UNIVERSITÀ DI PISA

Computer Engineering

Performance Evaluation of Computer Systems and Networks

Preemptive Priority Scheduler

Team members:

Francesco Martoccia

Salvatore Lombardi

Anna Fabbri

ACADEMIC YEAR 2021/2022

Contents

1 - Introduction

1.1 Problem Description

Consider a **preemptive priority scheduler** with two queues. The scheduler serves the jobs in the **high priority queue** whenever the latter is non empty. Jobs in the **low priority queue** are served only when the **high priority queue** is empty, and their service is stopped (**preempted**) in case the latter queue becomes non-empty. The service of preempted jobs has to be restarted from the beginning, (i.e., resuming is not allowed).
Study the response time of the low priority queue when the workload of **both** queues is varied. The job interarrival times are IID RVs (to be described later), and their service times have constant value.

More in detail, at least the following scenarios must be evaluated:

- The workload of **both** queues has constant interarrival times. Vary the service times until the load of the system approaches 100%.
- Exponential distribution of the job interarrival times.

1.2 Objectives

The objective of this project is to study the response time of the low priority queue when the workload of **both** queues is varied considering different scenarios, including those described above.

1.3 Performance Indexes

In order to assess the system's performance the indexes that were taken into account are:

- **E[R]**: Mean Response Time (mean time between the arrival and the departure of the same job).
- **E[N]**: Mean Number of Jobs in the system.

2 - Modeling

2.1 Assumptions

The assumptions that were made are the following:

- The **interarrival time** (the time between the arrival of 2 different jobs in the system) is modeled as a **RV** (Random Variable) with either an **exponential distribution** or a **constant value**.
- The **Mean interarrival time** of the **high** priority jobs is **greater** than the one of the low priority jobs.
- The **service time** of a job has a **constant value**.
- Both High Priority and Low Priority queues are **FIFO** and have **unlimited capacity**.
- **No propagation error** in sending the jobs to the scheduler.

2.2 Preliminary Validation

Before implementation, it is essential to conduct a preliminary validation phase to ensure the accuracy of the model. To accomplish this, an analysis of the previously established general assumptions is required.

- The second assumption is justified by the fact that, since we have to study the response time of **low jobs**, we need their **arrival rate** to be **higher than** that of **high jobs** to ensure that they can be processed by the scheduler.
  Assuming **otherwise**, on the other hand, **only high-priority jobs would be served most of the time**, and this would not allow the response time of low jobs to be properly studied.
- The choice of a **FIFO queue** ensures fairness in serving jobs based on their arrival order.
  The assumption of **unlimited queue capacity** simplifies the model by removing the need to consider the possibility of dropped or rejected jobs due to

queue overflow. This simplification is especially beneficial when analyzing the system under high-load scenarios.

- In a real scheduler, it is possible that, due to errors or factors of various kinds, there is a **loss of jobs**. The frequency of these errors or failures depends on the specific system and its configuration. This aspect is not taken into account in the considered model, to avoid adding too much complexity and to focus the analysis on the behaviour of the scheduler and the performance of the system.

2.3 Factors

The factors that can influence the performance of the considered system are:

- $\frac{\mathbf{1}}{\boldsymbol{\lambda}_{\mathbf{L}}}$: the **Mean Interarrival Time** of the Jobs with **Low** Priority.
- $\frac{\mathbf{1}}{\boldsymbol{\lambda}_{\mathbf{H}}}$: the **Mean Interarrival Time** of the Jobs with **High** Priority.
- $\frac{\mathbf{1}}{\boldsymbol{\mu}_{\mathbf{L}}}$: the **Mean Service Time** of the Jobs with **Low** Priority.
- $\frac{\mathbf{1}}{\boldsymbol{\mu}_{\mathbf{H}}}$: the **Mean Service Time** of the Jobs with **High** Priority.

3 - Implementation

3.1 Modules

The modules defined are the following:

- **Producer**: module that produces low or high priority jobs to send to the scheduler.
- **Scheduler**: module that handles the processing of jobs, taking into account their arrival order and their priority.

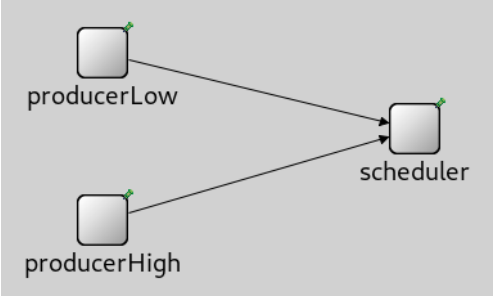

Figure 3.1: Network implementation in Omnet++

3.2 Messages

A new message format has been defined to represent the jobs, which can have high or low priority, to be processed by the scheduler. In particular, the message of type "Job" contains the following fields:

- **bool isHighPriority**: a boolean to check the priority of the job;
- **simtime_t queueArrival**: the arrival time of the job in the correct priority queue of the scheduler;
- **simtime_t serviceTime**: the time needed to process the job.

3.3 Modules' Behaviour

The network was built using **two instances** of the **Producer** simple module connected to **one instance** of the **Scheduler** simple module.

3.3.1 Producer

The Producer module is initialised with the parameters defined in the *omnetpp.ini* file.
At the start, if the arrival distribution is set as a **constant**, the **arrival time** of the job will be **equal to the value** chosen **for the arrival mean**. In the case of the **exponential distribution**, however, the **arrival time** of the job will be **generated randomly according to the arrival mean**. After these steps, a timer is set with the obtained arrival time.
When the timer expires, the fields of the **job message** are filled in with the **priority type** and the **service time**, which must be generated or taken as a constant value, as is the case for the arrival time.
Finally, the job message is sent to the scheduler module.

3.3.2 Scheduler

The scheduler module comprises **two FIFO queues**, implemented as *std::queue*, namely a high-priority queue and a low-priority queue.
Upon job arrival, the scheduler checks its priority and assigns it to the respective queue accordingly. **High-priority jobs are given precedence over low-priority jobs**, and processing begins immediately for them. Low-priority jobs, on the other hand, are only processed when the high-priority queue is empty.
In the event of a high-priority job arrival while a low-priority job is being executed, the progress of the latter is reset, allowing the high-priority job to commence processing (**preemption**). The low-priority job remains in the queue for subsequent processing.

4 - Verification

The step of verification is necessary because it allows us to assess the main characteristics of the model.

4.1 Degeneracy Test

The degeneracy test is used to analyze the behavior of the system in limit situations where the parameters are either set to 0 or to extreme values.
Three different configurations were used, all with constant values of interarrival time and service time:

Test 1

- $\mathbf{\frac{1}{\lambda_L} = 0s}$, $\mathbf{\frac{1}{\lambda_H} = 0s}$, $\mathbf{\frac{1}{\mu_L} = 0s}$, $\mathbf{\frac{1}{\mu_H} = 0s}$

In this case, the simulation did not go beyond t=0 because the interarrival times were 0, but the jobs in the high queue were instantly processed, since their service time was 0s.

Test 2

- $\mathbf{\frac{1}{\lambda_L} = 0s}$, $\mathbf{\frac{1}{\lambda_H} = 0s}$, $\mathbf{\frac{1}{\mu_L} = 0.1s}$, $\mathbf{\frac{1}{\mu_H} = 0.1s}$

In this situation the simulation time remained always at 0 because the interarrival times were set to 0s. Indeed, the jobs were never processed since their service time was 0.1s.

Test 3

- $\mathbf{\frac{1}{\lambda_L} = 0.1s}$, $\mathbf{\frac{1}{\lambda_H} = 0.1s}$, $\mathbf{\frac{1}{\mu_L} = 0s}$, $\mathbf{\frac{1}{\mu_H} = 0s}$

In this scenario both the low and high jobs are processed since they have a service time of 0s that is less than the interarrival times of 0.1s. The high jobs were processed before the low jobs because of the preemption.

4.2 Consistency Test

The consistency test is used to verify whether the system reacts consistently with the input or not.
Two different configurations were tested, both with exponential distribution of interarrival times and service times.

Test 1

- $\frac{1}{\lambda_L} = 0.2s$, $\frac{1}{\lambda_H} = 1s$, $\frac{1}{\mu_L} = 0.1s$, $\frac{1}{\mu_H} = 0.1s$

Test 2

- $\frac{1}{\lambda_L} = 0.2s$, $\frac{1}{\lambda_H} = 2s$, $\frac{1}{\mu_L} = 0.1s$, $\frac{1}{\mu_H} = 0.1s$

Both configurations were tested for **30 repetitions** and their comparison is shown in the figure 4.1.

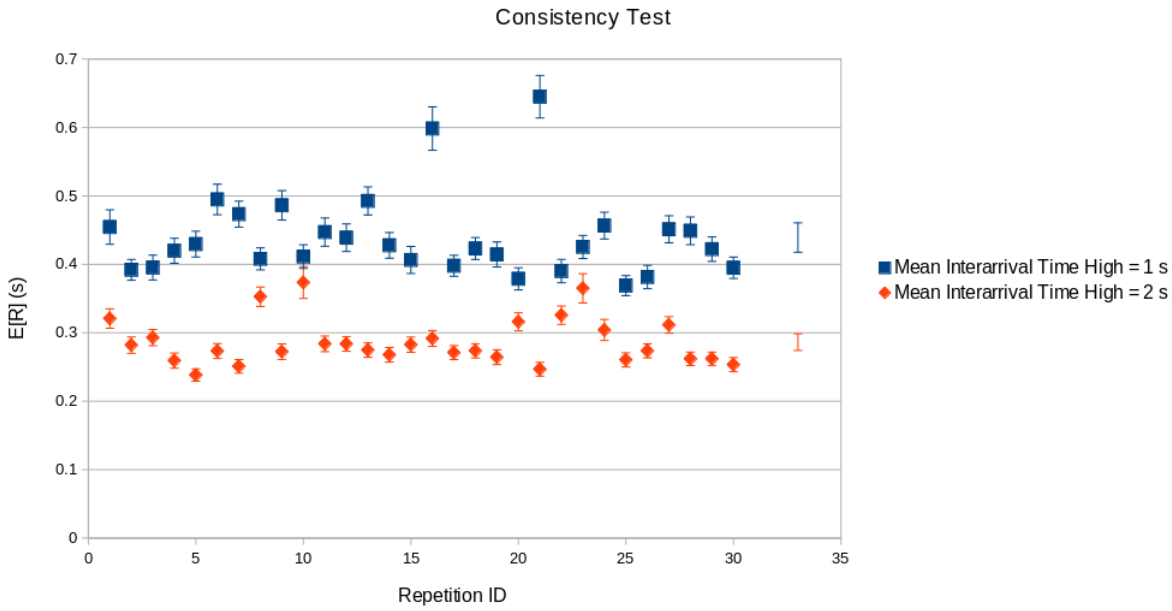

Figure 4.1: Results obtained from the consistency test

As we can see from the figure 4.1 above, the system behaves as expected in computing the value of response time. Indeed, its value is lower in the cases in which the mean interarrival time is higher.

4.3 Continuity Test

The aim of this test is to verify whether the output of the system changes slightly with small changes in the input.

Two different configurations were tested, both with exponential distribution of interarrival times and service times.
Both configurations were tested for **30 repetitions**.

Test 1

- $\frac{1}{\lambda_L} = 0.2s$, $\frac{1}{\lambda_H} = 1s$, $\frac{1}{\mu_L} = 0.1s$, $\frac{1}{\mu_H} = 0.1s$

Test 2

- $\frac{1}{\lambda_L} = 0.2s$, $\frac{1}{\lambda_H} = 1s$, $\frac{1}{\mu_L} = 0.11s$, $\frac{1}{\mu_H} = 0.1s$

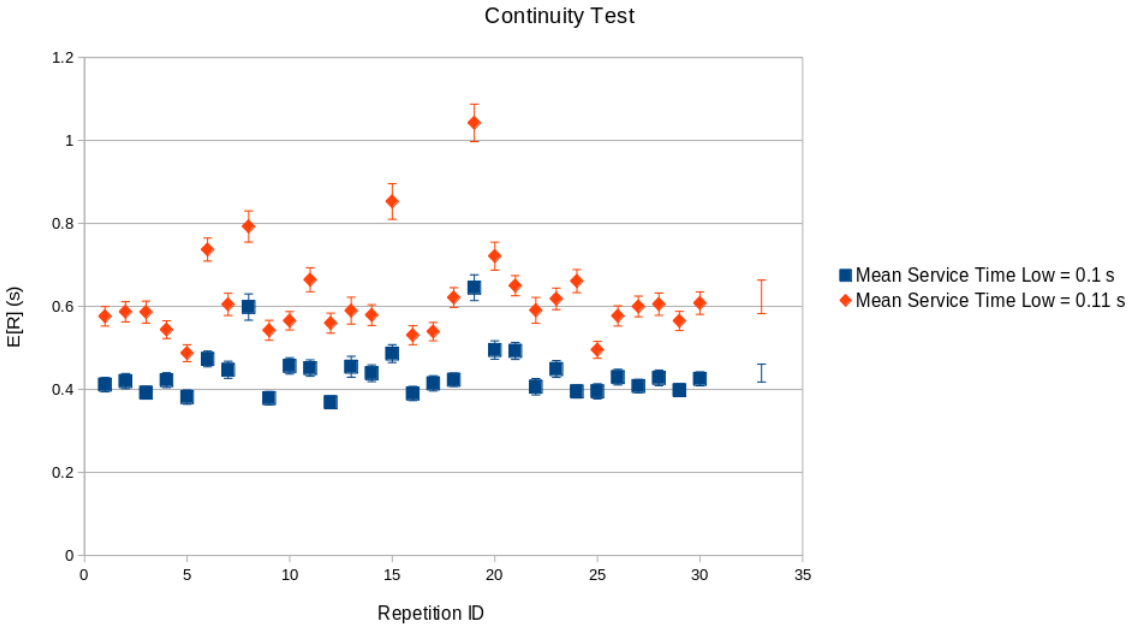

Figure 4.2: Results obtained from the continuity test

By looking at the results in the figure 4.2 we can see that slight changes in the input do in fact correspond to slight changes in the output.

