Politecnico di Torino

POLITECNICO DI TORINO
ICT for Smart Societies

MANAGEMENT AND CONTENT DELIVERY FOR SMART NETWORKS
2022/2023

Laboratoy REPORT

GROUP 8:

Maryam Bigonah s308977
Ali Mohammad Alizadeh s308885
Mohammad Eftekhari s307774

Lab 1: Network system simulation

The purpose of this lab is to model a router as a queuing system and analyze its behavior under different modifications to the arrival rates, service rates, buffer sizes, and types of queuing systems. The specific queuing systems that will be studied are MM1, MG1, and Multi-server system, with different variables for each server. The impact of different packet dispatching algorithms for the Multi-server system will also be investigated. Finally, some metrics will be derived to evaluate the system performance under various modifications.
The working process of a router can be modeled as a queuing system with packets as customers. Packets arrive at the router with different arrival rates, and the router services them by transmitting them over the network. The buffer can be considered as a waiting line for packets.
The simulation approach used in this lab is event scheduling, which is based on the future event scheduler (FES). The FES consists of a running loop that handles arrivals and departures, and calls the appropriate functions based on the current event.

The monitored metrics to understand the effects of different changes are:

- Number of transmitted packets: to evaluate the load in the system.
- Number of dropped packets: to evaluate the impact of service and arrival rates, and buffer size over time.
- Size of buffer: how it affects the system with constant or variable service or arrival rates.
- Average number of packets: to have a general understanding of the system.
- Average waiting and queuing delay: to see the effects of arrival rate and service rate on the time that each packet waits before being served.
- Loss probability: to see the fraction of losses during time under different circumstances.

Task 1:

The task is to investigate the system performance under different arrival rates and a constant service rate, and the effect on the chosen metrics. The simulation is done using an MM1 queue with infinite and finite buffer sizes. With a finite buffer size, losses can be monitored by observing whether new packets are dropped when the buffer is full. When the arrival rate of packets is greater than the service rate (lambda > mu), the buffer will queue up and the server will not be able to start processing packets immediately upon arrival. This can lead to both delays and losses, as the packets will have to wait in the buffer before being processed. In an infinite buffer, there are no losses, as the buffer can always accommodate new packets. However, the average delay experienced by packets in the system may be higher in an infinite buffer, as the packets may have to wait longer to be processed.

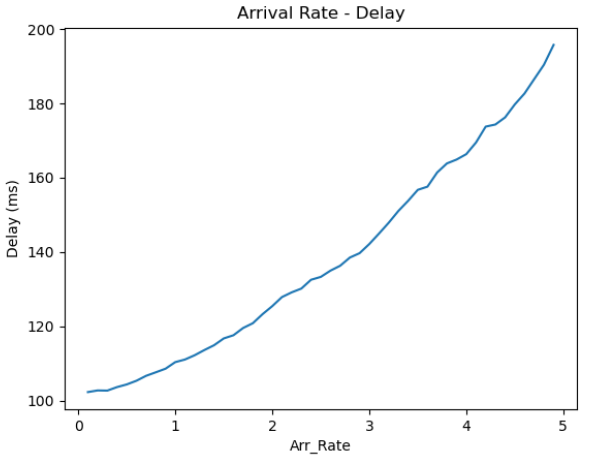

Figure 1.1 - Arrival rate vs Delay

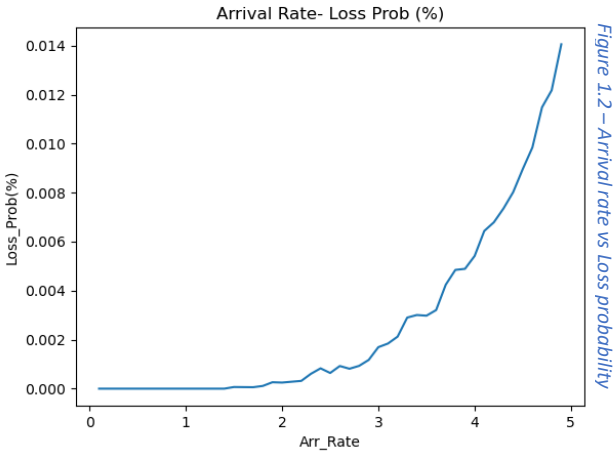

Figure 1.2 – Arrival rate vs Loss probability

As shown in Figure 1.1, the delay experienced by packets increases with increasing arrival rate. This is because the packets have to wait longer in the buffer before they can be served. Figure 1.2 describes the impact of increasing arrival rate on the loss probability. The loss probability is the probability that a packet will be dropped because the buffer is full. The loss probability increases with increasing arrival rate because the buffer is more likely to be full when there are more packets arriving.

