LAB Q

Scorca Francesco s288876

February 20, 2022

1 Introduction

The following work involves the analyses of different types of queues to collect some performance metrics, compared to the ones from the analytical models when possible. In all the simulations the initial random seed is fixed to 42, the metrics are averaged on 10 runs, providing confidence intervals with confidence level 0.95.

2 Task 1: MM1

This queue will be subject to two subsequent simulations, being necessary to fix a simulation time suited for the work-loads under analyses: once found it will be used for other paragraphs simulations as well.

2.1 Assumptions

- The inter-arrivals times are independent and exponentially distributed with average frequency λ.
- The service times are independent and exponentially distributed with average frequency $\mu = 0.1$.
- The queue capacity B is infinite.
- There is exactly 1 server.

2.2 Input parameters

- The work **load** $\rho = \frac{\lambda}{\mu}$. We test values from 0.09 to 0.96 in the first simulation with step 0.03, then from 0.2 to 0.8 with step 0.2 in the second. The condition $\rho < 1$ is ensured in order to have an ergodic system.
- The **simulation time**, the condition to stop the simulation, testing values from 10^5 to 2×10^6.
- The **threshold** to define the probability that the customers experience a delay lower than this. The values tested are 10, 50, 100: from $\frac{1}{\mu}$ to $\frac{10}{\mu}$.

2.3 Output metrics

- The **average delay**, as the ratio between sum of times customers spend in the queue and number of customers that leave it. It is possible to derive theoretically such value through Little's Law:

$$E[T] = \frac{E[N]}{E[\Lambda]} \tag{1}$$

  where $E[T]$ is the average time in the queue (average delay), $E[N] = \frac{\rho}{1-\rho}$ is the average number of customers, $E[\Lambda] = \lambda$ is the average arrival rate. Applying such a formula and simplifying it we have the analytical expression

$$E[T] = \frac{1}{\mu - \lambda} \tag{2}$$

- The **below** $threshold$ **probability**, the probability that the customers experience a delay lower than the input $threshold$.
- The **users distribution N**, where $\pi_i = P(N = i)$ is computed as the sum of intervals of time the queue has i clients and normalized by the simulation time. We consider values of i from 0 to 10, considering values greater than this as 10. Theoretically N is a random variable with geometric distribution:

$$\pi_i = (1 - \rho)\rho^i \tag{3}$$

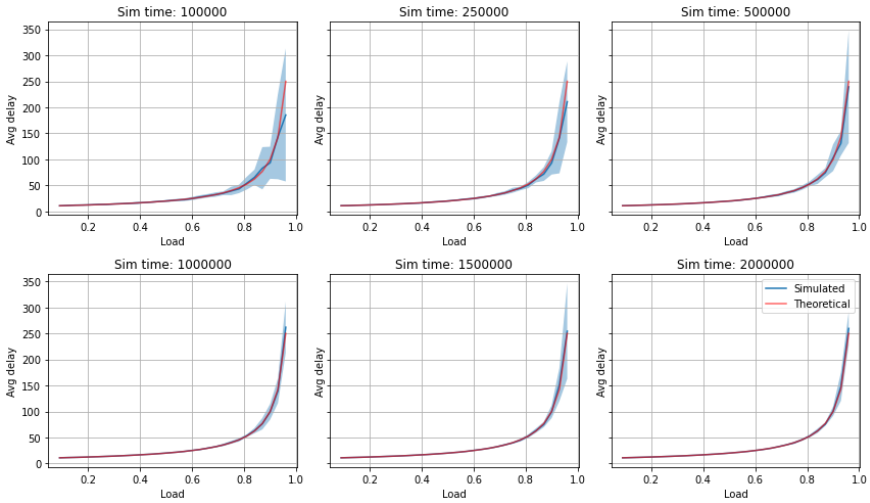

Figure 1: Average delay in function of ρ for the different simulation times.

2.4 Algorithm

The queue is modelled with a discrete event simulator with only state variable the **number of users**. We use the event scheduling approach with two types of events:

- **Arrival.** The number of users is increased by 1, the next arrival is scheduled and if the server is idle a customer is served, scheduling its departure time.
- **Departure.** The number of users is decreased by 1 and if there are customers in the waiting line a new one is served, scheduling its departure time.

