Assigment 1: Discrete Event Modeling

Danis Alukaev (Task 2)
d.alukaev@innopolis.university
B19-DS-01

February 18, 2022

Abstract

The paper presents yet another simulation of a M/M/2/6 queue. Significant efforts were directed to compare empirical results to analytical approach with respect to twelve performance metrics.

Introduction

According to the proposed rule for choosing the task, I was given a task 2. The formulation is as follows:

> On average, cars with products arrive at some base each 30 minutes. The average unloading time of one machine is 1.5 hours. Unloading is carried out by two teams of loaders. On the territory of the base, no more than 4 cars can be in line waiting for unloading. Determine the performance of the queuing system.

Let's reformulate the task in the Kendall's notation [1, pp.12-13]. The described system is M/M/2/6, i.e. it has infinite arrival source of elements, where they stay for an exponentially distributed time, the service times are exponentially distributed, the service is carried out according to the request's arrival of $c = 2$ servers, and the system capacity $K = 6$. The system is also characterized by an arrival intensity $\lambda = 2$ elements per hour (short, e.p.h.) and service intensity $\mu = \frac{2}{3}$ e.p.h.

1 Analytical solution

The given system is a continuous stochastic process represented by states, where each state $S(t_m)$ at discrete time t_m depends only on previous state $S(t_{m-1})$ and changes the state according to exponential random variable. In this case, all flows of events are simplest: ordinary, stationary, without aftereffect. Thus, this is the Markov chain (see Figure 1).

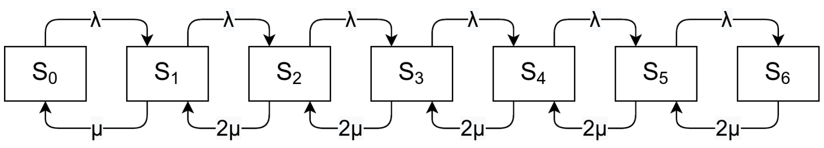

Figure 1: Graph of queue states

The states are labeled (see Table 1) according to number of elements that are served and resided in the queue, e.g. the state with subscript 0 stands for the case when there is no element in the system.

	S_0	S_1	S_2	S_3	S_4	S_5	S_6
Servers (in use)	0	1	2	2	2	2	2
Queue (occupied)	0	0	0	1	2	3	4

Table 1: Number of elements for states

1.1 Steady-state probabilities

Let $P_i(t)$ is the probability of the system to be at state S_i at time moment t, where $i \in [0..6]$. Note that the $\sum_{i=0}^{6} P_i(t) = 1$ holds for any t. In the current settings there can be build the system of Chapman–Kolmogorov differential equations (1).

$$\begin{cases} P_0' = \mu \cdot P_1 - \lambda \cdot P_0 \\ P_1' = 2\mu \cdot P_2 + \lambda \cdot P_0 - (\lambda + \mu) \cdot P_1 \\ P_2' = 2\mu \cdot P_3 + \lambda \cdot P_1 - (\lambda + 2 \cdot \mu) \cdot P_2 \\ P_3' = 2\mu \cdot P_4 + \lambda \cdot P_2 - (\lambda + 2 \cdot \mu) \cdot P_3 \\ P_4' = 2\mu \cdot P_5 + \lambda \cdot P_3 - (\lambda + 2 \cdot \mu) \cdot P_4 \\ P_5' = 2\mu \cdot P_6 + \lambda \cdot P_4 - (\lambda + 2 \cdot \mu) \cdot P_5 \\ P_6' = \lambda \cdot P_5 - 2 \cdot \mu \cdot P_6 \end{cases} \tag{1}$$

Since the marginal probabilities are constant, the correspondent derivatives are zeros. Recall that $\sum_{i=0}^{6} P_i(t) = 1$ holds for $\forall t$. Therefore, the system takes form of (2). Note that the system has a solution, because the rank of the matrix is equal to number of variables.