4.4 Monotonicity Test

This test consists in assessing the monotonicity of some of the performance indexes by using different combinations of factors. The configuration used was the following (with exponential distribution of mean interarrival time and mean service time of the jobs):

- $\frac{1}{\lambda_L} = 0.1s$, $\frac{1}{\lambda_H} = 5s$, $\frac{1}{\mu_L} = (0.1s, 0.11s, 0.12s, 0.13s, 0.14s)$, $\frac{1}{\mu_H} = 0.1s$

Each configuration was tested for **30 repetitions**. The analysed KPIs were the following:

Mean Response time

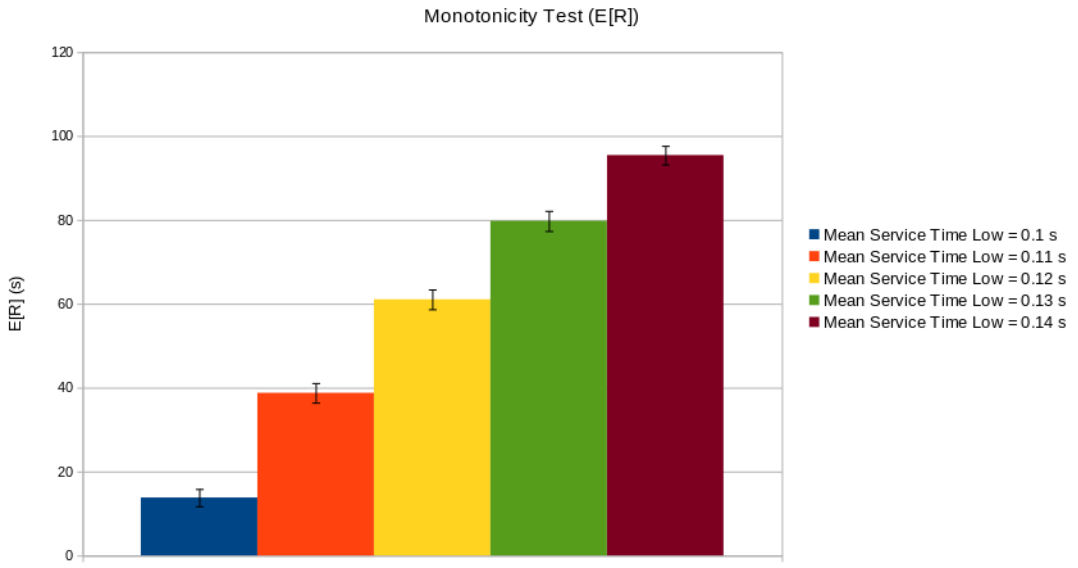

Figure 4.3: Results obtained from the monotonicity test (E[R])

The system was assessed using varying mean service times for low-priority jobs. As depicted in figure 4.3, it is evident that the response time consistently increased with higher mean service times.

Mean Number of low jobs in the system

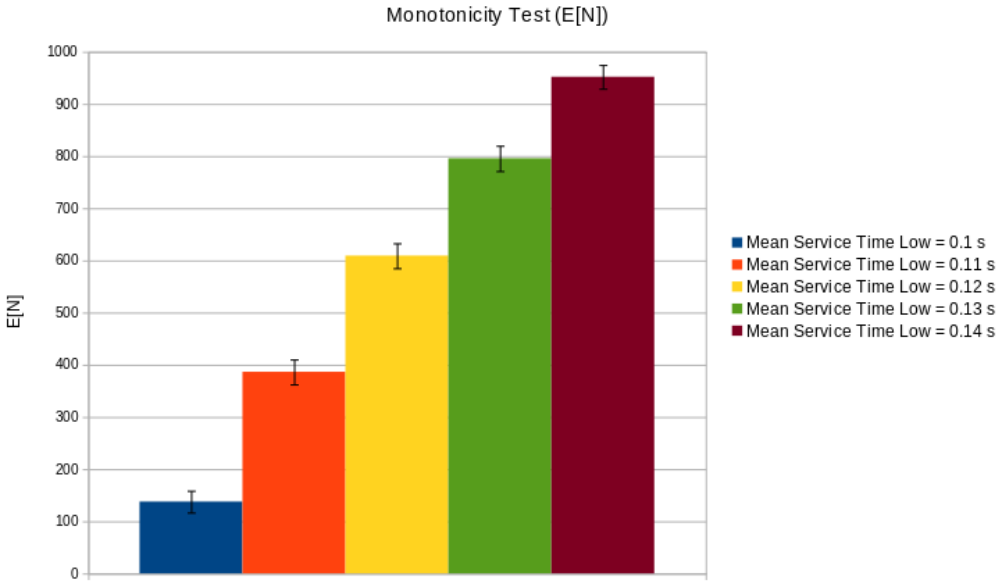

Figure 4.4: Results obtained from the monotonicity test (E[N])

Also in this case, as the mean service time of low-priority jobs increases, their mean number also rises, as evident from the figure 4.4.

4.5 Verification against Theoretical Model

The aim of this test is to compare the results obtained from a simplified version of the system with those calculated using a theoretical model of it.
The system has been simplified, considering only one of the 2 queues (the low priority one). This made it possible to study its behaviour by modelling it as an **M/M/1** system.
The configuration used for this test was the following (**30 repetitions** with exponential distribution of mean interarrival time and mean service time of the jobs):

- $\mathbf{\frac{1}{\lambda_L} = 0.5s}$, $\mathbf{\frac{1}{\lambda_H} = 10000s}$, $\mathbf{\frac{1}{\mu_L} = 0.1s}$, $\mathbf{\frac{1}{\mu_H} = 0.1s}$

The $\mathbf{\frac{1}{\lambda_H}}$ parameter was chosen in order to avoid the arrival of high priority jobs during the test. To do that, its value was set higher than the total duration of the simulation (600s in this case). We studied the system by considering the subsequent 2 KPIs:

Mean Number of jobs in the system

The value computed theoretically with the formulas was:

- $E[N] = \rho/(1-\rho) = 0.25$

The value obtained from the test was:

- $E[N] = 0.251984$

As we can see from the computation and from the figure 4.5, the E[N] value acquired corresponds with the one calculated theoretically.

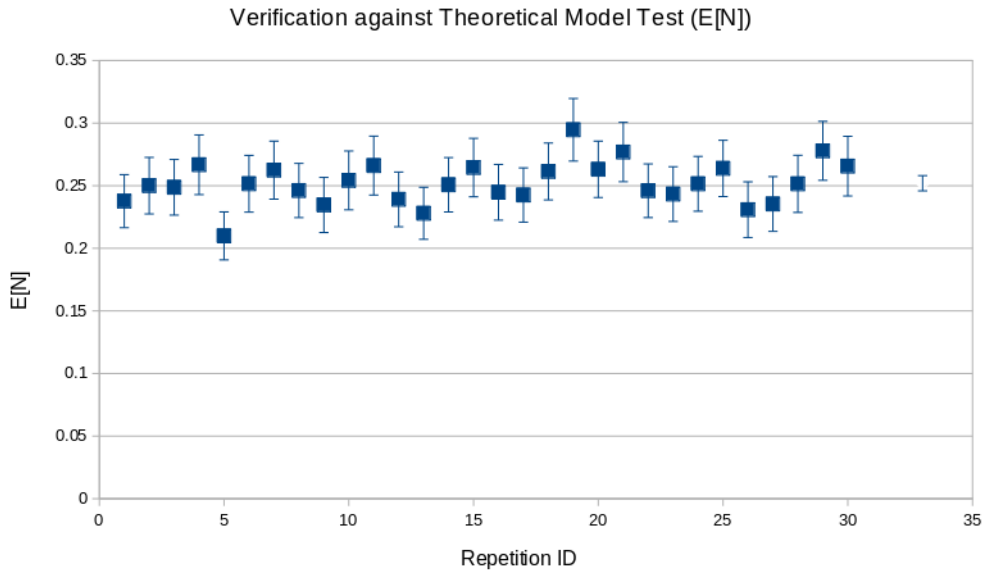

Figure 4.5: Results obtained from the theoretical test (E[N])

Mean Response Time

The value computed theoretically with the formulas was:

- $E[R] = E[N]/\lambda = 0.125$

The value obtained from the test was:

- $E[R] = 0.125205$

Also in this case, the values of E[R] correspond. This can be seen from the figure 4.6.

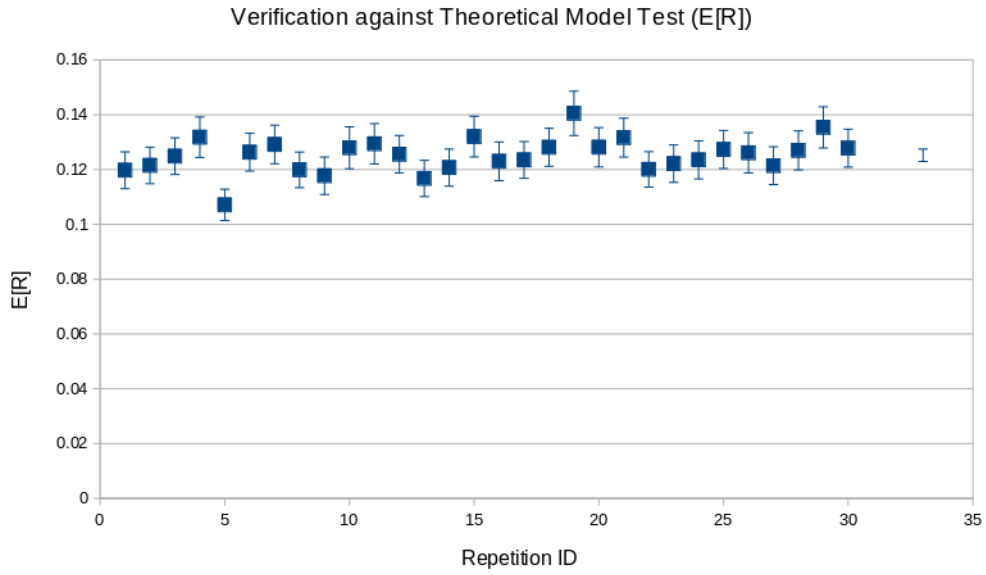

Figure 4.6: Results obtained from the theoretical test (E[R])

5 - Calibration

5.1 Factors Calibration

The aim of calibration is to find the best factor intervals to simulate a realistic situation.
Before performing the study for this phase, the following assumption was made: since we are considering a scheduler of jobs, in most real-life cases, the service time values of these jobs vary in a range from a few milliseconds to a few seconds. For this reason, we assumed constant mean service time for both kind of Jobs and evaluated the system with several values of mean interarrival time for Low Jobs and 2 values of mean interarrival time for High Jobs, to better understand the ranges to use in different load scenarios.
The system was tested for **30 repetitions** using the following configurations (with exponential distribution of mean interarrival time and constant mean service time of the jobs):

- $\frac{1}{\lambda_L} = (\mathbf{0.12s}, \mathbf{0.15s}, \mathbf{0.19s}, \mathbf{0.24s}, \mathbf{0.31s}, \mathbf{0.39s}, \mathbf{0.5s}, \mathbf{0.62s}, \mathbf{0.79s}, \mathbf{1s})$,
  $\frac{1}{\lambda_H} = \mathbf{0.2s}$, $\frac{1}{\mu_L} = \mathbf{0.1s}$, $\frac{1}{\mu_H} = \mathbf{0.1s}$

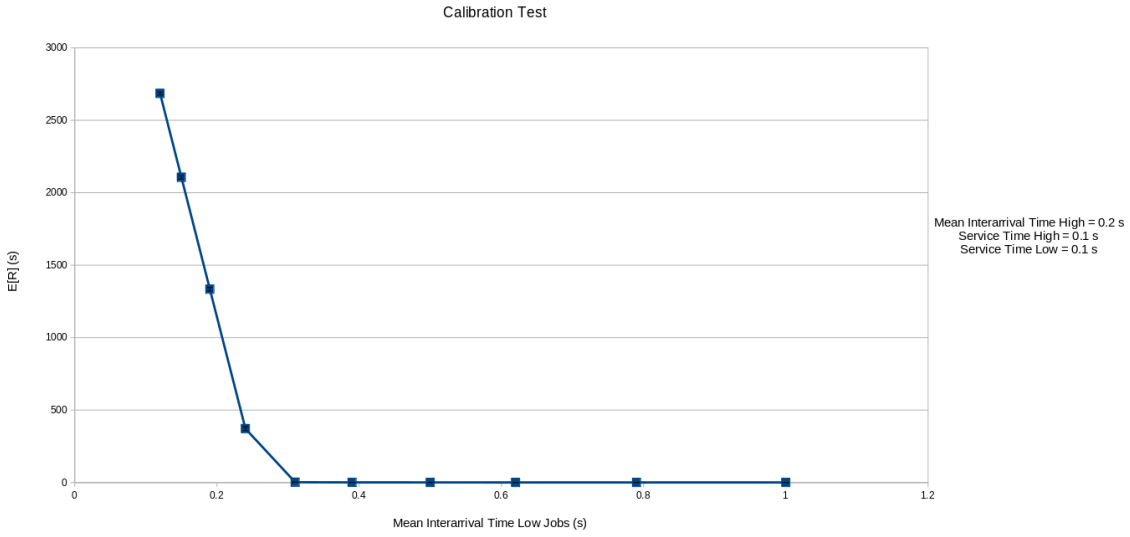

Figure 5.1: Calibration with Mean Interarrival Time of High Jobs = 0.2s

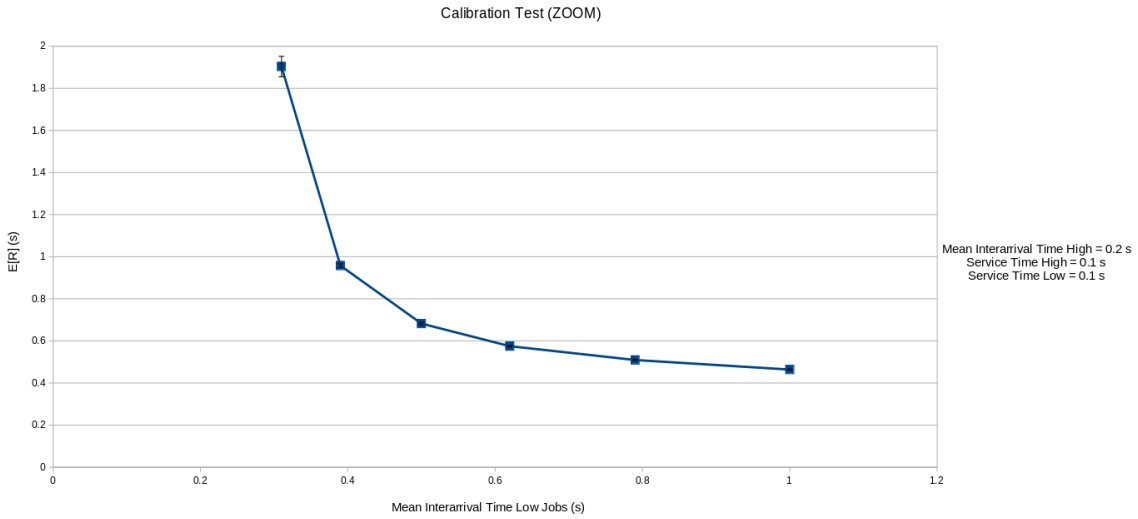

Figure 5.2: Calibration with Mean Interarrival Time of High Jobs = 0.2s (ZOOM)