The difference between delay in a finite buffer and an infinite buffer is that in a finite buffer, the delay is bounded by the time that the buffer takes to fill up. Once the buffer is full, any new packets that arrive will be dropped. In an infinite buffer, the delay can grow indefinitely because there is no limit on the number of packets that can be stored in the buffer.

Task 2:

The task is to consider a MM2 queue with a finite buffer and constant service rate and make a comparision with MM1 queue and then investigate the effect of different buffer sizes on the MM2 system.

a. The speed of losing packets are slightly slower than the MM1 case since there are two servers with the constant service rate serving the packets at a higher speed. As the arrival rate increases the fraction of lost packets become more and the faster the arrival rate the faster the growth of lost packets.

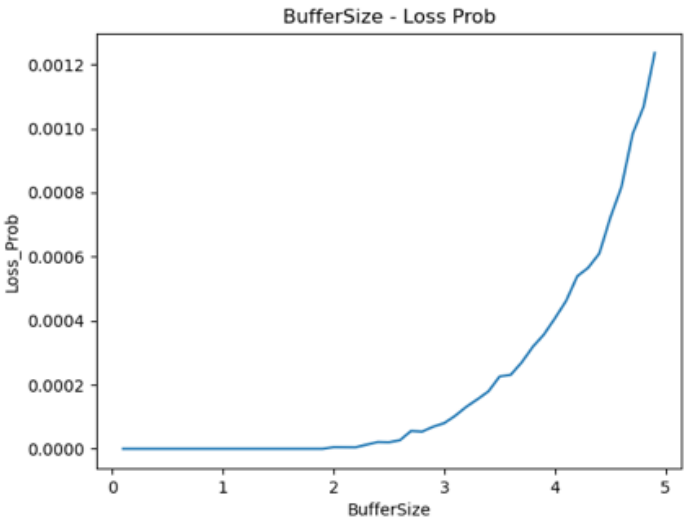

Figure 2.1 – Arrival rate vs Loss probability

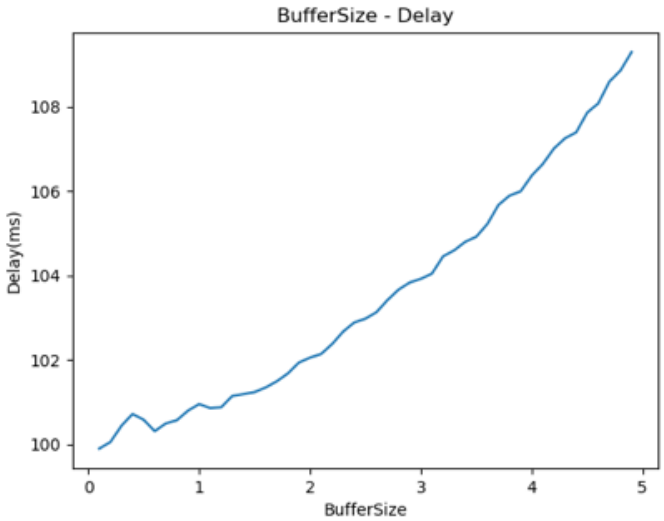

Figure 2.4 – Arrival rate vs Delay MM2 Finite buffer

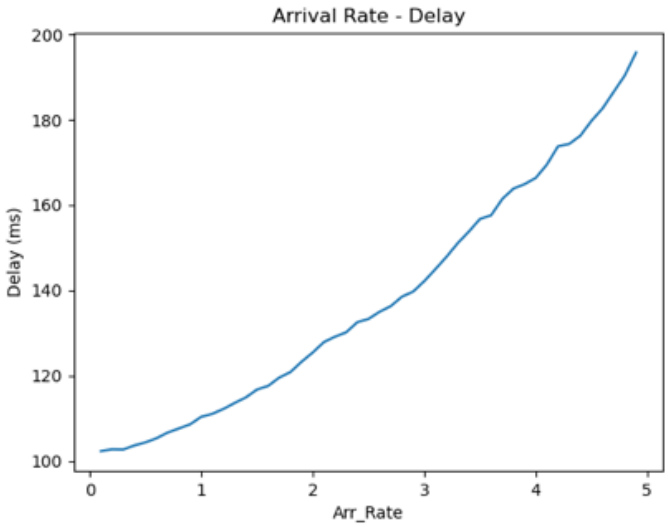

Figure 2.3 – Arrival rate vs Delay MM1

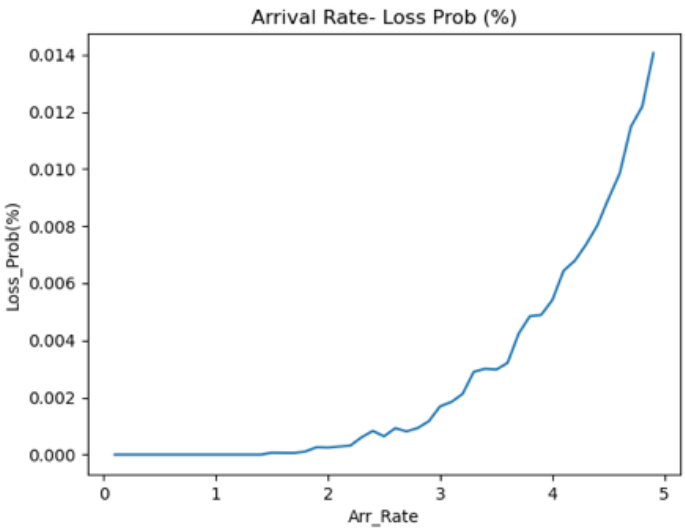

Figure 2.2 – Arrival rate vs Loss probability MM2

b. In case of difference buffer sizes as the buffer size increased the probability to lose packets decrease since there is more room to store packets in the queue.

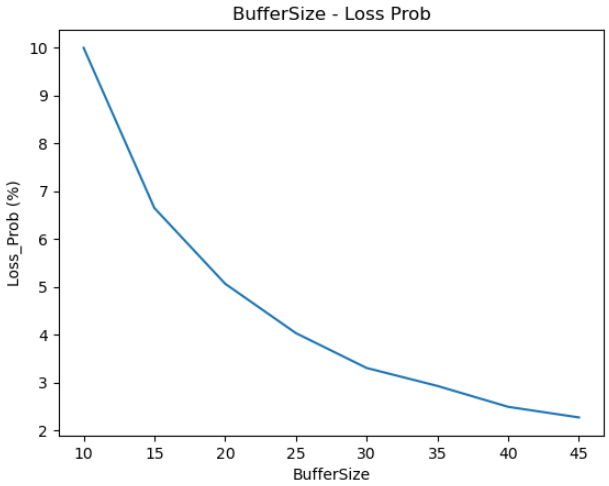

Figure 2.5 – Buffer Size vs Loss Probability MM2

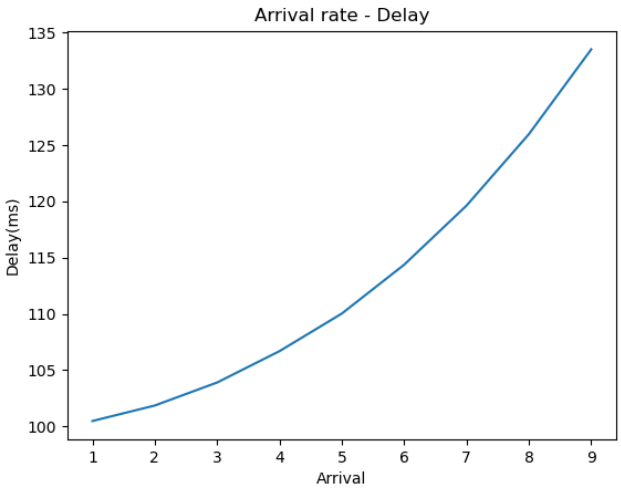

Figure 2.6 – Arrival rate vs Delay MM2 Infinite buffer

As shown in Figure 2.5, the loss probability decreases as the buffer size increases from 10 to 50. This is because the larger buffer can accommodate more packets before it becomes full, which reduces the likelihood that a packet will be dropped.

In the case of an infinite buffer, the number of packets waiting in the queue will continue to increase indefinitely. This will lead to higher queuing delay, as the packets will have to wait longer to be served. The higher the arrival rate (constant service rate), the faster the queue will fill up. Since there is no limit on the number of packets that can be stored in the queue, there can be infinite delay.

As explained during the course, the size of the buffer can be helpful with a few number of arrivals. However, in the case of high arrival rates, having a large buffer will not help mitigate congestion problems. This is because the buffer will simply fill up quickly, and the packets will still experience high delay. Using large buffers in routers seems like a good idea, but it is not as practical as it sounds.

Task 3:

In the case of a multi-server scenario, there are three algorithms for assigning packets to servers:

1- Random: This algorithm assigns packets to servers randomly.
2- Round Robin: This algorithm assigns packets to servers evenly, in a round-robin fashion.
3- Fastest Server: This algorithm assigns packets to the server with the fastest service rate, or the one with the lowest load.

