M/M/1 QUEUE SIMULATION

Gastón Amengual
Simulation
National Technological University
gastonamengual@icloud.com

ABSTRACT

Queues are time and resource consuming, and planning and allocating them can be a daunting task. However, many situations that consist of queues can be modeled and subsequently simulated, to observe the state of the system after a certain time. In this work, a particular case of queues will be studied, the M/M/1 model, for which a simulation will be developed, and its performance measures will be both graphically and analytically analyzed.

1 Introduction

Queues are a recurring situation in many scenarios, from people waiting in banks [1] and supermarkets [**?**], to the flow of patients in hospitals [2]. Without the support of the appropriate tools, decision-making on issues related to the behavior of these queues results in very complex analysis and very difficult to predict. It is then where the field of simulations comes into play, which became a fundamental part of the analysis in queuing theory, since they allow to run models of real scenarios and observe their evolution in a controlled environment, allowing to predict the effects of the decisions that will be taken.

First, the characteristics of discrete event simulations will be described. Then, the particularities of the *M/M/1* queuing model will be described, and how a simulation of it will be approached and evaluated. Finally, the results obtained will be presented, and the conclusions taken will be detailed.

2 Discrete-event Simulation

Discrete-event simulation [3] consists of modeling a system as it evolves, through a representation in which the variables change instantaneously at separate points in time, that is, the system changes only in number finite times in time. These points are those at which an event occurs. An event is defined as an instantaneous occurrence that can change the state of the system.

Due to the dynamic nature of discrete simulation models, the current value of simulated time must be tracked as the simulation progresses, and a mechanism that advances simulated time from one value to another is also needed. The variable that represents the current value of the simulated time is called *simulation clock* (henceforth *clock*). A mechanism for advancing the clock is called *next-event time advance*. With this method, the clock is initialized at 0, and the occurrence times of future events are determined. The clock then advances to the time of the occurrence of the most imminent event, at which point the status of the system is updated to account for an event that occurred. This process of advancing the clock from one event to another is continued until some stop condition is satisfied.

3 M/M/1 Queue

The **M/M/1 queue** represents the length of the queue on a system consisting of a single server and is one type of discrete event simulations.

The system operates as follows. Suppose a customer that logs into the system to obtain a service. When he or she arrives, he or she goes into service if the server is available, or joins the queue if the server is busy. Each time the server completes service for a customer, it begins serving the customer that has waited for the most (if customers are waiting), or it remains available until the next customer arrives (if there are no customers). The events are the arrival of a customer and the departure of a customer.

Throughout this work, the following notation will be used:

- t_i: time of arrival of the ith customer.
- $A_i = t_i - t_{i-1}$: interarrival time between (i-1)st and ith customer.
- S_i: time that the server spends serving ith customer.
- D_i: delay in the queue of the ith customer.
- $c_i = t_i + D_i + S_i$: time that ith customer completes service and departs.
- e_i: time of occurrence of the ith event (both arrival and departure), the ith value the simulation takes.

Moreover, this case study makes the following assumptions:

- t_i follows a Poisson process.
- A_i is a random variable that follows an exponential distribution with mean $E[X]_c = \frac{1}{\lambda_c}$.
- S_i is an independent random variable that follows an exponential distribution with mean $E[X]_s = \frac{1}{\lambda_s}$.
- The capacity of the queue is infinite.
- The queue discipline is FIFO (First In First Out).

4 Variable under study

Two approaches can be considered. In the first, the variable under study is the time it takes n given customers to use the server. In the second, the variable under study is the number of customers that use the server in a given time T.

In this case study, the first approach will be used, being then the variable under study ***T**: total time required for n customers to use the server*, where $T > 0$ and $n \in \mathbb{Z}^+$. Being dependent on n, T will be expressed as $T(n)$. Analytically, $T(n) = n \cdot \lambda_c$, that is, the number of customers times the rate of time between arrivals. $T(n)$ is measured in the same unit as λ_c.

The **sample average of the total time** in k simulations is defined as $\overline{T}(n) = \dfrac{1}{k} \sum_{i=1}^{k} T(n)_i$.

5 Performance measures

Let k be the number of runs in which the system simulation is run.

