UNIVERSITÀ DI PISA

DEPARTMENT OF COMPUTER ENGINEERING

Edge Computing
Project Documentation

January 8, 2025

Team:

Cavedoni F.
Monaci M.
Pinna F.

ACADEMIC YEAR 2024/2025

Contents

1 Introduction **1**
- 1.1 Problem Description . . . 1
- 1.2 Objectives . . . 2
- 1.3 Performance Metrics . . . 2

2 Modeling **3**
- 2.1 System Parameters . . . 4

3 Implementation **6**
- 3.1 Defined Modules . . . 6
- 3.2 Module Behavior . . . 6
  - 3.2.1 EdgeNetwork . . . 6
  - 3.2.2 BaseStation . . . 7
  - 3.2.3 User . . . 7

4 Verification **8**
- 4.1 Degeneracy Test . . . 8
  - 4.1.1 Scenario 1 - No Users . . . 8
  - 4.1.2 Scenario 2 - Very High Number of Users . . . 8
  - 4.1.3 Scenario 3 - Very Low Service Rate . . . 9
- 4.2 Consistency Test . . . 11
- 4.3 Continuity Test . . . 12
  - 4.3.1 Increment of Users ($N = 100, 250, 500$) . . . 12
  - 4.3.2 Increment of Requests' Load ($\lambda = 1/0.1, 1/0.5, 1$) . . . 12
  - 4.3.3 Increment of Size Rate ($\mu = 100, 1000, 10000$) . . . 13

5 Calibration **15**
- 5.1 Factors Calibration . . . 15
  - 5.1.1 Fixed Factors . . . 15
  - 5.1.2 Varying Factors . . . 16
- 5.2 Warmup Time Analysis . . . 17

6 Simulation Experiments **18**
6.1 How increasing the umber of users affects the system 18
6.2 The effect of forwarding packets on the system 21
6.2.1 Size rate variation - Method A 21
6.2.2 Size rate variation - Method B 28

7 Conclusions **35**

Chapter 1

Introduction

In recent years, the rapid growth of mobile devices and wireless networks has led to a dramatic increase in the demand for efficient and low-latency computational services. Traditional cloud computing architectures, while powerful, often suffer from latency issues due to the physical distance between end-users and centralized data centers. To address these challenges, *edge computing* has emerged as a promising paradigm by decentralizing computational resources closer to the users, significantly reducing latency and improving service quality. This report focuses on evaluating the performance of a cellular network enhanced with edge computing capabilities. The system under study consists of M base stations arranged in a 2D floorplan of size $L \times H$ according to a regular grid. Each base station is equipped with computational resources and processes user-generated tasks following a *First Come First Served* (FCFS) policy. Additionally, all base stations are interconnected via a mesh topology, allowing communication and task forwarding between them. Users are distributed across the floorplan and generate computational tasks at regular intervals. These tasks are sent to the geographically closest base station, which can either process the task locally or forward it to a less-loaded neighboring base station. This introduces a trade-off between processing delays and task-forwarding latency, which depends on the system's load and user distribution.

1.1 Problem Description

We consider N users placed at random locations `(x, y)` within the same 2D floorplan of size $L \times H$, where the coordinates `x` and `y` are random variables to be defined based on specific distributions. Each user generates a new computational task request every T seconds, and each task consists of I instructions to be executed. Both T and I are exponentially distributed random variables. A user sends each new task request to its closest serving base station, which can process the task using one of the following methods:

- **(A)** Serve the request locally at the receiving base station.

- **(B)** Forward the request to the less-loaded base station in the network. If this option is chosen, an additional fixed latency of D milliseconds is added to the total processing time.

1.2 Objectives

This project aims to achieve the following objectives:

- Evaluate the time required to complete a computing task for various values of N, T and I, comparing the two methods:
  - **Method (A):** Local task execution at the receiving base station.
  - **Method (B):** Task forwarding to the less-loaded base station.
- Assess the performance of the system under the following user placement scenarios:
  - **Uniform distribution:** `x` and `y` are uniformly distributed random variables in the range:

$$x \in [0, \text{width}], \quad y \in [0, \text{height}]$$

  - **Lognormal distribution:** `x` and `y` follow a lognormal distribution with defined parameters.

1.3 Performance Metrics

To evaluate the system's performance, the following metrics are considered:

- **Response time:** The time taken for a task to leave the system after being processed.
- **Queue length:** The number of tasks waiting to be processed at a base station.
- **Packet Dropped:** The number of tasks that are forwarded to an another basestation due to the option B.
- **Packet Dropped:** The number of tasks that are dropped due to insufficient resources.

The analysis presented in this report provides insights into the trade-offs and performance implications of local versus collaborative task processing in edge computing-enabled cellular networks. These findings contribute to the understanding and optimization of edge computing architectures for modern applications.

Chapter 2

Modeling

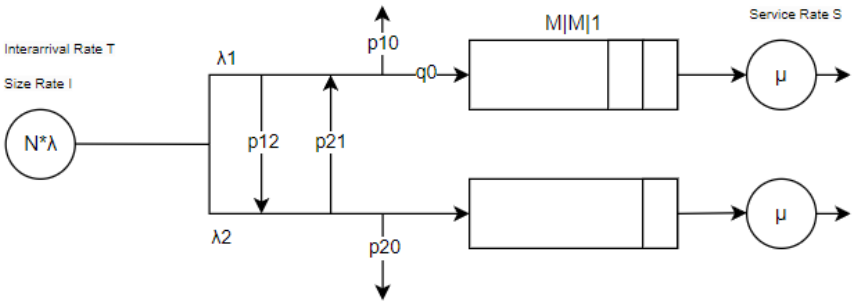

Figure 2.1: System modeling scheme.

The system under study is represented by the scheme shown in Figure 2.1. Each base station is modeled as an `M/M/1/K` queuing system, where λ_i represents the arrival rate of tasks at base station i, and μ is the service rate. Both λ_i and μ are exponentially distributed random variables, which aligns with the Markovian assumptions commonly used in queuing theory. The queues are of finite length, with a maximum capacity of K slots per base station. Any tasks arriving at a base station with a full queue are discarded. The system consists of N users generating tasks at an interarrival rate λ. These tasks are routed to one of the available base stations according to their proximity and load. The system supports task forwarding between base stations, as indicated by the probabilities p_{ij}, which represent the likelihood of a task being forwarded from base station i to base station j. Forwarding introduces an additional latency, which is assumed to be constant for all base stations. As illustrated in the scheme, each task is processed according to a *First Come First Served* (FCFS) policy. The finite queue length K ensures that the system reflects realistic constraints, where resources are limited, and overloading results in task

loss. Additionally, tasks may originate from users distributed either uniformly or according to a lognormal distribution, allowing for the evaluation of the system's performance under diverse user distributions.

2.1 System Parameters

Below, we provide a detailed explanation of all the parameters used in the system model, with specific emphasis on their dependencies and characteristics:

- N**: Number of users**
  The total number of users in the system. These users are distributed across the 2D floorplan according to either a *uniform* or a *lognormal* distribution. Higher values of N result in increased load on the base stations.

- K**: Queue length**
  The maximum number of tasks a base station can hold in its queue at any given time. If the queue is full, additional incoming tasks are discarded. This parameter models resource constraints at the base station level.

- λ_i**: Interarrival rate for base station** i
  The task arrival rate for each base station i depends directly on the user distribution. In the case of a *uniform distribution*, users are evenly spread across the floorplan, leading to nearly balanced λ_i values across all base stations. In contrast, for a *lognormal distribution*, certain base stations may experience higher arrival rates due to user clustering in hotspots, resulting in uneven λ_i values.

- μ**: Instruction rate (fixed)**
  The rate at which instructions are executed at each base station is fixed across the system and represents the computational capacity of the edge servers.

- **Size rate: Packet size distribution (exponential)**
  The size of each task (in instructions per packet) follows an *exponential distribution*. This variability in task size directly impacts the service rate for each base station, making it effectively exponential, as the time to serve a task depends on its size.

- p_{ij}**: Forwarding probability**
  The probability that a task is forwarded from base station i to base station j. This depends on the relative load between base stations and is used to balance the computational workload across the system.

- D**: Forwarding delay**
  The additional latency incurred when a task is forwarded to a less-loaded base station. This delay is assumed to be constant for all tasks and base stations and represents the propagation time in the mesh network.

- $\mu_{\log}$: **Mean of the lognormal distribution**
  The mean value of the lognormal distribution used to define the clustering of users in the floorplan. This parameter determines the overall central tendency of user placement in the floorplan.

- $\sigma_{\log}$: **Standard deviation of the lognormal distribution**
  The standard deviation of the lognormal distribution, controlling the spread of user placement. Higher values of $\sigma_{\log}$ result in more dispersed user locations, while lower values create tighter clusters.

Chapter 3

Implementation

The implemented system in OMNeT++ is organized into modular components to reflect the structure and behavior of the edge computing-enabled cellular network. Each module has a specific responsibility, ensuring a clear and maintainable architecture.

3.1 Defined Modules

The following modules have been defined for the simulation:

- **EdgeNetwork (Compound Module)**
  This is the top-level module representing the entire system. It hosts the following simple modules:
  - **BaseStation (Simple Module):** Represents a single base station in the network. It is responsible for receiving, processing, and, if necessary, forwarding packets generated by users.
  - **User (Simple Module):** Represents a user generating computational tasks. Each user sends packets (with lengths determined by a specified distribution) to its nearest base station.

3.2 Module Behavior

The behavior of each module is described in detail below:

3.2.1 EdgeNetwork

- Acts as the parent module, hosting all base stations and users within the simulation.
- Stores the parameters of all contained modules, allowing them to be retrieved during the simulation using parent pointers.

- Provides the spatial layout of base stations and users, ensuring that the correct associations (e.g., nearest base station) are maintained.

3.2.2 BaseStation

- Receives packets from users in the form of `cPacket` objects and processes them based on the specified scenario:
  - **Locally Managed:** If the queue has sufficient free slots, the base station enqueues the packet for local processing, ignoring the state of other base stations.
  - **Forwarding:** Upon receiving a packet, the base station evaluates the load of all other base stations and forwards the packet to the one with the lowest queue load. If no other base station has a lower load, the packet is processed locally.
- Records the following statistics:
  - **Dropped Packets:** The number of packets dropped due to full queues.
  - **Forwarded Packets:** The number of packets forwarded to other base stations.
  - **Queue Length:** The number of packets in the queue over time, used to compute averages.
  - **Response Time:** The time taken for packets to be processed, allowing for average response time computation.

