Simulation Report: *A Simple Queuing Network Case*

Mingjie HU 21210690193

We are interested in a tandem queuing system with tow stations (see Figure1). Station 1 is a M/M/2 queue with service-time distribution being $Exp(1)$, and station 2 is a M/M/1 queue with service-time distribution being $Exp(2)$. Assume that the arrival to the system follows a Poisson process with rate λ and the system follows the first-come first serve queuing rule. Our main task is to implement a discrete-event simulation algorithm to simulate the system and analyze the average sojourn time under different service-time distribution assumption.

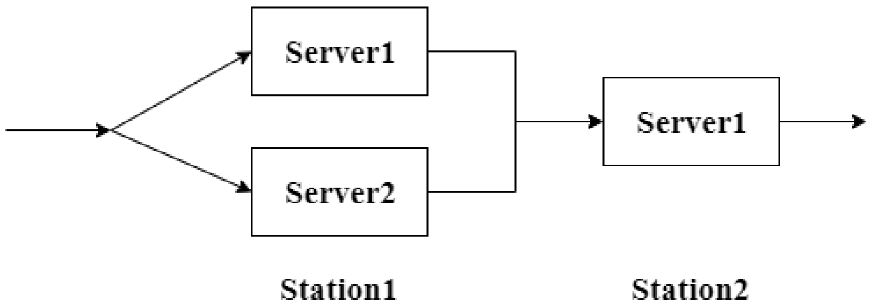

Figure 1: Queuing Network Structure

We define the state of the system as the number of customers at station 1 and station 2. The events in the system include: customer arrives at station 1 (denoted as 0), customer departure from server 1 at station 1 into station 2 (denoted 1), customer departure from server 2 at station 1 into station 2 (denoted 2), and customer departure from station 2 (denoted 3). The simulation clock jumps from event time to event time. When the simulation clock reaches the time threshold, the simulation ends. We design Algorithm 1 to simulate the queuing network.

To facilitate the interpretation of the algorithm, we define the following notation: λ represents customer arrival rate, μ_1, μ_2 denotes the service rate of single server in station 1 and the service rate of station 2, respectively, the simulation time threshold is T. $clock$ denotes current simulation time and $last_e vent_t ime$ denotes the time of last event. $next\_station1\_arrive$ represents the next time when customer arrives at station 1. $next\_station1\_server1\_finish, next\_station1\_server2\_finish$ represents the next time when server 1 and server 2 of station 1 completes the service respectively. $next\_station2\_finish$ represents the next time when station 2 completes the service.

We implement the Algorithm 1 in python. The original code can be found in the appendix or in github: `https://github.com/xiaohu8888/Queueing-System-Simulation.git`.

We use Algorithm 1 to simulate the queuing network in Figure 1. Firstly, we set $\lambda = 1.8$ and collect 1000 observations of the customer sojourn time after a warm-up period of 3000 time units. **Figure 2 shows the average sojourn time distribution of 1000 customers arriving after 3000 time units (with random seed=2). The average sojourn time under this setting is 11.67 time units**. However, this result is not robust. Under different random seed settings, the average sojourn time of 1000 customers may vary greatly. Therefore, we collect 10000 observations of the customer sojourn time after a warm-up period of 3000 time units. **Figure 3 shows the average sojourn time distribution of the 10000 customers. The average sojourn time under this setting is 10.34 time units**.

Algorithm 1: Queuing Network Simulation

Input: λ, μ_1, μ_2, T.

Output: Customer sojourn time.