5.1 Average expected waiting time in queue

If $\overline{D}(n) = \dfrac{\sum_{i=1}^{n} D_i}{n}$, $D_i \geq 0$ is the expected waiting time in a queue n customers, then

$\overline{\overline{D}(n)} = \dfrac{1}{k} \sum_{i=1}^{k} \overline{D}(n)_i$ is the average expected waiting time in a queue of n customers.

5.2 Average expected customers in queue

Let $Q(t)$ be the number of customers in the queue in time t, $t \geq 0$. For any time t between 0 and $T(n)$, $Q(t)$ is a non-negative integer. If p_i is the expected proportion of the time in which $Q(t) = i$, then

$\overline{Q}(n) = \sum_{i=0}^{\infty} i p_i$

If Tc_i is the total time in which the queue has i length, then $T(n) = Tc_0 + Tc_1 + Tc_2 + ...$ and $\overline{p}_i = Tc_i / T(n)$. It follows that

$\overline{Q}(n) = \dfrac{\sum_{i=0}^{\infty} i Tc_i}{T(n)}$

The numerator is the area under the curve of $Q(t)$. Then,

$$\overline{Q}(n) = \frac{\displaystyle\int_0^{T(n)} Q(t)dt}{T(n)}$$

The area under the curve is incremented every time in the area under the rectangle under $Q(t)$ in the last interval from the last event, that is the width *time since the last event* multiplied by the height, *customers in the queue*.

$\overline{\overline{Q}(n)} = \frac{1}{k} \sum_{i=1}^{k} \overline{Q}(n)_i$

is the average expected customers in the queue.

5.3 Server usage proportion

It is the proportion of time in which the server is busy.

Let $B(t)$ be a function such that

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

$$\overline{U}(n) = \frac{\displaystyle\int_0^{T(n)} B(t)dt}{T(n)}$$

The area under the curve is incremented every time t in the area under the rectangle under $B(t)$ in the last interval from the last event, this is, the width *time since the last event* multiplied by the height *state of the server*.

$\overline{\overline{U}(n)} = \frac{1}{k} \sum_{i=1}^{k} \overline{U}(n)_i$

is the average server usage proportion.

6 Simulation description

The following constants are defined:

- Number of event types.
- n
- λ_c
- λ_s

The following variables are defined

- Clock (T)
- Current server state (0 - idle | 1 - busy)
- Customers in queue
- Time since the last event
- Arrival time of customers in queue
- Event list: [Time of next arrival | Time of next departure]
- n_i: Number of customers served
- Total wait in queue
- Area under the curve of $Q(t)$
- Area under the curve of $B(t)$

Step 1 - Initialization

Clock, *Customers in queue*, *Time since the last event*, n_i are set to 0, *Current server state* in Idle, and *Event list*: [Clock + A_i | $1e30$]. The first next event will always be an arrival.

Step 2 - Next event

The times of *Event list* are compared, and the next event type with a smaller occurring time is set. *Clock* advances to the time of the next event.

Step 3 - Performance measures update

Time since the last event is set as *Clock* - *Time of last event*. *Area under* $Q(t)$ is updated as *Time since the last event* by *Customers in queue*. *Area under* $B(t)$ is updated as *Time since the last event* by *Current server state*. *Time of last event* is updated to *Clock*.

Step 4 - Event occurrence

Arrival event

When a customer arrives, it is checked that n_i+ *Customers in queue* $\neq n$. If true, then no more arrivals are allowed, and *Time of next arrival* is set to $= 1e30$. If false, the next A_i is calculated. If *Current server state* is *Busy*, *Customers in queue* is incremented in 1, and *Time of customer in queue* is updated. If *Current server state* is *Idle*, *Wait* is set to $= 0$ for the current customer, *Total wait in queue* is updated, n_i is incremented in 1, *Current server state* is set to *Busy*, c_i is calculated for the customer.

Departure event

The queue state is checked. IF *Customers in queue* $= 0$, *Current server state* is set to *Idle*, and *Time of next departure* to $= 1e30$. If *Customers in queue* $\neq 0$, *Customers in queue* decrements in 1, *Wait* of entering-service customer is calculated, n_i increments in 1, c_i is scheduled, and the customers move in the queue (if there is one) one place up, updating *Arrival time of customers in queue*.