3.2.3 User

- Generates packets with:
  - **Length:** Packet lengths follow an exponential distribution.
  - **Rate:** The packet generation rate also follows an exponential distribution.
- Sends each packet to the nearest `BaseStation`, determined by the Euclidean distance.[1]

[1] This ensures that users are associated with geographically closest base stations, a common assumption in cellular networks.

Chapter 4

Verification

This chapter describes the tests performed to verify the stability and behavior of the system under various scenarios, including extreme and degenerate cases. The results of these simulations help validate the robustness, consistency, and correctness of the implementation.

4.1 Degeneracy Test

This test evaluates the system's stability and behavior under extreme or edge-case conditions. The goal is to ensure that the system remains robust and operates without failures or unexpected behaviors when subjected to these scenarios. By testing the system in degenerate cases, we validate its ability to handle both minimal and excessive loads reliably.

4.1.1 Scenario 1 - No Users

In this degenerate scenario, no users are present in the system. As expected, the system remains stable because there are no packets to process or forward. This verifies the correctness of the system's behavior in edge cases with no load.

4.1.2 Scenario 2 - Very High Number of Users

This scenario tests the system under an extreme load with a very high number of users ($N = 500$) and only two base stations ($M = 4$).

The system demonstrates robustness under these conditions, maintaining a stable *queue length* and consistent *response time*. The results are shown in Figure 4.1 and Figure 4.2, confirming that the system can handle significant load without degenerating.

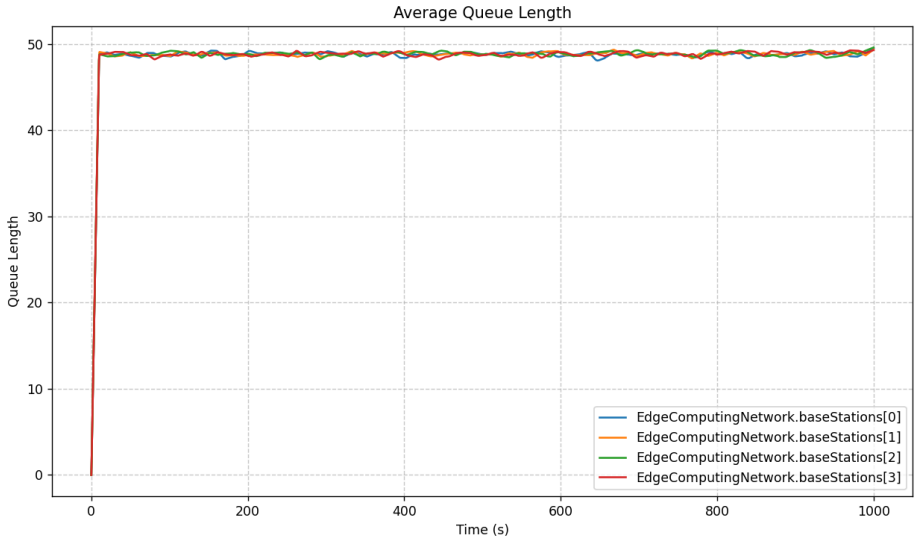

Figure 4.1: Average queue length under high load.

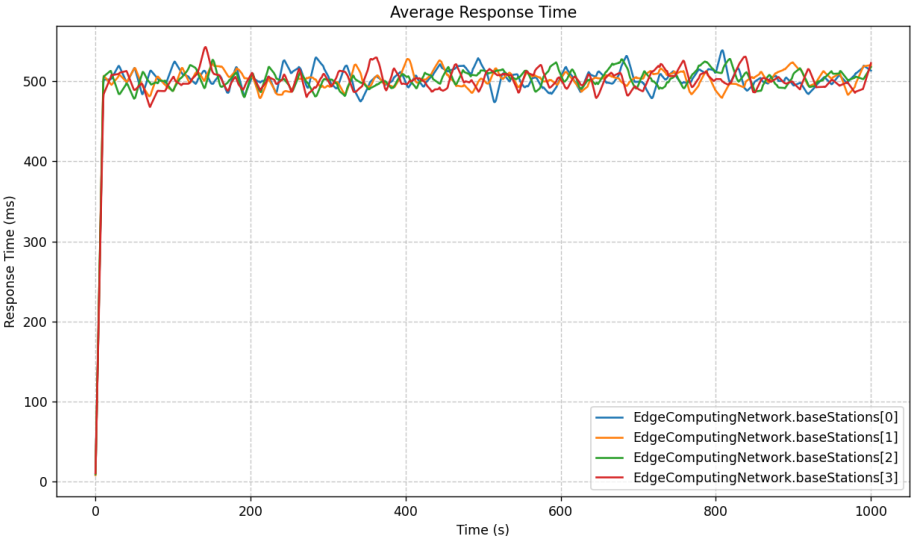

Figure 4.2: Average response time under high load.

4.1.3 Scenario 3 - Very Low Service Rate

This scenario evaluates the system when the service rate is significantly lower than the interarrival rate. The simulations show that the queue length stabilizes very fast (Figure 4.3), while the response time exhibits a *linear growth* over time

(Figure 4.4). This behavior is consistent with expectations under such extreme conditions.

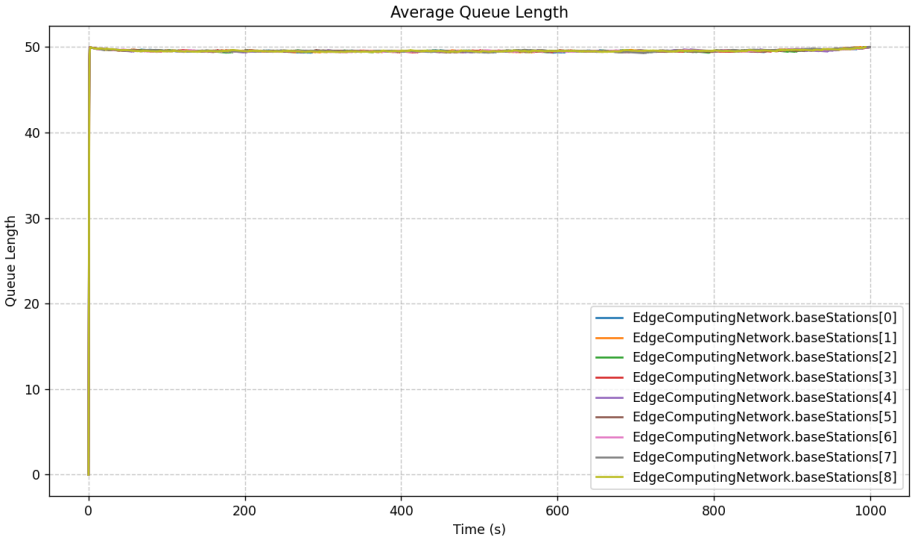

Figure 4.3: Queue length stabilizes under low service rate.

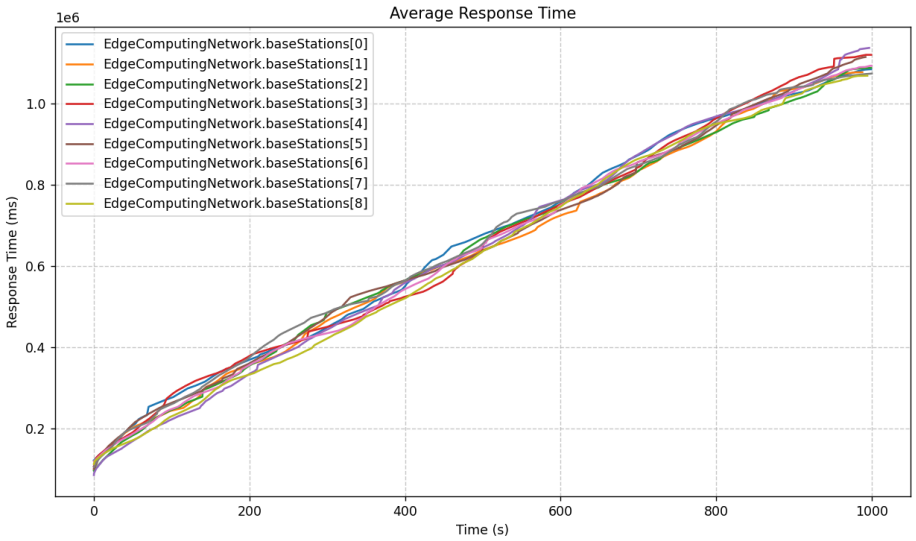

Figure 4.4: Linear growth in response time under low service rate.

4.2 Consistency Test

This test verifies that the system produces coherent results under both configurations: *Locally Managed* and *Forwarding*. The results indicate similar mean response times, with the forwarding configuration showing a slight advantage. However, the forwarding configuration also exhibits lower variability, as seen in Figure 4.5 and Figure 4.6.

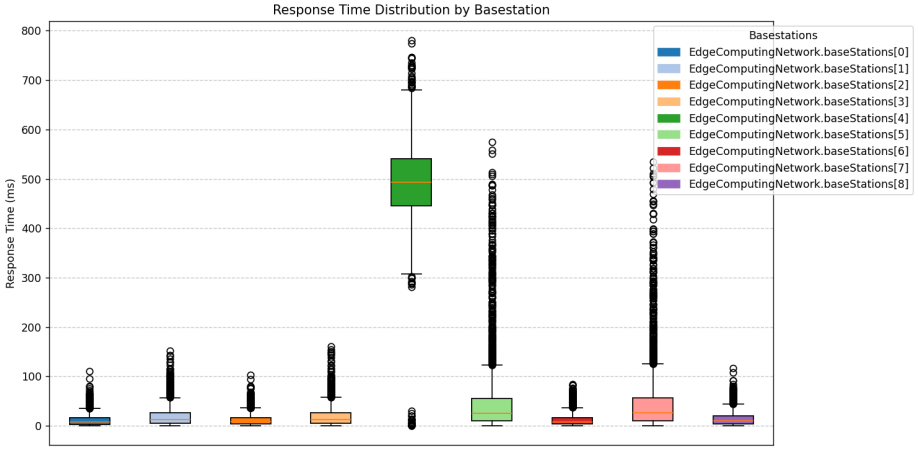

Figure 4.5: Response time: Locally managed.