- $\frac{1}{\lambda_L} = (\mathbf{0.12s}, \mathbf{0.15s}, \mathbf{0.19s}, \mathbf{0.24s}, \mathbf{0.31s}, \mathbf{0.39s}, \mathbf{0.5s}, \mathbf{0.62s}, \mathbf{0.79s}, \mathbf{1s})$, $\frac{1}{\lambda_H} = \mathbf{1s}$, $\frac{1}{\mu_L} = \mathbf{0.1s}$, $\frac{1}{\mu_H} = \mathbf{0.1s}$

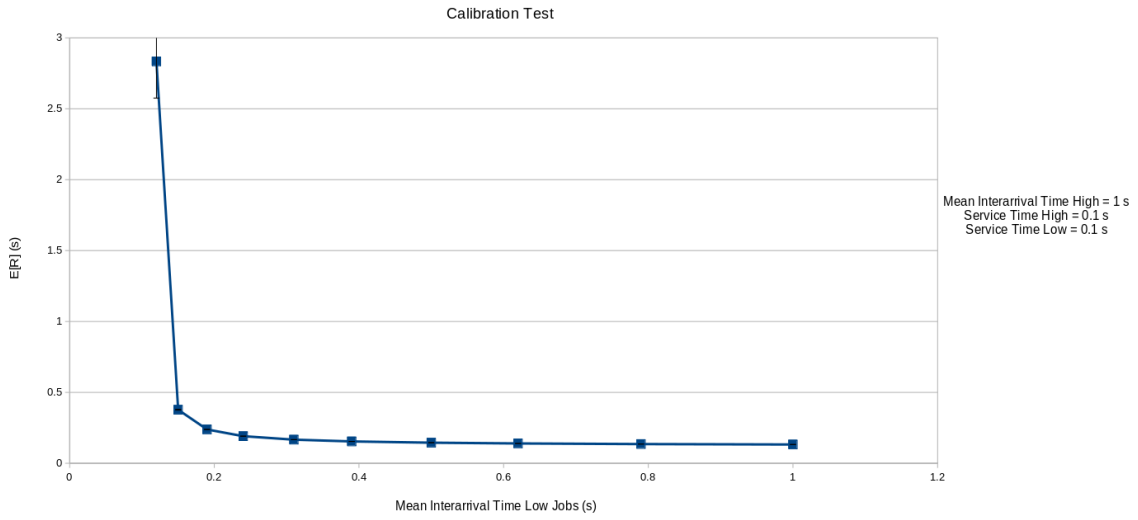

Figure 5.3: Calibration with Mean Interarrival Time of High Jobs = 1s

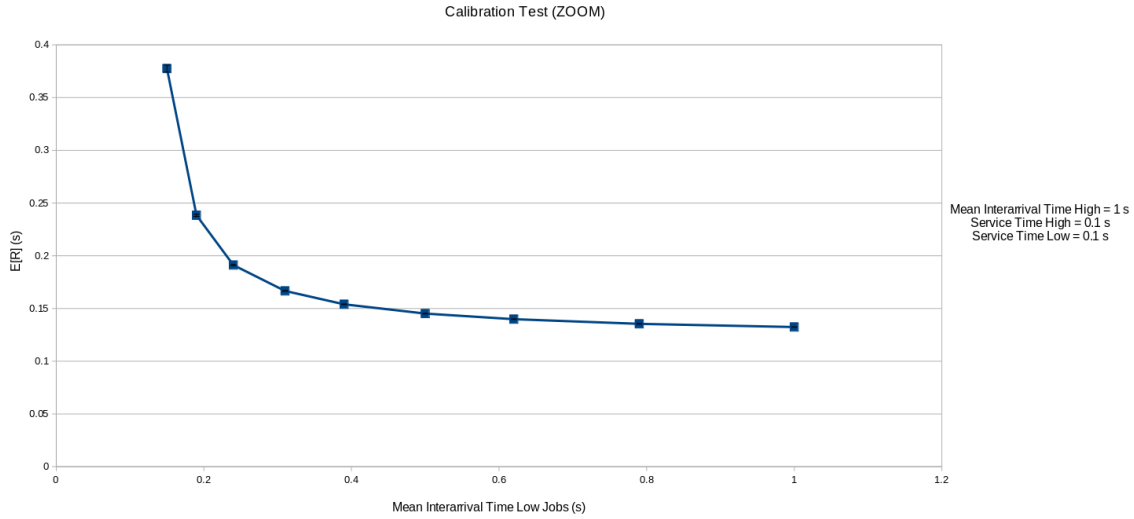

Figure 5.4: Calibration with Mean Interarrival Time of High Jobs = 1s (ZOOM)

Considering the results obtained through factors calibration, it was possible to identify three different possible load scenarios for our system, which were used to conduct the experiment phase.
In particular, the ranges of mean interarrival time of Low Jobs that were defined are the following:

- **High Load Scenario: [0.12s, 0.31s]**
- **Medium Load Scenario: [0.39s, 0.5s]**
- **Low Load Scenario: [0.79s, 1s]**

5.2 Warm Up Time Calibration

To correctly calibrate the warm-up time, several simulations were conducted. The KPI considered was the Response Time and the configuration used was (**10 repetitions** with exponential distribution of mean interarrival time and constant mean service time of the jobs):

- $\mathbf{\frac{1}{\lambda_L} = 0.12s}$, $\mathbf{\frac{1}{\lambda_H} = 0.2s}$, $\mathbf{\frac{1}{\mu_L} = 0.05s}$, $\mathbf{\frac{1}{\mu_H} = 0.05s}$

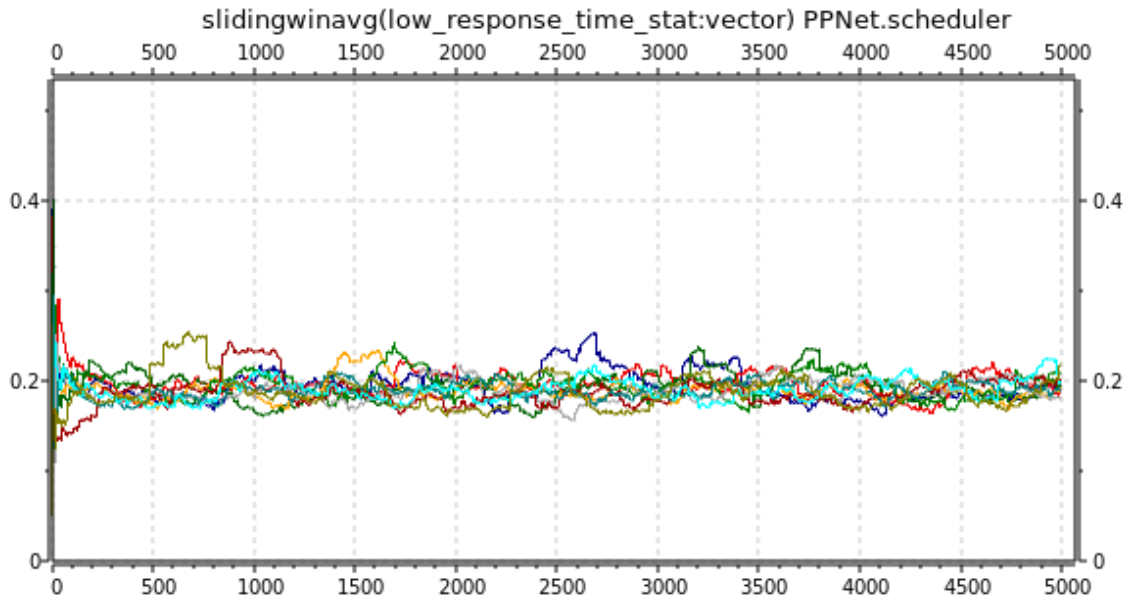

Figure 5.5: Sliding average of Response Time

As we can see from the image above, the Mean Response Time becomes more stable after **3000s**, and, although it is never completely stable, it remains within a well-defined range. For this reason we choose it as warm-up time value for the experiments conducted. In order to collect enough data, we decided to set the **simulation time limit** to **10000s**.

6 - Simulation Experiments

6.1 $2^k r$ Factorial Analysis

To evaluate the contribution of factors (defined in section 2.3) on the Response Time, a $2^k r$ factorial analysis has been performed. In particular, based on the Factors Calibration's results, we identified 3 different scenarios (**High Load, Medium Load, Low Load**).
For the experiments, we considered 5 repetitions ($\mathbf{r = 5}$) and the following 4 factors ($\mathbf{k = 4}$):

- $\frac{1}{\lambda_L}$: **Mean Interarrival Time Low**
- $\frac{1}{\lambda_H}$: **Mean Interarrival Time High**
- $\frac{1}{\mu_L}$: **Mean Service Time Low**
- $\frac{1}{\mu_H}$: **Mean Service Time High**

Moreover, for each configuration used, we set the **interarrival time** distribution as **exponential** and the **service time** distribution as **constant**. For the effects obtained through the factorial analysis, we considered a **confidence level of 95%**.

6.1.1 High Load Scenario

- **Mean Interarrival Time Low [0.12, 0.31]**
- **Mean Interarrival Time High [0.24, 0.31]**
- **Mean Service Time Low [0.15, 0.2]**
- **Mean Service Time High [0.15, 0.2]**

We assessed the **normality of the residuals** and the **homoskedasticity**.
In this case, residuals followed a normal distribution, as can be seen in the QQ plot in figure 6.1.

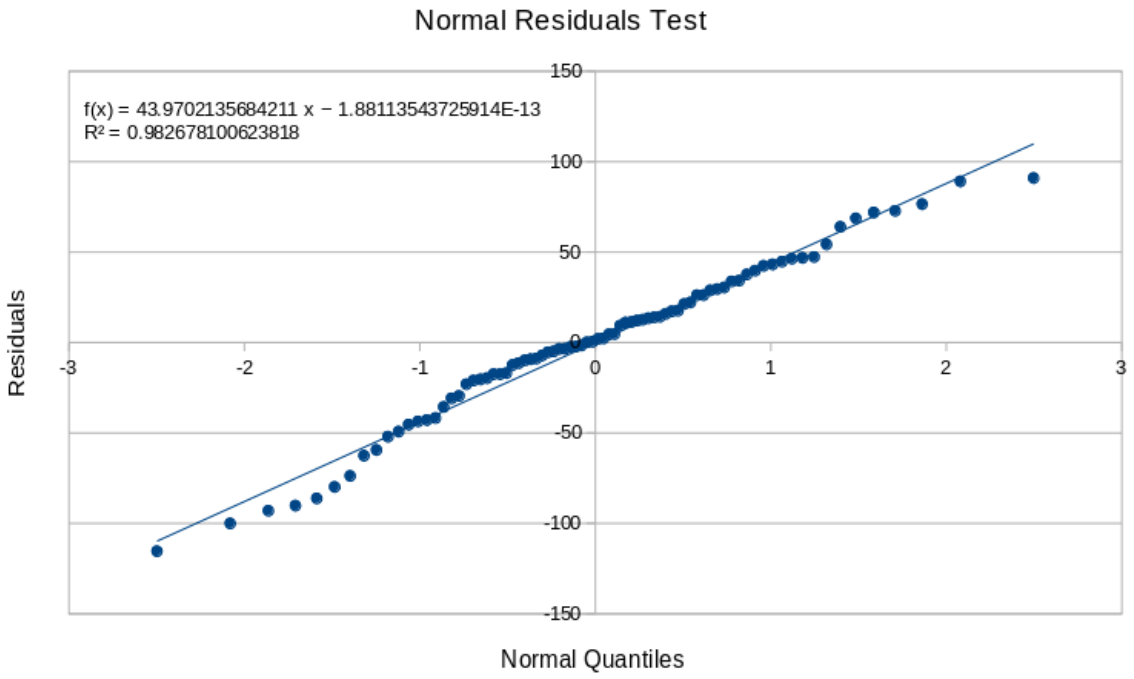

Figure 6.1: High Load QQ Plot for testing the normal hypothesis

In the plot of **residuals vs predicted response** (figure 6.2) there was a trend, but, since the errors were 2 or 3 order of magnitude below the predicted response, we can ignore it, so the homoskedasticity hypothesis was respected. However, since the CIs include zero, the results obtained are not significant at the specified level of confidence.

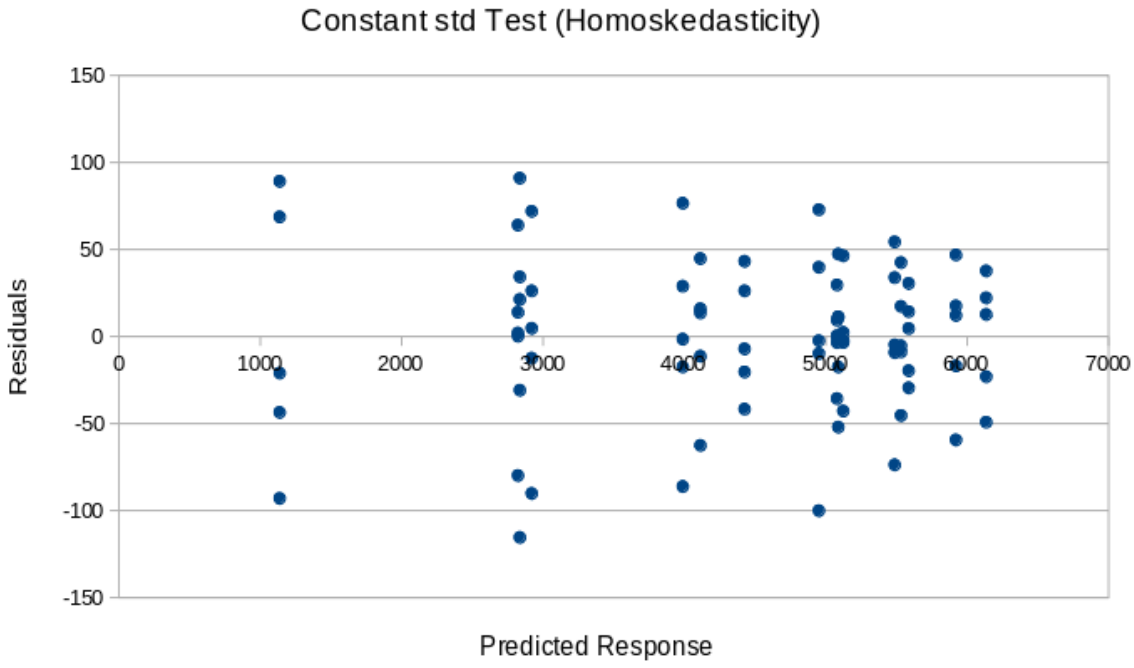

Figure 6.2: High Load Homoskedasticity test

We performed the factorial analysis again using a **logarithmic transformation** of the data, but also in this case, results were not usable.