$$\begin{cases} \mu \cdot P_1 = \lambda \cdot P_0 \\ 2\mu \cdot P_2 + \lambda \cdot P_0 = (\lambda + \mu) \cdot P_1 \\ 2\mu \cdot P_3 + \lambda \cdot P_1 = (\lambda + 2 \cdot \mu) \cdot P_2 \\ 2\mu \cdot P_4 + \lambda \cdot P_2 = (\lambda + 2 \cdot \mu) \cdot P_3 \\ 2\mu \cdot P_5 + \lambda \cdot P_3 = (\lambda + 2 \cdot \mu) \cdot P_4 \\ 2\mu \cdot P_6 + \lambda \cdot P_4 = (\lambda + 2 \cdot \mu) \cdot P_5 \\ \lambda \cdot P_5 = 2 \cdot \mu \cdot P_6 \\ \sum_{i=0}^{6} P_i = 1 \end{cases} \tag{2}$$

The derivation of marginal probabilities $P_i, i \in [0..6]$ for birth-death process is the known problem and presented in [1]. For simplicity, let $\rho = \frac{\lambda}{\mu}$ is traffic intensity and $a = \frac{\rho}{c} \neq 1$ is traffic intensity per each server. Ultimately, there should be obtained following formulas [1, pp.94-95]:

$$P_0 = [\frac{\rho^c}{c!} \cdot \frac{1 - a^{K-c+1}}{1-a} + \sum_{n=0}^{c-1} \frac{\rho^n}{n!}]^{-1} \tag{3}$$

$$P_n = \begin{cases} \frac{\lambda^n}{n!\mu^n} P_0, & \text{for } n \in [0..c] \\ \frac{\lambda^n}{c^{n-c}c!\mu^n} P_0, & \text{for } n \in [c..K] \end{cases} \tag{4}$$

According to (3) and (4), there were computed marginal probabilities:

$$\begin{cases} P_0 \approx 0.0157869 \\ P_1 \approx 0.0473606 \\ P_2 \approx 0.0710409 \\ P_3 \approx 0.1065614 \\ P_4 \approx 0.1598421 \\ P_5 \approx 0.2397632 \\ P_6 \approx 0.3596448 \end{cases} \tag{5}$$

1.2 Performance measurements

The performance of a given system can be determined based on the 12 metrics proposed by G.Saakyan [2].

1. **Probability of reneging (rejecting) without waiting**
   The probability that the new element will be rejected. Such a situation appears when both servers are busy and queue is full, i.e. when the current state is S_K.

$$P_r = P_K \approx 0.3596448$$

2. **Relative queue throughput**
   The ratio of the average number of elements served by the system to the average number of elements requested the service per unit of time.

$$\Lambda_r = 1 - P_r \approx 0.6403552$$

3. **Absolute queue throughput**
   The average number of elements the system can serve per unit of time.

$$\Lambda_a = \lambda \cdot \Lambda_r \approx 1.2807104 \text{ e.p.h.}$$

4. **The mean busy period of the system**
   The average time the system serving the elements.

$$E\delta_r = \frac{1 - P_0}{\lambda P_0} \approx 31.1718292$$

.

5. **Utilization factor**
   The ratio of average time the system serving the elements with all servers occupied to the entire unit of time.

$$U_n = \frac{(1 - P_K)\rho}{c} \approx 0.9605328$$

6. **Mean queue length**
   The average number of elements waiting for service in the queue.

$$\overline{Q} = \frac{P_0 \rho^c a}{c!(1-a)^2}[1 - a^{K-c+1} - (1-a)(K-c+1)a^{K-c}] = 2.5841182$$

7. **Distribution at the moment of arrival into the system**
   The probability distribution that there are $n \in [0..5]$ customers in the system given a customer is about to enter into the system, i.e. $\Pi_n = \frac{P_n}{1-P_K}$.

$$\{\Pi_i = \frac{P_i}{1 - P_K} : i \in [0..5]\} = \{0.0246533, 0.0739599, 0.1109398, 0.1664098, 0.2496147, 0.3744222\}$$

8. **Mean waiting time**
   The average time the element waits for service in the queue.

$$\overline{W} = \frac{\overline{Q}}{\lambda(1 - P_K)} = \sum_{k=c}^{K-1} \frac{(k - c + 1)}{c\mu} \Pi_k = 2.0177193$$