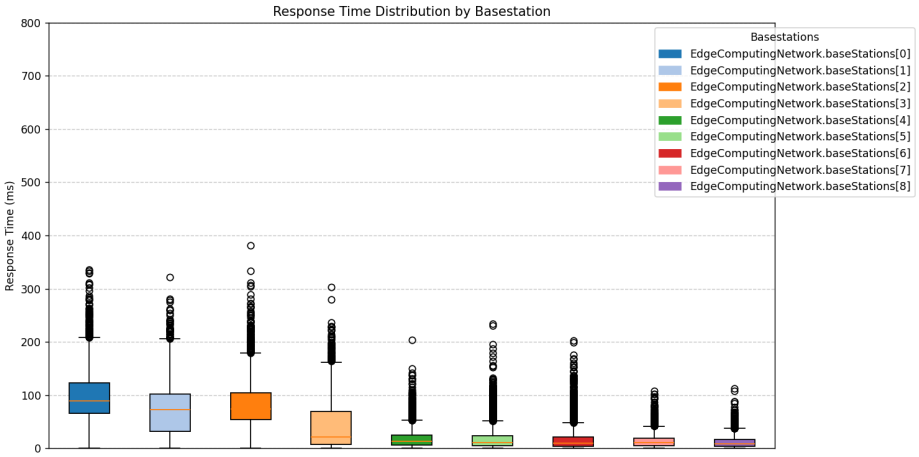

Figure 4.6: Response time: Forwarding.

4.3 Continuity Test

This test ensures that the system responds smoothly when parameters are varied gradually, without abrupt discontinuities in behavior.

4.3.1 Increment of Users ($N = 100, 250, 500$)

As expected, the mean response time increases with the rising number of users. It remains relatively stable between $N = 100$ and $N = 250$, but exhibits a sharp and significant increase as the number of users reaches $N = 500$.

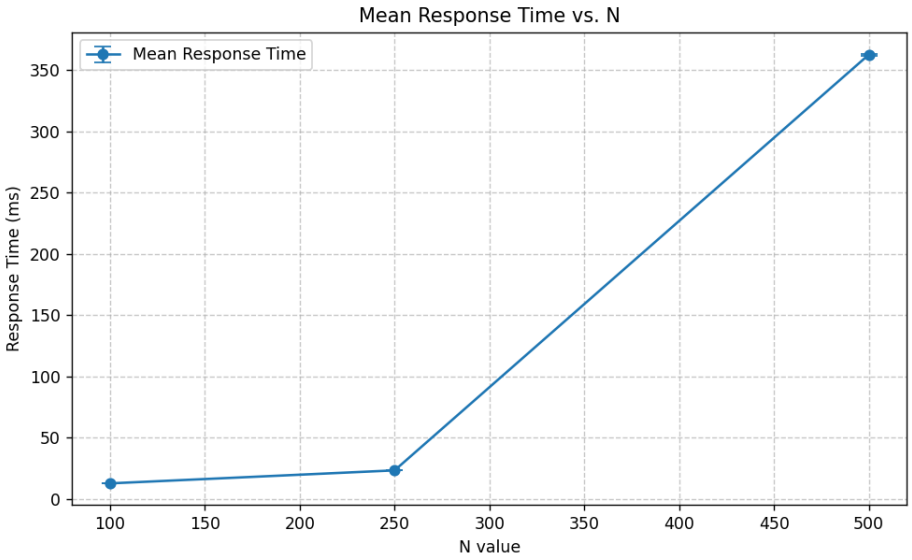

Figure 4.7: Response time for varying numbers of users.

4.3.2 Increment of Requests' Load ($\lambda = 1/0.1, 1/0.5, 1$)

At low values of λ, the mean response time remains relatively low and stable. However, as λ approaches $1/0.1$, the mean response time experiences a sharp increase, nearly an order of magnitude higher. This growth is also accompanied by increased variability in the response time.

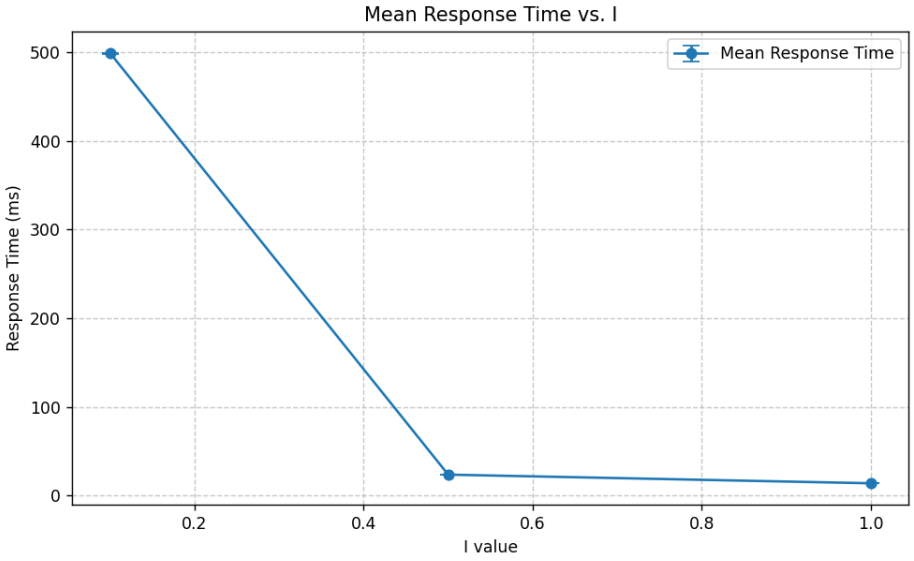

Figure 4.8: Response time for varying interarrival rates.

4.3.3 Increment of Size Rate ($\mu = 100, 1000, 10000$)

The response time increases significantly as the size rate grows. Notably, for $\mu = 10000$, the response time is two orders of magnitude higher compared to the lower size rates. This trend highlights the strong dependency of response time on the size rate of each packet, as larger packets require substantially more processing time.

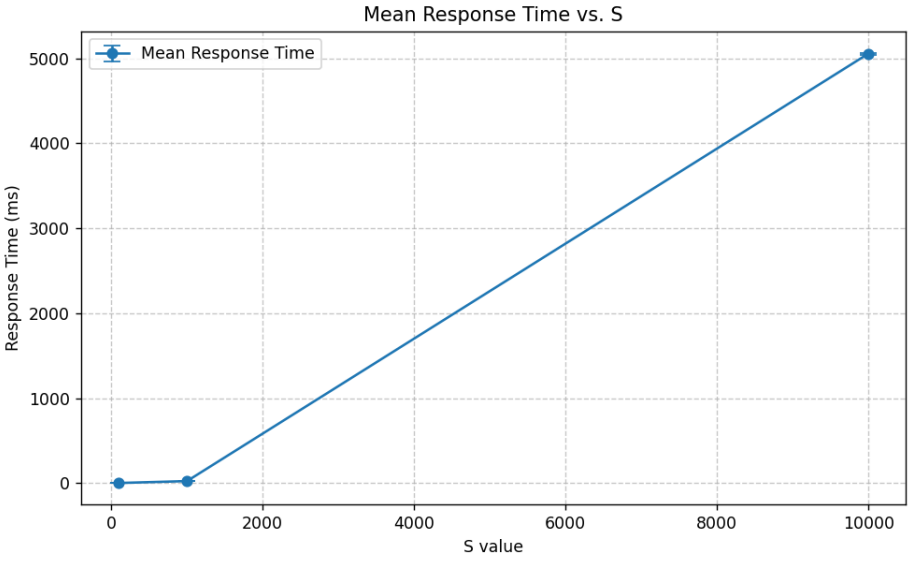

Figure 4.9: Response time for varying service rates.

Chapter 5

Calibration

In this chapter, we discuss how the factors and parameters have been chosen throughout the development of our system. We distinguish between *fixed factors* and *varying factors* (representing different scenarios).

5.1 Factors Calibration

The objective of this section is to specify the intervals of the key factors in order to correctly reproduce realistic conditions in our system.

5.1.1 Fixed Factors

The following factors remain constant throughout our experiments:

- **Number of base stations (M)**: 9
- **Dimensions of the area**: height = 1800 m, width = 1800 m
- **Service rate**: 10^5 (packets/second)
- **Delay**: 50 ms
- **Queue size**: 50
- **Distribution parameters**:
  - Mean: $\log\left(\frac{1800}{2}\right)$
  - Standard Deviation: 0.4

The number of base stations ($M = 9$) aims to reflect a moderately sized area with multiple sites, balancing suburban and urban deployments. The dimensions of the area ($1800\,m \times 1800\,m$) were chosen to represent a typical environment where base stations are neither too sparse nor too dense. The service rate of 10^5 packets/second, along with a 50 ms delay and a queue size of 50, is

used to simulate a realistic cellular-network-like environment where moderate buffering and short delays are expected. Finally, the log-normal distribution (characterized by the specified mean and standard deviation) models the spatial distribution of users. This distribution is particularly suitable for analyzing cases where user concentration is uneven across the area, such as in scenarios with hotspots and sparse regions.

5.1.2 Varying Factors

The following factors vary to represent different scenarios and test system performance under varying conditions:

- **Number of users (N)**:
  - $\boldsymbol{N = 100}$: Represents a lightly loaded scenario, simulating a low-density environment with fewer active users.
  - $\boldsymbol{N = 250}$: A moderately loaded scenario, reflecting a typical environment with a balanced user density.
  - $\boldsymbol{N = 500}$: A heavily loaded scenario, used to evaluate system performance under high user density and demand.
- **Interval rate**: This parameter determines how frequently users generate requests (i.e., the arrival rate). We consider two scenarios:
  - **Medium-case scenario**: **1/0.5**, representing a moderate traffic load with steady user activity.
  - **Extreme-case scenario**: **1/0.1**, emulating peak traffic conditions with frequent user requests.
- **Size rate**:
  - **Medium-case scenario**: $\mathbf{10^3}$: Corresponds to smaller data packet sizes, typical of lightweight applications or systems with low bandwidth requirements.
  - **Extreme-case scenario**: $\mathbf{10^4}$: Represents larger data packet sizes, increasing system load and reflecting heavier traffic conditions.