initialization: $clock = last\_event\_time = 0$, $next\_station1\_arrive$=exponential($1/\lambda$), $next\_station1\_server1\_finish$=$+\infty$, $next\_station1\_server2\_finish$=$+\infty$, $next\_station2\_finish$=$+\infty$;
while $clock < T$ **do**
  Determine the next event and update the clock to the time of the next event;
  if *next event* =0 **then**
    Update customer waiting time: $clock - last\_event\_time$;
    if *server 1 of station 1 is idle and station 1 is not empty* **then**
      First customer in station 1 enters server 1 and $service\_time = exponential(\frac{1}{\mu_1})$;
      Update $next\_station1\_server1\_finish = clock + service\_time$;
    end
    if *server 2 of station 1 is idle and station 1 is not empty* **then**
      First customer in station 1 enters server 1 and $service\_time = exponential(\frac{1}{\mu_1})$;
      Update $next\_station1\_server2\_finish = clock + service\_time$;
    end
    if *server 1 of station 1 is idle and station 1 is empty* **then**
      Update $next\_station1\_server1\_finish = clock + service\_time = +\infty$;
    end
    if *server 2 of station 1 is idle and station 1 is empty* **then**
      $next\_station1\_server2\_finish = clock + service\_time = +\infty$;
    end
    $next\_station1\_arrive$=exponential($1/\lambda$)
  end
  if *next event* =1 **then**
    Update customer waiting time: $clock - last\_event\_time$;
    if *station 1 is not empty* **then**
      First customer in station 1 enters server 1 and $service\_time = exponential(\frac{1}{\mu_1})$;
      Update $next\_station1\_server1\_finish = clock + service\_time$;
    else
      Update $next\_station1\_server1\_finish = +\infty$;
    end
    if *station 2 is idle and not empty* **then**
      First customer in station 2 enters and $service\_time = exponential(\frac{1}{\mu_2})$;
      Update $next\_station2\_finish = clock + service\_time$;
    end
    if *station 2 is idle and empty* **then**
      Update $next\_station2\_finish = +\infty$;
    end
  end
  if *next event* = 2 **then**
    Similar to next event = 1
  end
  if *next event* = 3 **then**
    Update customer waiting time: $clock - last\_event\_time$;
    if *station 2 is not empty* **then**
      First customer in station 2 enters server and $service\_time = exponential(\frac{1}{\mu_2})$;
      Update $next\_station2\_finish = clock + service\_time$;
    else
      Update $next\_station2\_finish = clock + service\_time + \infty$;
    end
  end
  $last\_event\_time = clock$
end

In order to obtain a more accurate estimate of the average sojourn time, we set the time threshold $T = 10000$ and repeat the experiment 50 times with different random seed setting. Figure 4 shows distribution of the average sojourn time of different experiments. **The average sojourn time for the 50 experiments is 10.05 time units. The 95% confidence interval is** $[7.67, 12.42]$.