The performance of these algorithms was evaluated under different criteria, and they were compared with different buffer sizes.

Random Assignment Algorithm: Each packet is assigned to an available server randomly. As the arrival rate increases, the uneven dispatching of packets results in different queue sizes for each server and different loads on each server. This can lead to higher packet loss, as some servers may become overloaded.

Round Robin Assignment Algorithm: Each packet is assigned to an available server in turn. This results in a more balanced distribution of packets among the servers, as the load on each server is almost the same. However, the round robin algorithm may not be optimal in all cases, as it does not take into account the service rates of the different servers.

Fastest Server Assignment Algorithm: Each new packet is assigned to the server with the highest service rate. This ensures that the packets are processed as quickly as possible, and it can help to reduce packet loss. The fastest server assignment algorithm is generally considered to be the most efficient of the three algorithms.

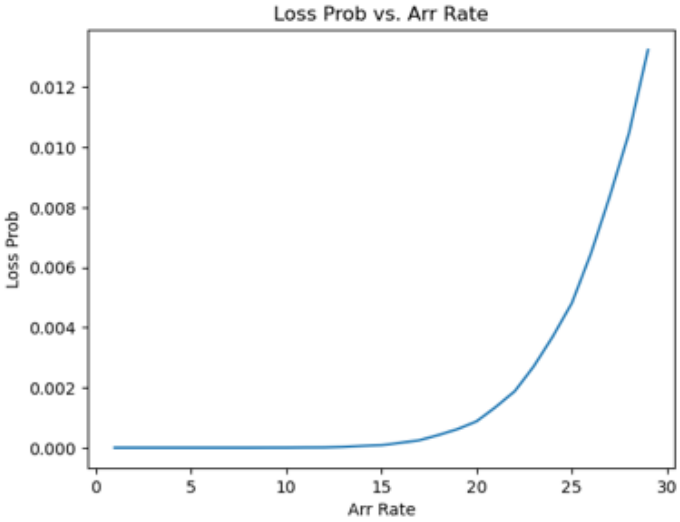

Figure 3.1 – Arrival rate vs Loss probability Random

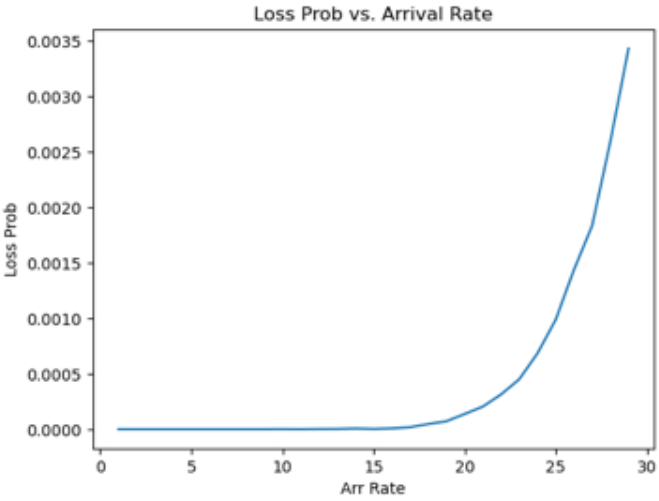

Figure 3.3 – Arrival rate vs Loss probability RR

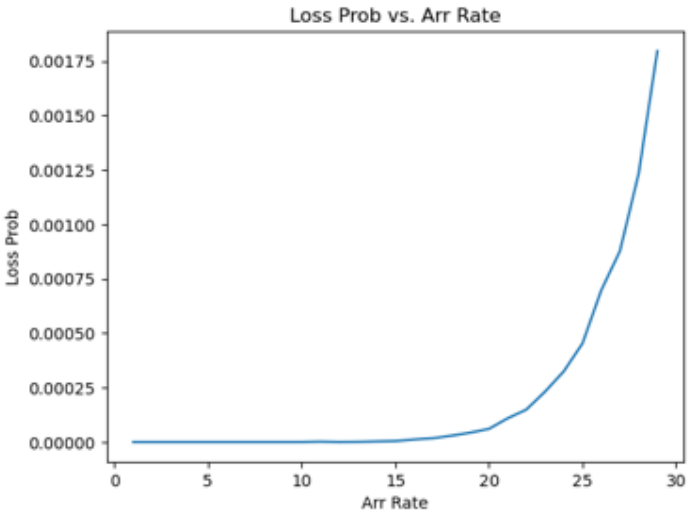

Figure 3.5 – Arrival rate vs Loss probability Fastest