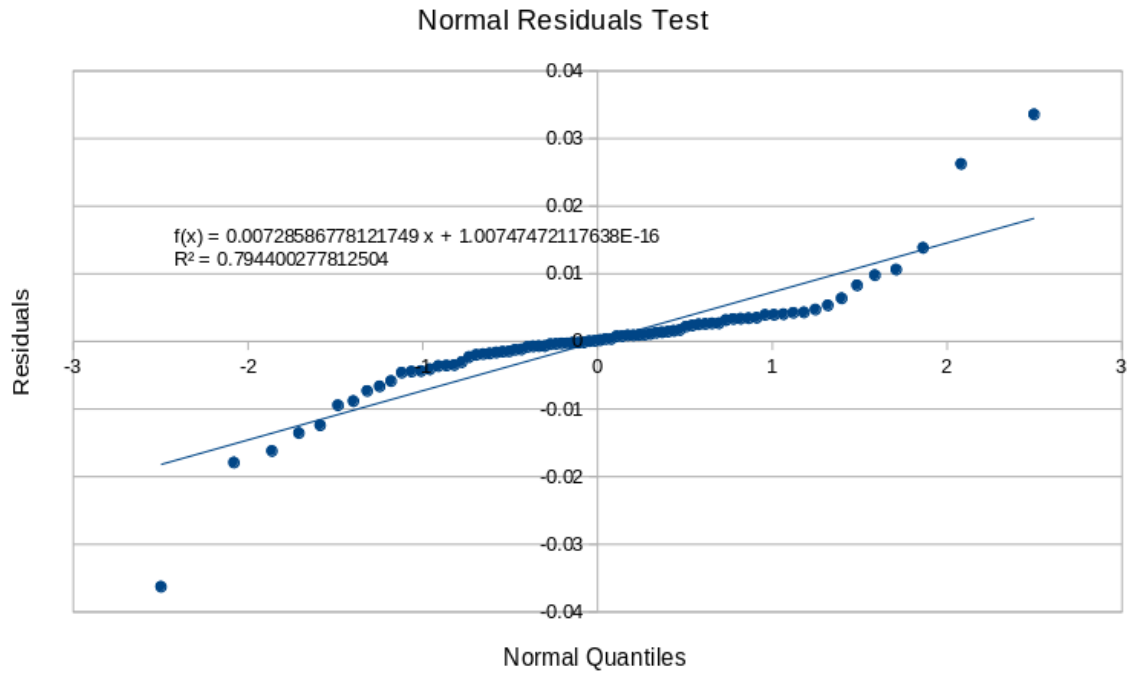

Figure 6.3: High Load QQ Plot for testing the normal hypothesis with a logarithmic transformation of the data

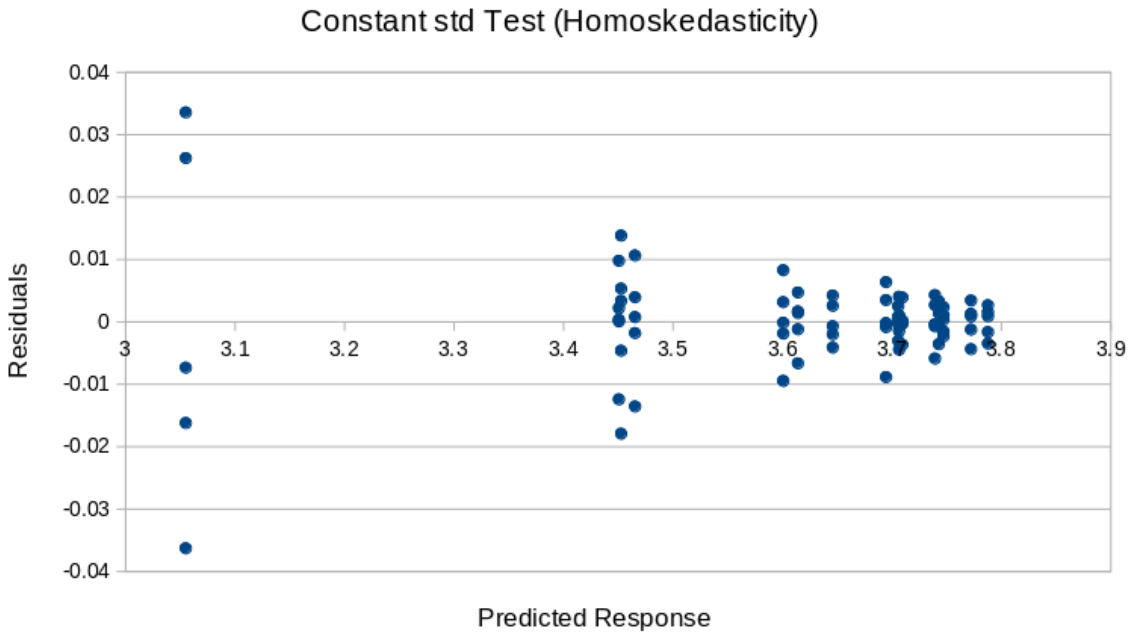

Figure 6.4: High Load Homoskedasticity test with a logarithmic transformation of the data

6.1.2 Medium Load Scenario

- **Mean Interarrival Time Low [0.39, 0.5]**
- **Mean Interarrival Time High [0.62, 0.79]**
- **Mean Service Time Low [0.1, 0.2]**
- **Mean Service Time High [0.1, 0.2]**

In this scenario the normal hypothesis was not respected because, as we can see from the QQ plot in figure 6.5, the residuals didn't have a linear trend, but followed an heavy-tailed distribution.

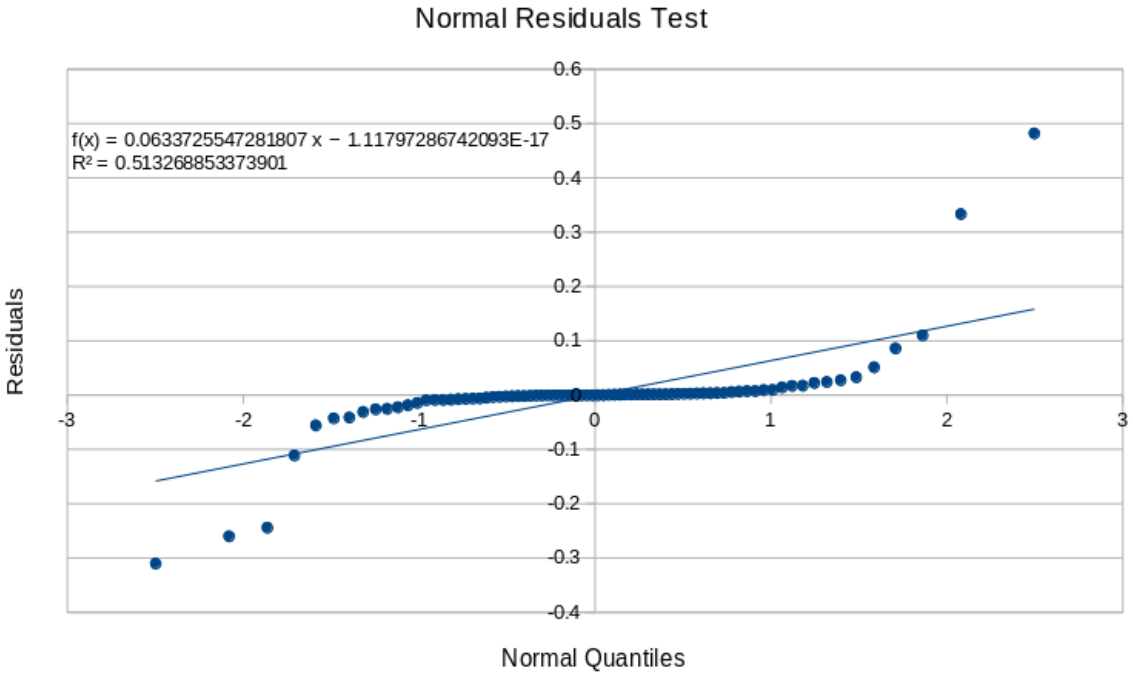

Figure 6.5: Medium Load QQ Plot for testing the normal hypothesis

Moreover, the results obtained from the Homoskedasticity test shows a trend, as can be seen from the figure 6.6

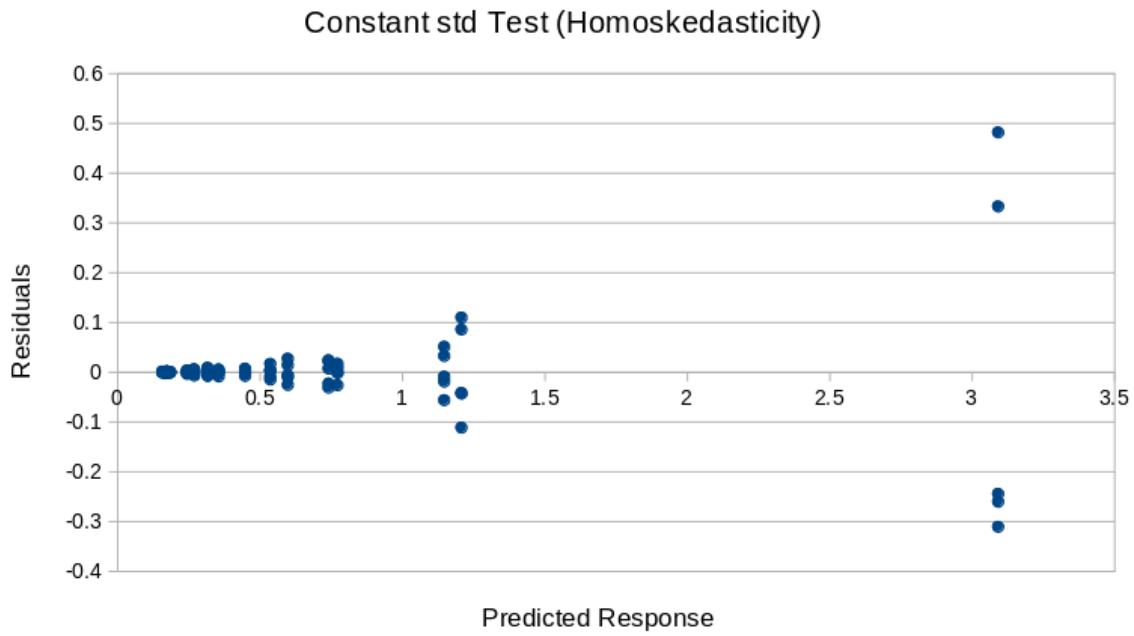

Figure 6.6: Medium Load Homoskedasticity test

Also after a logarithmic transformation of the data, the obtained values were not considerable for the analysis.

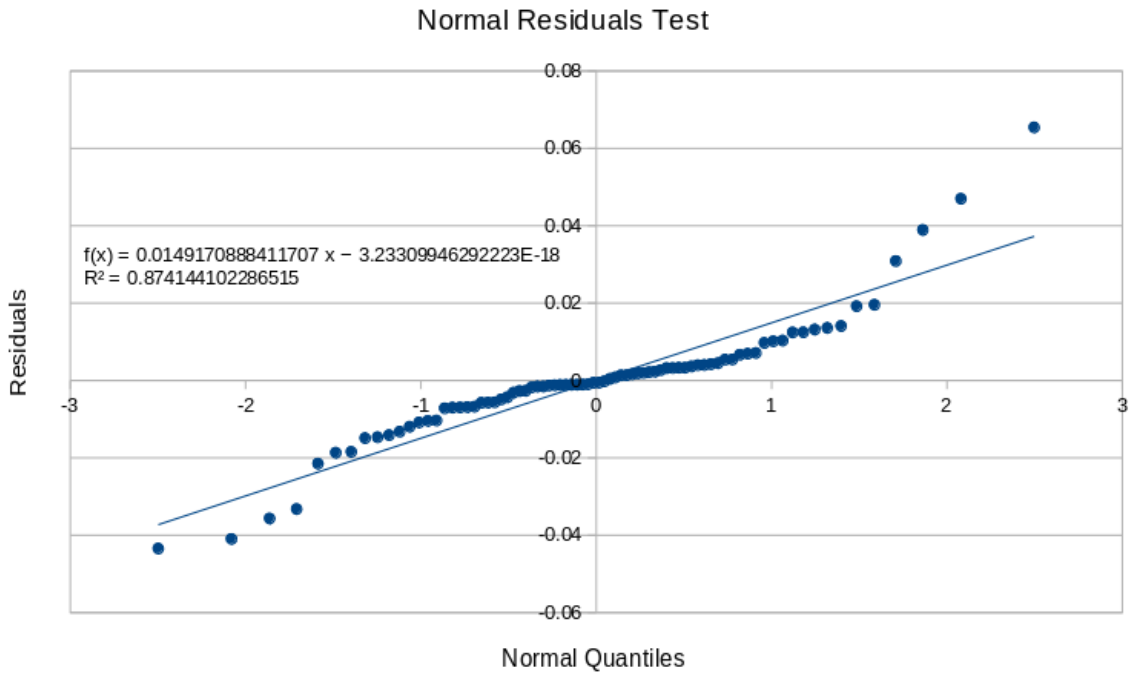

Figure 6.7: Medium Load QQ Plot for testing the normal hypothesis with a logarithmic transformation of the data

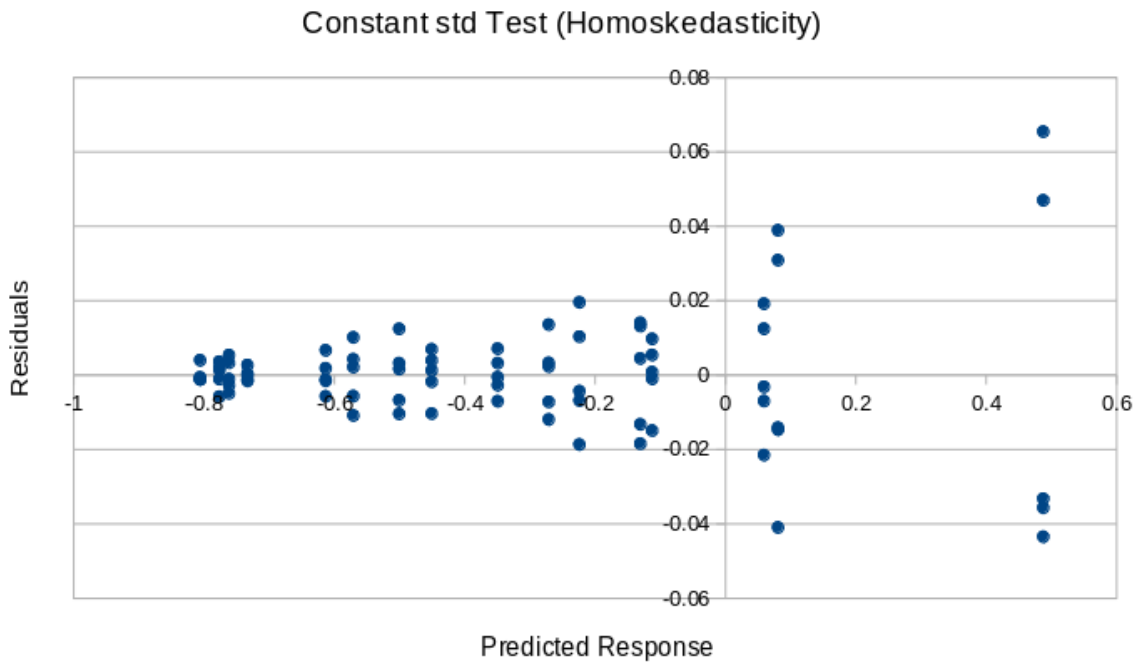

Figure 6.8: Medium Load Homoskedasticity test with a logarithmic transformation of the data

6.1.3 Low Load Scenario

- **Mean Interarrival Time Low [0.79, 1]**
- **Mean Interarrival Time High [0.79, 1]**
- **Mean Service Time Low [0.05, 0.2]**
- **Mean Service Time High [0.05, 0.2]**

In the first factorial analysis we conducted, the residuals were not normal, since their QQ plot (figure 6.9) fitted an heavy-tailed distribution, so the obtained data was not usable in this case.