The combination of these parameters enables us to evaluate the system under a broad range of realistic conditions, from moderate loads with fewer users and smaller data sizes to extreme loads with higher user counts, larger data sizes, and higher arrival rates. These variations ensure that the analysis is comprehensive and captures critical performance metrics under both typical and challenging conditions.

5.2 Warmup Time Analysis

Warmup time refers to the period during which the system transitions from an initial transient state to a steady state. Properly setting the warmup time ensures that the system's performance metrics are evaluated only during the steady-state period, avoiding bias caused by the initial transient. In our analysis, we observed the response time as a function of time. As shown in Figure 5.1, the response time stabilizes and reaches a steady state within 100 seconds. The transient period, characterized by larger variations in response time, does not exceed 100 seconds across all experiments. Based on these observations, the warmup time is empirically set to 100 seconds. This ensures that any data collected during the transient state is excluded from the analysis, leading to more accurate and reliable results.

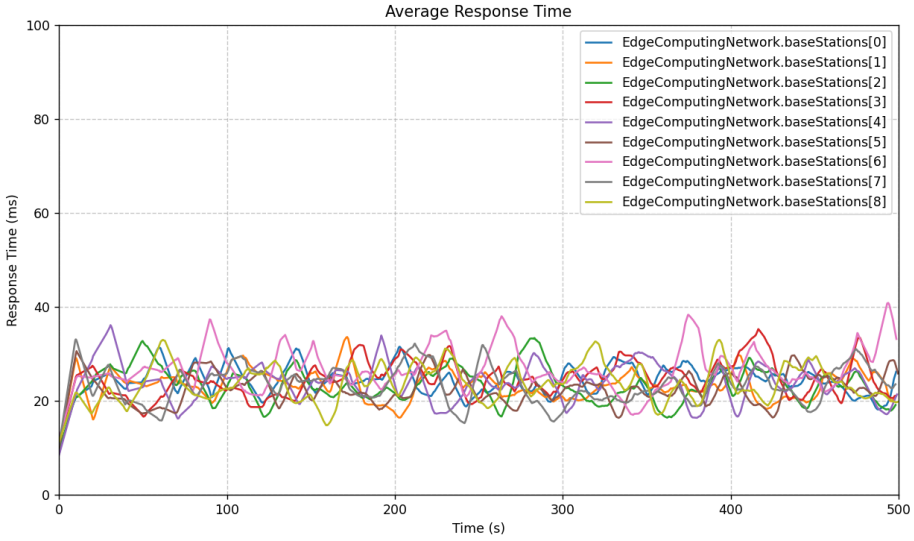

Figure 5.1: Response time as a function of time. The transient period does not exceed 100 seconds, indicating the warmup time.

Chapter 6

Simulation Experiments

The experiments take into account 3 factors:

- **N**: (100, 250, 500)
- **Interarrival rate**: $(\frac{1}{0.1}, \frac{1}{0.5})$
- **Size rate**: $(\frac{1}{10^3}, \frac{1}{10^4})$

Considering the average case as a reference:

- $N = 250$
- $I = \frac{1}{0.5}$
- $S = \frac{1}{10^3}$

6.1 How increasing the umber of users affects the system

A system of telecommunications should be robust to the of the number of users, meaning it has to remain functional if the situation becomes more or less crowded.

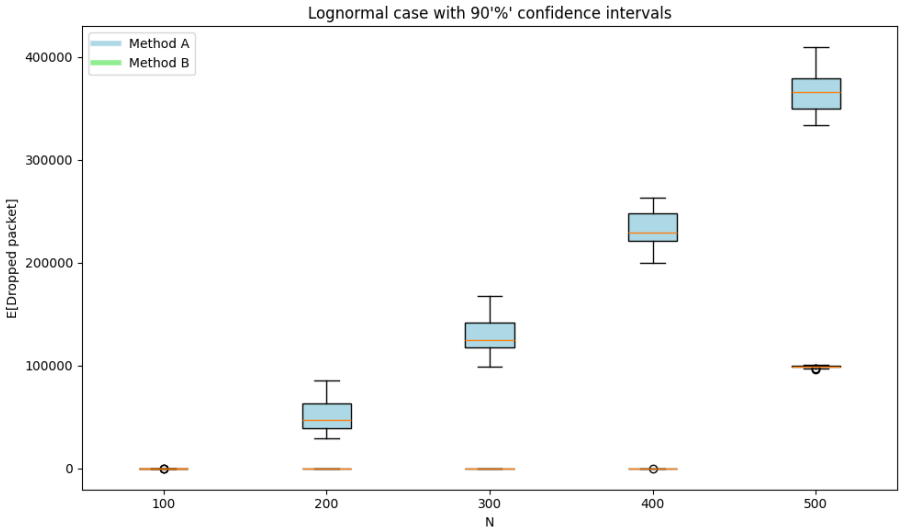

Figure 6.1: Distribution of dropped packets for lognormal distribution

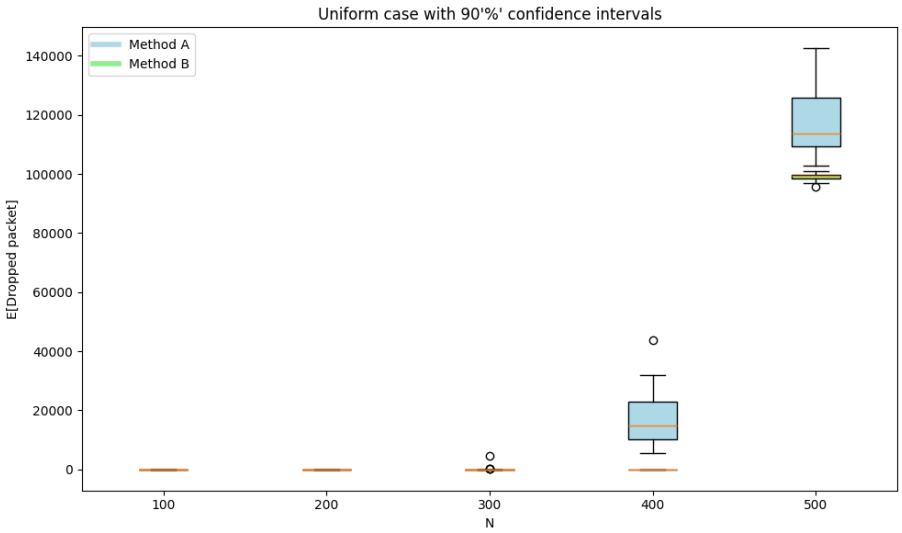

Figure 6.2: Distribution of dropped packets for uniform distribution

By looking at the distribution of dropped packets we can say that (as we could expect) with an increasing number of users the number of discarded packets raises, being manageable in the uniform case for both methods except when the number of users becomes 500. The lognormal case follows a worse response for method A and presents situation similar to the previous one for method B.

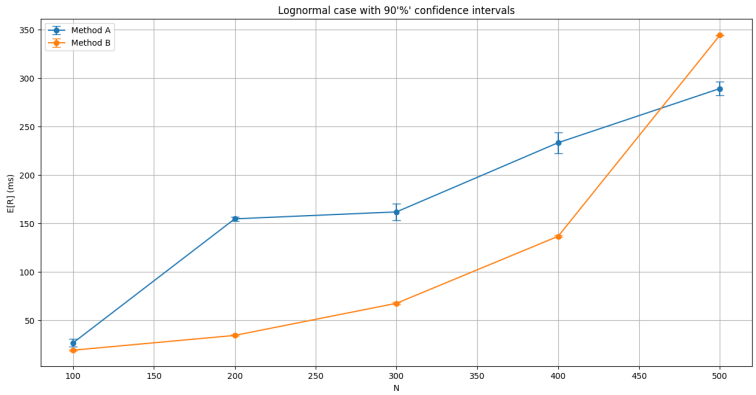

Figure 6.3: Mean response time trend per number of users (lognormal)

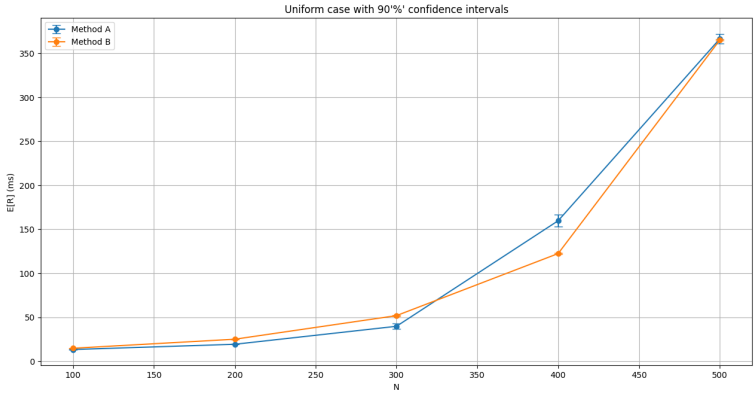

Figure 6.4: Mean response time trend per number of users (uniform)

Overall, in the lognormal case method B is better in all cases but in the last one (500 users). This might be caused by the fact that base stations always try to forward packets, overloading the system.

The uniform case sees both methods perform the same way, with method B being the best one only in the 400 users case.

6.2 The effect of forwarding packets on the system

We test our system in both cases A and B to see the effects of forwarding packets wether the packets are generated acording to a uniform distribution or a lognormal distribution.

