Single Server Queuing System Simulation

MD. Irfanur Rahman Rafio
Department of Computer Science and Engineering
Islamic University of Technology
Dhaka, Bangladesh
ID: 190041125
irfanurrahmanrafio@gmail.com

***Abstract*—Single Server Queuing System is a basic simulation model that can be used to predict the behavior of a system with a single server serving jobs in a single queue. In this experiment, we simulate a Single Server Queuing System of our own and calculate its results. We then address the performance of this system by making a comparison of the simulated results with the analytical results. We analyze how the output variables deviate in the simulation result from the analytical values by graphically representing them. Thereafter, we verify the simulation and estimate the correctness of our simulation results.**

***Index Terms*—Single Server Queuing System, Simulation, Server, Arrival Event, Departure Event, Exponential Distribution**

I. Introduction

A Single Server Queuing System is a simple simulation modeling system. It consists of a single serving station that can serve only one job at a time, and a queue of infinite length that will supply jobs to the server. However, in practical implementations, the queue is of finite length.

The Single Server Queuing System is essential in conceptually understanding more complex simulation systems such as Multi-Server Queuing System, and Inventory System. The core concept of SSQS revolves around queuing theory which is a set of quantitative mathematical techniques used to construct queuing systems.

In the upcoming section, we will have a look at the description of the SSQS which consists of the state variables, events, input, and output variables of the system. Then we will go through the simulation program, its classes, and their relationships. The flowcharts of some of the major functions of SSQS will be provided. Then, the simulation and analytical solution's data will be presented, and a graphical comparison will be drawn. Finally, we will conclude the graphical comparison of the simulation and analytical results.

II. System Description

A. Problem Statement

Let's consider a Single Server Queuing System for which the inter-arrival times a_1, a_2, ... are independent and identically distributed (IID) random variables. A job that arrives and finds the server idle enters service immediately, and the service times s_1, s_2, ... of the successive jobs are IID random variables that are independent of the inter-arrival times. A job that arrives and finds the server busy joins the end of a single queue. Upon completing service for a job, the server chooses a job from the queue (if any) in a first-in, first-out (FIFO) manner. The simulation will begin in the 'empty-and-idle' state; i.e., no job is present and the server is idle. At time 0, we will begin waiting for the arrival of the first job, which will occur after the first inter-arrival time, a_1, rather than at time 0. We wish to simulate this system until a fixed number (n) of jobs have completed their delays in queue; i.e., the simulation will stop when the n-th job leaves service. Here, the time the simulation ends is thus a random variable, depending on the observed values for the inter-arrival and service-time random variables.

B. Input Variables

In our simulation, we have assumed that the inter-arrival times are distributed exponentially with mean $1/\lambda$, the service times are distributed exponentially with mean $1/\mu$, and $\lambda < \mu$. The simulation will stop when n=100 jobs leave service.

$$\lambda = \text{average arrival rate}$$
$$\mu = \text{average service rate}$$

C. State Variables

The state of the system in a certain point of time t is represented by two variables: server status x(t) and queue length q(t).

$$x(t) = \begin{cases} 0, & \text{if server is idle at time t} \\ 1, & \text{if server is busy at time t} \end{cases}$$

$$q(t) = \text{queue length at time t}$$

D. State Space

The system state can be denoted by a pair of numbers where the first element represents the server status and the second one represents the queue length. So, the state space of the system can be defined as a set of pairs consisting of the valid states of the system.

$$\begin{aligned} \text{State Space, } X &= \{(0,0),(1,0)\} \cup (\{1\} \times \mathbb{N}) \\ &= \{(0,0),(1,0),(1,1),(1,2),(1,3),...\} \end{aligned}$$

E. Event Set

The system has two events: Arrival(a) and Departure(d). Arrival either changes the server status from idle to busy, or increases the queue length. Departure either changes the server status from busy to idle, or decreases the queue length.

$$\text{Event Set, } E = \{a, d\}$$

F. Feasible Event Set

When the system is idle, Departure event is not possible. Otherwise, any event is feasible in any point of time.

$$\text{Feasible Event Set, } F(t) = \begin{cases} \{a\}, & \text{if } x(t) = 0 \\ \{a, d\}, & \text{if } x(t) = 1 \end{cases}$$

