Lab Q: Simulating a queue

Khudayar Farmanli
Politecnico Di Torino
Data Science and Engineering
Computer Aided Simulations and Performance Evaluation
Torino, Italy
s276310@studenti.polito.it

***Abstract*—In this homework we simulate single server event-based priority queue in python by using *queue* library. We consider different conditions and circumstances and see how they affect results.**

I. Tasks

- Plot Average Delay vs Load:
  1) Observe impact of simulation time.
  2) Change seed see what happens.
- Derive measures:
  1) Probability that server is Idle.
  2) Probability of Delay below a given value.
  3) Number of Users distribution.
- Finite waiting line:
  1) Probability that customer is lost.
  2) Average delay comparison with normal case.
- Server with failures:
  1) Effect of Failures and Repairs on performance.
  2) Measure the performance.

II. Assumptions

- The waiting line is infinite or finite based on defined condition.
- Customers come and first one enters to service (FIFO principle).
- After being serviced customer leave the server.
- If *Waiting Line* defined by finite number, customer can be lost.
- If *Failure* happens customers in waiting line and the one in service are spending more time based on *Repairment* time.

III. Input Parameters

- Seed - to initialize random number generator
- Service time - service time of each client defined by :

$$random.expovariate(lambd = 1.0/SERVICE)$$

- Average Service time - fixed number
- Load - load of the queue
- Inter-arrival time - time between each arrivals, defined by:

$$random.expovariate(lambd = 1.0/ARRIVAL)$$

- Average Inter-arrival time - defined by :

$$\lambda = \frac{Service}{Load}$$

- X - r.v. the time when failure happens, defined by:

$$random.expovariate(0.025/SERVICE)$$

- Y - r.v. the time taken for repairment, defined by:

$$random.expovariate(0.025/SERVICE)$$

- Simulation time - condition to stop simulation

IV. Output parameters

- Average Delay theoritical - defined by below formula:

$$1.0/(1.0/SERVICE - 1.0/ARRIVAL)$$

- Average Delay empirical - defined by:

$$\frac{Totalsumdelay}{totalnum.departures}$$

- Probability Idle
- Probability delay is below given value
- Probability customer is lost

V. Task 1.1 Measures

In this task the main point is to check how simulation time and changing seed affect simulation in general.

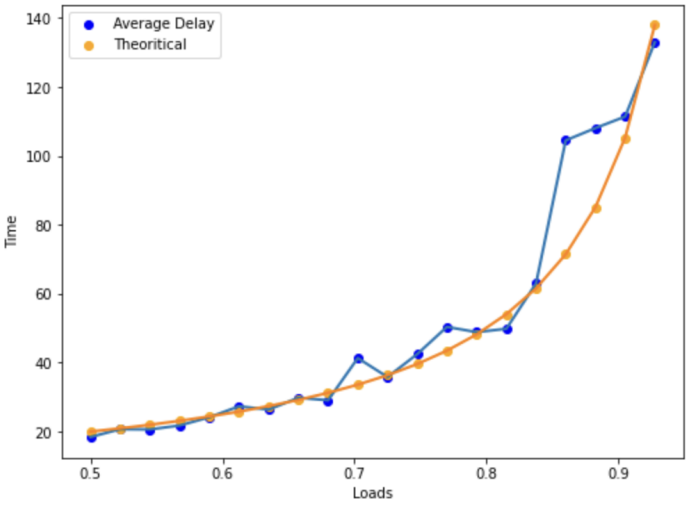

Fig. 1: Average Delay with respect to Load where seed = 40, simulation time = 50000, t = 0.55 second (execution time).

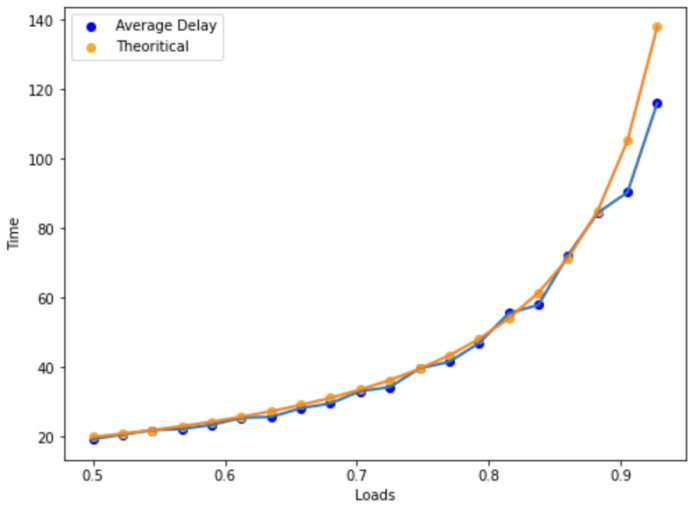

Fig. 2: Average Delay with respect to Load where seed = 40, simulation time = 500000, t = 05.28 second (execution time).