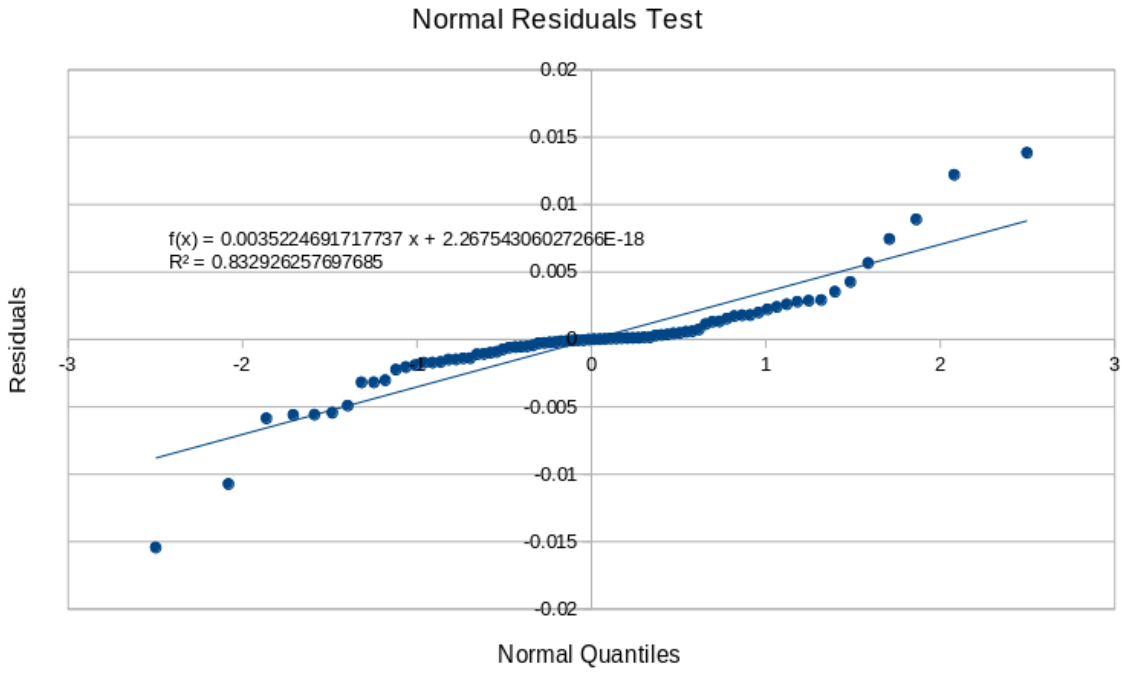

Figure 6.9: Low Load QQ Plot for testing the normal hypothesis

In addition to that, the residuals vs predicted response plot showed a trend.

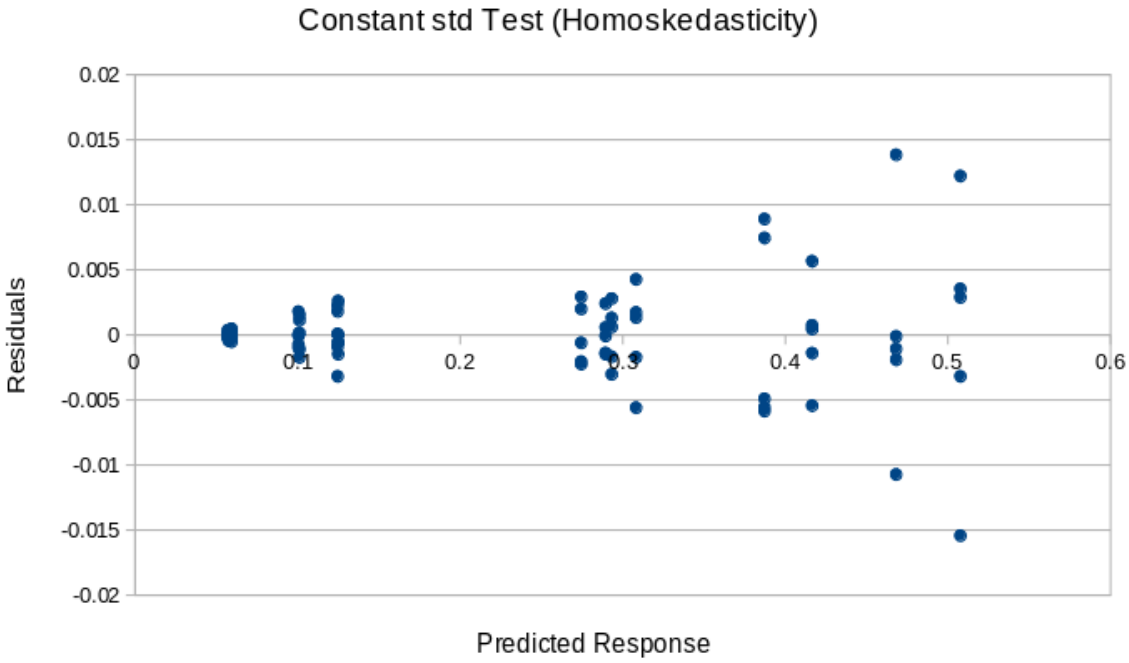

Figure 6.10: Low Load Homoskedasticity test

After a logarithmic transformation of the data, we obtained useful results. In this case, the normal hypothesis for residuals was respected since they followed a linear trend, as shown in figure 6.11.

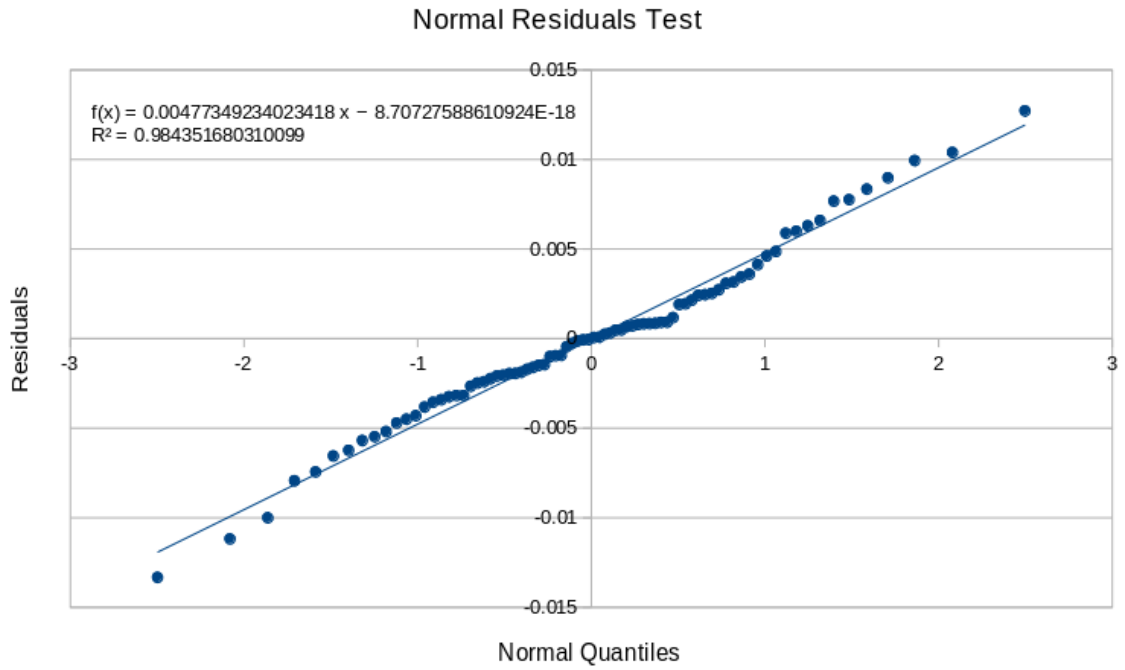

Figure 6.11: Low Load QQ Plot for testing the normal hypothesis with a logarithmic transformation of the data

The homoskedasticity hypothesis was also respected, because, although the plot of residuals vs predicted response had a trend (figure 6.12), the errors were 2 or 3 order of magnitude below the predicted response, so it was negligible.
Finally, the CIs did not include zero, so the data produced are significant.

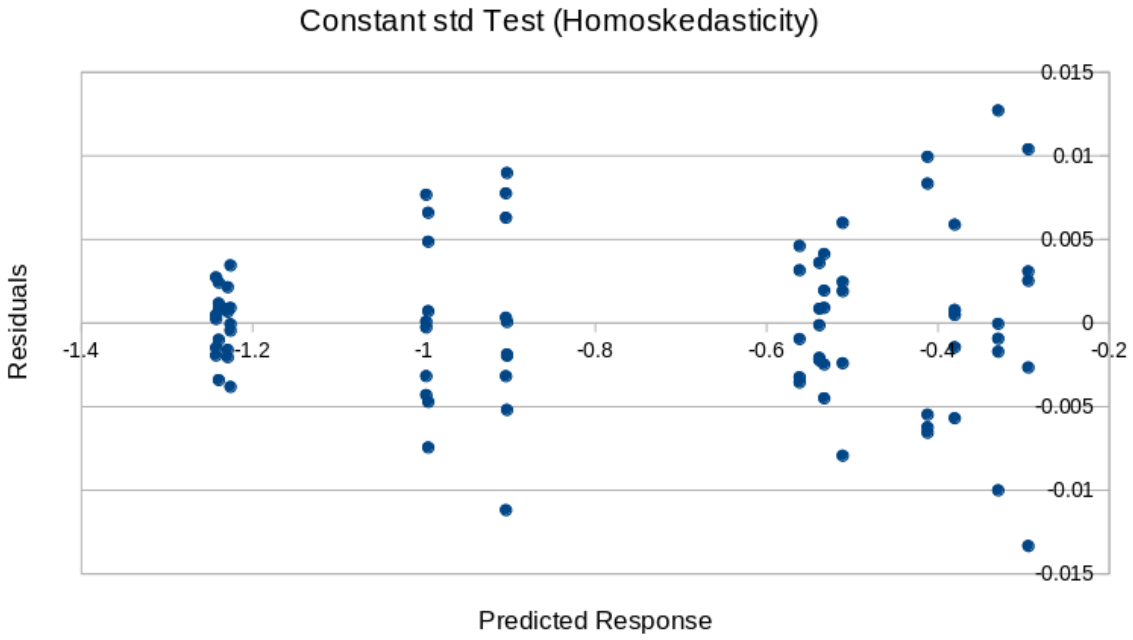

Figure 6.12: Low Load Homoskedasticity test with a logarithmic transformation of the data

The following table contains the data relative to the impact of all the possible combinations of the 4 considered factors:

	Response Time	
Factors	qi	Impact on Variability (%)
A	-0,0075978	0,0480%
B	-0,0272866	0,6201%
C	0,32351364	87,1719%
D	0,11654077	11,3122%
AB	5,824418 x 10^{-5}	2,825508 x 10^{-6} %
AC	-0,00635298	0,0336%
AD	-0,00121611	0,0012%
BC	-0,00075871	0,0004%
BD	-0,01688182	0,2373%
CD	-0,02562770	0,5470%
ABC	0,00018830	2,9533 x 10^{-5}%
ABD	0,00022772	4,3192 x 10^{-5}%
ACD	-0,00156275	0,0020%
BCD	0,00283528	0,0066%
ABCD	0,00038635	0,0001%

Table 6.1: Results of $2^k r$ analysis on the Response Time

A = Mean Interarrival Time Low, B = Mean Interarrival Time High, C = Mean Service Time Low, D = Mean Service Time High

From the table 6.1 we can see that the most significant factor for the Response time of Low Jobs is the **Mean Service Time of Low Jobs**, that accounts for the **87.17%** of the variation. This was expected because in this kind of scenario the queues are almost always empty, so the response time is typically equal to the service time of the jobs. The second factor in terms of variation is the Mean Service Time of the High Jobs, that accounts for the 11.31%. The effects of the other factors are negligible.

6.2 Response Time Experiments

In these experiments we studied the Response Time of the system by using **8** different values for the **Mean Service Time** of **Low Jobs** and **4** for the one of **High Jobs**. Moreover, we used a **CI of 95%** for the results obtained.
The Mean Interarrival Time of the High Jobs was set **higher** than for Low Jobs. This was done to allow the Response Time of Low Jobs to be studied.

The experiments were performed with **3** different **load levels** (**low, medium and high**). Each one was analysed with either a **constant** or **exponential interarrival time distribution**, while the **Mean Service Time** was considered **constant** in both cases. In the configurations with an exponential interarrival time, **35 repetitions** were performed.
Before analyzing the various scenarios, we can state that in cases where the Mean Interarrival Time was considered to be exponential, the response time of low jobs was greater than in cases with constant distribution. This behaviour is probably due to the unpredictability of the frequency of arrival of jobs. Indeed, bursty arrivals of jobs can lead to the congestion of the queue, delaying the processing of low priority jobs.

6.2.1 Low load Scenario

- $\mathbf{\frac{1}{\lambda_L} = 1}$, $\mathbf{\frac{1}{\lambda_H} = 1.6s}$,
  $\mathbf{\frac{1}{\mu_L}} = (\mathbf{0.05s, 0.15s, 0.25s, 0.35s, 0.45s, 0.55s, 0.65s, 0.75s})$,
  $\mathbf{\frac{1}{\mu_H}} = (\mathbf{0.15s, 0.35s, 0.55s, 0.75s})$

It can be seen from the following plots that, in the case in which the Interarrival Time is constant, the Response Time diverges to very high values earlier than when it is considered exponential.
This is because, when the Interarrival time is constant and Service Time values begin to increase, we have a situation where the low jobs queue fills up and no more jobs within it are processed. This happens sooner than in the exponential case.
In the exponential scenario (figure 6.13), unlike the constant case (figure 6.14), when the number of low jobs in the queue is very high, the scheduler is still able to process some of them, since they may have high arrival time values. For this reason, additional values of Response Time can be measured in this case, unlike in the constant one.

