COMPENG 4DK4 Lab 1 Report

Aaron Pinto
pintoa9

Raeed Hassan
hassam41

September 23, 2022

Random Number Generator Seeds

For the experiments in this lab, we used the same set of 18 random number seeds for all experiments. Experiment 2 instructs us to use at least 10 different random number generator seeds and that our *McMaster IDs* as had to be among these seeds. We used our *McMaster IDs* and shifted them by one digit at a time to create 9 different seeds from each our IDs, for a total of 18 different seeds. All the random number generator seeds can be seen in Table 1. In the C code used for the experiments, leading zeroes are removed.

400188200	400190637
001882004	001906374
018820040	019063740
188200400	190637400
882004001	906374001
820040018	063740019
200400188	637400190
004001882	374001906
040018820	740019063

Table 1: Random Number Generator Seeds

Experiment 2

If we set `SERVICE_TIME` to a value of 10, we need to set $0 <$ `ARRIVAL_RATE` < 0.1 to satisfy $0 <$ `ARRIVAL_RATE` $\times$ `SERVICE_TIME` < 1. We selected the following values for `ARRIVAL_RATE`: 0.001, 0.01, 0.03, 0.05, 0.07, 0.09, 0.095, 0.099. We doubled the variable `NUMBER_TO_SERVE` from the default to 50×10^6 to 100×10^6. We modified the provided code to run to loop through each arrival rate with all 18 random number generator seeds, and print out the average results of the seeds for each arrival rate. A plot of the average mean delay vs. arrival rate is shown in Figure 1.

At low arrival rate values, we see the mean delay approaches 10. The mean delay axis intercept at these low arrival rates is the service time, because customers will have shorter queue times when the arrival rate is low. Therefore the majority of their delay will become the service time, as queue times become shorter as arrival rates decrease. As the arrival rate increases and approaches the right-hand inequality in Expression 1, the mean delay begins to increase exponentially. As the arrival rate increases, there are on average more customers in the system at any time, and customers must wait in queue for longer before being serviced. The value of the arrival rate axis asymptote is 0.1, which is the upper limit of `ARRIVAL_RATE` that satisfies Expression 1 when the value of `SERVICE_TIME` is set to 10. The mean delay curve that is obtained is an exponential curve where the mean delay increases slowly at low values of the arrival rate but begins to increase significantly as the arrival rate reaches the upper arrival rate axis asymptote.