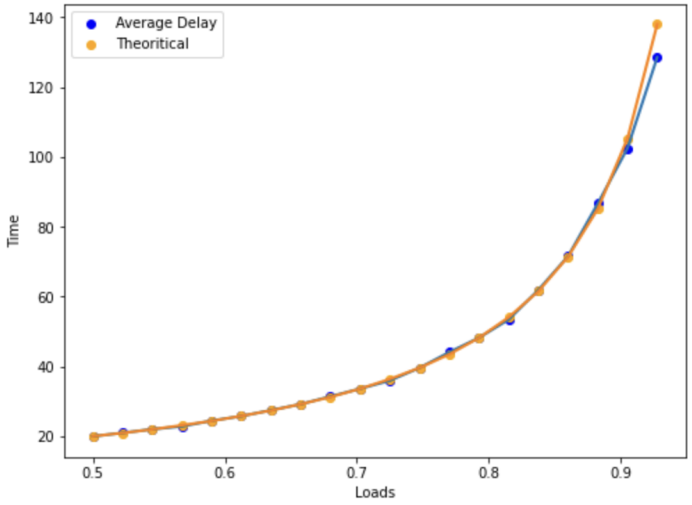

Fig. 3: Average Delay with respect to Load where seed = 40, simulation time = 5000000, t = 01 minute 52.29 second (execution time).

From above graphs we can easily observe that changing simulation time creates immense impact on general process. Since the number is small, average delay empirical and theoritical even though show the same pattern do not overlap as in the case of higher simulation time. In last graph we see they are overlapping to a great extent. So, we can say that increasing simulation time will provide better results. Normally, the trade-off here is execution time.

In case of changing seed there is not something special to show in terms of graphs. Since seeds are for initiating random number generator and repeat the pattern after some period, changing seed only provides a little bit different pattern but not something remarkable in the long run.

VI. Task 1.2 Derive Measures

In this section we need to check probability that server is Idle which means the customer comes and encounters that server is idle and enters it and probability that delay is below considered value, both per each load and the distribution of the number of users.

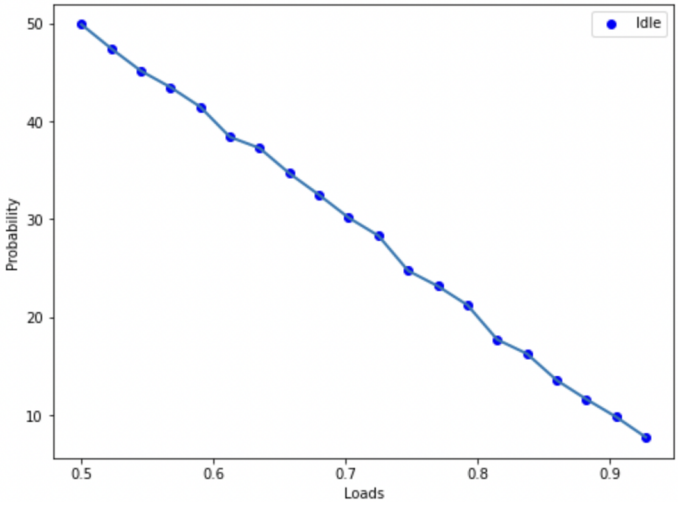

Fig. 4: Probability that server is Idle where seed = 40, simulation time = 500000.

As we can observe from above graph, idleness of server negatively affected by increased number of load. Smaller number of load means customer has higher chance to see server is idle, by increasing number of load it decreases tremendously.

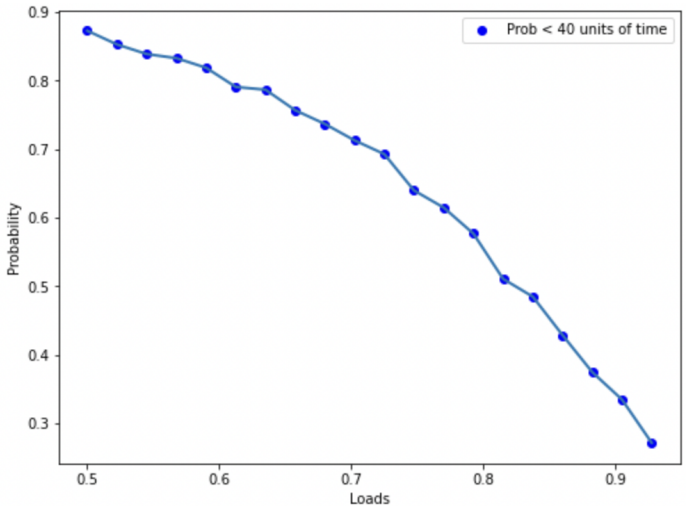

Fig. 5: Probability that delay is below than 40 units of time, simulation time = 500000

I defined maximum number of delay as 40 units of time to see the probability that delay is below this value. It is easy to understand that providing more load will increase delay more and eventually the probability will be lesser, so Figure 5 shows this phenomenon apparently. With increasing number of loads

we get decreasing number of probabilities that delay is below 40 units of time.

As we defined inter arrival times exponentially distributed users also distributed exponentially.

VII. Task 1.3 Finite Waiting Line

Here we need to check providing finite capacity to waiting line provides what kind of results in terms of customer loss (if customer sees waiting line is full he or she leaves the queue without entering) which will be measured by probability and Average Delay to compare results with infinite case (previous results)and see the changes.