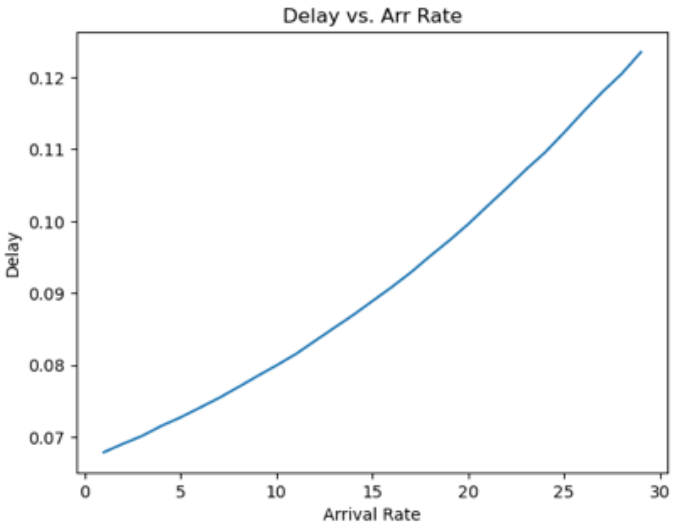

Figure 3.1 – Arrival rate vs Delay Random

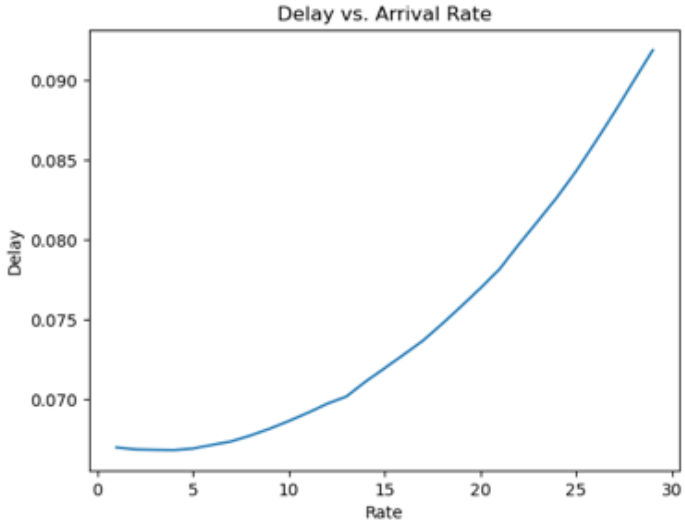

Figure 3.4 – Arrival rate vs Delay RR

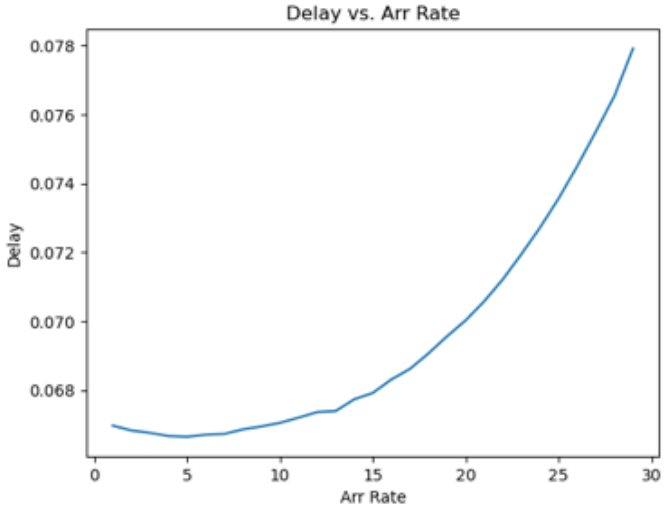

Figure 3.6 – Arrival rate vs Delay Fastest

In any case, with a finite buffer, all servers will experience losses if the arrival rate exceeds the capacity of all servers together. However, the amount of loss will vary depending on the packet dispatching algorithm.

Random assignment: This algorithm will have the most losses, as the servers are chosen randomly. This means that some servers may get more packets than others, and the queues on these servers will fill up faster.

Round robin: This algorithm will have fewer losses than random assignment, but it will still have some losses. This is because the server with the lowest speed will have more load, and this can lead to delays and losses.

Fastest server: This algorithm will have the fewest losses, as it ensures that the packets are sent to the servers that can process them the fastest. This helps to keep the queues from filling up, and it can reduce delays and losses.

In conclusion, the best way to reduce losses in a multi-server system with a finite buffer is to use the fastest server dispatching algorithm. This algorithm will ensure that the packets are processed as quickly as possible, and it can help to reduce delays and losses.