9. **Mean response time**
   The average time the element resides inside the system.

$$\overline{T} = \overline{W} + \frac{1}{\mu} = 3.5177193$$

10. **Probability the request is immediately accepted**
    The probability that the new request is serviced without waiting.

$$P_a = \sum_{i=0}^{c-1} P_i = 0.0631475$$

11. **Distribution of the waiting time**
    The probability distribution function for the waiting time in queue.

$$F_W(t) = 1 - \sum_{n=c}^{K-1} \Pi_n \sum_{i=0}^{n-c} \frac{(c\mu t)^i e^{-c\mu t}}{i!} = 1 - 2.0662555 \cdot (\frac{4}{3}t)^i e^{-\frac{4}{3}t}$$

12. **Mean number of customers in the system**
    The average number of cars that are served or reside in the queue.

$$\overline{N} = \overline{Q} + \rho(1 - P_K) = 4.5051838$$

2 Simulation model

The implementation is written in Python programming language of version 3.8.12. As a framework for simulation modeling there was used SimPy 4.0.1. Navigate to the directory with unzipped files submitted to Canvas. Start Jupyter notebook from there. Make sure that you have permissions to modify the directory. Further, activate the environment with appropriate version of Python and SimPy. Congratulations, now you can observe my code cell by cell.

On setup program will create a directory called "logs", where log files for each simulation will be stored. The name of log file is a conjunction of date and time, when simulation was started. Each log entry has a form of "TIME ID ACTION", where the "TIME" is a time moment inside the environment, "ID" is an id of a car, and "ACTION" is either "accepted" or "unloaded" or "left" or "rejected".

2.1 Design

In the frame of work there were introduced two abstractions *ProductBase* and *Car* establishing settings for a simulation. Besides, there are multiple methods that complement the functionality, and make the code reusable.

Method *get_random_interval* returns a time interval taken from exponential distribution with a given mean value at random.

ProductBase represents a product base, to which suppliers send trucks with goods. It possesses the loader resources, maintains queue, and implements the unloading functionality. The time required by loaders is taken via *get_random_interval* with mean of 90 minutes.

Car represents the truck carrying the goods from suppliers. It implements arriving mechanism:

- If the queue is full, the truck is rejected by the product base.
- Otherwise
  - If the server is available, the truck is accepted by a product base for unloading.
  - Else, the truck's id is added to queue, so that the loaders will proceed with unloading when the turn came.

Method *setup* is in charge of establishing the flow of events. In the endless loop it creates cars arriving to product base (which is instantiated there). Each car invokes method *arrive* (see prev. paragraph). The time intervals between arrivals are taken via *get_random_interval* with mean of 30 minutes.

Method *simulate* sets up the logging, creates an environment, and runs the *setup* method as a process in the environment.

3 Convergence estimation

Let's start with formalizing term "stationary" for a numerical implementation: the observations $X = \{x_0..x_T\}$ said to converge in time moment T iff $|x_T - x_i| < \epsilon$, for $i \in [1..5]$. The value x_T here is a numerical approximation of a limit on the integer realization and $\alpha = |x_T - x_i|$, for $i \in [1..5]$ is a convergence rate.

To conduct experiments in the same notebook there was implemented sandbox. It allows collection of the empirical observations for performance metrics 1, 2, 3, 6, 8, 9, 10, 12. The x-axis was set to be the time $t \in [1..20000]$, whereas the y-axis devoted for the mean value of metric at a given moment. There were tested different configurations, but sufficient enough appears the number of simulations $N = 100$ and $\epsilon = 0.0001$.

Figures 1-9 demonstrate the convergence of an empirical result (blue line) to the analytical result (red line) from the left and convergence rate α (blue line) to the chosen ϵ (red line) on the right. As you can see most of the empirical results converge to the analytical solution in an exponential manner.