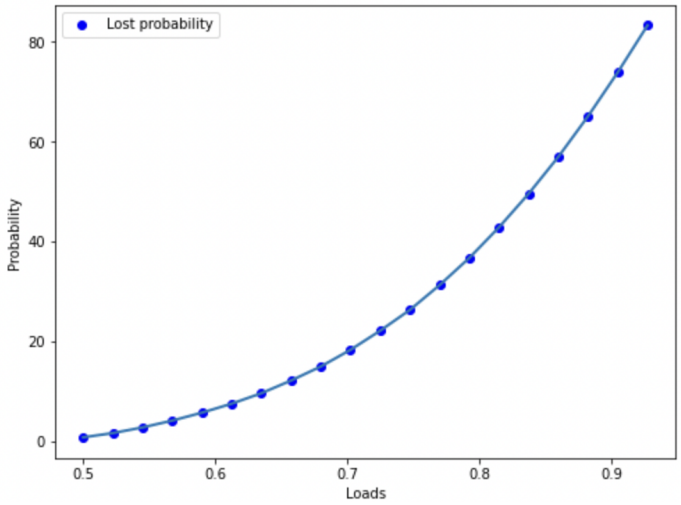

Fig. 6: Probability that customer is lost with respect to Load where seed = 40, simulation time = 500000.

In my particular case I defined maximum number of customer in waiting line 5 and checked how it is affected by load. As it can be seen from Figure 6, increasing load increases probability that customer can be lost which is normally expected results. In starting case where load = 0.5 the probability is almost 0 and in the highest case it is almost around 85 percent.

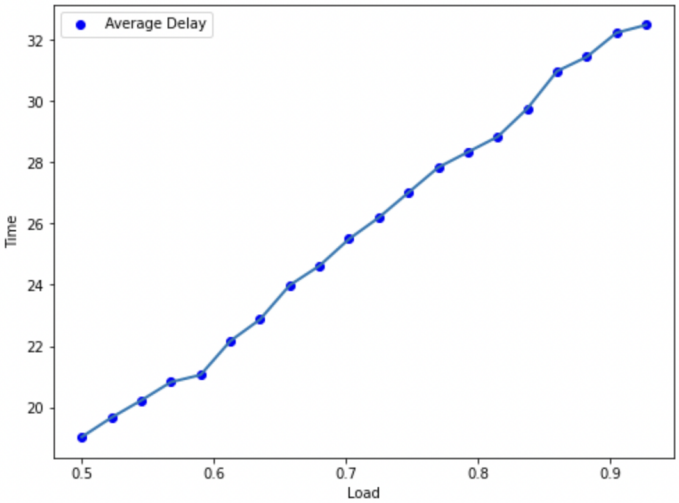

Fig. 7: Average Delay with respect to Load where seed = 40, simulation time = 500000.

Above graph shows average delay with respect to particular load for finite waiting line. What we see here is that average probability is lesser than infinite waiting line ones. Especially, with the higher number of load the difference becomes more noticeable. Of course, it is understandable since waiting line is finite we discard additional newcomer clients and that highly affects delay, less customer in waiting line lesser delay.

VIII. Task 2 Server with Failures

Here we introduce novelty that from time to time failure can happen in server which is defined by random variable X and the time taken to repair server is defined by random variable Y. So, this novelty affects simulation in a different way.

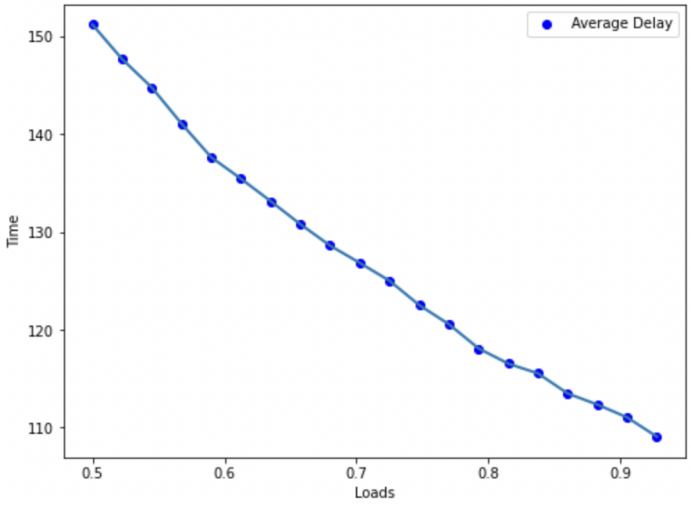

Fig. 8: Average Delay with respect to Load where seed = 40, simulation time = 1750.

What I observed from this phenomenon is that average delays increase through loads with respect to non-failure case which is expected since repairment time also added to each client's waiting time and average service time also increases because of the same issue. The time spent for execution also increases immensely because of computational complexity of the process, that is why I defined simulation time very low with respect to previous cases.

IX. Conclusion

In conclusion, based on experiments we can say that results totally overlap with expectations and provide solid explanation to theoritical knowledge.