CS-350 - Fundamentals of Computing Systems
Homework Assignment #3 - EVAL Solutions

Due on October 16, 2023 — Late deadline: October 18, 2023 EoD at 11:59 pm

Prof. Renato Mancuso

Renato Mancuso

EVAL Problem 1

In this EVAL assignment we will understand how seemingly insignificant changes in the input traffic distribution can lead to measurable differences in the behavior of queues and thus in the perceived service.
IMPORTANT: For this EVAL, make sure you download the latest version of the client binary! To confirm that this is the correct version, the client must print a line at startup that states:
`[#CLIENT#] INFO: CS350 Client Version 3.0`

a) In the previous EVAL assignment, you discovered that the client sends requests to the server with inter-arrival times that are exponentially distributed with mean $1/\lambda$ where λ is the arrival rate passed via the parameter `-a`. Also, the request lengths are exponentially distributed with mean $1/\mu$ where μ is the service rate passed via the parameter `-s`. Great! But as it turns out, that's just the *default* behavior of the client. Let's discover what else the client can do.

Which distribution the client uses is controller with an extra parameter `-d <dist. number>`, where `<dist. number>` is a number from 0 to 2. When passing `-d 0` you will be using the default exponential distribution. But what are the other two distributions? And does the `-d <dist.number>` parameter control both inter-arrival and service times?

To begin answering these questions, run your server and client with the following parameters (let it run, it will take about 5 minutes):

```
./server_lim -q 1000 2222 & ./client -a 4.5 -s 5 -n 1500 -d 1 2222
```

Notice that the client is requested to generate traffic according to distribution 1. Just like you did in HW2, plot the experimental data of inter-arrival times and request lengths and recover the type and parameters of the distributions used by the client when `-d 1` is passed.

Hint: *there is a some guesswork involved in recovering the distribution parameters. Start by looing at the shape of the distribution produced by the collected data and make a guess about which distribution might be. Setting the mean will be easy if you think about it. If there is a standard deviation to set, explore integer fractions or multiples of the mean.*

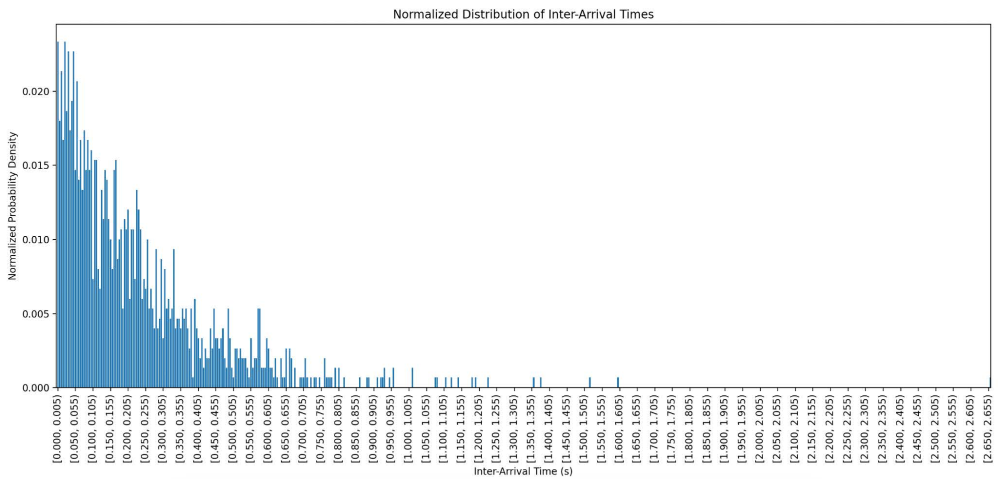

Figure 1: Normalized distribution of inter-arrival times for d=1

The plot above depicts the normalized distribution of the inter-arrival time between any two subsequent as measured by the sent timestamps of the requests when $d = 1$. The bin size is 0.005 seconds. The y-axis represents the ratio of the number of inter-arrival times that fall in the bin to the total number of samples.

Problem 1 continued on next page. . .

From the plot, it follows an exponential distribution with a mean and standard deviation of $1/\lambda$. In this case, the inter-arrival times $1/\lambda$ follow an exponential distribution with a standard deviation and mean with $1/4.5$ or an arrival rate of 4.5 requests per second, which is indicated by the `-a` parameter in the client.

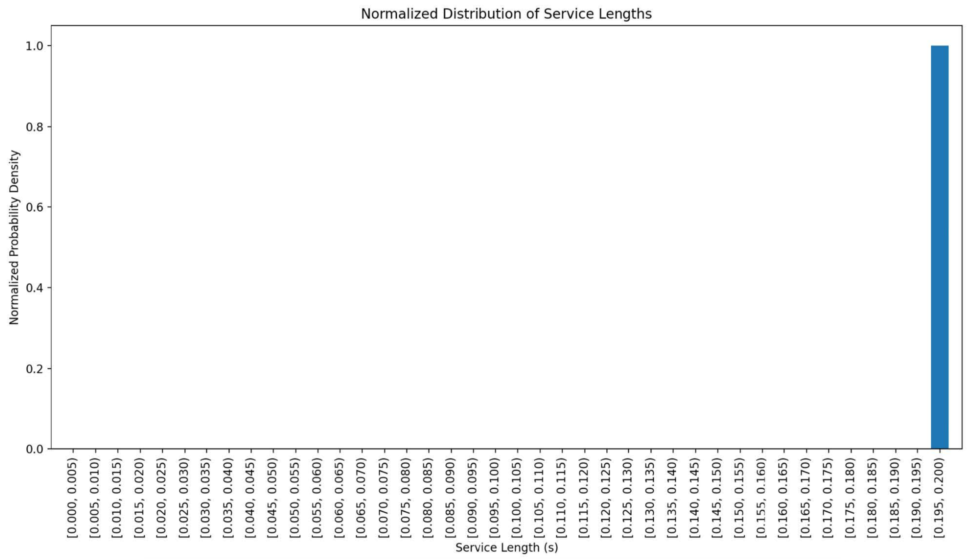

Figure 2: Normalized distribution of service length for d=1

The plot above depicts the normalized distribution of the service lengths (i.e., request lengths). It is observed that the service time for all 1500 requests are constant at 0.200 seconds. Thus the service time is $D = 0.2$ seconds, thus the service rate $\mu = 1/D = 5$ requests per second. This is indicated by the `-s` parameter in the client.

In conclusion, distribution number `-d` 1 is representative of a M/D/1 server model, in which the inter-arrival times are exponentially distributed and the service times are constant (deterministically distributed).

Problem 1 continued on next page. . .

b) Do the same as above to recover the parameters of distribution number 2. In a similar way as above, collect and post-process the server output data generated by the following parameter:

```
./server_lim -q 1000 2222 & ./client -a 4.5 -s 5 -n 1500 -d 2 2222
```

When decoding the distributions used by the client and their parameters, make conclusive statements about the distribution type for both inter-arrival times and service lenghts, and explicitly state their mean and standard deviation parameters.

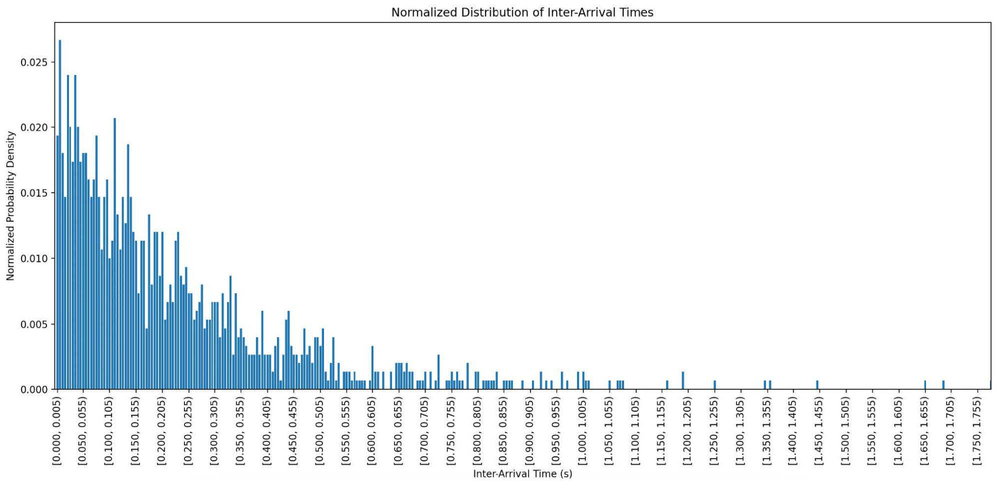

Figure 3: Normalized distribution of inter-arrival times for d=2

We observe the same arrival distribution as in (a). The inter-arrival times are exponentially distributed with a mean and standard deviation of $1/\lambda$ where $\lambda = 4.5$ requests per second.

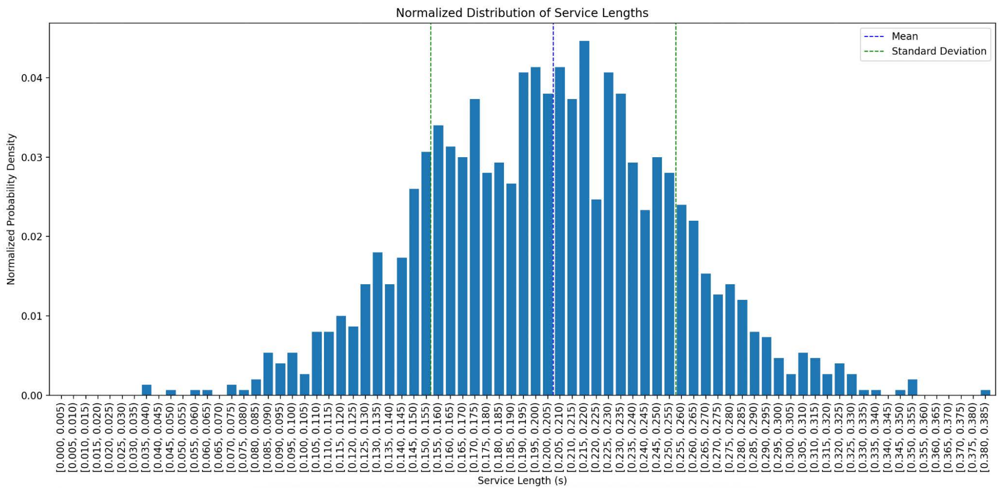

Figure 4: Normalized distribution of service length for d=2

From the service lengths plot, we observe that the service times follow a bell-shaped curve, a normal distribution. Thus, the service lengths are generically distributed. The mean of the distribution is 0.2 and the standard deviation is approximately 0.05.

The average of the service lengths from the 1500 requests was calculated to be 0.202195. Then, $\mu = 1/T_s$ where T_s is the average service time. $\mu = 1/0.202195 = 4.95$, which is close to the service rate of 5 that was passed in via the parameter `-s`.

Problem 1 continued on next page. . .

In conclusion, distribution number 2 is representative of a $M/G/1$ server model, in which the inter-arrival times are exponentially distributed and the service times are generically distributed (in this case, the 'G' represents the normal distribution).

Problem 1 continued on next page...

c) We are now ready to see how different distributions affect the quality of service in our simple FIFO server. Let us focus on the comparison between an exponential distribution and whatever distribution 1 is. Run and collect experimental data for the following template command:

```
./server_lim -q 1000 2222 & ./client -a <arr. rate> -s 20 -n 1500 -d 0 2222
```

where `<arr. rate>` is varied from (and including) 10 up until 19. Notice that these experiments will ask the client to use an exponential distribution (`-d 0`). Use this set of experiments to produce a plot of the average response time as a function of the server utilization—in a way similar to what you did in HW1.

Overlap on the same plot produced above the curve obtained by post-processing in the same exact way the results coming from running the following template command:

```
./server_lim -q 1000 2222 & ./client -a <arr. rate> -s 20 -n 1500 -d 1 2222
```

where once again `<arr. rate>` is varied from (and including) 10 up until 19. What can you conclude about the quality service perceived by the user when the load (a.k.a. its utilization) is comparable and only the distribution of the traffic characteristics changes?

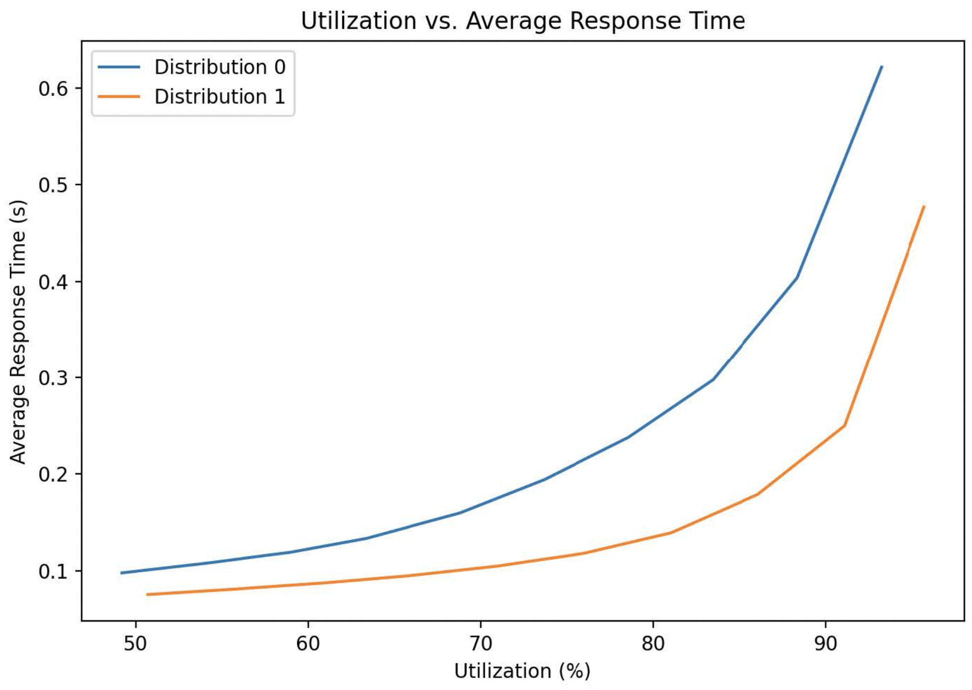

Figure 5: Average response time as a function of server utilization

The plot above depicts the trend of the average response time as a function of the service utilization for distribution 0 (service times are exponentially distributed) and distribution 1 (service lengths are constant or deterministically distributed), respectively. We observe that for any given percentage of utilization, the exponential distribution has a larger average response time compared to that of distribution 1. Consequently, the quality of service perceived by the user is likely to be higher with distribution 1 compared to distribution 0. The server takes less time on average to respond to the client requests (i.e., lower average response time).

Problem 1 continued on next page. . .

d) Let us now explore what happens when the queue has a constrained size. To do that, run the following command:

```
./server_lim -q 10 2222 & ./client -a 18 -s 20 -n 1500 -d 0 2222
```

Post-process the server output to understand what happened to our requests. First, calculate the ratio of requests that get rejected over the total. Next, plot the distribution of the inter-rejection time, i.e. the time that elapses between a rejection and the next. What does that distribution look like? Do not recover the parameters of the distribution, but simply share your insights on the shape of the distribution.

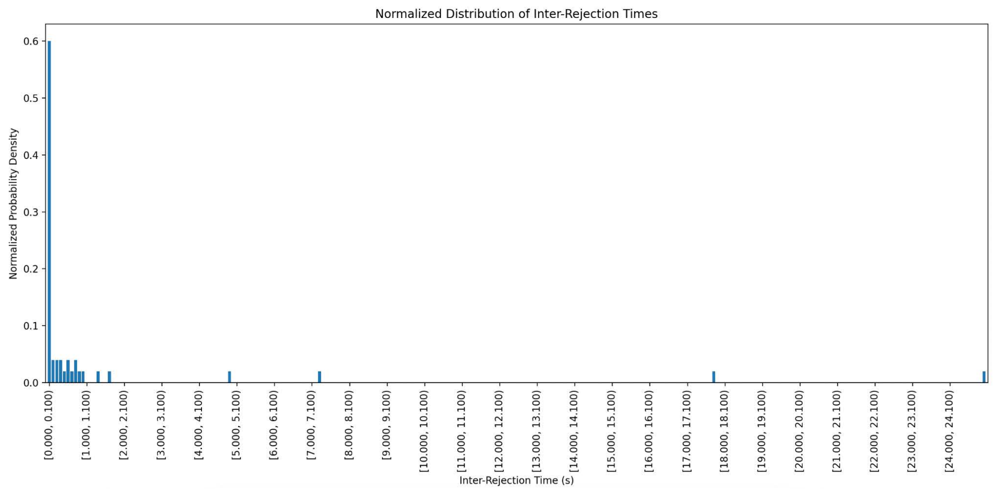

Figure 6: Normalized distribution of inter-rejection times for q=10

The plot above depicts the normalized distribution of the inter-rejection times (i.e., the time that elapses between a rejection and the next) in distribution 0. In particular, it displays the probability distribution of the amount of time elapsed for the server to reject the next request when the queue is full, as measured by the receipt timestamps of such requests.