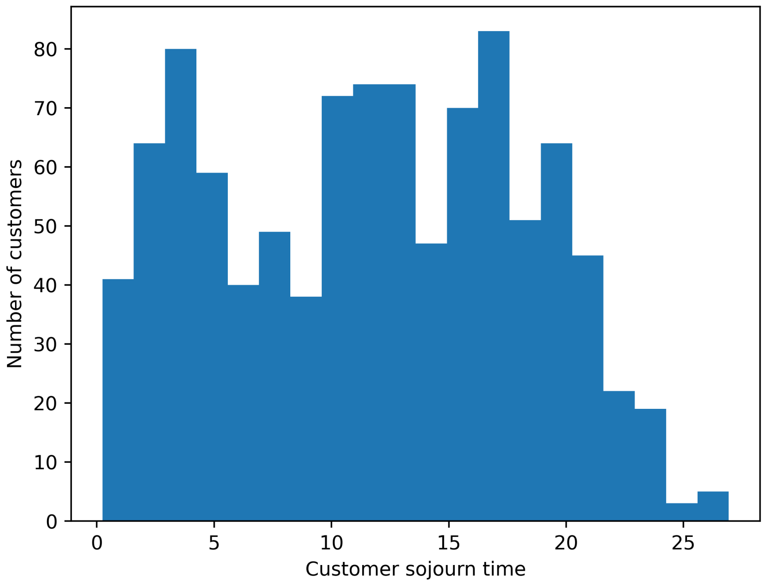

Figure 2: Histogram of the 1000 customers' sojourn time

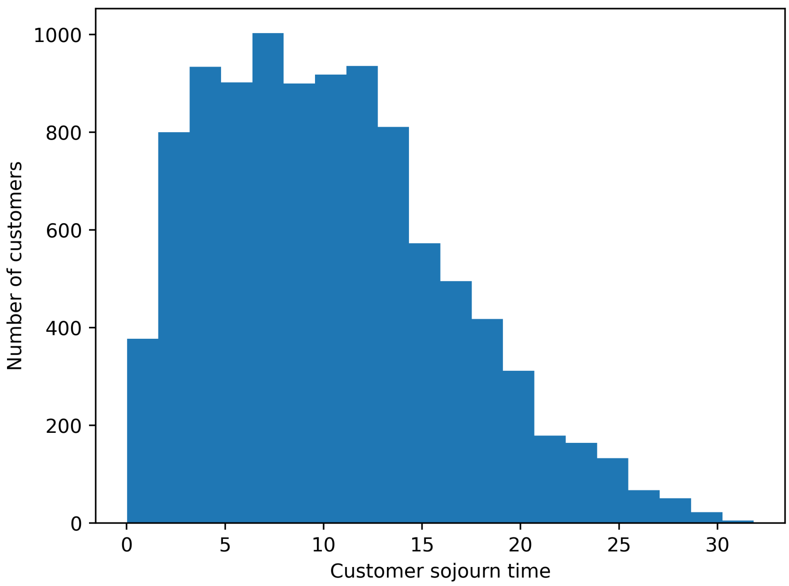

Figure 3: Histogram of the 10000 customers' sojourn time

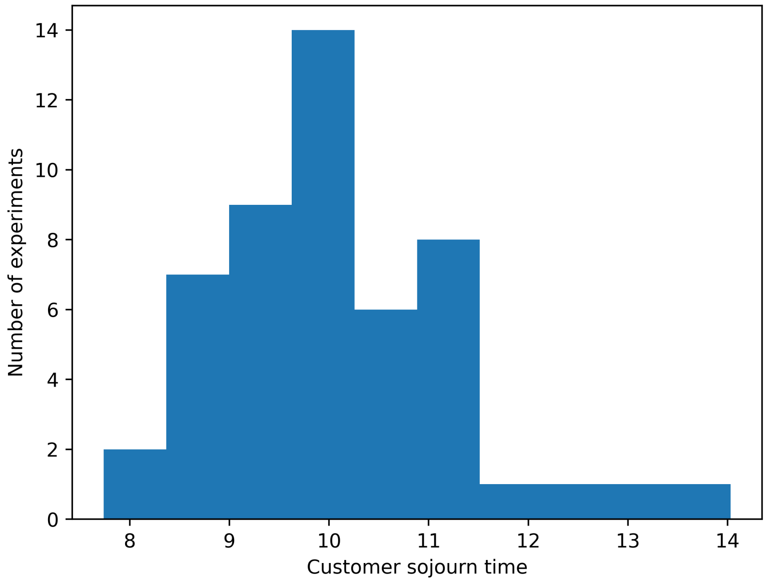

Figure 4: Average sojourn time of different experiments

Next, we use Jackson Network formula to calculate the theoretical value of the average sojourn time. Jackson Network is a class of queuing network with K arbitrarily connected (M/M/m) queues. Our queuing network structure can be viewed as an M/M/2 queue connected with an M/M/1 queue.

We first analyze the steady state of two queuing systems independently. For the M/M/2 queue, let m denote the number of servers, (m=2 in our problem), P_i represents the probability that there are i customers in the system when the system is in steady state, P_Q denotes the probability that an arriving customer has to wait in queue. Then we have the solution to the balance equation:

$$P_i = \begin{cases} P_0 \frac{(m\rho_1)^i}{i!}, & \text{for } i \leq m \\ P_0 \frac{m^m \rho_1^i}{m!}, & \text{for } i > m \end{cases}$$

where $\rho_1 = \frac{\lambda}{m\mu_1}$, because $\sum_{i=0}^{\infty} P_i = 1$, we have that:

$$P_0 = \left[\sum_{i=0}^{m-1} \frac{(m\rho_1)^i}{i!} + \frac{(m\rho_1)^m}{m!(1-\rho_1)} \right]^{-1}$$

$$P_Q = \frac{P_0 (m\rho_1)^m}{m!(1-\rho_1)}$$

According to Little's Rule, the average number of customers in the system is:

$$N_1 = \lambda T_1 = \frac{\lambda}{\mu_1} + \frac{\lambda P_Q}{m\mu_1 - \lambda} = m\rho_1 + \frac{\rho_1 P_Q}{1-\rho_1}$$

In our problem, $m = 2, \rho_1 = \frac{1.8}{2\times 1} = 0.9$, putting them into the formula, we can get: $P_Q = 0.8526$, $N_1 = 9.4734$. For the M/M/1 queue, the arrival rate equal to external arrivals (in our problem, it's zero) plus customers who finish service at station 1 and are then routed to station 2 for the next stage of service, therefore, $\lambda_2 = 0 + \lambda = 1.8$. The average number of customers in the system is: $N_2 = \frac{\rho_2}{1-\rho_2} = 9$. **For Jackson Network, the average sojourn time equals to**:

$$T = \frac{N}{\lambda} = \frac{N_1 + N_2}{\lambda} = 10.263$$

We next consider that the service time of server 2 follows a normal distribution with mean $\mu = 0.5$ and standard deviation $\sigma = 0.15$. In the process of randomly generating the service time, if the time is negative, it will be generated again. Repeat the experiment above, we can get the following results. **Figure 5 shows the average sojourn time distribution of 1000 customers arriving after 3000 time units (with random seed=2). The average sojourn time under this setting is 10.34 time units. Figure 6 shows the average sojourn time distribution of the 10000 customers. The average sojourn time under this setting is 7.01 time units**. This indicates that under this distribution of service time, the average sojourn time is shorter.