G. State Equations

The state equations of the system:

$$x(t^+) = \begin{cases} x(t) == 0 ? \, 1 : x(t), & \text{arrival at time t} \\ q(t) == 0 ? \, 0 : x(t), & \text{departure at time t} \\ x(t), & \text{otherwise} \end{cases}$$

$$q(t^+) = \begin{cases} x(t) == 0 ? \, 0 : q(t) + 1, & \text{arrival at time t} \\ q(t) == 0 ? \, 0 : q(t) - 1, & \text{departure at time t} \\ q(t), & \text{otherwise} \end{cases}$$

H. Statistical and Output Variables

1) Variables associated with job i:

$$\begin{aligned} \text{arrival time} &= a_i \geq 0 \\ \text{service start time} &= b_i \geq a_i \\ \text{queue delay} &= d_i = b_i - a_i \\ \text{service time} &= s_i \geq 0 \\ \text{system delay} &= w_i = d_i + s_i \end{aligned}$$

2) Variables associated with system:

$$\begin{aligned} \text{total run-time} &= T \\ \text{traffic intensity} &= \rho = \lambda/\mu \\ \text{server utilization} &= \bar{u} \end{aligned}$$

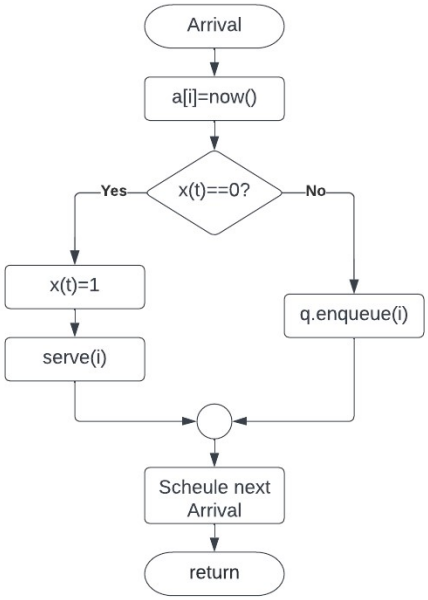
I. Output Equations

Output equations for measuring the performance of the system:

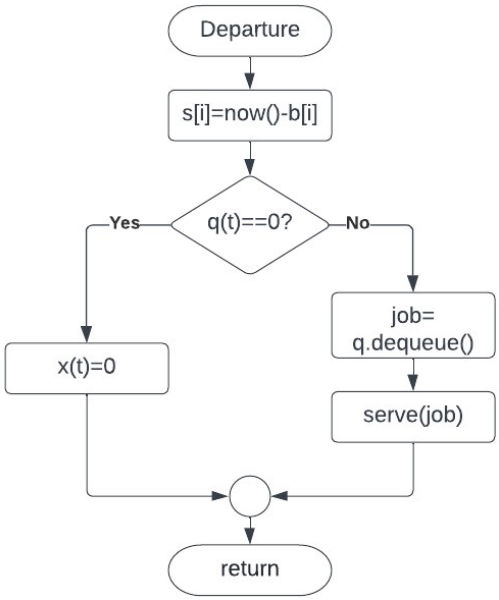
$$\begin{aligned} \text{maximum system delay, } w_{max} &= \max_{i=1}^{n}(w_i) \\ \text{average system delay, } \bar{w} &= \frac{\sum_{i=1}^{n}(w_i)}{n} \\ \text{maximum queue length, } q_{max} &= \max_{i=1}^{n}(q_i) \\ \text{average queue length, } \bar{q} &= \frac{\sum_{i=1}^{n}(q_i)}{n} \\ \text{server utilization, } \bar{u} &= \int_0^T x(t)dt \end{aligned}$$

J. Event Routines

1) Arrival Event: When a new job arrives in the system, we save its arrival time. Then we check if the server is idle. If it is, we change its status to busy and we immediately serve the job. If not, we add the job to the queue. After that we schedule the arrival of the next job. For determining its time, we apply the inverse transform method using the inter-arrival rate as a parameter.



2) Departure Event: After the service of a job ends, we save its service time. Then we check if the queue is empty. If the queue is found empty, there is no job to service at the moment. So, we change the server status to idle. Otherwise, we pick the first job from the queue and start serving it.

