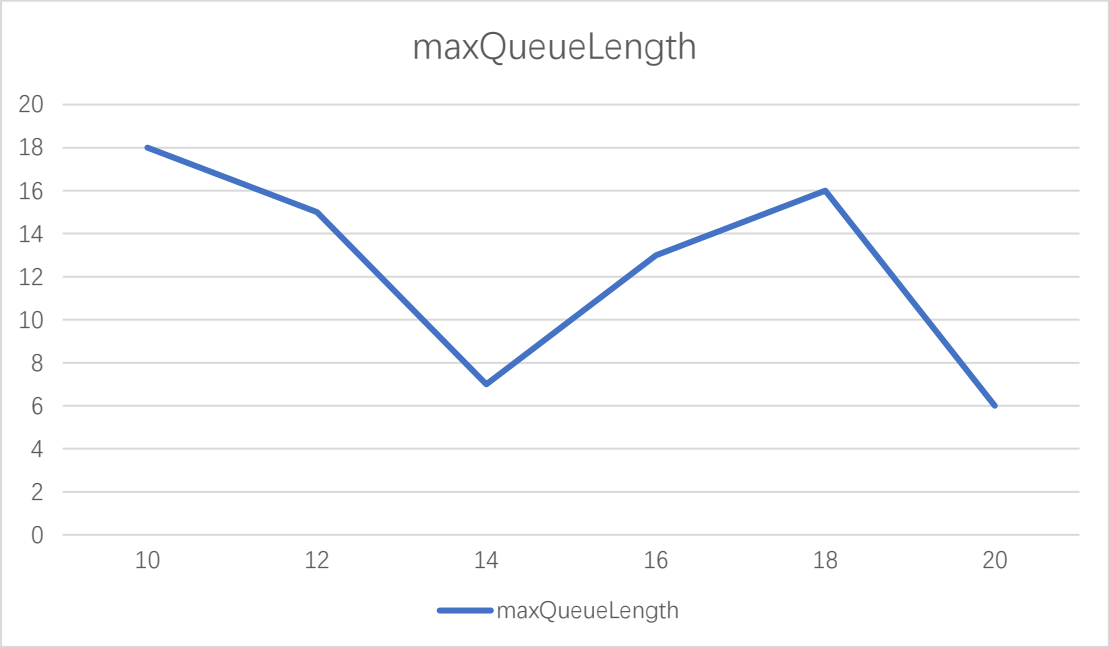
It can be concluded from this graph that the maximum queue length is becoming short as the probability increases. The queue length can very high for a smaller probability. When the probability reaches 0.3, the queue length dramatically drops. What's more, the length of the queue cannot be too long because the patience of the customers is limited.

The second parameter to test is the patience of the customers. The more patient the customers are, the long the queue is supposed to be. The relationship between waiting time and the maximum queue length is listed below.



It can be concluded from this graph that the queue length is generally longer as the patience of customers increases, although there is some fluctuation in the graph.

The third test of the simulation is the number of counter. In intuition, the line of queue length should drop dramatically as the number of counter increases. However, it is not always that case. The result varies when running the simulation. However, it seems that the result is highly random. This is one of the result from different simulations. It can be figured out that the max queue length is not always dropping. In contrary, it is even growing with more counter. It can be concluded that although more counters can help customers to reduce the time to wait, the actual case depends on some extra factors.



Appendix

ABCmod Table:

Entities:

Resource: Ser[N]: Counter	
Represents the place serving customers	
Attributes	
availableTime	The next time for the current counter to be available

Consumer: Class: Customer	
Represents people visiting the bank	
Attributes	
patience	Time that can wait in a queue

type	The behavior of the customer in the bank
isVIP	If the customer is a VIP

Queue: Unary: Queue	
Represents the queue people waiting in	
Attributes	
personNum	The number of people in the queue
index	If it is queue for service table or counter

Behavior:

Event: Arrival	
Represents the arrival of a customer	
Attributes	
arrivalTime	The time of the arrival of the customer
customer	Some attributes of a customer

Event: Free	
Represents a customer can be served by another counter	
Attributes	
serveTime	The time of the service of the counter

Activity: wait	
Represents a customer is waiting in the queue	
Attributes	
Initial Event	Arrival
Duration	Waiting in the queue
Terminal Event	Free

Activity: Service	
Represents a Counter serving a customer	
Attributes	
Initial Event	Pop from the queue
Duration	Serving the customer
Terminal Event	Free

Action: Quit from queue	
Represents a customer has been waiting for too long and finally quit from the queue	
Attributes	
queueLength	queueLength = queueLength - 1

References

[1] N.M. van Dijk, “To Pool or Not to Pool? The Benefits of Combining Queuing and Simulation,” Proc. Winter Simulation Conference (WSC’02), vol. 2, Dec. 2002, pp. 1469-1472, doi:10.1109/WSC.2002.1166421.

[2] Najmeh Madadi, Arousha Haghighian Roudsari, Kuan Yew Wong, Masoud Rahiminezhad Galankashi, “Modeling and Simulation of a Bank Queuing System”, 2013 Fifth International Conference on Computational Intelligence, Modelling and Simulation, DOI 10.1109/CIMSim.2013.41209

[3] Arbez, G., & Birta, L. G. (2016, December). A tutorial on ABCmod: an activity based discrete event conceptual modelling framework. In Proceedings of the 2016 Winter Simulation Conference (pp. 88-102). IEEE Press