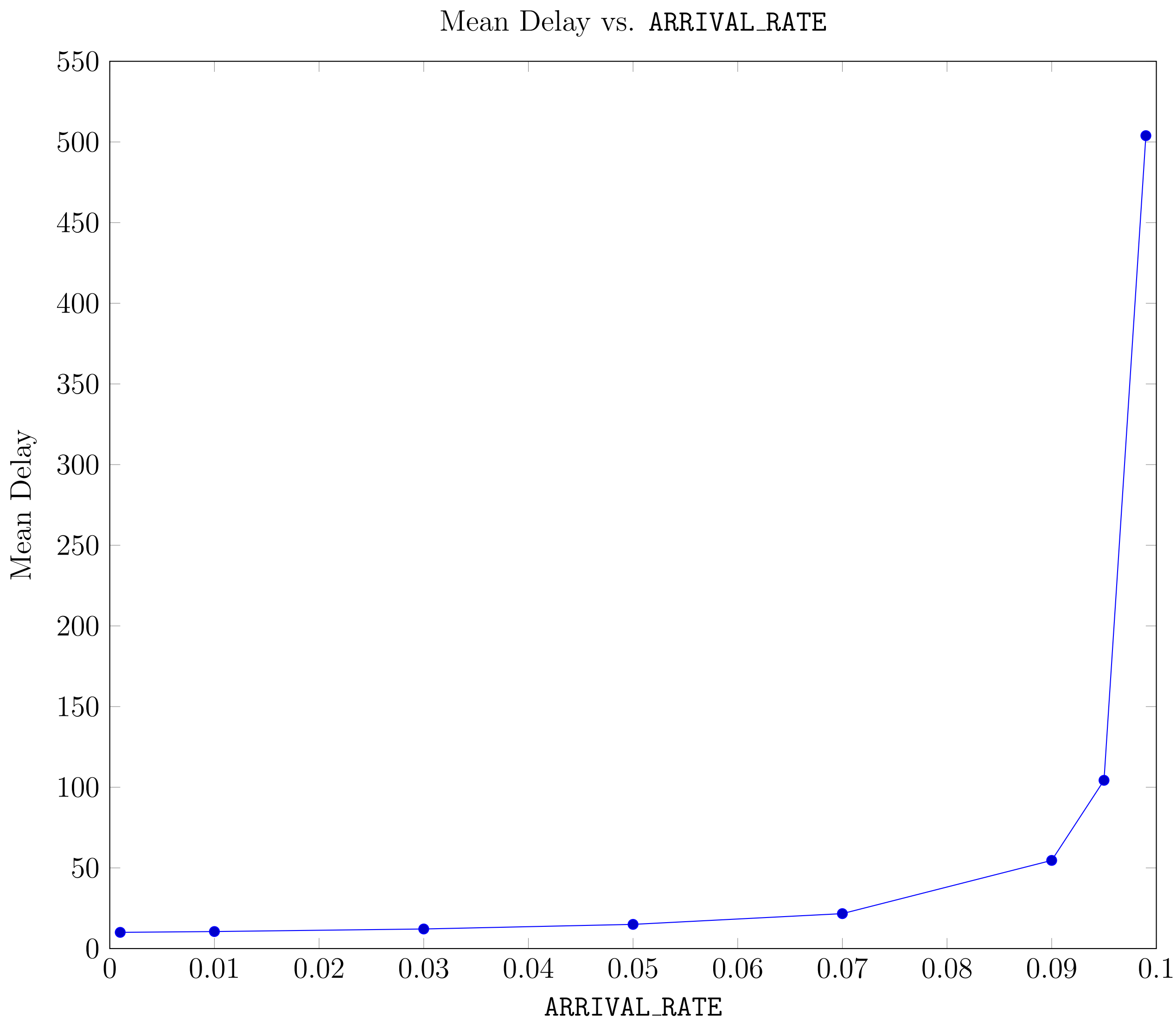

Figure 1: Experiment 2: Mean Delay vs. Arrival Rate

Experiment 3

We kept the same value of `SERVICE_TIME` and increased the `ARRIVAL_RATE` to 0.2, such that the product of `SERVICE_TIME` $\times$ `ARRIVAL_RATE` $= 2$. At 10,000 customers served the mean delay has increased to over 50,000, significantly more than largest value seen when the arrival rate was < 0.1. As we increase the run length (number of customers to serve), the mean delay begins to increase proportionally with the number of customers to serve as seen in Table 2. As the system becomes saturated, every customer entering the system will enter a queue and will need to wait a mean delay that is proportional to the number of customers already waiting in queue (each additional customer increases the mean delay by the same amount). The condition in Expression 1 is necessary to prevent such a scenario where increasing the number of customers will affect the mean delay, unlike the case where the arrival rate satisfies Expression 1. For example, we can see in Table 3 that the mean delay is not affected by the number of customers to serve. Satisfying this condition also ensures that we are able to empty out the queue and reach a steady state, otherwise the server will never be able to catchup as more customers arrive than the server can service.

Number of Customers to Serve	Mean Delay
10000	50160.191256
20000	99962.602135
50000	249997.485025
100000	499914.794732

Table 2: Experiment 3: Mean Delay when Arrival Rate = 0.2

Number of Customers to Serve	Mean Delay
10000	14.989001
20000	14.989132
50000	14.982623
100000	14.972480

Table 3: Experiment 3: Mean Delay when Arrival Rate = 0.05

Experiment 4

When setting the `SERVICE_TIME` to 30, we similarly adjusted our `ARRIVAL_RATE` values by dividing the rates used in experiment 2 by 3. We kept all other parameters the same (including the random number generator seeds) and reran the simulation. A plot of the mean delay vs. arrival rate of both experiments 2 and 4 are shown in Figure 2. We can see that in this experiment, with a larger service time the mean delay increases significantly faster at the same arrival rates.