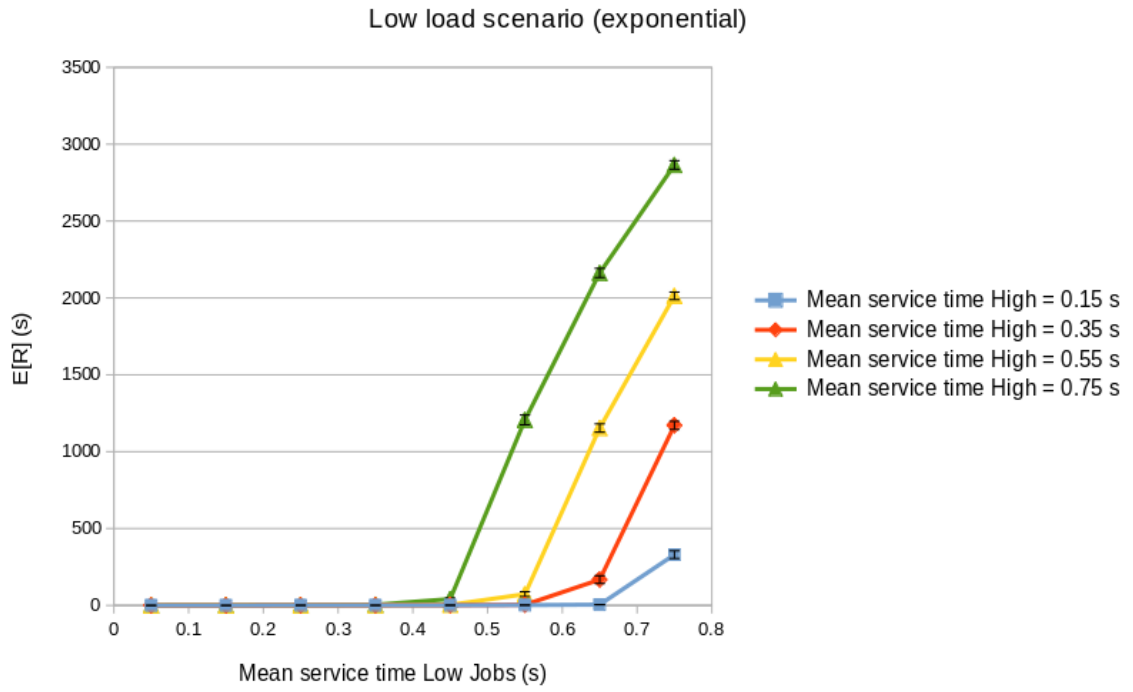

Figure 6.13: Mean Response Time Low Load Scenario (exponential)

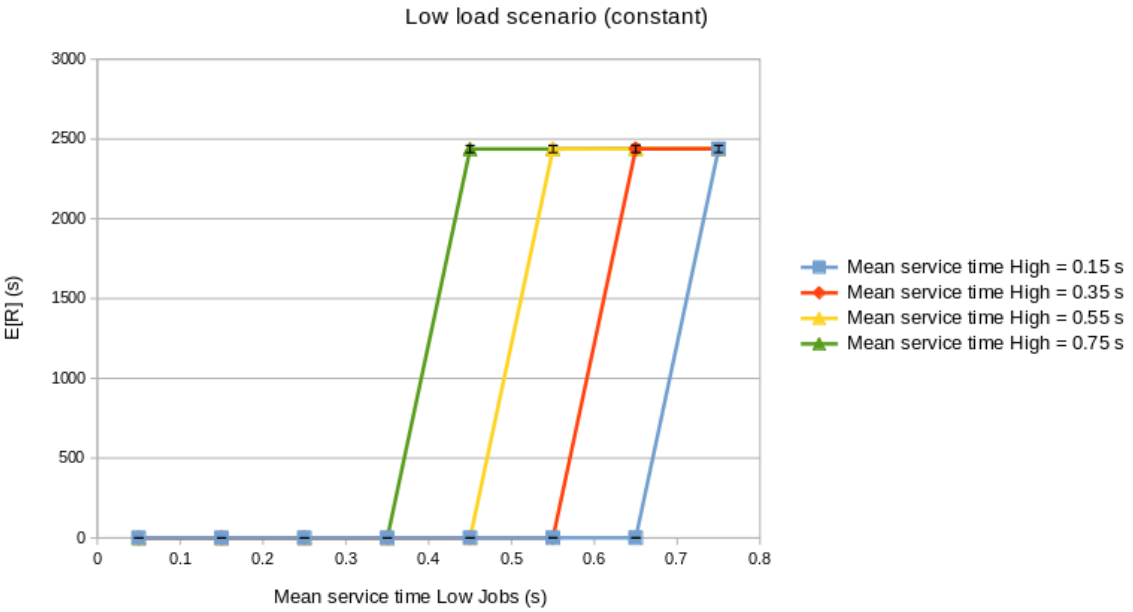

Figure 6.14: Mean Response Time Low Load Scenario (constant)

From the following figures it can be seen that, in general, the response time of the exponential case (figure 6.15) reaches higher values than the one of the constant case (figure 6.16).

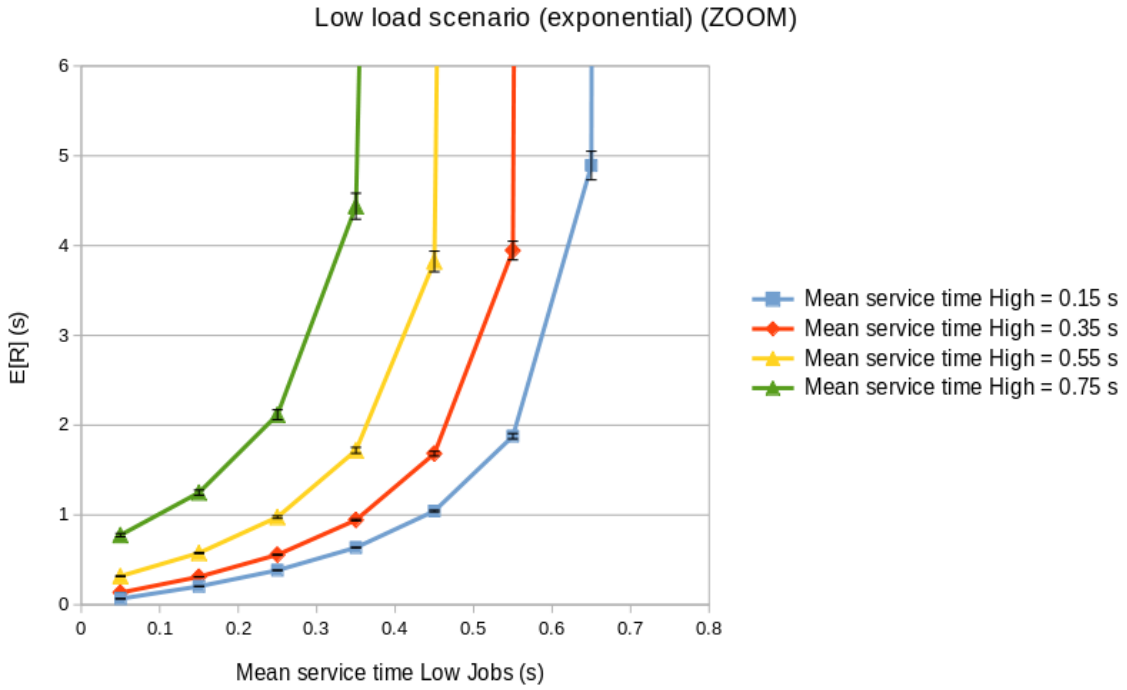

Figure 6.15: Mean Response Time Low Load Scenario (exponential) (ZOOM)

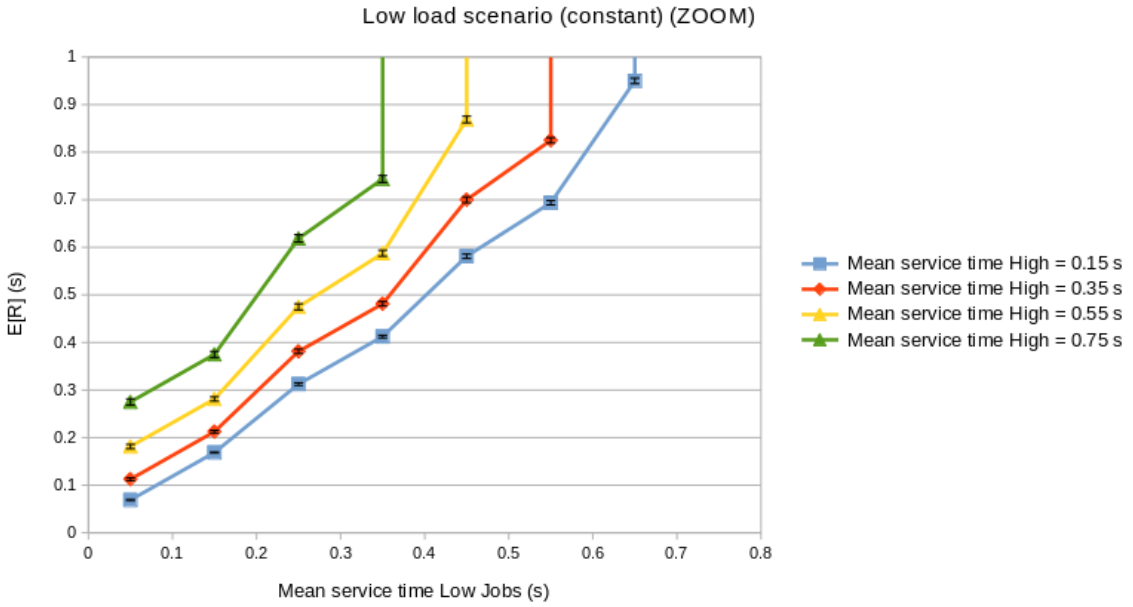

Figure 6.16: Mean Response Time Low Load Scenario (constant) (ZOOM)

6.2.2 Medium load Scenario

- $\frac{1}{\lambda_L} = 0.45s$, $\frac{1}{\lambda_H} = 0.65s$,
  $\frac{1}{\mu_L} = (0.05s, 0.06s, 0.08s, 0.11s, 0.15s, 0.19s, 0.25s, 0.33s)$,

$$\frac{1}{\mu_H} = (\mathbf{0.1s}, \mathbf{0.14s}, \mathbf{0.21s}, \mathbf{0.31s})$$

About the medium load scenario, analysing the figures below, the same conclusions can be drawn as in the previous case.

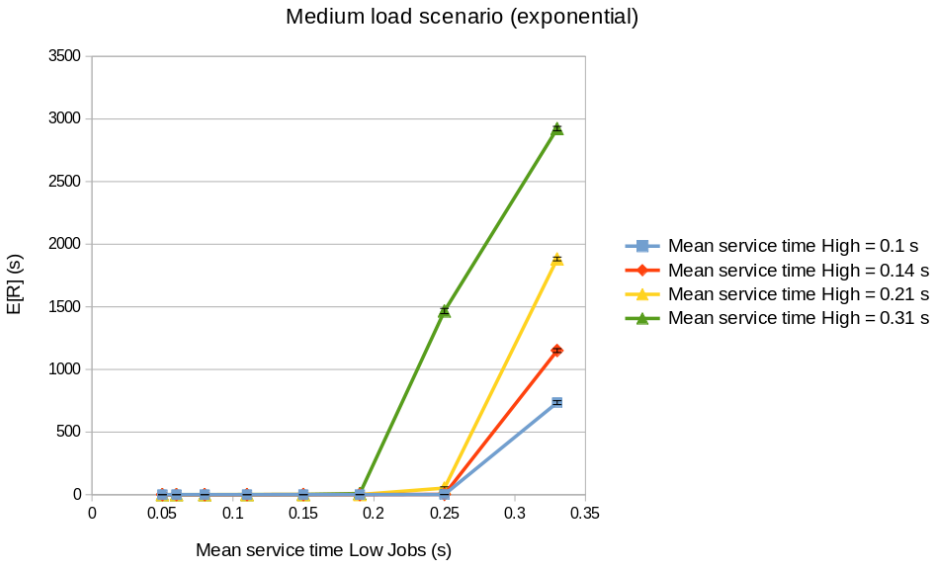

Figure 6.17: Mean Response Time Medium Load Scenario (exponential)

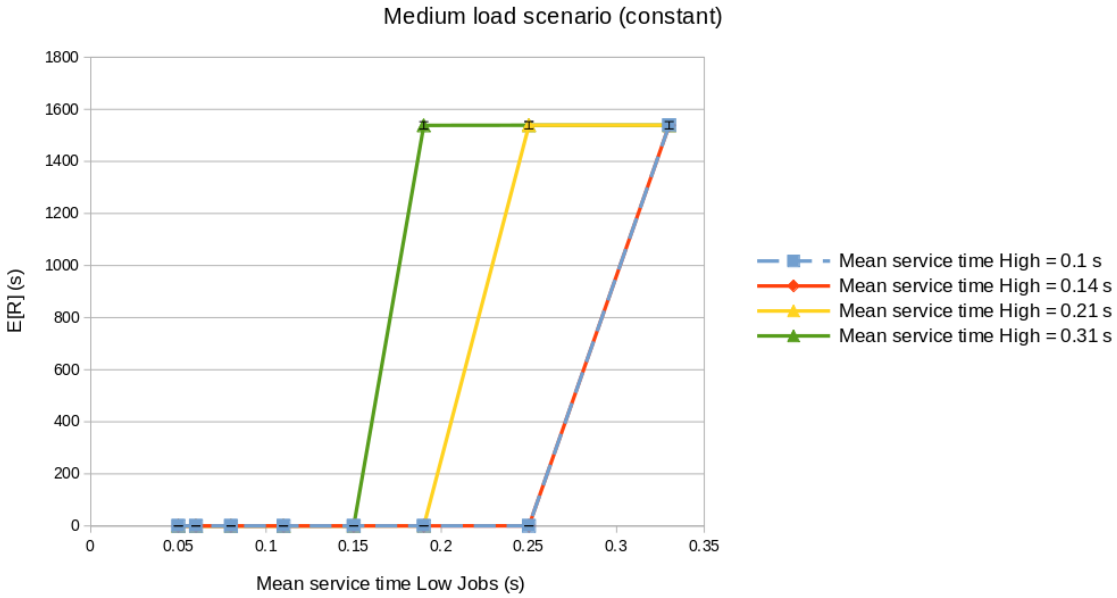

Figure 6.18: Mean Response Time Medium Load Scenario (constant)