Task 4:

To investigate the effect of changing the service time distribution in the MG1 model, a uniform distribution was chosen instead of the exponential distribution. As expected, the growth of the delay time and loss probability is not exponential. Instead, it is much faster. This is because the uniform distribution has a wider range of values than the exponential distribution. This means that there is a greater chance of having longer service times, which can lead to longer delays and more losses.

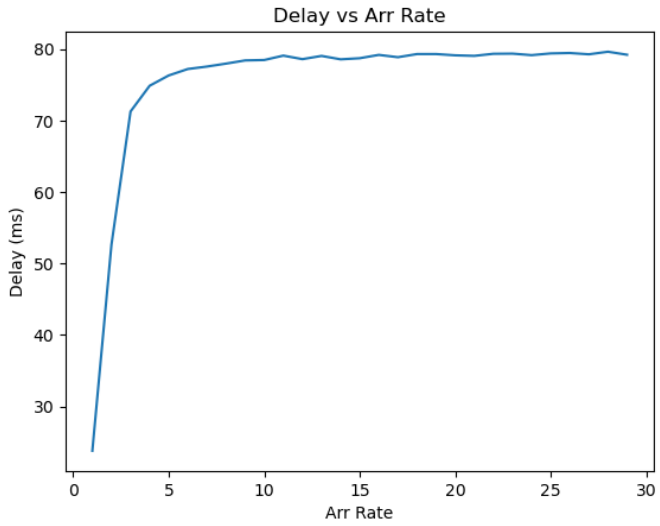

Figure 4.1 – Arrival rate vs Delay UNIFORM

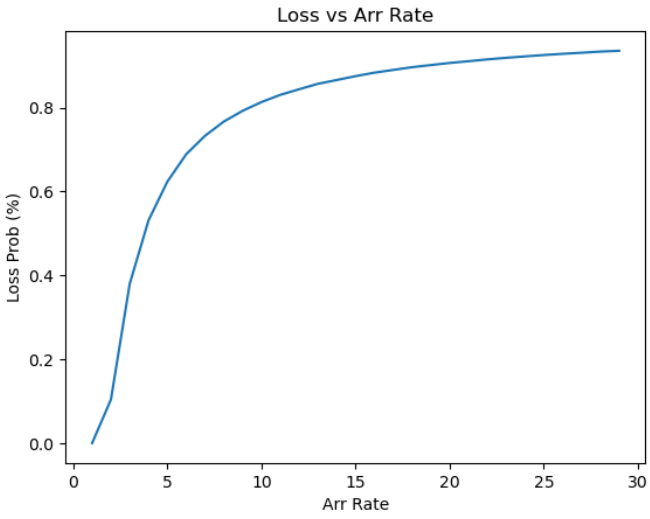

Figure 4.2 – Arrival rate vs Loss probability UNIFORM

Lab 2: Industrial Internet of Things (IIoT) system simulation

The purpose of this lab is to simulate a portion of an Industrial Internet of Things (IIoT) system modeled as a queuing system and analyze its performance under variable configurations. The lab explores how different scenario configurations and network parameter settings affect system behavior and performance. The system represents an IIoT system where sensors collect real-time data at the edge, undergo local processing, and may be forwarded to a Cloud Data Center. In the queuing system, customers correspond to IIoT data packets arriving at the MicroData Center, while service represents local and Cloud data processing. The waiting line represents the buffer where packets are stored before processing tasks.

Task 1:

According to the lab explanation local and cloud servers both have finite buffers and the fraction of packets of type A and B are both 0.5. packet loss at the cloud is observed, the lost packets are those of type A which are forwarded to the cloud due to full buffer or finished processing and type B which are the packets specified as type B at the beginning. This loss is the result of full buffer at the cloud. As is shown in Figure 2.1.1 at the beginning we have a sharp increase in terms of packet loss and after a while the loss probability growth has become slower and stabilized with a lower rate. It can be said that transient duration is between 2000 and 4000 which before this range we have sudden increase and after that the slower growth is experienced. the warm-up period refers to the initial phase of the simulation where the system is allowed to stabilize before collecting performance metrics. During this period, the system transitions from an initial state to a steady-state, ensuring that the collected data accurately represents the system's behavior. To remove this period there can be multiple approaches such as:

- Run the simulation for a longer duration: By extending the simulation time, you allow the system to stabilize and reach a steady-state, effectively eliminating the need for a separate warm-up period.
- Discard data from the initial phase: Collect data from the simulation but exclude the data collected during the initial phase, which is considered the warm-up period. This approach assumes that the system has reached a steady-state after a certain point in time.
- Using a large number of replications: This means running the simulation multiple times with different random seeds. This will help to reduce the impact of the initial conditions on the results and give you a more accurate estimate of the true steady state behavior of the system.