Experiment 5

Experiment 6

Listing 1: Modifications to Experiment 6 Code

```
/* System state variables. */
int number_in_system = 0;
double next_arrival_time = 0;
double next_departure_time = 0;
double new_service_time = 0;

/* If this customer has arrived to an empty system, start its
   service right away. */
if (number_in_system == 1) {
```

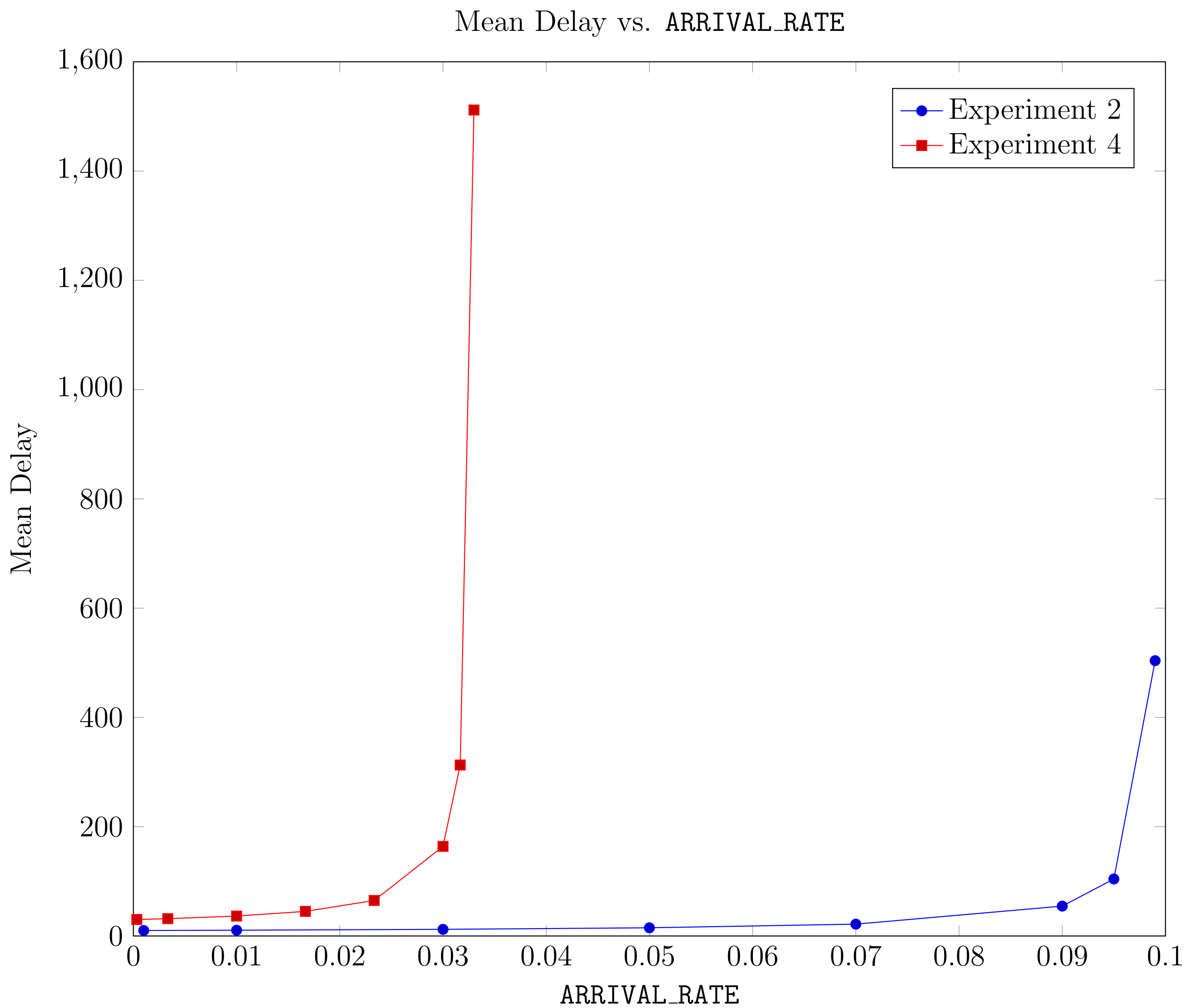

Figure 2: Experiment 4: Mean Delay vs. Arrival Rate

```
        new_service_time = exponential_generator((double)
            SERVICE_TIME);
        next_departure_time = clock + new_service_time;
}

number_in_system--;
total_served++;
total_busy_time += new_service_time;

/* If there are other customers waiting, start one in service
    right away. */
if (number_in_system > 0) {
```