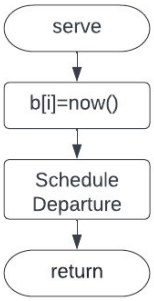


III. Simulation Program Description

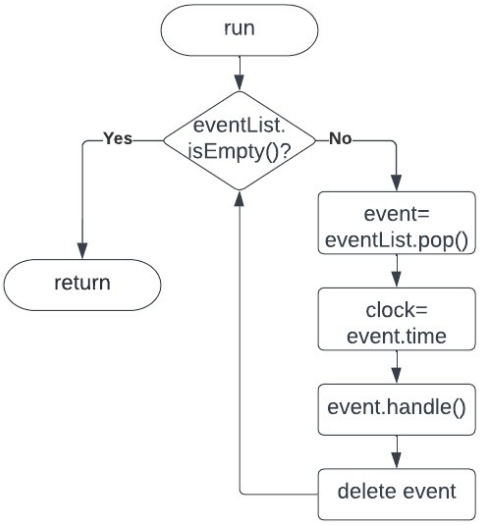
The program consists of three major classes: Simulator, Server and Event. Minor classes include the Job class, a FIFO Queue and a min Heap.

The event class is abstract. Its child classes represent the arrival and departure event. Each of those classes have a handler function which manipulates the state of the Server.

The server class consists of the system variables, the queue, and the statistical variables. Handling an event involves updating the statistical variables and creating changes to the system variables. There is a utility function in the Server class that performs the task of serving the jobs.



Finally, we have the Simulator class which mainly comprises of the event list and system clock. The event list is a min heap where the upcoming events are stored. The Simulator has a run function which pops an event from the event list and handles it. In every occurrence of a new event, the clock is updated.



The full source code of the simulation program can be found here: https://github.com/rafio-iut/Simulation-and-Modeling-Lab/tree/main/Single%20Server%20Queueing%20System

IV. Result

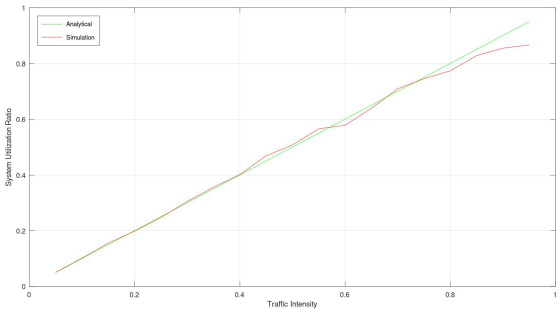
TABLE I
SSQS Simulation Results

$1/\lambda$	$1/\mu$	ρ	w_{max}	$\bar{w}$	q_{max}	$\bar{q}$	$\bar{u}$	$1-\bar{u}$	T
5	0.25	0.05	1.24187	0.249687	0.166667	0.00258284	0.0505365	0.949463	498.241
5	0.5	0.1	2.65773	0.501835	0.5	0.0111389	0.101575	0.898425	498.679
5	0.75	0.15	3.70862	0.746121	1	0.025697	0.155399	0.844601	487.231
5	1	0.2	5.0496	1.00395	1.06667	0.0470178	0.198424	0.801576	493.337
5	1.25	0.25	6.70612	1.2673	1.76667	0.0780003	0.247247	0.752753	510.568
5	1.5	0.3	7.49599	1.48942	2.1	0.122917	0.304517	0.695483	496.035
5	1.75	0.35	8.26174	1.77052	2.66667	0.18675	0.355936	0.644064	500.396
5	2	0.4	9.81749	1.99267	3.1	0.245846	0.401192	0.598808	506.916
5	2.25	0.45	11.1518	2.29394	3.63333	0.389036	0.468599	0.531401	499.149
5	2.5	0.5	12.5119	2.58936	3.9	0.500325	0.507179	0.492821	512.639
5	2.75	0.55	13.4743	2.79364	4.43333	0.724461	0.565587	0.434413	495.979
5	3	0.6	14.3506	2.95533	5	0.727851	0.577994	0.422006	504.553
5	3.25	0.65	15.4372	3.25131	5.5	1.04619	0.638927	0.361073	516.156
5	3.5	0.7	19.9515	3.63896	6.93333	1.56561	0.70912	0.29088	508.456
5	3.75	0.75	21.3722	4.00301	7.3	1.80357	0.745059	0.254941	516.391
5	4	0.8	27.41	5.05222	8.9	2.6851	0.773338	0.226662	528.91
5	4.25	0.85	26.0539	4.78189	8.86667	2.75117	0.82786	0.17214	512.067
5	4.5	0.9	39.102	6.33196	10.8333	4.06171	0.854646	0.145354	526.618
5	4.75	0.95	52.7852	8.08923	14.4	5.5026	0.866256	0.133744	553.395