6.2.1 Size rate variation - Method A

Case $S = \frac{1}{10^3}$

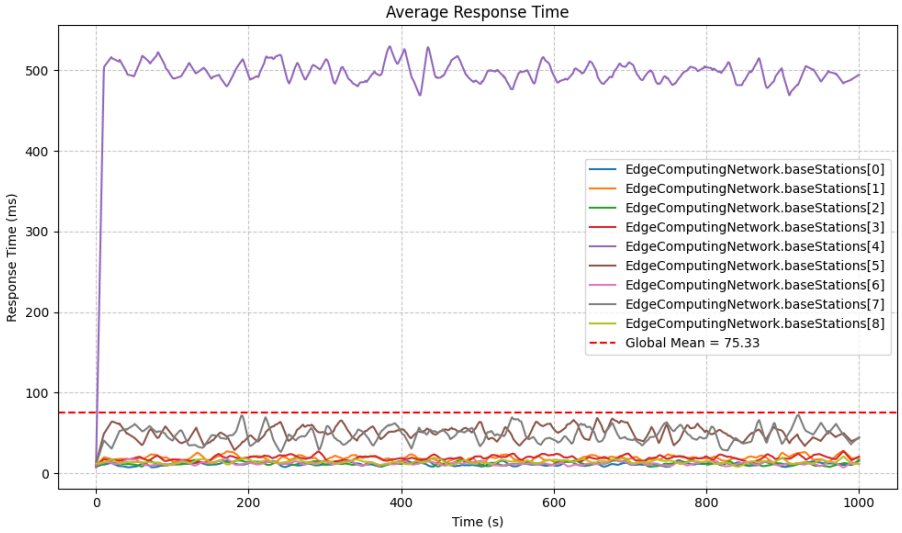

Figure 6.5: $E[R]$ for the lognormal distribution

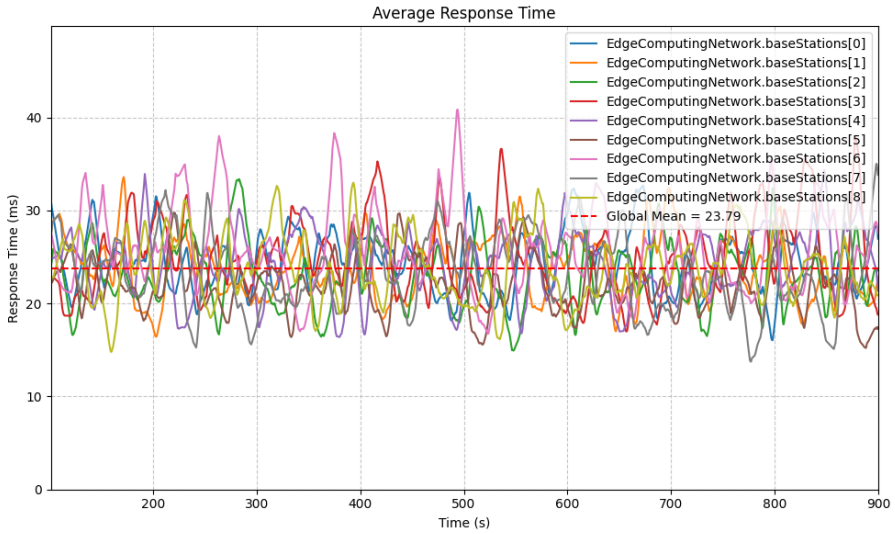

Figure 6.6: $E[R]$ for the uniform distribution

The difference between the two distribution is remarkable, as in the lognormal case we find a higher mean response time and queue length.

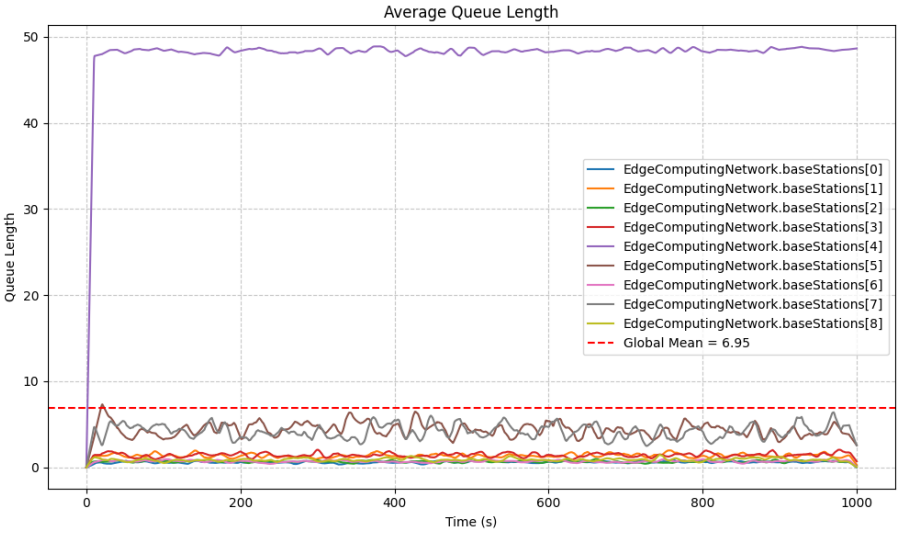

Figure 6.7: $E[qlen]$ for the lognormal distribution

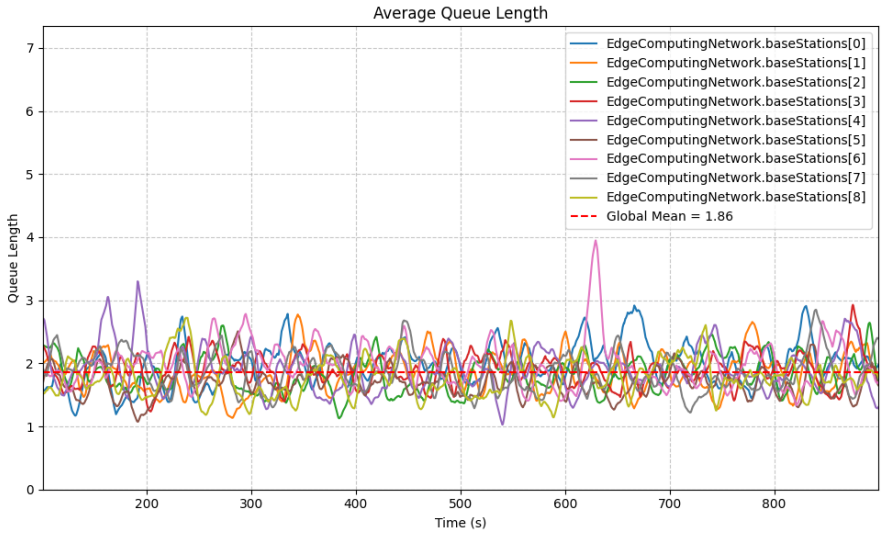

Figure 6.8: $E[qlen]$ for the uniform distribution

In the uniform distribution the values are more compact around the mean, unlike the lognormal case, where is evident that there are one or more base stations dealing with a heavier load.

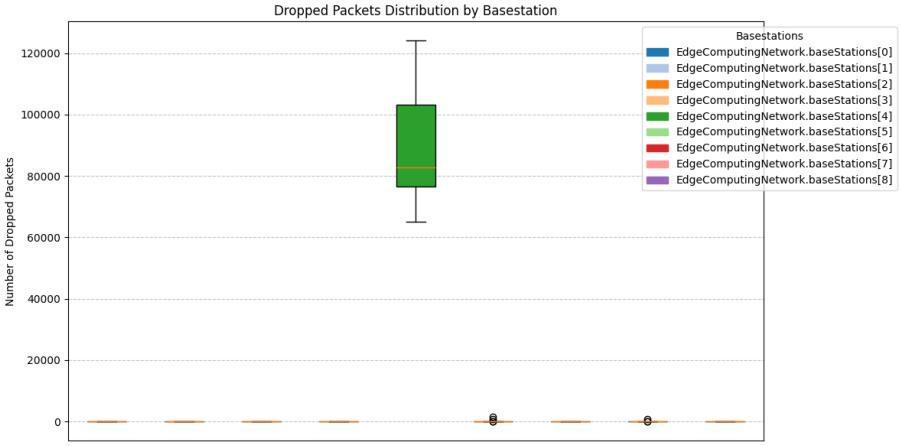

Figure 6.9: Dropped packets distribution (lognormal)

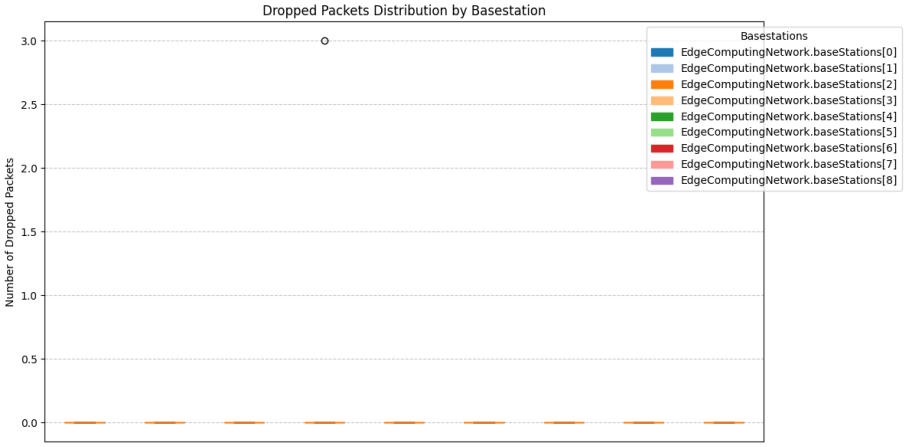

Figure 6.10: Dropped packets distribution (uniform)

Since base stations use the policy where packets are served locally, a higher queue length might mean a higher number of discarded packets. Even though, the system can still sustain the load in the uniform case as the number of discarded packets is lower than 10.

Not the same for the lognormal case, where the total of discarded packets is over 2.5×10^5 units.