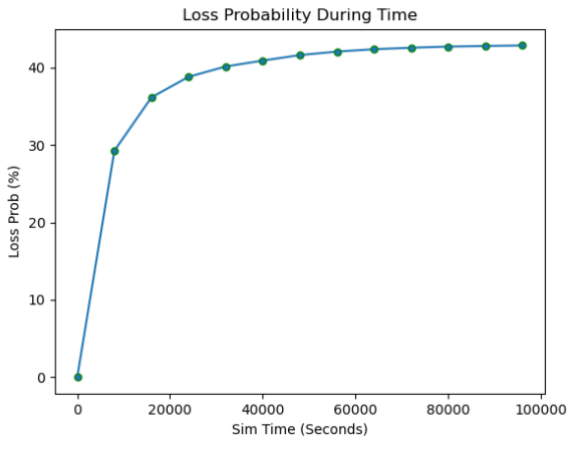

Figure 2.1.1 Loss Probability Cloud

Task 2:

In this task, we study the effect of edge buffer size on the overall system behavior. We expect that increasing the buffer size at the local server will lead to a lower growth in terms of packet loss. This is because the service rate at the edge node is lower than the cloud, so having more buffer will help to mitigate the loss problem.

a) As shown in Figures 2.2.X, the packet loss probability decreases with increasing buffer size. This is because a larger buffer can accommodate more packets before it becomes full, which reduces the likelihood that a packet will be dropped.

However, a larger buffer also means that packets have to wait longer in the queue before they can be processed. This can lead to an increase in the overall delay experienced by the packets. In the experiment, three different buffer sizes were chosen: 1, 500, and 2000. The results show that the packet loss probability decreases as the buffer size increases. However, the overall delay also increases, and the increase is greater for larger buffer sizes. This suggests that there is a trade-off between packet loss and delay. A larger buffer can help to reduce packet loss, but it can also lead to an increase in delay. The optimal buffer size will depend on the specific application and the desired balance between packet loss and delay.

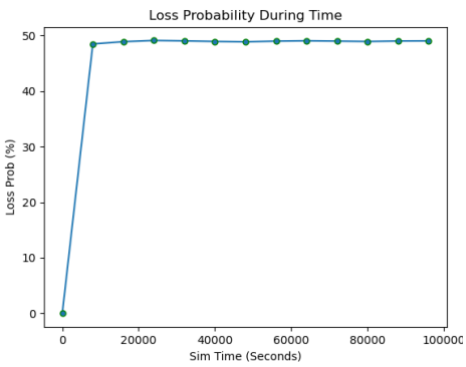

Figure 2.2.1 Loss Probability Edge Buffer 1

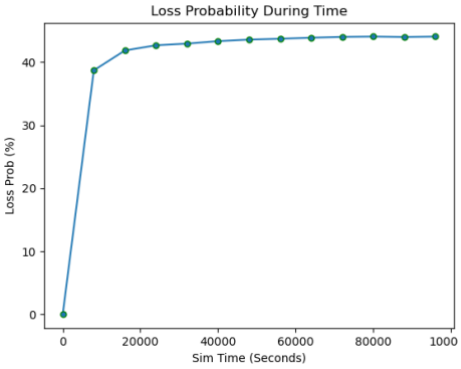

Figure 2.2.2 Loss Probability Edge Buffer 500

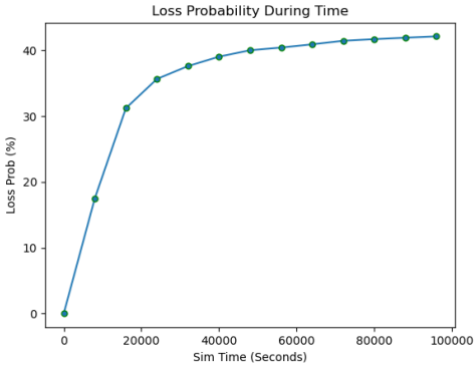

Figure 2.2.3 Loss Probability Edge Buffer 2000

b) The loss rate decreases at a slower rate as the buffer size increases in the cloud, as was the case for the micro data center. This is because a larger buffer can store more packets before they are dropped, which reduces the likelihood of packet loss. However, a larger buffer also means that packets have to wait longer in the queue before they can be processed, which can lead to an increase in delay.