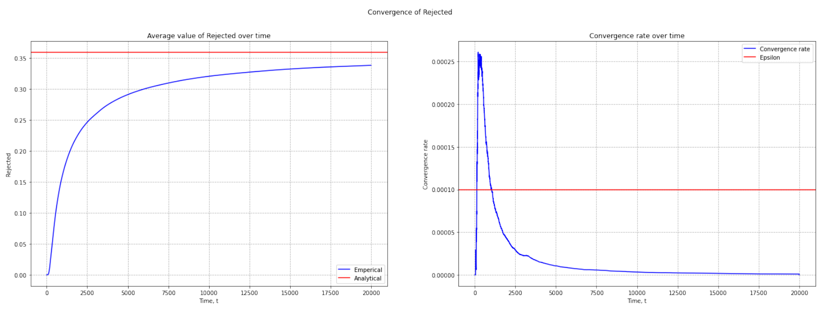

Figure 2: Probability of reneging (rejecting) without waiting

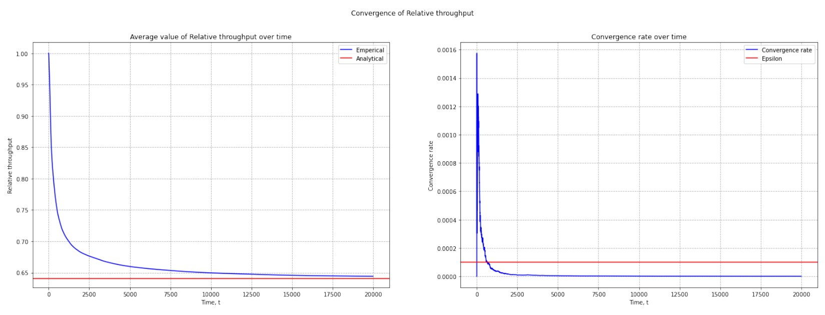

Figure 3: Relative queue throughput

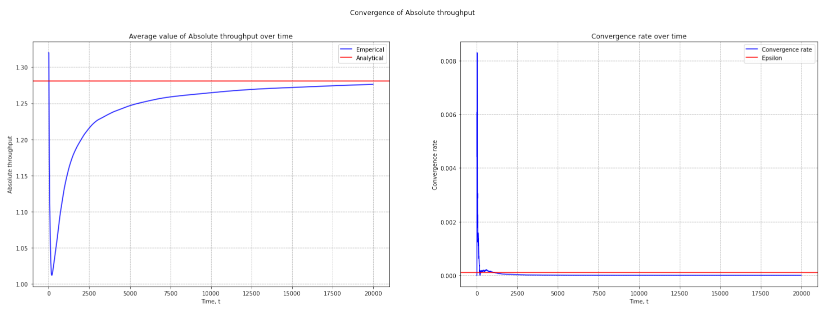

Figure 4: Absolute queue throughput

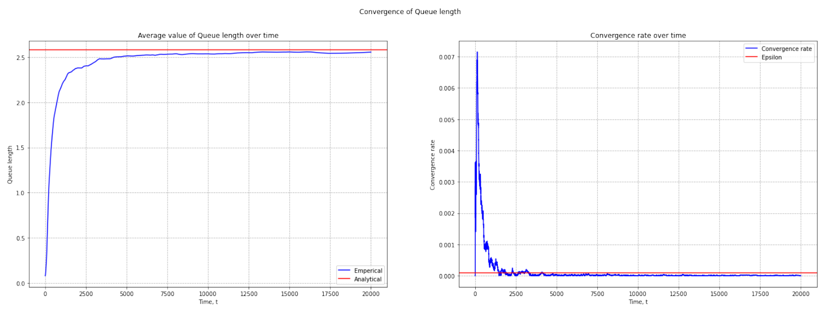

Figure 5: Mean queue length

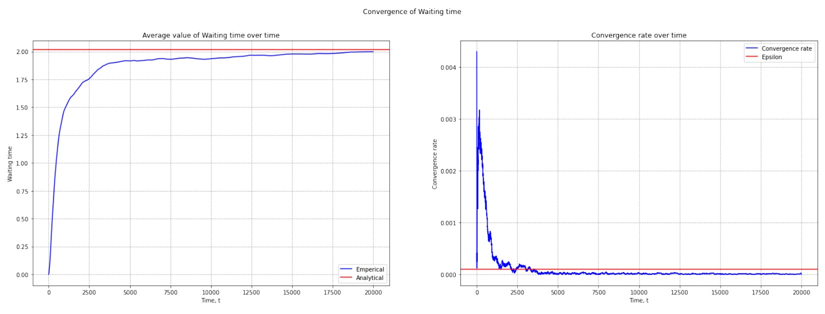

Figure 6: Mean waiting time

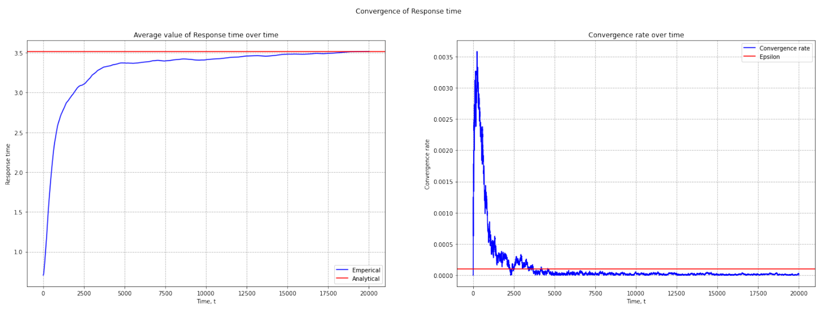

Figure 7: Mean response time

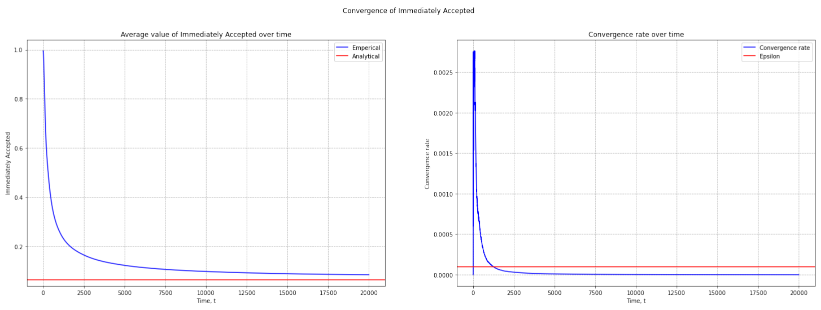

Figure 8: Probability the request is immediately accepted

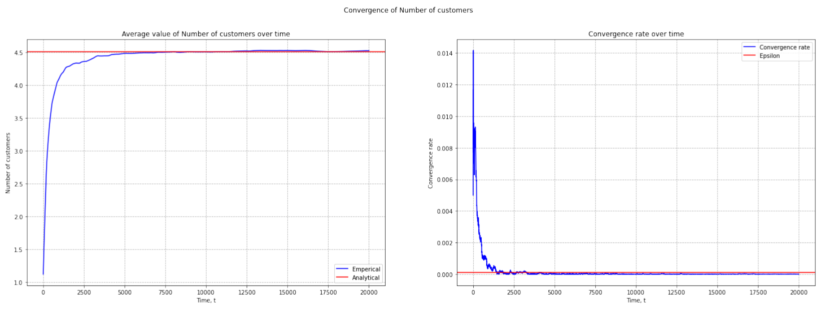

Figure 9: Mean number customers in the system

To measure the error of numerical approximation, there was used Mean-Absolute Error (MAE). There results for 10000 simulations presented in Table 2.

Metric	MAE
Mean Queue length	$2.56 \cdot 10^{-8}$
Mean Number customers	0.0625
Mean Waiting time	0.0430
Mean Response time	0.0952
Probability of reneging (rejecting) without waiting	0.0902
Probability the request is immediately accepted	0.0218
Relative queue throughput	0.0097
Absolute queue throughput	0.0249

Table 2: Number of elements for states

References

[1] J. Sztrik, *Basic Queueing Theory*, 1st ed., Saarbrucken, Germany, GlobeEdit, 2016.

[2] G. Saakyan, *Queuing Theory*, Shahty, Russia, South Russian State University of Economics and Service, 2006.