Step 2, **Step 3**, y **Step 4** are repeated until $n_i = n$, *Customers in queue* $= 0$, and *Time of next departure* $= 1e30$,

7 Methodology

3 experiments were carried out, each one consisting of 500 replicates, $\lambda_c = 1$ y $\lambda_s = 2$. In the first experiment, $n = 15$, in the second, $n = 100$, and in the third $n = 1000$. In every replicate, the simulation described in Section 3.2 was run.

8 Results

In table 1 the values of the performance measures and the variable under study are shown, along with their respective standard deviations for every n corresponding to each experiment.

The values taken by the simulated T are similar to the values calculated analytically. With $\lambda_c = 1$ and $n = 1000$, the analytical result is 1000. After running an experiment with those parameters, it was observed that approximately 90% of the data is between 950 y 1050, with an Std. of 30, approximately. Similarly, with $n = 100$, 95% of the data is 80 and 120, with a Std. of 10, approximately.

In relation to the performance measures, it is noted that the simulated values are similar to the analytical ones, being these $\frac{\lambda_c}{\lambda_s} = 0.5$ para $\overline{D}(n)$ and $\overline{Q}(n)$, and $\frac{1}{\lambda_s} = 0.5$ for $\overline{U}(n)$

Figure 1 illustrates the histograms of the performance measures and the variable under study, varying n in 15, 100, y 1000. The histogram of the Total Time was normalized to observe the change in data dispersion.

Figure 2 shows the box and swarm plots of the performance measures and the variable under study for every run.

λ_c	λ_s	n	$\overline{\overline{D}(n)}$	Std($\overline{D}(n)$)	$\overline{\overline{Q}(n)}$	Std($\overline{Q}(n)$)	$\overline{\overline{U}(n)}$	Std($\overline{U}(n)$)	$\overline{T}(n)$	Std($\overline{T}(n)$)
1	2	15	0.368	0.35	0.383	0.39	0.49	0.15	16	3.7
		100	0.463	0.25	0.471	0.29	0.494	0.07	101	9.6
		1000	0.496	0.084	0.496	0.01	0.5	0.022	1000	31.2

Table 1: Performance measures

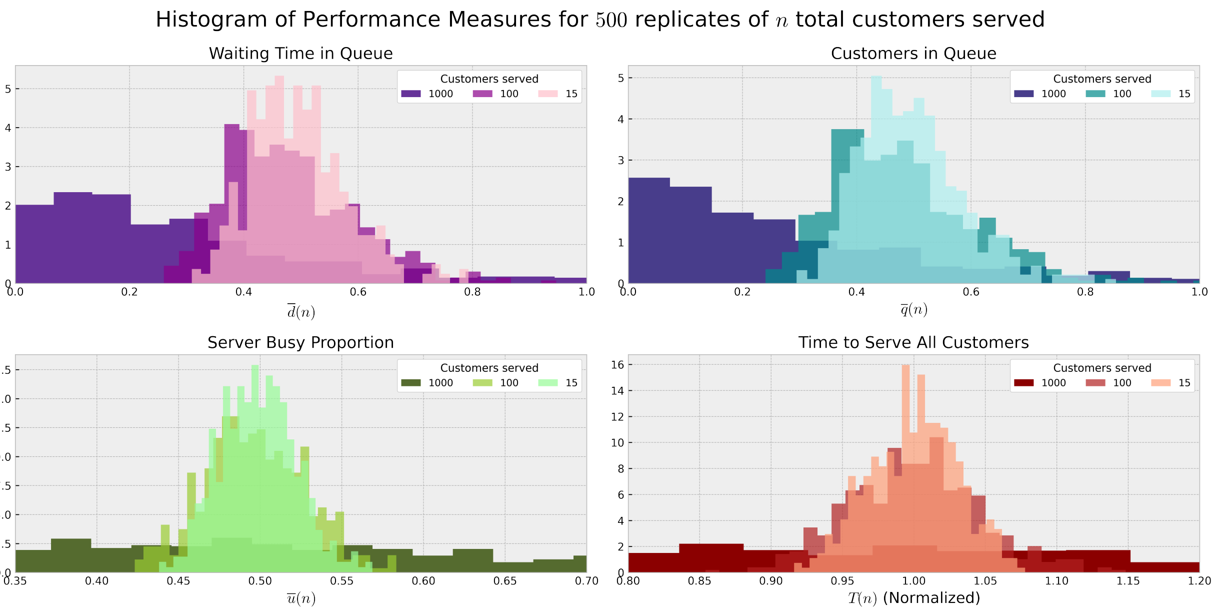

Figure 1: Histograms of performance measures with $n = 15$, $n = 100$ and $n = 1000$