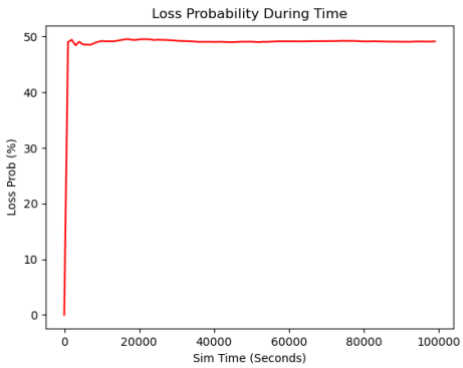

Figure 2.2.4 Loss Probability cloud Buffer 1

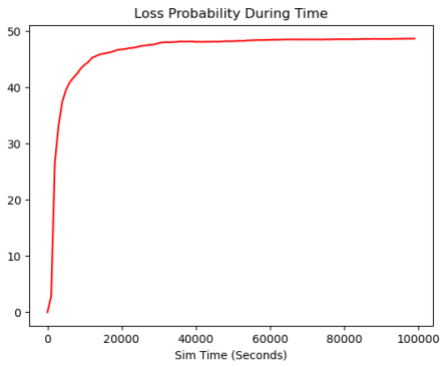

Figure 2.2.5 Loss Probability Cloud Buffer 500

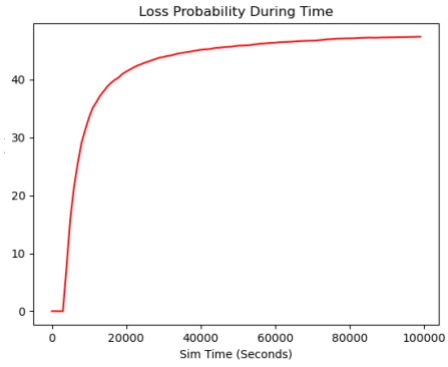

Figure 2.2.6 Loss Probability Cloud Buffer 1

c) For the task C there were 3 different fractions chosen to monitor the changes of the system with values [0.1, 0.5, 0.9]. The higher the fraction the more packets are sent to the micro data center and the lower the fraction the higher the number of packets sent to the cloud data center. As described in the following Figures 2.2.X with increasing the f ratio the number of the packets of type A increases and this results in higher loss probability

In general, the delay and loss probability of packets will increase as f increases. This is because type B packets require more complex processing, which takes longer and increases the likelihood of packets being lost.

Here are some specific examples of how the delay and loss probability of packets can vary with f:

- If f = 0.1, then 90% of the packets will be type A and 10% will be type B. The delay and loss probability of type A packets will be low, and the delay and loss probability of type B packets will be high.
- If f = 0.5, then 50% of the packets will be type A and 50% will be type B. The delay and loss probability of both type A and type B packets will be medium.
- If f = 0.9, then 90% of the packets will be type B and 10% will be type A. The delay and loss probability of type A packets will be high, and the delay and loss probability of type B packets will be very high.

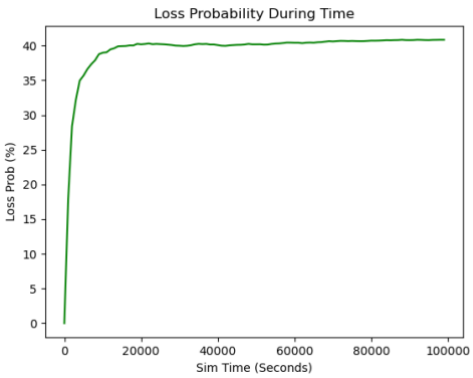

Figure 2.2.7 Loss Probability Fraction 0,1

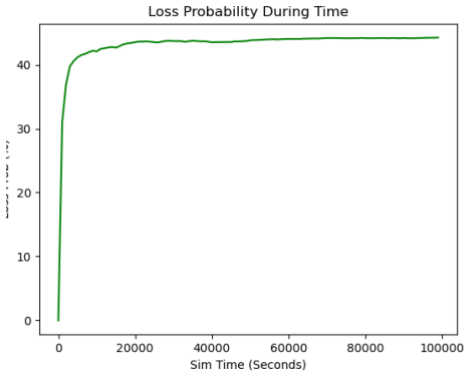

Figure 2.2.8 Loss Probability Fraction 0,5

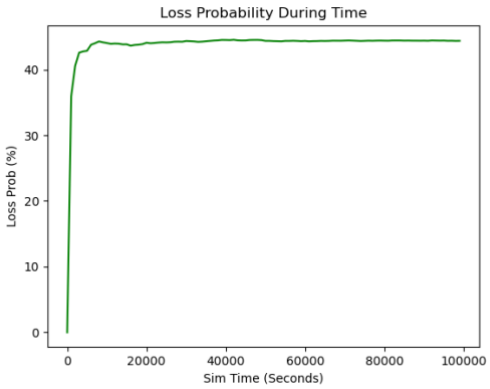

Figure 2.2.9 Loss Probability Fraction 0.9