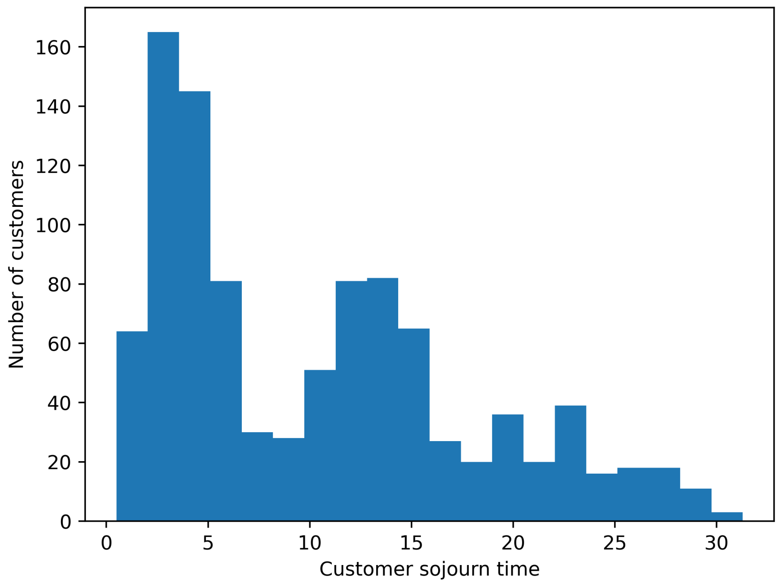

Figure 5: Histogram of the 1000 customers' sojourn time

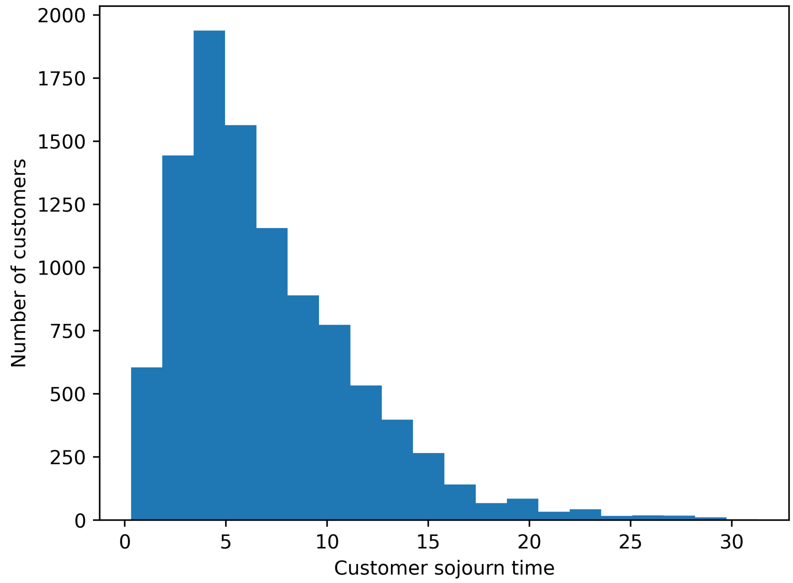

Figure 6: Histogram of the 10000 customers' sojourn time

Finally, we consider that the arrival rate λ satisfies that $0 < \lambda < 2$, **Figure 7 shows the average sojourn time against different λ. We can find that as λ grows up, the average sojourn time also increases**. When λ tends to 2, which means ρ goes to 1, the average sojourn time grows very large instantaneously.

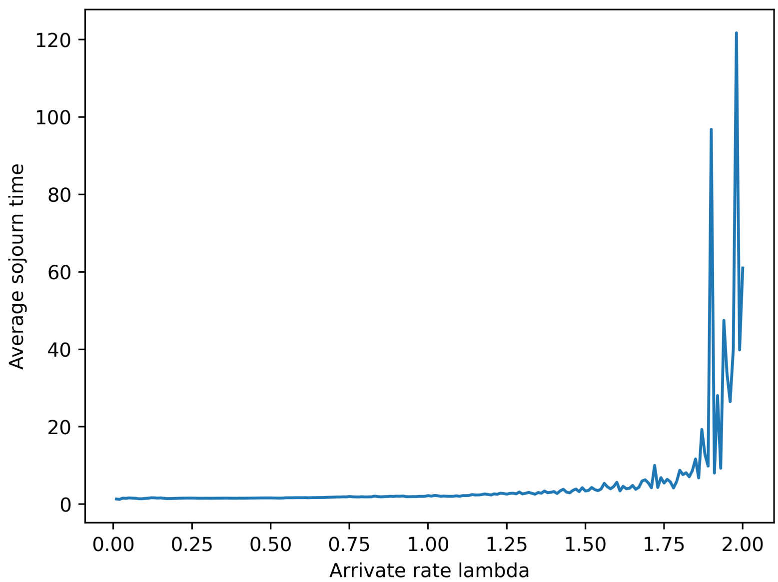

Figure 7: Average sojourn time of different lambda