Case $S = \frac{1}{10^4}$

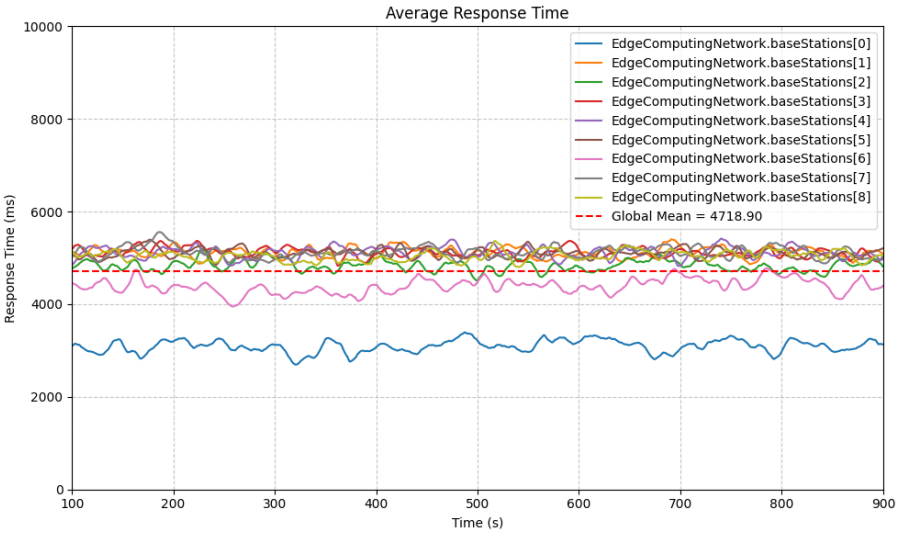

Figure 6.11: $E[R]$ for the lognormal distribution

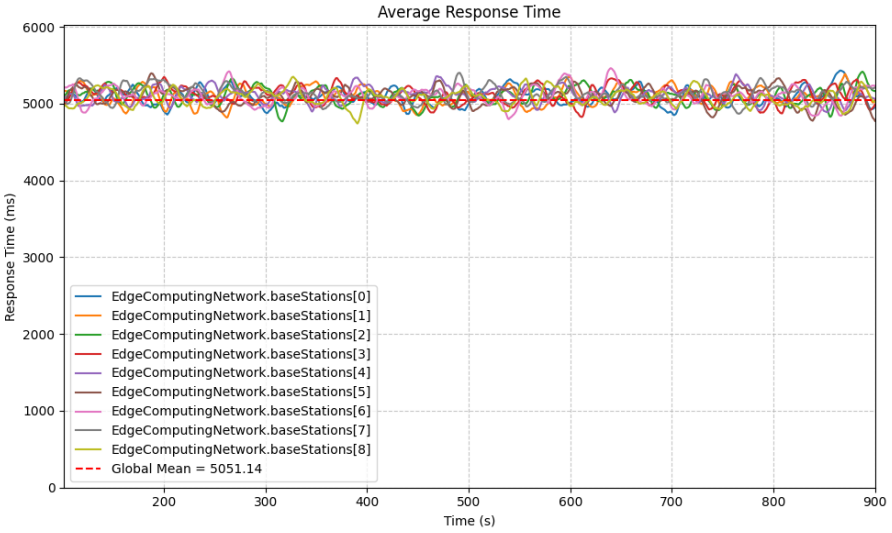

Figure 6.12: $E[R]$ for the uniform distribution

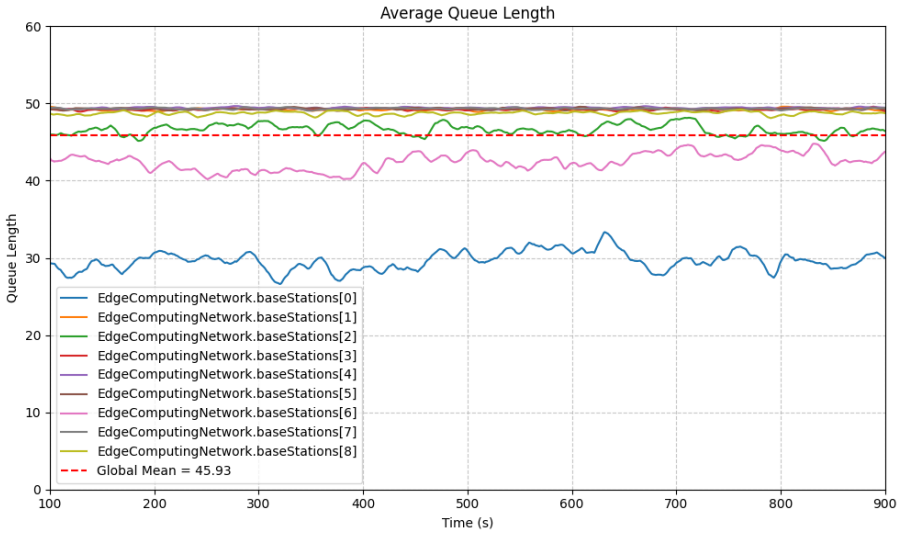

Figure 6.13: $E[qlen]$ for the lognormal distribution

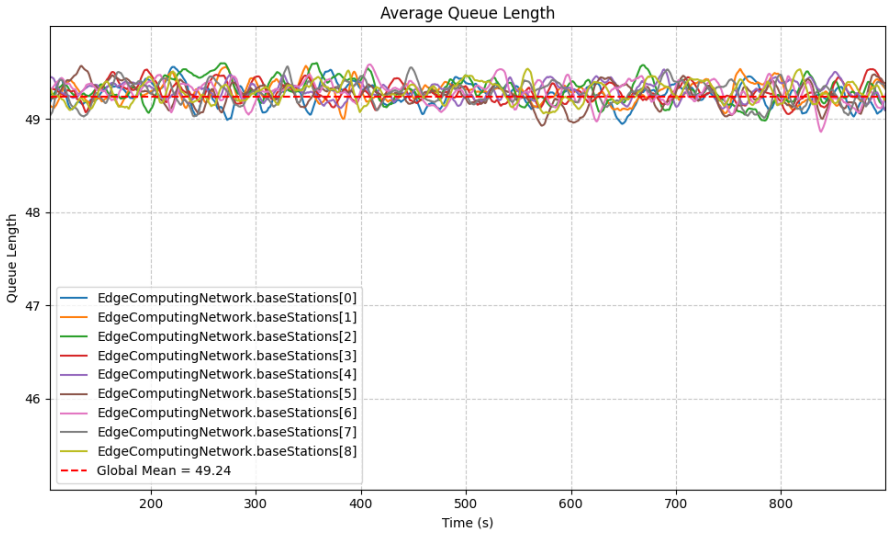

Figure 6.14: $E[qlen]$ for the uniform distribution

This experiment shows how the system can better sustain the load in the log-normal case dealing with bigger packets. Unlike the previous experiment, this time we find similar behaviours.

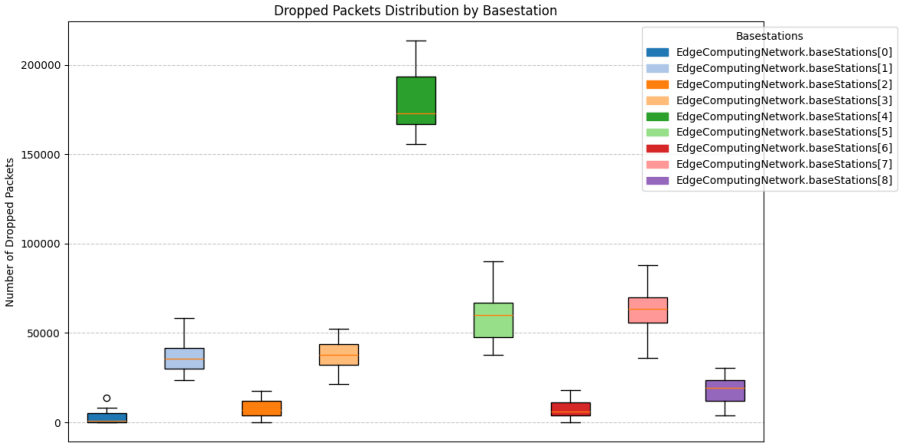

Figure 6.15: Dropped packets distribution (lognormal)

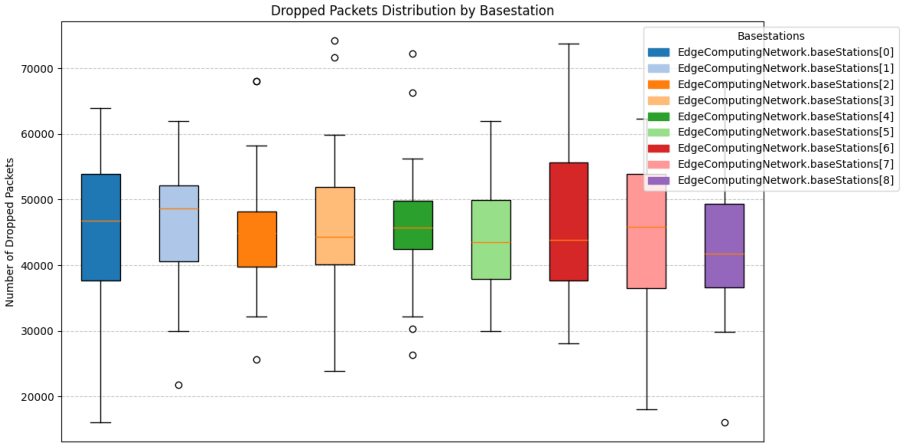

Figure 6.16: Dropped packets distribution (uniform)

Also the distribution of dropped packets is similar, presenting more concentrated values in the lognormal case.

