Lab2 – Queueing Theory and Its Applications

Objective: Through this lab exercise, students should learn: (1) how to apply queueing theory to model a simple web application system; (2) how to use fundamental laws to investigate the relations of system quantities and gain insights about the system performance; (3) how to design and implement a simulation study based on the queueing model; and (4) how to analyze simulation results statistically and comparing them to analytical results derived from queueing theory.

Part A: Consider a system modeled as shown in the figure below. A web request send to web server must wait in queue if the server is busy, otherwise it will begin the processing at the server. When the server finishes processing the request, it sends a response to the user. A user will think for a while before generating next request.



Fig. L2 - a model of a simple web application system.

(a) If there are 100 active users each with 20 seconds average think time (IID). The average system

response time (IID) is 10 seconds which is the sum of queueing time and processing time. What is the throughput of the web server (responses/second)?

(b) If average queueing time is 9.75 second, what is the average processing time? What is the web server utilization (busy time/total time)?

(c) Based on your study, without increasing system response time, what is the maximum number of active users the system can support?

Provide analytical solutions to questions (a) to (c) using queueing theory. Show the calculation steps and which fundament laws you applied.

Ans:

1. The information we have:

- There are 100 active users.

- Each user has an average think time of 20 seconds

- The average system response time (W) is 10 seconds.

So,  requests per second

Then,

So, on average, there are 50 requests in the system at any given time.

To calculate the throughput (responses per second), we need to consider that the system is not overloaded. Therefore, the throughput is equal to the arrival rate:

Throughput X = responses per second.

So, the throughput of the web server is 5 responses per second.

We know that the average system response time is 10 seconds, including queueing time and processing time. If the average queueing time is 9.75 seconds, then the average processing time S = W – d = 10 – 9.75 = 0.25 seconds.

Due to the process time S = 0.25 seconds, each second the server can process 1/0.25 = 4 requests. However, the arrival rate = 5, which is higher than the process capacity of the server. So, the Utilization rate is 100%.

Any number of activate users increasing will rise the system response time, because the arrival rate is higher than the capacity of the server.

Part B: Considering the model shown in Fig. L2, construct a simulation program use the knowledge you learned about simulation models so far (refer to textbook figure 1.3) and answer the above question in (a) to (c) using your simulation results. To simplify the simulation, please use the following assumptions and data set for now (they may not be the best data samples to reflect reality; we will reconsider it after we cover random variables and their distributions):

1. Assume 100 active users with the “empty and idle” initial condition which means there are no requests in queue and web server is idle at t=0.

2. Assume the following: (i) user think time is a random number in [18, 22] seconds following uniform distribution; (ii) system process time for each request is a random number in [0.20, 0.30] seconds following uniform distribution.

3. Assume, at t=0, all users are thinking. Then they generate next requests (arrival list) randomly described by assumption 2(i). The first request goes to server without waiting; others may wait in queue if server is busy.

4. Let the simulation runs for 3 hours (T=3 hours) then stop the simulation.

5. Repeat runs certain times to estimate the mean, variance and standard deviation. Computer a confidential interval with 95% confidence.

Analysis:

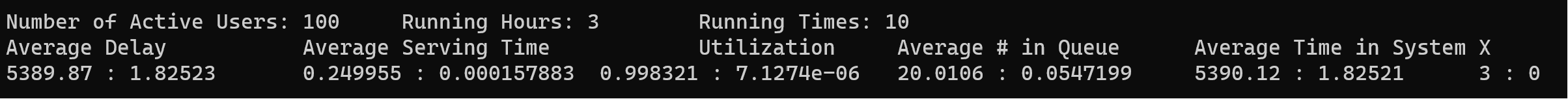
1. Implementation

My code simulates a queueing system with a single server and single queue by using c++. Four classes and one structure are created in the system. Class Server is use to store server status, the User class is used to imitate the behaviors of the users, the UniformRandom class is used to produce uniform random numbers, and the EventQueue is used to store event information for the event queue.

Two functions, arrival and departure, are the processes of arrival and departure respectively. A function called one\_shot is used to simulate once, which simulates the time span set by the variable of hour\_running. The main function calls one\_short as many time as the setting of variable of running\_times, and produce the report.

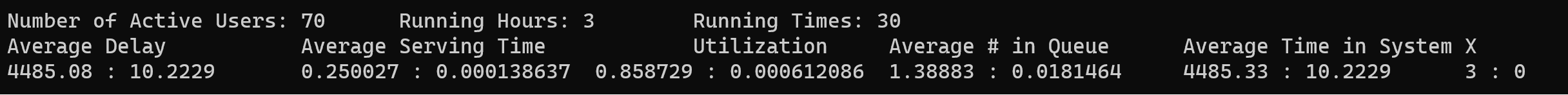
1. Result analysis

When setting the hour\_running to 3 hour and running\_time to 10, the output is shown as bellow:



Each statistical variable is shown by two numbers separated by a colon, where the number on the left side of the colon is the mean of the specific statistical variable, and the right side of the colon is the standard deviation of the specific statistical variable. One observes a high average delay and 0.998 utilization rate. It confirms that 100 active users make the server overloading.

After changing the number of active users to 70, we can find the utilization rate decreasing and lower average delay, as shown below:



From the above result, using arrival rate to approximate throughput is inappropriate, because high arrival rate will cause congestion, but not improve the throughput.

Confidential Interval:

(Using the first running data)

Average delay:

Average serving time:

Utilization:

Average requests in queue:

Average time in System:

Throughput: