Fundamentals of Computing Systems Homework Assignment #3 - EVAL

1. EVAL Problem 1

   (a) **Distribution of inter-arrival times and request length of -d 1 distribution**

   Using Python code my homework2, I parsed the .txt file to isolate only the length of the client requests. To display the distribution, I used matplotlib.

   I created three different distributions: normal, exponential, and uniform. Just as in HW2, I generated 10000 data for each distribution and plotted the inter arrival time and other three distributions. In terms of mean of three different plots, since the -a value of the experiment is 4.5, I used 1/4.5 as the mean of each distribution.

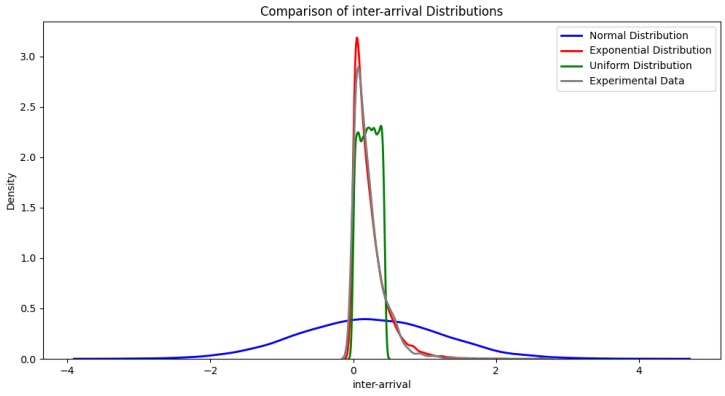

Figure 1: Comparison of inter-arrival distribution of -d 1 with other three distribution

In terms of similarity to my result plot, exponential distribution comes closest. In the same way that the -d 0 distribution has exponential inter-arrival times, so does the -d 1 distribution.

Now let's check the request length distribution.

Initially, I plotted the request length the same way as the inter-arrival value by just changing the mean of each distribution to 1/5 since the -s value I used for this experiment is 5.

Below is what I got for my result.

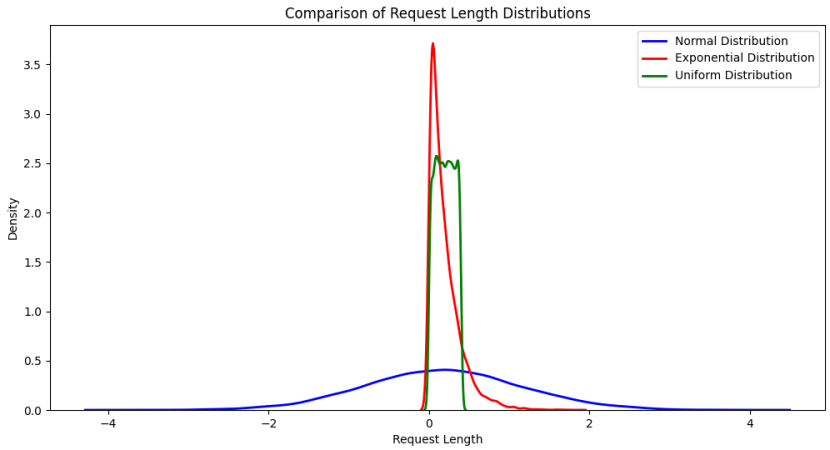

Figure 2: Comparison of request length Distribution with other three distribution - 1

For this graph I could not see the distribution of my data. I tried to re-run it a few times but I could not get the result. So I looked at the result output.

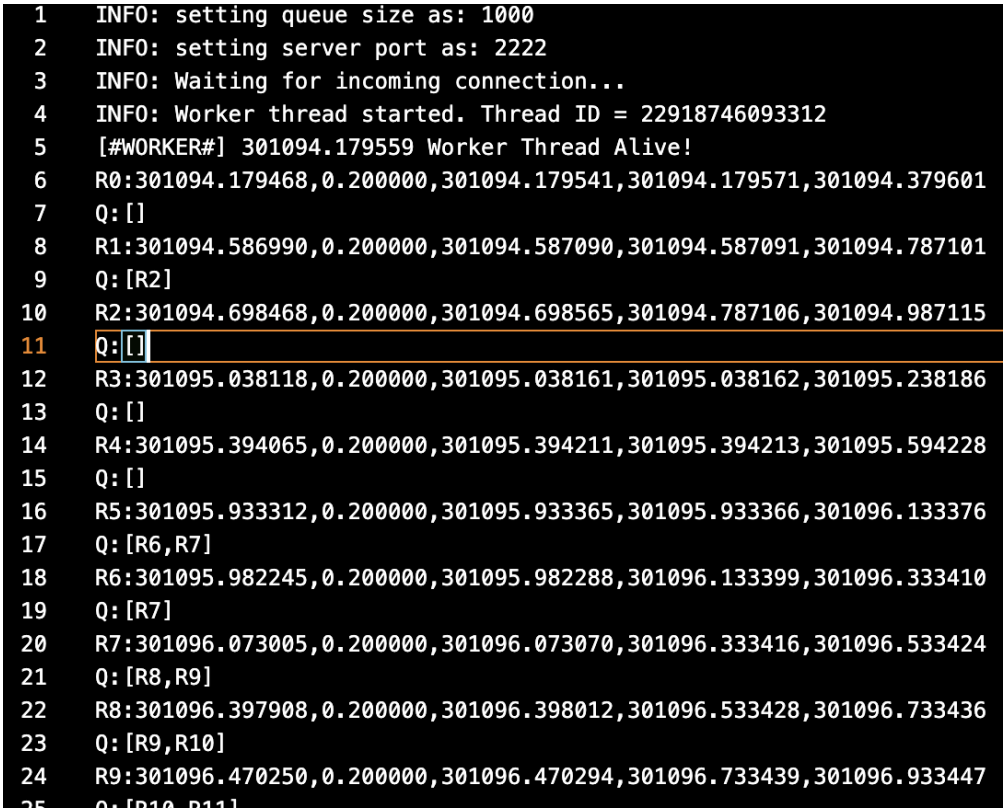

Figure 3: request length of result-a output

The structure of the output is

R<request ID>:<sent timestamp>,<request length>,<receipt timestamp>,<start timestamp><completion timestamp>.

So the second column is the request length of each request. From the observation, the request length of this outputs are all 0.200000 sec. So the client request deterministic length of request (0.2000 sec) to server every time. This is the reason my I could not see the result of plot. Because there is no distribution but actual uniform data with mean of 0.2. So I draw the vertical line with mean of 0.2.

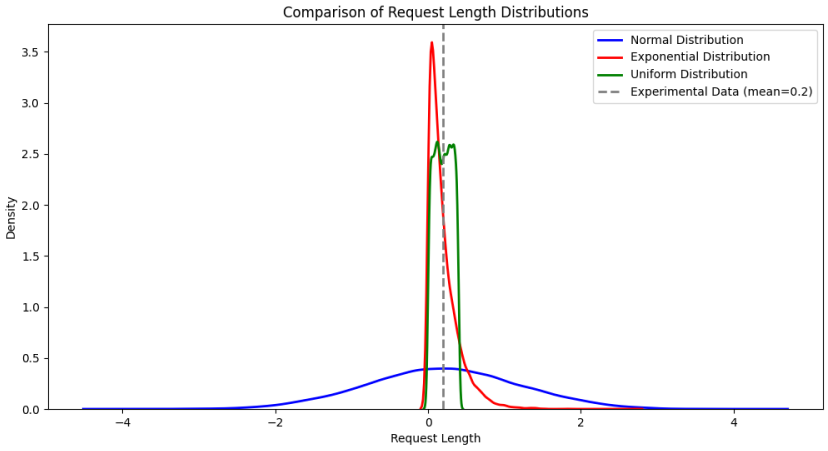

Figure 4: Comparison of request length Distribution with other three distribution - 2

With the data I plotted, the uniform distribution has the same mean in the center. Additionally, the request lengths of the result output are all 0.2. So it is uniform. As such, I conclude that the **-d 1 distribution has an exponential arrival time and a uniform request length distribution.**

(b) **Distribution of inter-arrival times and request length of -d 2 distribution**

Using the same method as part a), I plotted the distribution of -d 2 along with three different distribution means of 1/4.5 for comparison.

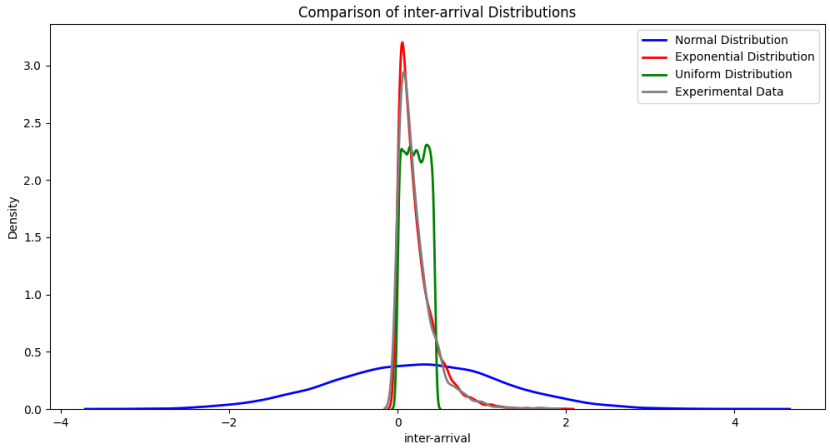

Figure 5: Comparison of inter-arrival distribution of -d 2 with other three distribution

It is similar to the exponential distribution. So We know the -d 2 arrival rate distribution is exponential just as -d 0 and -d 1.

I plotted the request length as the same way.

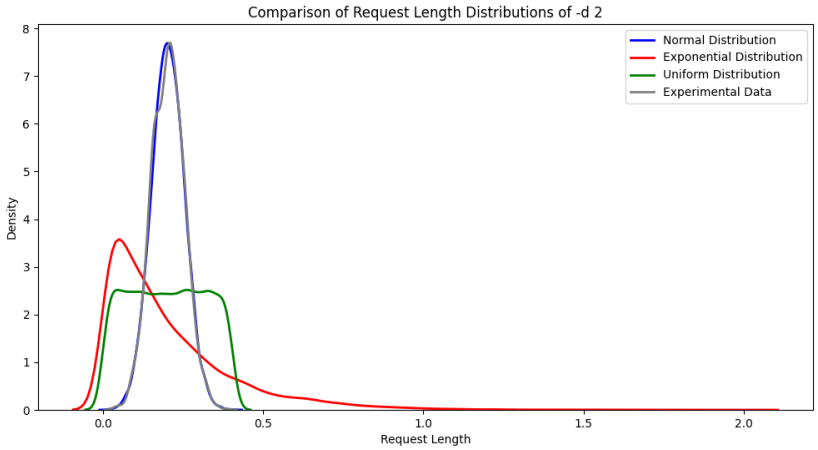

Figure 6: Comparison of request length distribution of -d 2 with other three distribution

In this comparison, I need to include mean and standard deviation of data point to generate the plot. After I generate the the other distribution including the mean

and standard deviation data. The plot is similar to the normal distribution. So I know the **-d 2 distribution is exponential for arrival rate and normal distribution for request length.**

(c) **An analysis of average response times based on server utilization for -d 0 and -d 1 distributions**

To make it easier for me to see the data, I've generated Average Response time and utilization relationship tables based on -a changes for -d 0 and -d 1.

	-d 0		-d 1	
-a	Avg Response (sec)	Utilization (%)	Avg Response (sec)	Utilization (%)
10	0.09765434200025629	0.49242891862028426	0.07526528599877687	0.5073897490552648
11	0.10766877933366535	0.5415280671683889	0.08076348999992479	0.5580536852867641
12	0.11904623266604418	0.5906053839427221	0.08708669533402039	0.6086872918069409
13	0.1332708073334846	0.6396512843400524	0.09463290666583149	0.659354800682165
14	0.1594046386674745	0.6886833398869285	0.10453065066738054	0.7100073367468578
15	0.1940192926666932	0.7377041603377111	0.11772975333349314	0.7606343199328207
16	0.23773647066605433	0.7866459048552443	0.13844868933285276	0.8112517486108155
17	0.2966538760002004	0.835622560835726	0.17838526399973004	0.8617199812210957
18	0.4019218626661847	0.8845417747230884	0.2499972646665216	0.9120576827576802
19	0.6179984780004015	0.9334549583063444	0.46995691466645806	0.9581351025568905

Table 1: Avg Response VS Utilization

Based on this statistic, I plot these two comparison data.

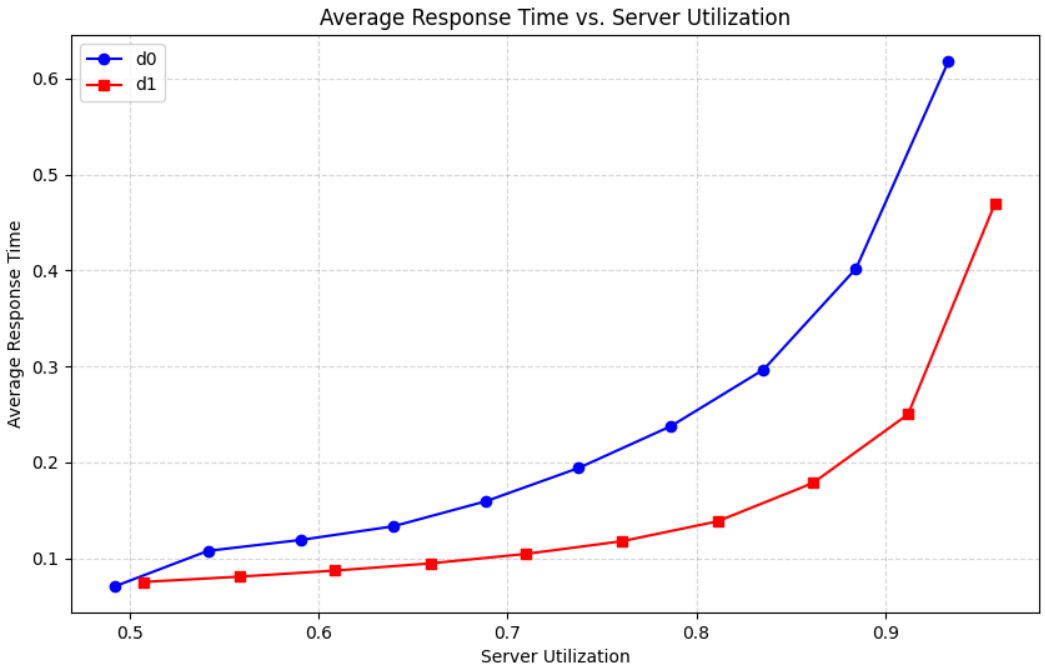

Figure 7: Comparison of request lengths vs Utilization of -d 0 and -d 1

The analysis of the data reveals that the M/M/1/k model (-d 0), based on my discovery from HW2 and limited queue size k, characterized by exponential inter-arrival and service time distributions, exhibits significantly more variability and less predictability as utilization increases compared to the M/G/1/k model (-d 1), which has a uniform service time distribution based on my discovery from part a), for d1 generic server model is "Uniform distribution".

At lower utilization levels (below 0.6), both systems provide comparable average response times, indicating that under light traffic, the M/G/1/k model (-d 1), which has minimal impact on user-perceived performance. However, as utilization rises, especially beyond 0.6, the differences become increasingly pronounced. The M/M/1/k model (-d 0) experiences a rapid escalation in average response times, reaching up to 0.62 at around 93% utilization. This behavior is due to the inherent variability of the exponential distribution, which leads to unpredictable delays, causing significant fluctuations in response time and resulting in lower

service efficiency. Users in this scenario would experience more inconsistent performance, with sudden spikes in wait times, making the system feel sluggish and less reliable.

In contrast, the M/G/1/k model (-d 1) with a uniform service time distribution maintains much more stable and predictable performance even as the system approaches full capacity. At 96% utilization, the average response time reaches around 0.47, demonstrating that the system can handle heavy traffic more efficiently without overwhelming the server.

This consistent behavior is crucial for maintaining a quality user experience, as it reduces the likelihood of unexpected delays and congestion. As a result, the M/G/1/k model (-d 1) delivers a smoother, more reliable experience under high loads, making it preferable in scenarios where predictable and steady service quality is essential. Accordingly, a uniform service time distribution would be far more effective than an exponential distribution for applications requiring consistent response times and a positive user experience.

(d) **Queue with a constrained size of d0 model.**

After I ran the command and plot the inter-rejection distribution, I got the following plot and ration.

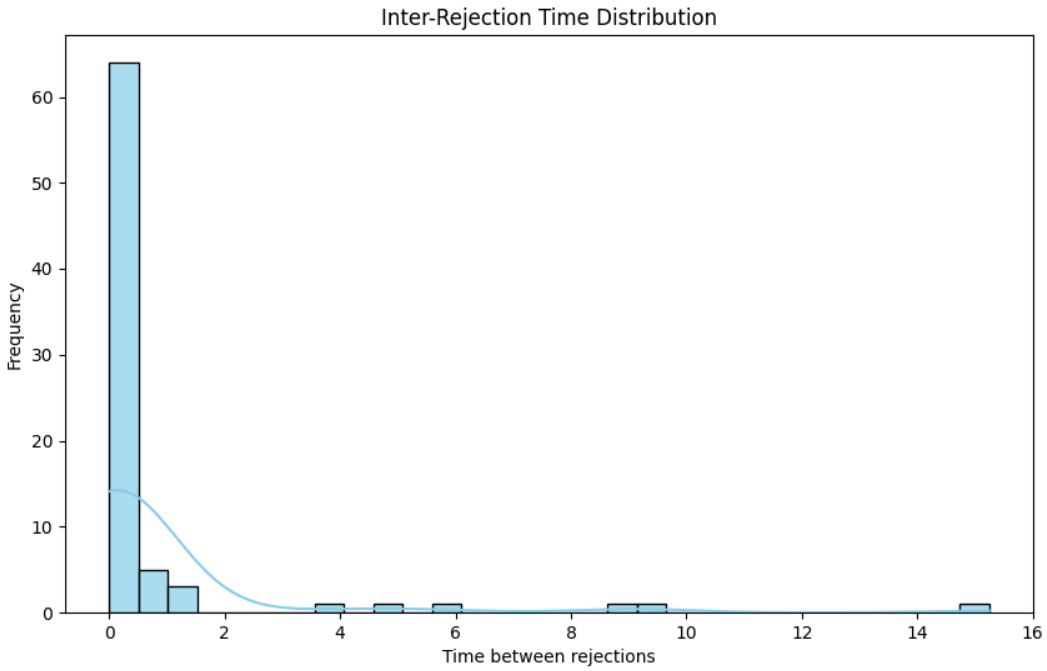

Figure 8: Inter-Rejection Time Distribution

The rejected request ratio over total is **0.0527**. I got 79 rejection out of 1500 request. Based on the plot, the inter-rejection time distribution appears left-skewed. This means that many rejection events happened quickly, while there were some intervals with a longer gap between rejections.

The system experiences bursts of rejection activity when under heavy load, indicating times when the server is overwhelmed and unable to process requests quickly enough. Due to the arrival rate (-a 19.6) approaching the service rate (-s 20), the system approaches saturation and rejections become more frequent. The short inter-rejection times highlight periods of intense competition for server resources, resulting in a noticeable decrease in service quality as requests are often denied or delayed.

(e) **Queue with a constrained size of d1 model.**

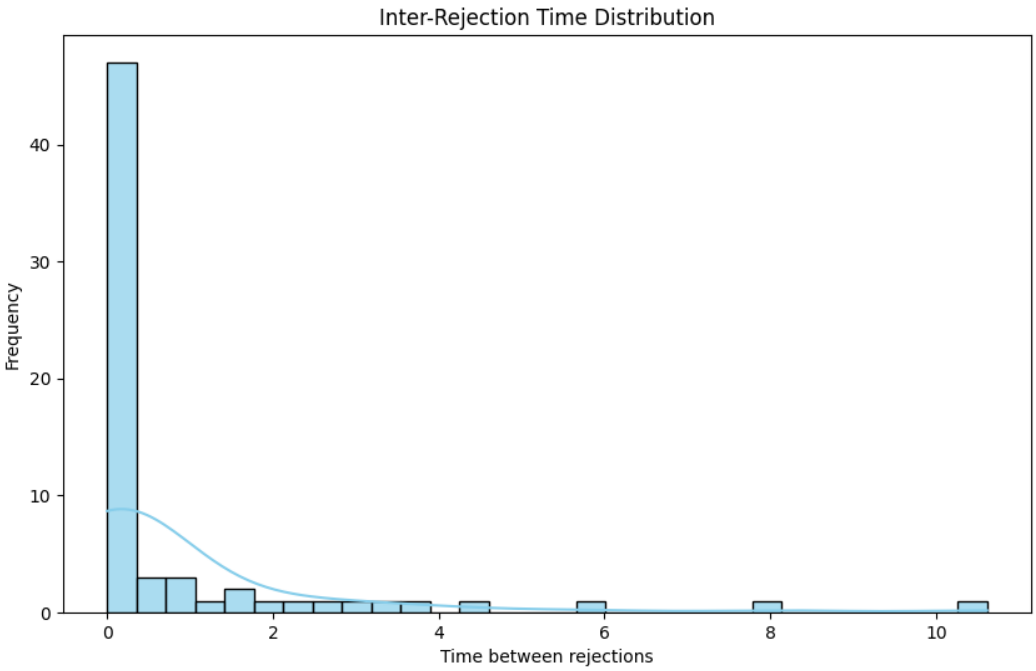

Figure 9: Inter-Rejection Time Distribution

For d1 model I got the **0.0447** ratio of rejected request. It is lower than the d0 model. Total number is 67, so 12 requests lower than d0.The d1 model handles incoming requests more effectively, offering a better quality of service to its users than the (-d 0) model, which had a higher rejection rate.

The shape of the distribution in the (-d 1) model may further support this conclusion. If the (-d 0) model's inter-rejection time distribution exhibits a higher frequency and more less skewed shape.

In conclusion, the (-d 1) model, with its lower rejection ratio and potentially smoother distribution of inter-rejection times, appears to provide better service to its users. By effectively managing the incoming load and minimizing the likelihood of request rejections, the (-d 1) model enhances the overall user experience, highlighting the importance of choosing the right traffic characteristics for optimizing service delivery.