6.2.2 Size rate variation - Method B

Case $S = \frac{1}{10^3}$

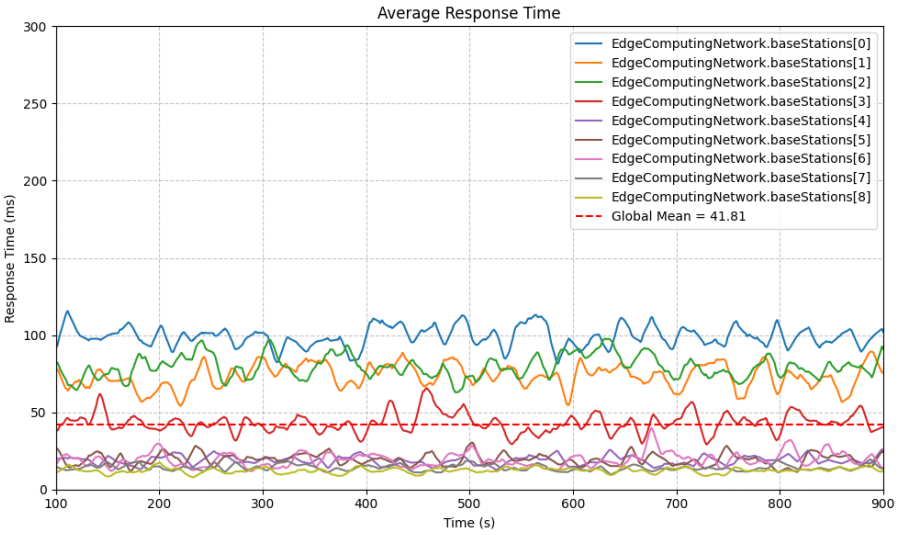

Figure 6.17: $E[R]$ for the lognormal distribution

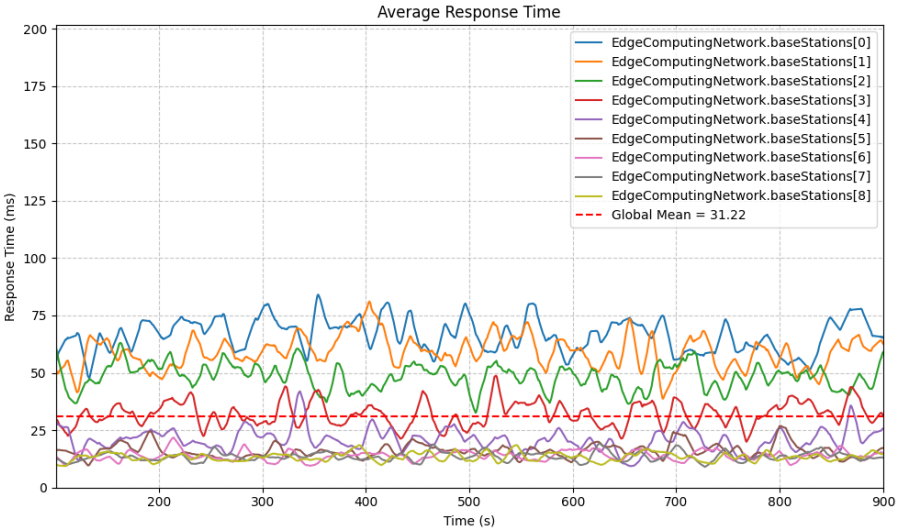

Figure 6.18: $E[R]$ for the uniform distribution

As we can see the performance of the system improves with respect to the method A, but only when dealing with the lognormal distribution. The mean response time in the uniform case gets higher due to the forwarding directed to the same couple of base stations, this behaviour is responsible to raise the mean response time.

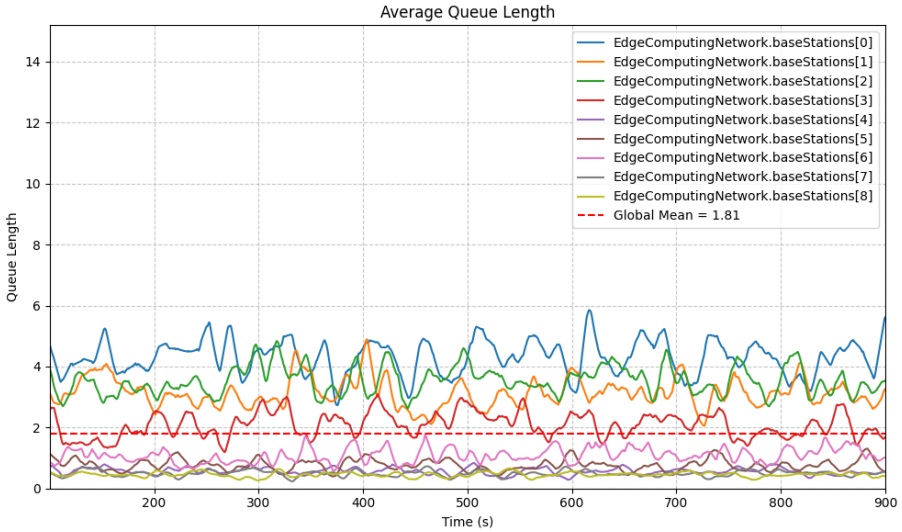

Figure 6.19: $E[qlen]$ for the lognormal distribution

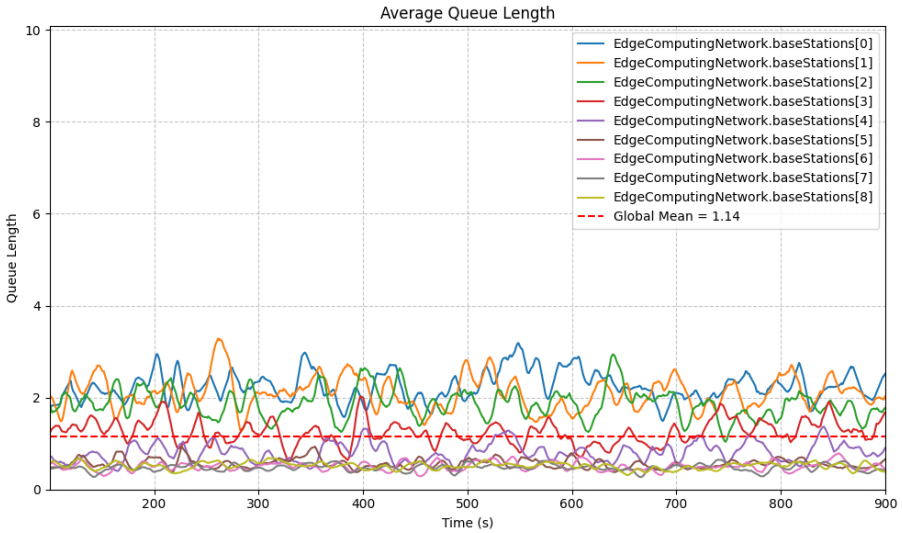

Figure 6.20: $E[qlen]$ for the uniform distribution

The same reasoning can be applied to the mean leangth of each queue, as the plots show the exact same behaviour.

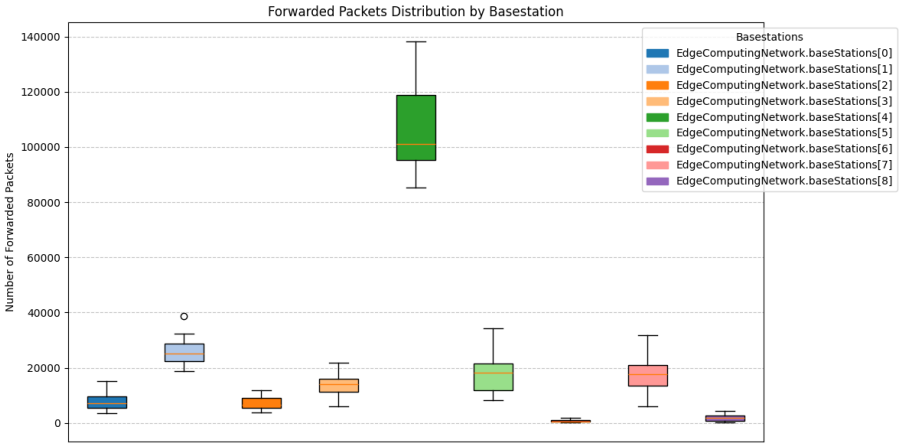

Figure 6.21: Forwarded packets distribution (lognormal)

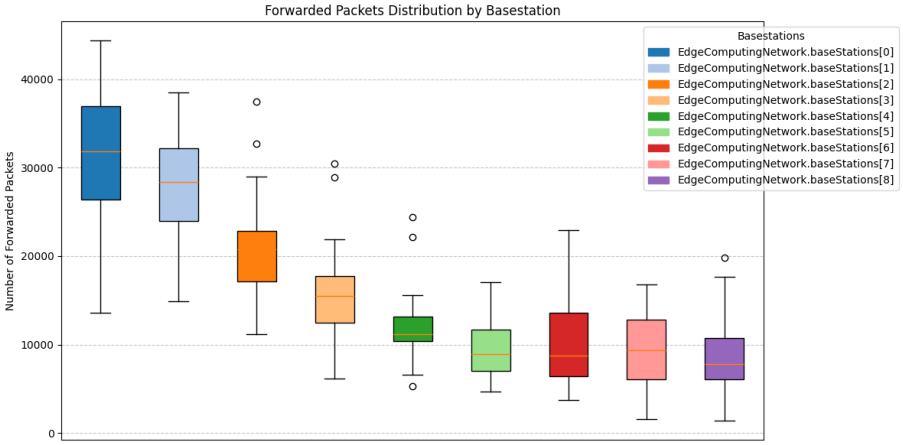

Figure 6.22: Forwarded packets distribution (uniform)

It is curious that in the lognormal case packets tend to be forwarded to one particular base station unlike the uniform case, where are forwarded more evenly. The lognormal case presents a higher number of forwarded packets (5.9×10^6 vs 4.3×10^6).

Dropped packets are not mentioned as there has been none throughout the experiment.