Such events are scheduled in a **Future-Event Set (FES)** and are executed in their time order in the **Event loop** until the simulation time is reached.

2.5 Results

Figure 1 shows the behavior of the average delay in function of ρ for the different ending conditions tested[1], comparing it with the theoretical curve derived from (2). It is possible to notice how the superposition starts to be accurate from simulation times greater than 5×10^5. In fact, in the first two curves for the highest load values we observe a consistent under-estimation of the delay. Furthermore, it is possible to notice that the system has not converged yet by looking at the confidence intervals, which starts to have a negligible depth on all the load tested only from simulations times greater than 10^6: we use such value for all the following simulations.

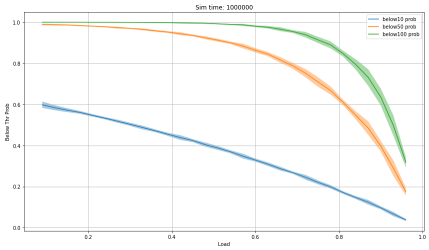

Figure 2: Probability of delay being under a threshold in function of ρ.

Figure 2 focuses on the probability that the delay is below 1, 5 and 10 times the average service time $\frac{1}{\mu} = 10$. The probabilities reduces with the increase of the load as it is intuitive, since it implies higher delays. The confidence intervals behavior is analogous to the previous. Furthermore, we can argue that the delays distribution is asymmetrical, in particular by looking that at $\rho = 0.8$ we have average delay equals to 50, but the probability that

[1] Although tests are conducted for a finite set of values we represent them with continuous curves because of the density of these.

delay is under this value is 0.6, hence the distribution is skewed towards lower values. Once again, this is intuitive since the time spent in the queue will depend on the arrivals rate and departure rates, exponentially distributed, hence presenting this type of skewness.

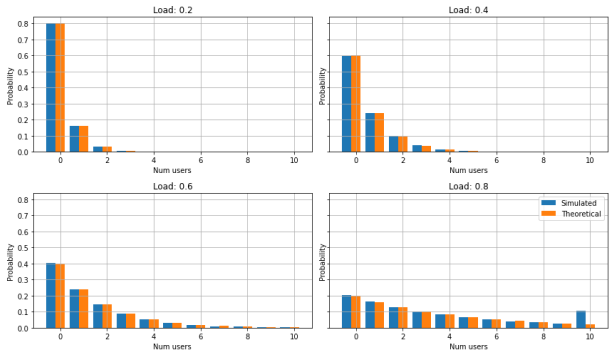

Figure 3: Users distributions for different ρ. The last bin is to interpret as $N \geq 10$.

Finally, figure 3 compares the users distributions obtained through simulations with the theoretical ones from (3). The last bin is to interpret as such a number being greater or equal than 10. It is possible to observe that for $\rho = 0.8$ the probability of $N \geq 10$ is not negligible anymore, arriving to 10%, hence if dimensioning a real system with such a load having a queue capacity less than 10 could bring problems in terms of customers' losses, as we will deepen in the following paragraph. Furthermore, it is possible to focus on $N = 0$, representing the idle status of the server: this decreases linearly with the increase of the load, mirroring the theoretical behavior: $\pi_0 = 1 - \rho$.

3 Task 1: MM1B

3.1 Assumptions

The assumptions are the same of the MM1 queue, with the difference that the queue capacity B is finite.

3.2 Input parameters

- **B**, from 1 to 10.
- ρ, from 0.2 to 0.8, with step 0.2. Although in this case the system is ergodic also for $\rho > 1$, we reuse the same values to have a comparison with the MM1 queue.

3.3 Output metrics

- the **average delay**. The theoretical behavior is obtained once again applying Little's Law with

$$E[N] = \sum_{i=1}^{B} i\rho^i \frac{1-\rho}{1-\rho^{B+1}}$$
$$E[\Lambda] = \lambda - \lambda\pi_B$$

- the **lost customer probability** computed as the ratio between the lost customers and the arrived customers. Theoretically, this is equal to the probability of the queue being in state B: $\pi_B = \rho^B \frac{1-\rho}{1-\rho^{B+1}}$.