The x-axis represents a set of time bins in 0.100 increments, respectively, of seconds. The y-axis represents the normalized count of inter-rejection times (a fraction of the total inter-rejection times) that fall in its respective bin.

The ratio between the number of rejected requests and the total number of requests was calculated to be 0.034. From the plot, we observe that the leftmost inter-rejection time bin has an approximate probability density of 0.6, which indicates that most rejected requests occurred within a tenth of a second of each other. In other words, requests that are rejected when the queue is full occur within a very short period of time of each other rather than being evenly spaced out. Many of the inter-rejection time bins with non-zero probabilities occur towards the left side of the distribution. The interval between inter-rejection time bins with non-zero probabilities and the next grows larger, as we read the graph from left to right. These larger inter-rejection times may occur when the queue randomly becomes full. Based on the shape of the distribution, the inter-rejection times follow an exponential distribution.

e) Repeat the same analysis as above, when distribution number 1 is used instead, thus by running the following command:

```
./server_lim -q 10 2222 & ./client -a 18 -s 20 -n 1500 -d 1 2222
```

If you compare the new rejection rate and shape of the distribution, would you conclude that the new system (the one that uses `-d 1`) offers a better or worse service to its users?

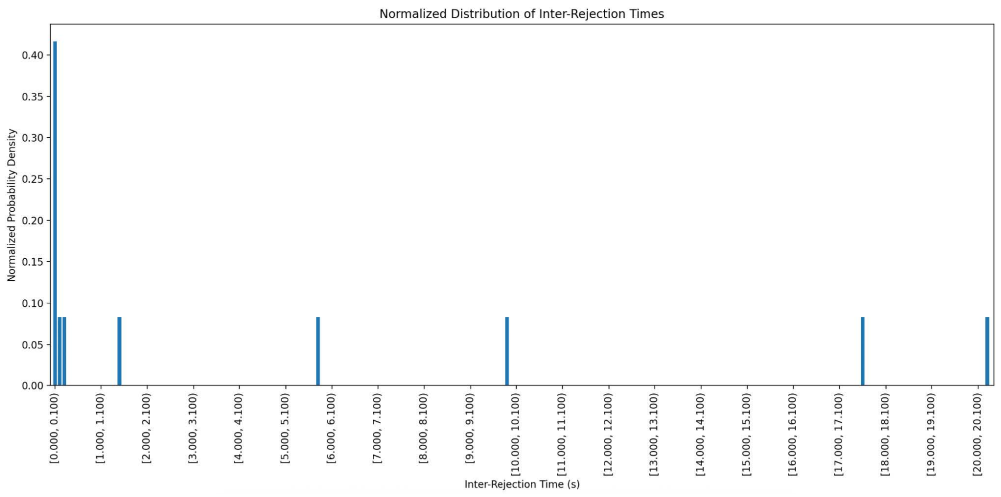

Figure 7: Normalized distribution of inter-rejection times for q=10 and d=1

The ratio between the number of rejected requests and the total number of requests was calculated to be 0.00866. From the plot, we observe that the leftmost inter-rejection time bin has an approximate probability density of 0.45, which indicates that requests that are rejected when the queue is full occur within a very short period of time of each other rather than being evenly spaced out.

Many of the inter-rejection time bins with non-zero probabilities are towards the left side of the plot. The interval between inter-rejection time bins with non-zero probabilities and the next grows larger, as we read the graph from left to right. These larger inter-rejection times may occur when the queue randomly becomes full.

Based on the shape of the distribution, the inter-rejection times follow an exponential distribution. For distribution 0, the rejection rate was calculated to be 3.4% and its inter-rejection times follow an exponential distribution. For distribution 1, the rejection rate was calculated to be approximately 0.87% and its inter-rejection times follow an exponential distribution.

Consequently, distribution 1 offers better service to its users because there are less rejections overall (i.e., less requests are rejected), even with the same exponential distribution type. This implies that the server is able to respond to the majority of incoming requests and will better meet client expectations in distribution 1.