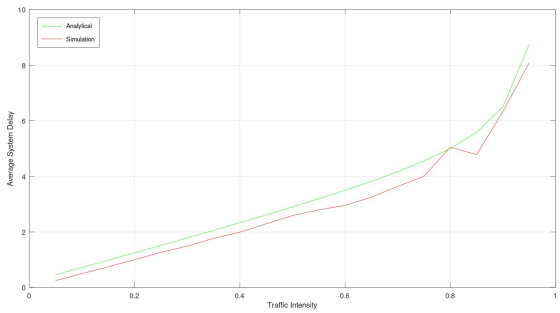
The simulation results are taken by running the simulation 30 times for different values of traffic intensity ρ. The arrival mean was fixed to 5 and by changing the service rate, we got different values of traffic intensity. The traffic intensity ranges from 0.05 to 0.95 with a 0.05 increase in each step. The same values of traffic intensity were used to find the analytical results. We used the theoretical formulas from the analytical solution of an SSQS to determine the values of the output statistics.

V. Analysis

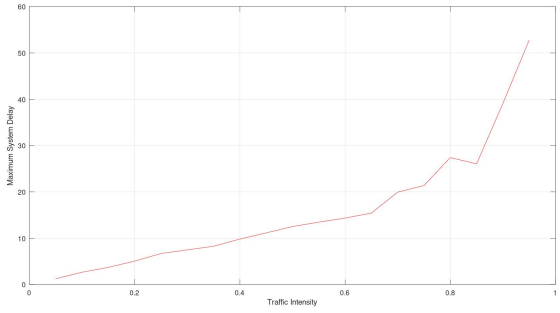
1) Utilization Ratio: The analytical server utilization of the system is equal to the traffic intensity. The simulation results display a graph which is almost a straight line with slope 1. This almost exactly matches with the analytical results.



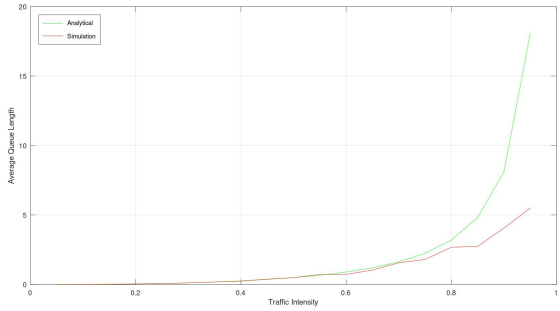
2) Average System Delay: The analytical average system delay is equal to $\frac{\rho^2}{1-\rho}$. Here, the simulation and analytical results look identical.



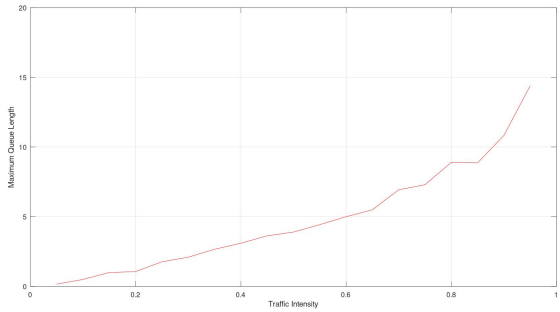
3) Maximum System Delay: Although the maximum system delay is a probabilistic statistic, its graph resemble a curve somewhat similar to the average system delay curve.



4) Average Queue Length: The simulation and analytical results are similar up to 0.8 traffic intensity but the analytical values increased sharply afterward. Both of them has the characteristics of an exponential graph.



5) Maximum Queue Length: The simulation results show that the maximum queue length is quite higher than the average which is an indication of the high variance of the exponential random variable.



VI. Conclusion

The performance of the simulation was almost identical to the values we found from the analytical solution. The server utilization graph for both the simulation and analytical solution were identical but the other graphs deviated for traffic intensities higher than 0.8. For such high values of traffic intensity, the analytical solution was producing greater values of the output statistics. There is, however, a justification for that. The simulation ran for a finite number of items for a finite duration. But the analytical solution does not consider such constraints. As traffic intensity approaches 1, the analytical statistical values tend towards infinity.

Overall, the simulation succeeded in producing results within the expected range and so the performance of the SSQS simulation was satisfactory.