3.4 Algorithm

The simulator has the same structure of the one for the MM1 queue. The only difference is the **Arrival** event performs the previously described routine only if the number of users is inferior to B, otherwise the queue is full and we register the loss of a customer.

3.5 Results

Figure 4 shows the average delays for different loads and queue capacities, comparing the results from the simulator with the ones from the analytical model. We observe a perfect superposition of the curves with quite deep confidence intervals only when the load and the capacity assume their highest values. All the curves show an increasing behavior starting from 10: this is the value of the average service time and must coincide with the average delay when $B = 1$, since the customer starts to be served exactly when entering the curve. Indeed, the increasing trend is due to the fact the customers need to wait for more and more clients to be served when the capacity increases. However, we observe a saturation point in all the curves, increasing with the load. This is explainable by looking again at the users distribution for the MM1 (figure 3): the saturation point corresponds to the last number of clients after which we have a negligible probability. Indeed, we do not observe such saturation point in the last curve, since the probability of having more than 10 clients

is not negligible. The same results are mirrored in figure 5, whose focus is the probability of loosing a customer. In all cases it tends to 0 in correspondence of the points where the average delay saturates, never approaching it in the last curve as expected.

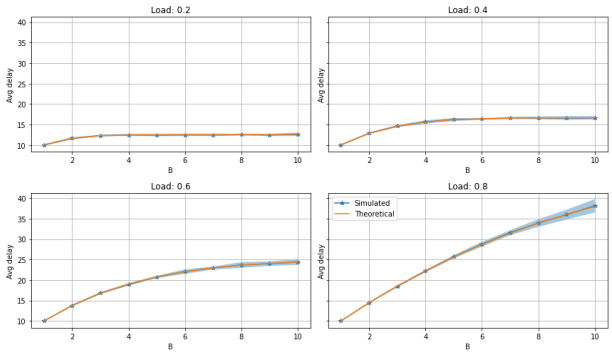

Figure 4: Average delay in function of the queue capacity for the different ρ.

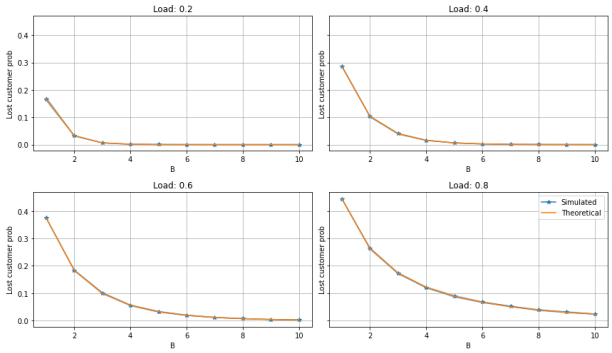

Figure 5: Probability of loosing a customer in function of the queue capacity for the different ρ.

4 Task 2

In this case we have a queue analogous to the MM1 in paragraph, with possible failures, requiring for a reparation. Thus, another status variable is needed, representing if the server is up or down.

4.1 Assumptions

The assumptions are the same of paragraph 2.1, with the addition of the fact that **inter-failure** time and the **repair time** are independent and exponentially distributed with average frequencies respectively γ and σ.

4.2 Input parameters

- The failure **risk**, namely the ratio between the average inter-failure frequency and the average inter-arrival time: $\frac{\gamma}{\lambda}$. Higher values of this parameter will correspond to more frequent failures. We test values from 0.5 to 5, with step 0.5.
- The failure **severity**, namely the ratio between the average inter-failure frequency and the average repair frequency: $\frac{\gamma}{\sigma}$. Higher values of this parameter will correspond to more durable failures. The values tested are 0.5, 1 and 2.

4.3 Output metrics

- The **average delay**.
- The **lost customer probability**.
- The **server down probability**, computed as the ratio of the time the server is down and the total simulation time.
- The **average number of users** in the queue, computed as the sum of users' number weighted for their time and normalized by the simulation time (10^6). We compute it for all the time and for only the time the server is down.