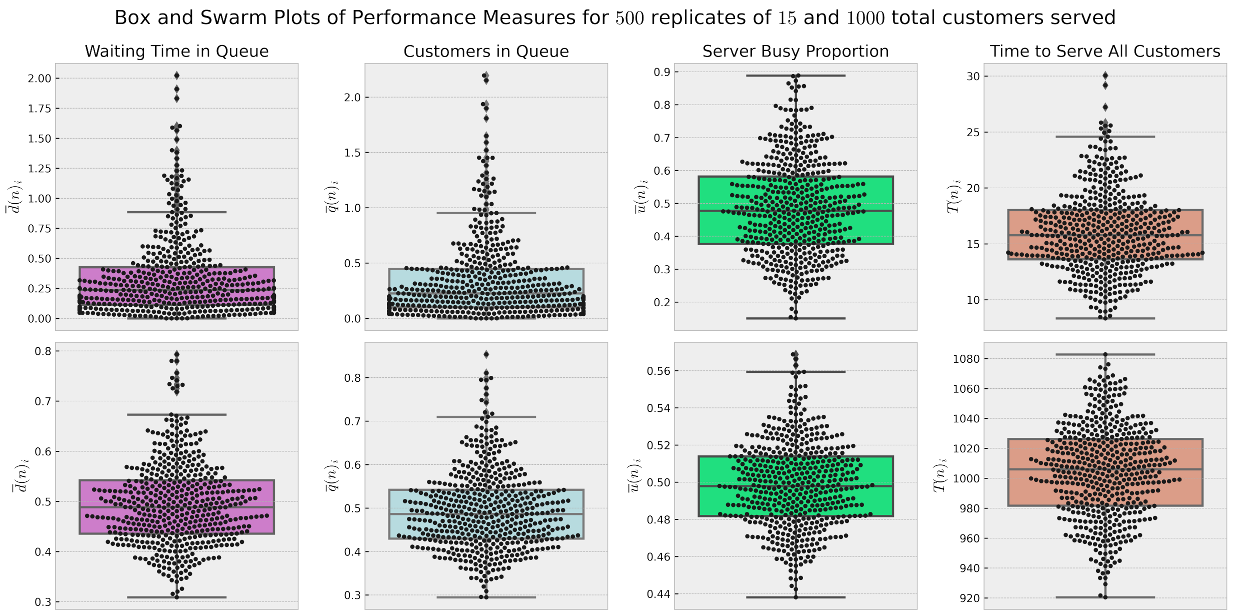

Figure 2: Box and swarmplots of performance measures with $n = 15$ (upper row) and $n = 1000$ (lower row)

9 Conclusions

In the first place, it is concluded that both the performance measures and the variable under study T take values similar to those proposed by analytical methods. It was also observed that the performance measures stabilize after a certain number of runs. Analyzing the moving average and the performance measures, it is concluded that, after a certain time, the measures in the run n present insignificant or null variations for the run $n - 1$ (the standard deviations decrease to the order of 10^{-2}, and that the values of these measures stabilize at a value (0.5 in this case, for all performance measures). This steady-state occurs at approximately 500 runs.

Second, it was observed that the distribution of the averages of the performance measures after a few runs (between 5 and 30) is exponential. However, as the number of runs increases, the data set begins to take the form of a normal distribution. This fact is justified by the Central Limit Theorem. By calculating 500 times the average of a random variable (regardless of its distribution as long as it is randomly independent and identically distributed), the data is normally distributed.

10 Future work

In section 3.1, two approaches were presented regarding the study variable, of which the first was addressed. The analysis of the simulation taking as the variable under study the number of clients in the queue that use the server in a given time is left pending as future research.

References

[1] Adesoji Farayibi. Investigating the application of queue theory in the nigerian banking system. *Available at SSRN 2836966*, 2016.

[2] Steven Allder, Kate Silvester, and Paul Walley. Understanding the current state of patient flow in a hospital. *Clinical medicine*, 10(5):441, 2010.

[3] Averill M. Law and W. David Kelton. Simulation modeling and analysis. 1982.