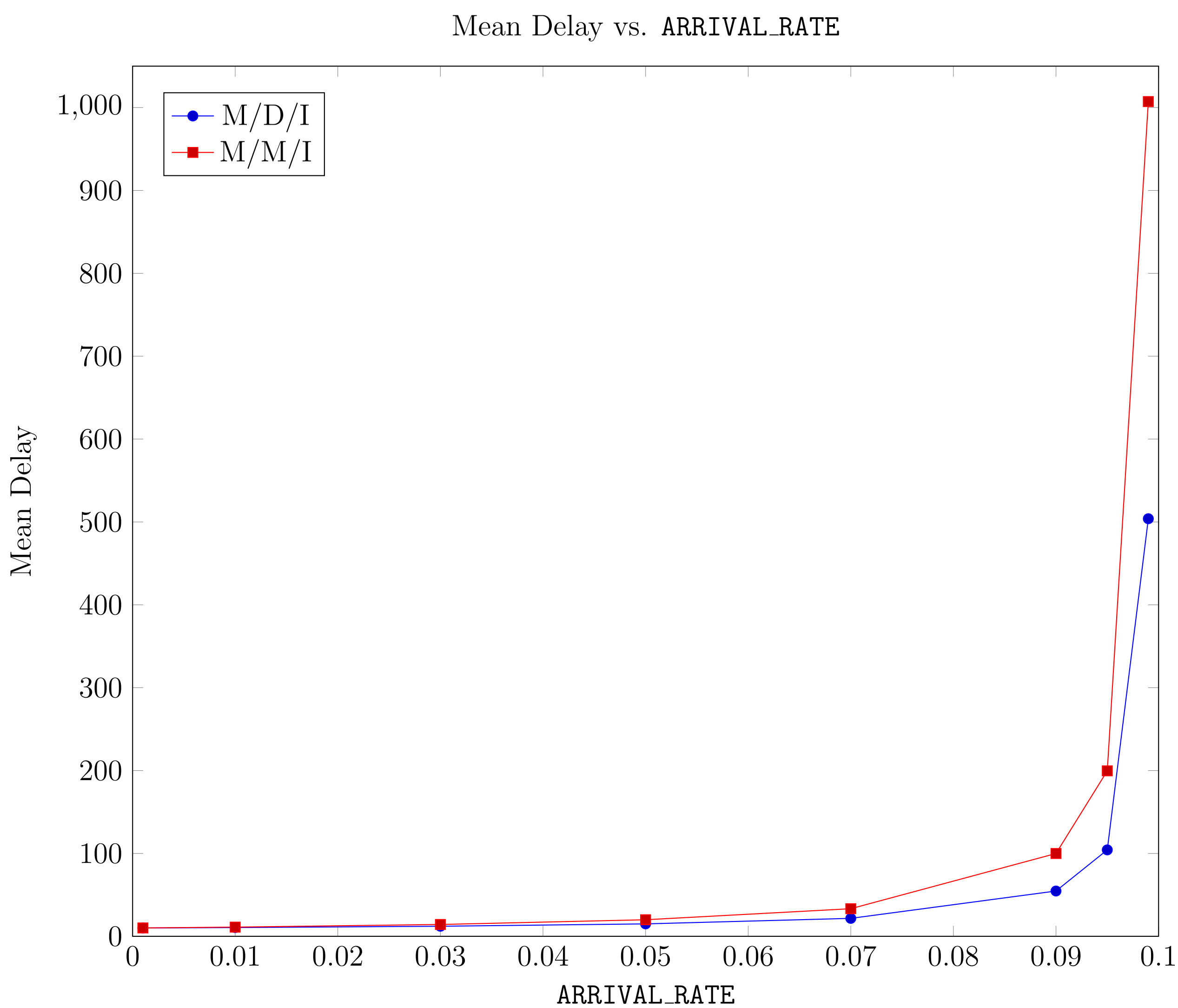

Figure 3: Experiment 6: Mean Delay vs. Arrival Rate

```
        new_service_time = exponential_generator((double)
          SERVICE_TIME);
        next_departure_time = clock + new_service_time;
}
```

Experiment 7

Experiment 8