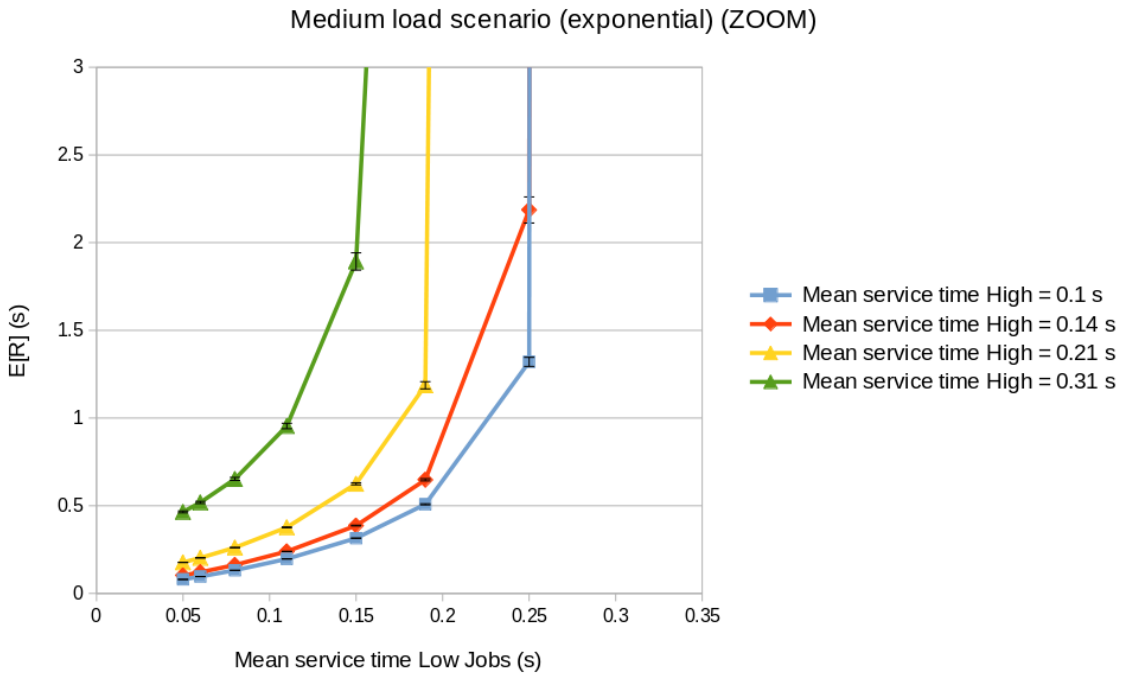

Figure 6.19: Mean Response Time Medium Load Scenario (exponential) (ZOOM)

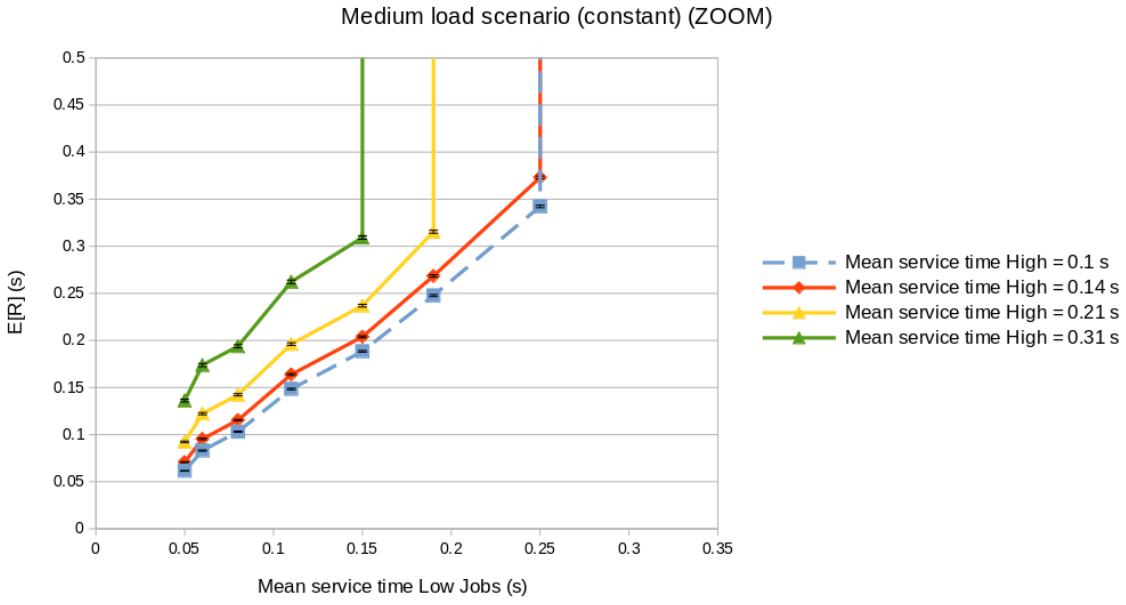

Figure 6.20: Mean Response Time Medium Load Scenario (constant) (ZOOM)

6.2.3 High load Scenario

- $\frac{1}{\lambda_L} = 0.2s$, $\frac{1}{\lambda_H} = 0.4s$,
  $\frac{1}{\mu_L} = (0.05s, 0.06s, 0.07s, 0.09s, 0.11s, 0.13s, 0.16s, 0.2s)$,
  $\frac{1}{\mu_H} = (0.08s, 0.1s, 0.13s, 0.16s)$

In this scenario, if we consider the case with the exponential Interarrival Time of jobs (figure 6.21), the Response Time tends to diverge to very large values for smaller values of service time of low jobs with respect to the constant case (figure 6.22).

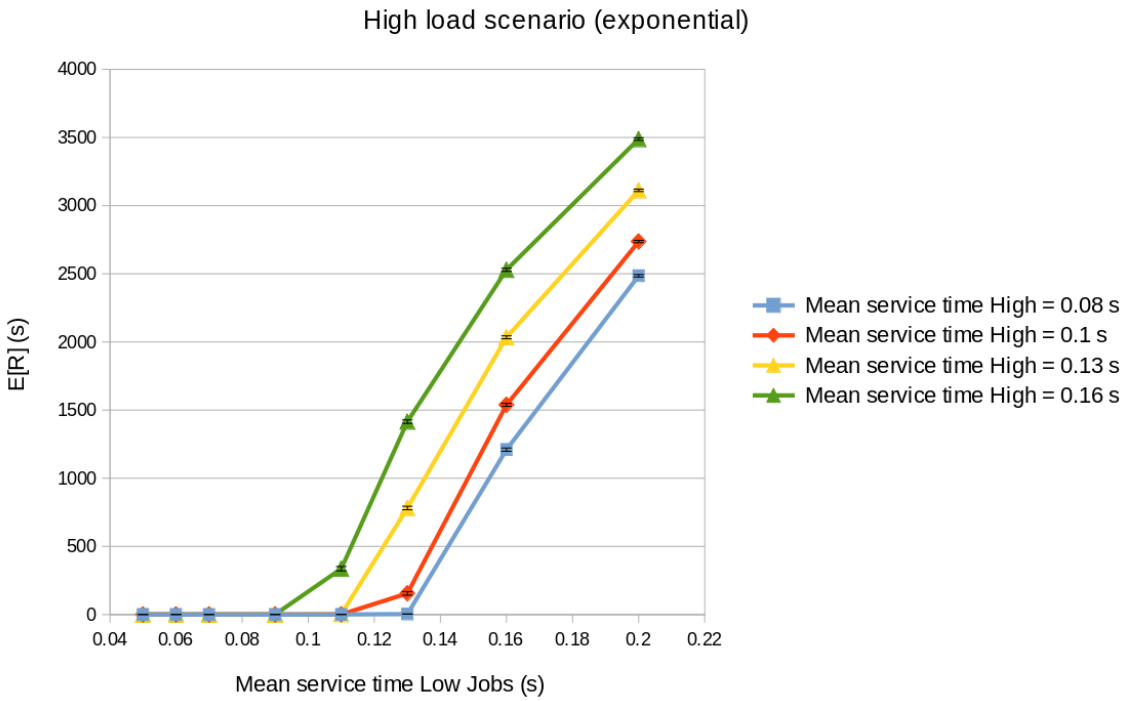

Figure 6.21: Mean Response Time High Load Scenario (exponential)

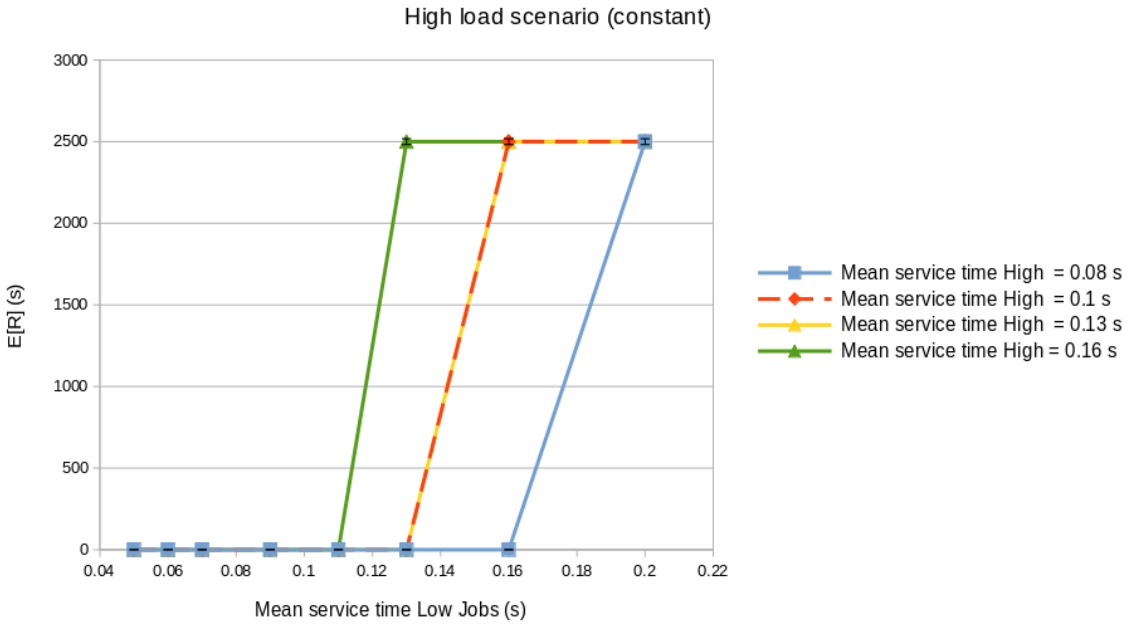

Figure 6.22: Mean Response Time High Load Scenario (constant)

Moreover, from the figure 6.23, we can see that, also in this exponential case, the Response Time is greater than in the constant one (figure 6.24).

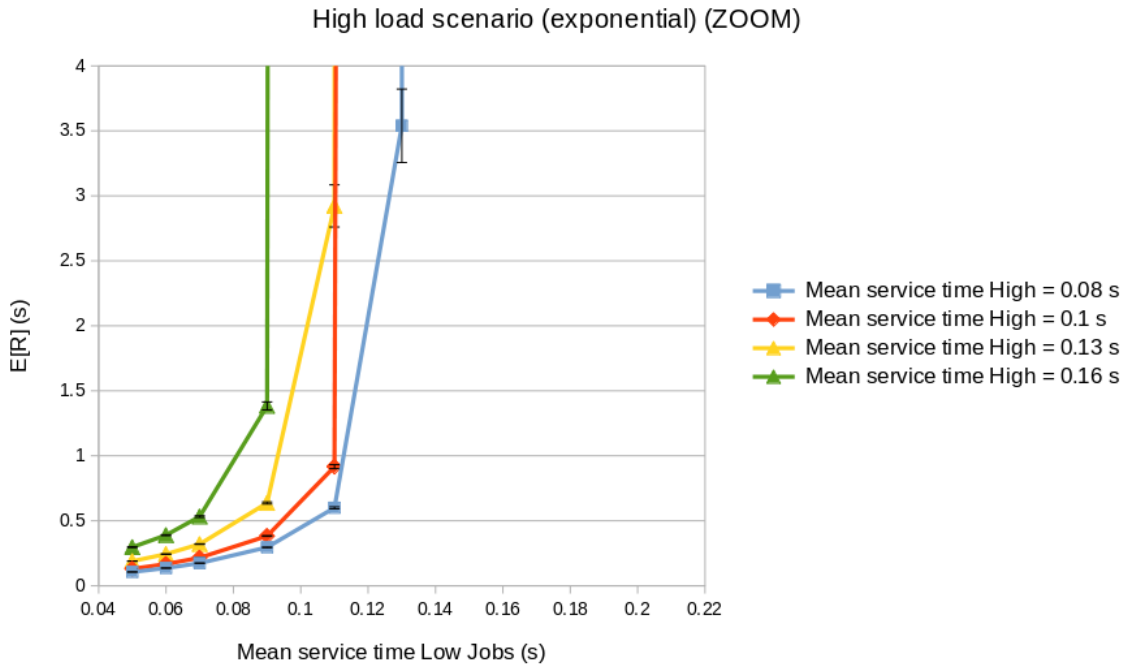

Figure 6.23: Mean Response Time High Load Scenario (exponential) (ZOOM)

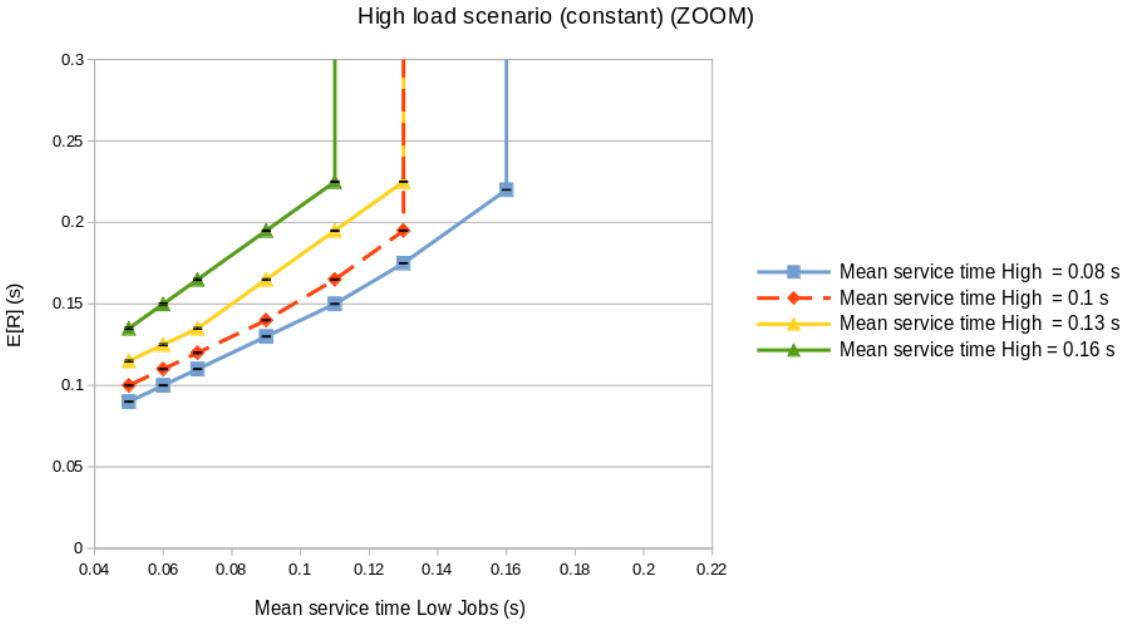

Figure 6.24: Mean Response Time High Load Scenario (constant) (ZOOM)

6.3 Extra case

To evaluate the impact of exponential service time instead of constant service time, we conducted an additional medium-load scenario test.
In this case, the Interarrival Times of Low Jobs (**8 values** with exponential distribution) and those of High Jobs (**4 values**) were varied, with exponential Service Time in one case and constant in the other. For every configuration **35 repetitions** were performed, in order to obtain meaningful results.