Case $S = \frac{1}{10^4}$

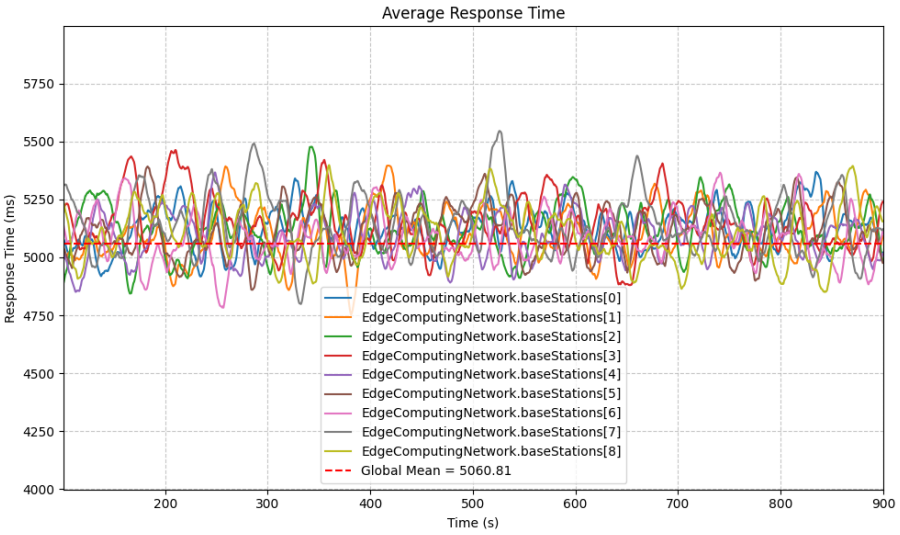

Figure 6.23: $E[R]$ for the lognormal distribution

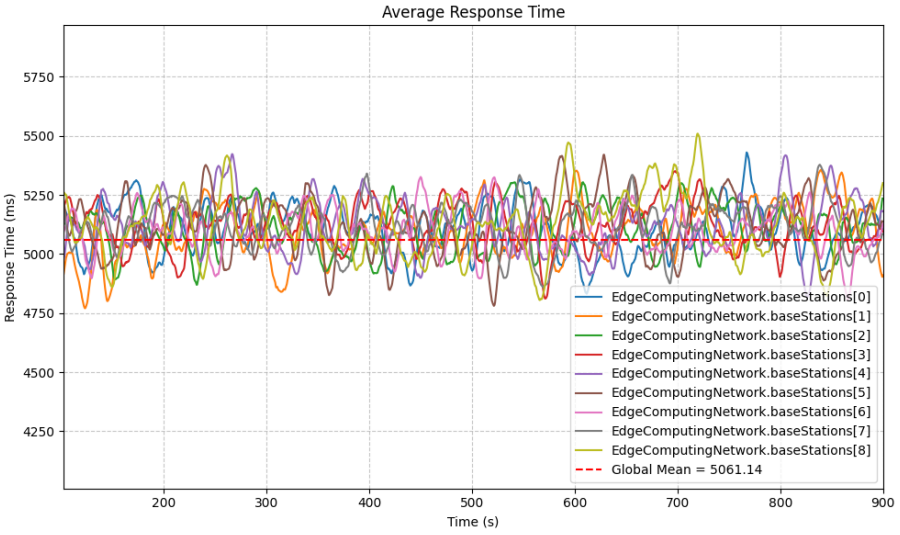

Figure 6.24: $E[R]$ for the uniform distribution

In this case it is cear that the system is stressed out in both the distribution with almost equal mean response time. All the values of the base stations are

concentrated around the global mean, which represents the fact that all of them respond at almost the same speed.

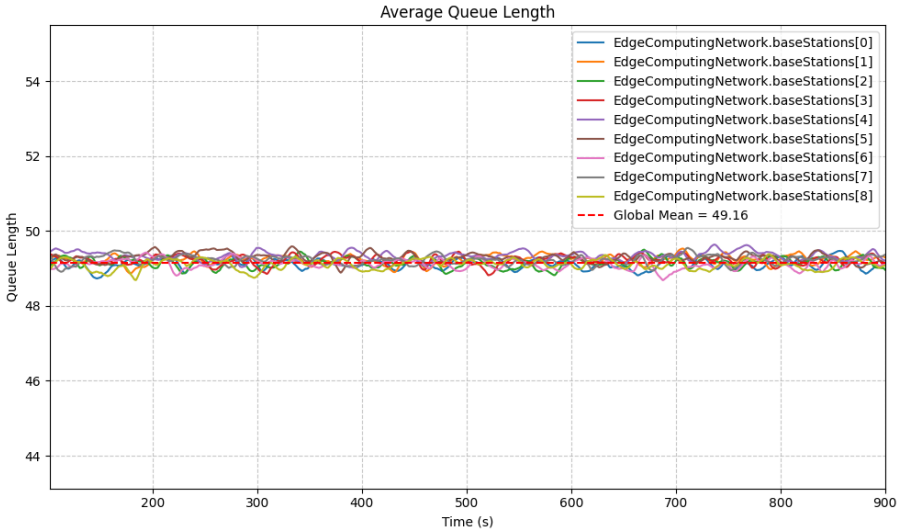

Figure 6.25: $E[qlen]$ for the lognormal distribution

The same reasoning can be applied to the mean leangth of each queue, as the plots show the exact same behaviour.

Since for both cases the plots are the same, it is shown the one relative to the lognormal distribution only.

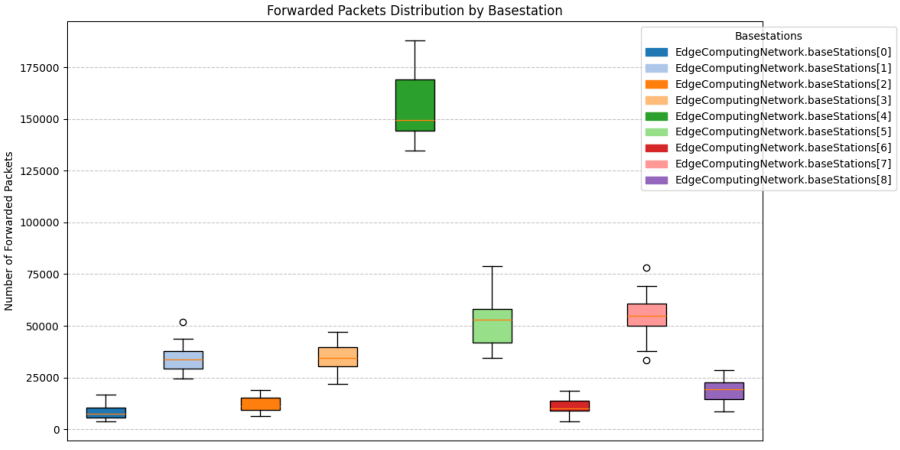

Figure 6.26: Forwarded packets distribution (lognormal)

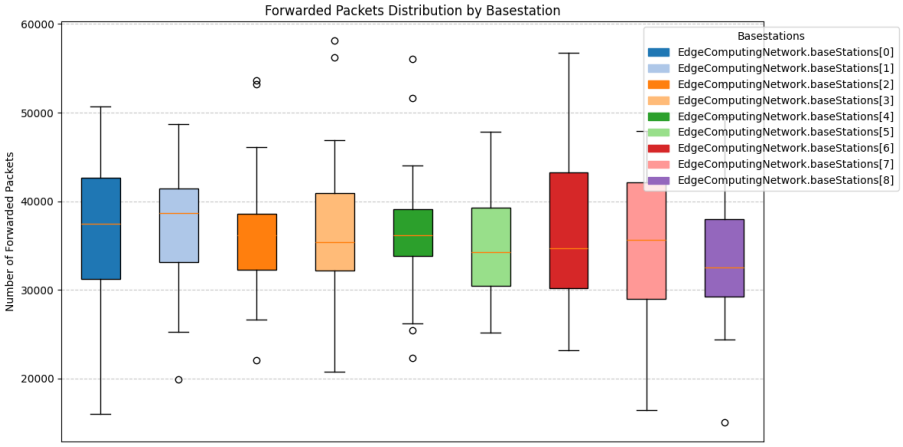

Figure 6.27: Forwarded packets distribution (uniform)

It is shown the same behaviour as in the method A, the only difference is that in the uniform case packets are forwarded more evenly among all the base stations.

We report a total of 9.7×10^6 forwarded packets for the uniform and 11.4×10^6 for the lognormal.

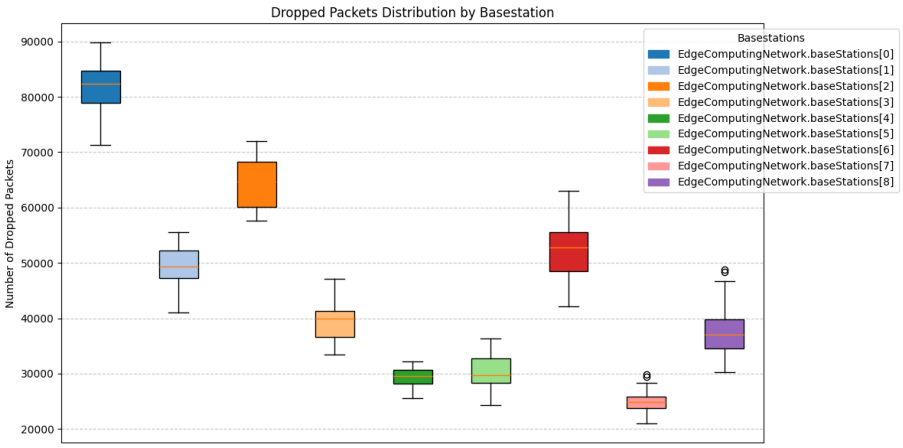

Figure 6.28: Dropped packets distribution (lognormal)

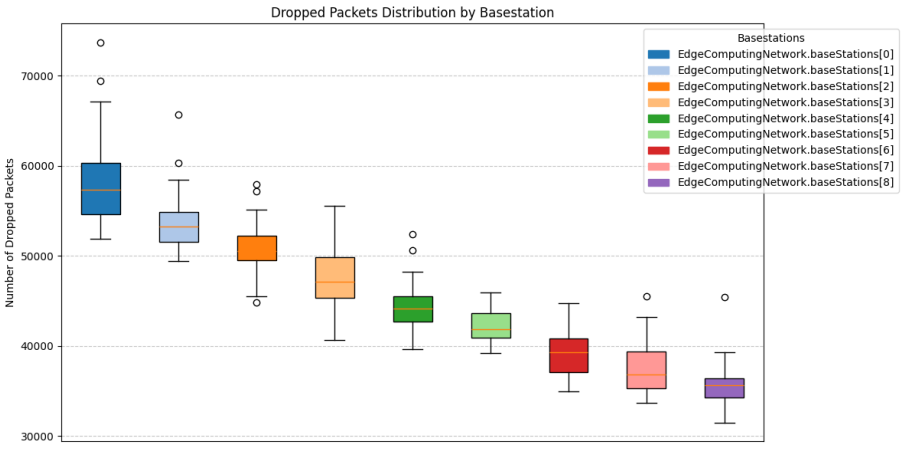

Figure 6.29: Dropped packets distribution (uniform)

As we can expect base stations that forward the majority of the packets are the ones that discard few of them and vice versa. We encounter this phenomenon for both the distributions.

We report the same number of discarded packets both for the lognormal and the uniform distribution.

Chapter 7

Conclusions