4.4 Algorithm

The simulator previews some modifications on the one described in paragraph 2.4. The **Arrival** event performs the previously described routine only if the server is up, otherwise it updates the number of lost customers. A **Departure** is allowed only if the server is up, otherwise is postponed of the previewed "repair time": the client is frozen in the server. Furthermore, two new events are defined to manage the server status state variable: **Failure**, putting down the sever and scheduling a repair event and **Repair**, putting up the server and scheduling the next failure.

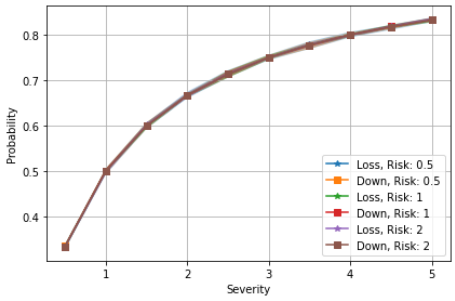

Figure 6: Probability of loosing a customer and of the server being down.

4.5 Results

Figure 6 show the probabilities of loosing a customer and of the server being down: they are superposed since we can experience a loss only when the server is down. We observe that the curves are independent from the risk: the only determinant is the relation between failures and repairs frequencies. Regarding this one, it asymptotically approaches the 100% with the knee in the curve being between 1 and 2: even if the repair time is considerably inferior to the the average time between failures its impact on the system is always severe.

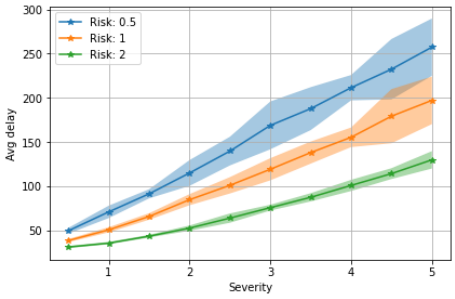

Figure 7: Average delay with respect to failure risk and severity.

Figure 7 shows the average delay behavior. Fixing the failure risk, we observe a linear increase in the delay if the failure requires more time for its reparation, which is intuitive since customers in service are frozen in the queue.

Figure 8 shows the average number of users in the whole simulation and only when the server is down. The red line represents the theoretical value in a MM1 with no failure possibility. The graphs are indistinguishable because of the specific type of failure modelled, resulting in the system being frozen. It is possible to hypothesize that this would happen for every metric, since all the customers-related events happen when the server is up, leading to the fact that the overall behaviors are the same of the ones given different status of the server. Analysing the metric trend, we can observe that it generally increases as the severity does, explainable by looking at the increase in the average delay (Fig 7): as users spend more time in the queue the server is supposed to spend less time in idle status (N = 0), thus increasing the average number of users. However, the curves could not show this behavior for severities between 0.5 and 1, for which we know that the system has a dramatic increase in loss probability (Figure 6): in this range this could increase faster than the average delay, implying a server idle more often. Furthermore, we observe a decrease of the average number of clients as the risk increases, from the 4 of MM1 case (risk = 0). Although the loss probability does not depend on risk, this impacts the initial conditions. Indeed, the system reaches a given number of customers in the queue before the first failure which will be lower if failures are more frequent. This allows also to explain why the average delay decreases when risk increases (Figure 7): having shorter queues, customers will be served before.

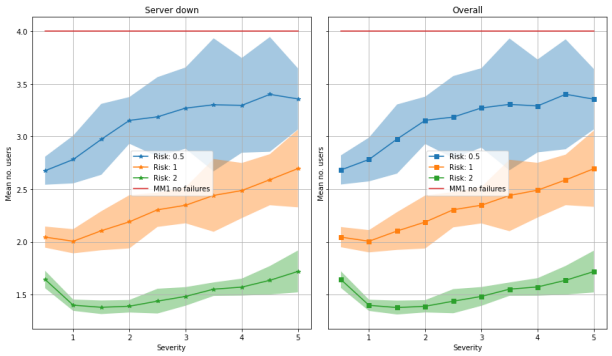

Figure 8: Average number of users: on the left when the server is down, on the right considering all the simulation. The red line represents the theoretical value in a MM1 with no failure possibility.