- $\frac{1}{\lambda_L} = (\mathbf{0.3s}, \mathbf{0.35s}, \mathbf{0.40s}, \mathbf{0.45s}, \mathbf{0.50s}, \mathbf{0.55s}, \mathbf{0.60s}, \mathbf{0.65s})$,
  $\frac{1}{\lambda_H} = (\mathbf{0.5s}, \mathbf{0.55s}, \mathbf{0.60s}, \mathbf{0.65s})$,
  $\frac{1}{\mu_L} = \mathbf{0.2s}$, $\frac{1}{\mu_H} = \mathbf{0.2s}$

As can be seen from the figures below, the impact of having an exponential Service Time is substantial. While keeping other parameters unchanged, it leads to a significant increase in Response Time.

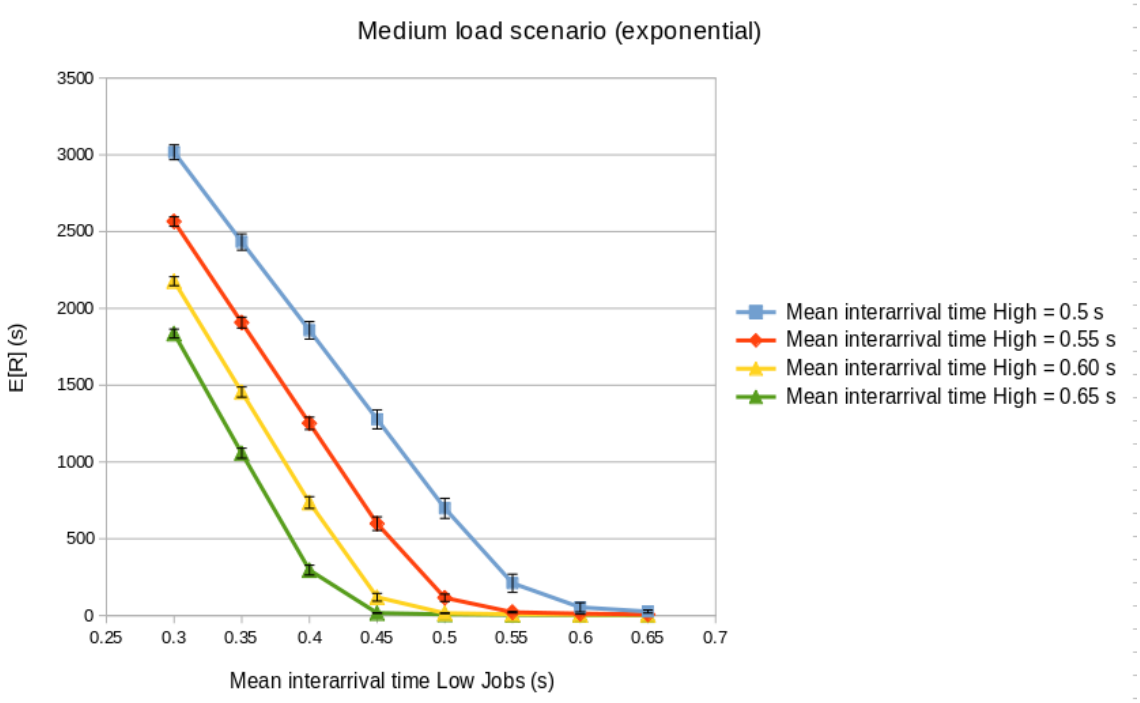

Figure 6.25: Extra case: Response Time exponential scenario

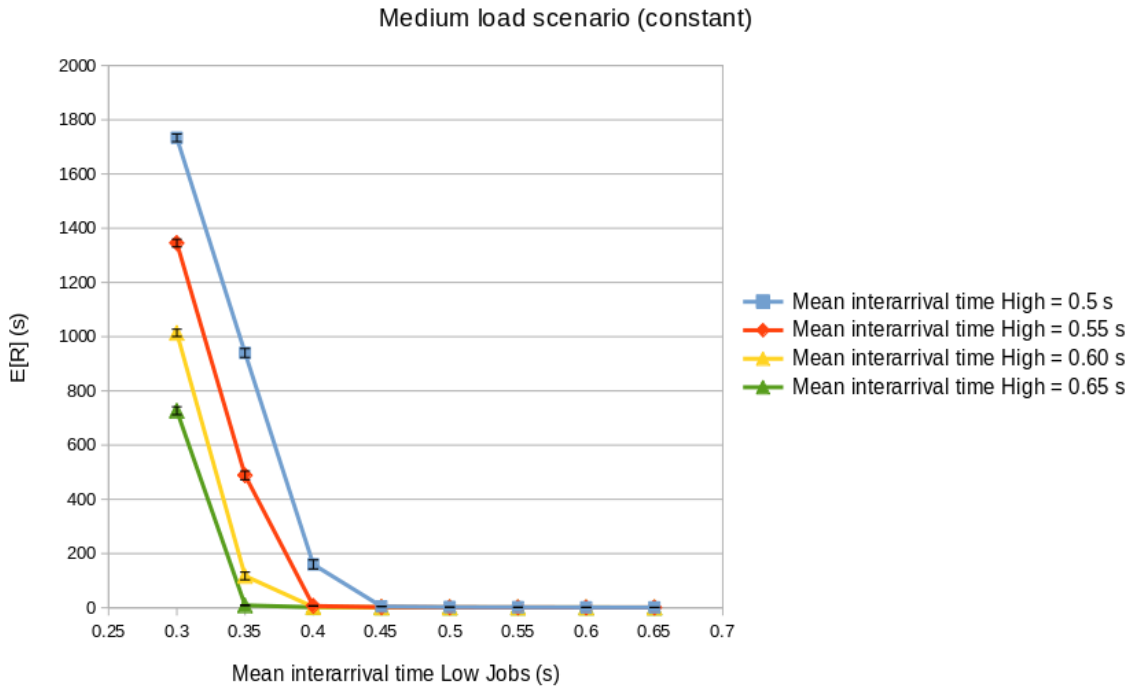

Figure 6.26: Extra case: Response Time constant scenario

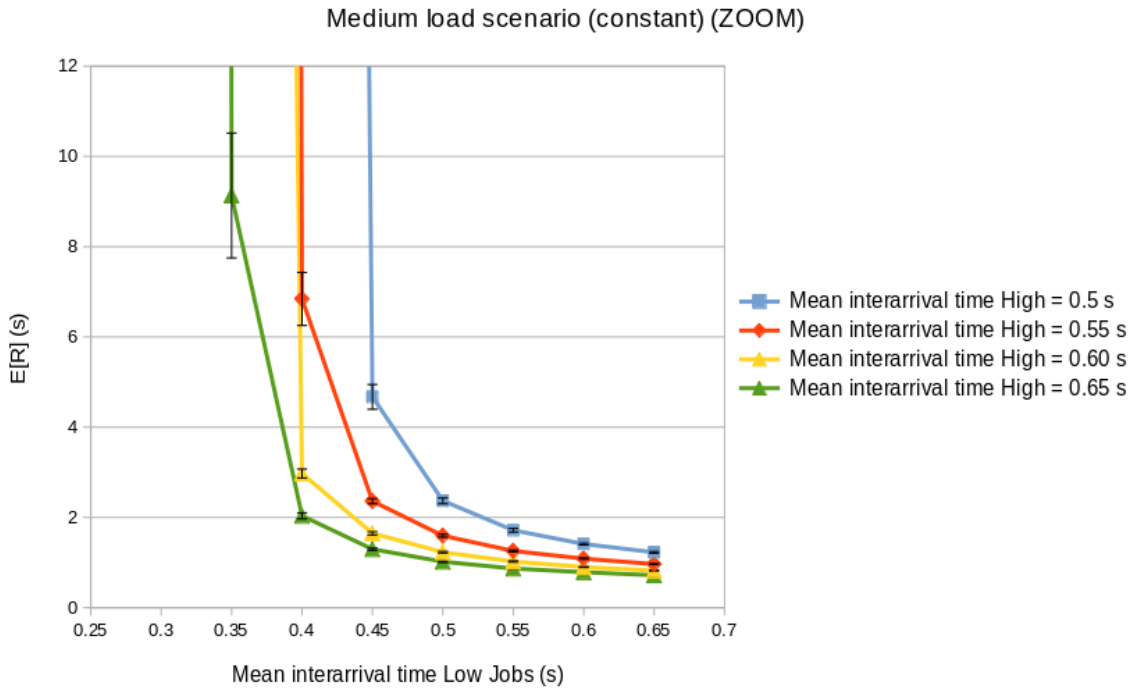

Figure 6.27: Extra case: Response Time constant scenario (ZOOM)

The impact of exponential Service Time on the **number of jobs in the queues** was also evaluated.

The figures below show the results relative to **E[N]** obtained from the first run of the configuration with $\frac{1}{\lambda_L} = \mathbf{0.45s}$ and $\frac{1}{\lambda_H} = \mathbf{0.55s}$.
It can be seen the **Mean Number of Low Jobs in the system** (**green points**) and the **Mean Number of High Jobs in the system** (**yellow points**).
In the constant case (figure 6.28) the Number of Jobs in the queues does not exceed a certain threshold, while in the exponential case (figure 6.29), it tends to diverge. In particular, in the constant case, the scheduler manages to process jobs, even when their number peaks. In the exponential case, at a certain point, the scheduler is not able to handle the incoming jobs and the number of jobs in the Low priority queue tends to grow very rapidly.

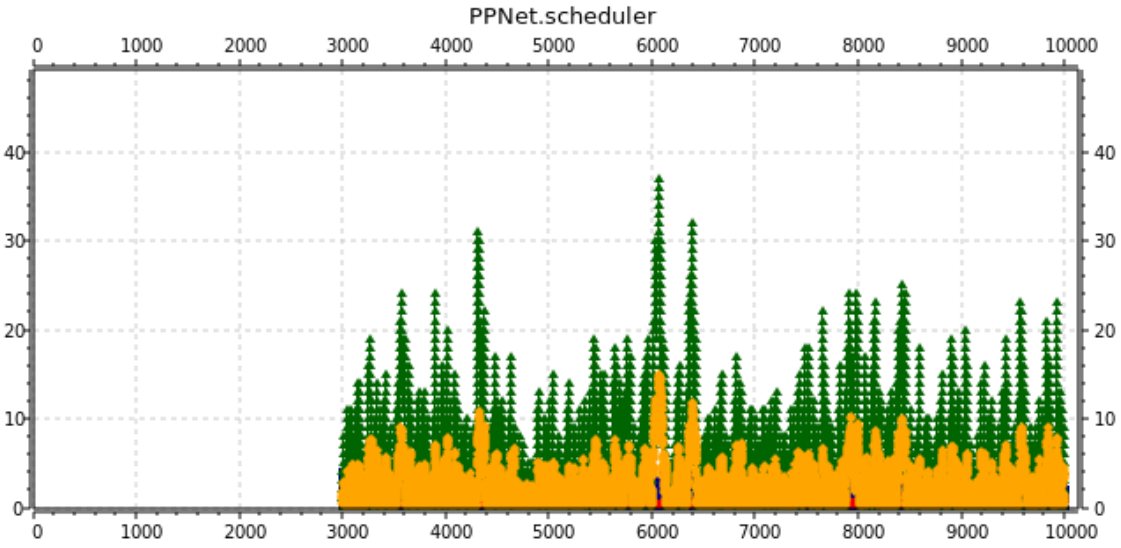

Figure 6.28: Extra case: Mean Number of Jobs constant scenario

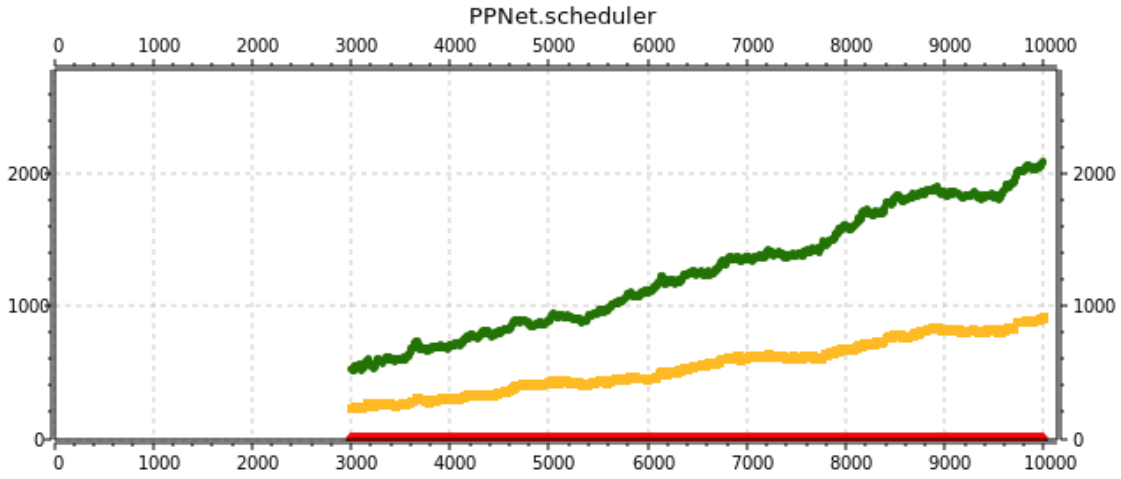

Figure 6.29: Extra case: Mean Number of Jobs exponential scenario

7 - Conclusions

The system has been evaluated in 3 different load scenarios: **Low Load**, **Medium Load** and **High Load**.
From the experiments conducted and the analysis of the results obtained, the following conclusions can be drawn:

- In all 3 scenarios, considering constant service time values, it can be observed that in the cases with **exponential** distribution of job **interarrival times**, the **response time** values obtained were on average **higher than in the constant case**.

- As far as the low load scenario is concerned, the experiments and results obtained from the $2^k r$ factorial analysis showed that the **factor most influencing** the response time is the **mean service time of the low jobs**. In this case, the response time of the low jobs tends to be very close to their service time, since the queues are empty most of the time.

- Considering a medium load case, with exponential distribution of interarrival times of jobs, the **impact of having an exponential service time**, as opposed to a constant one, **is substantial**. In particular, with other factors unchanged, there is a significant **increase in Response Time**.
  Moreover, in this case, the **mean number of jobs in the low priority queue** is also **strongly impacted**. This is because the possible arrival of jobs with a large service time may lead to the scheduler no longer being able to handle them and the queue to a rapid growth.