UNIVERSITÀ DI PISA

Computer Engineering

Performance Evaluation of Computer Systems and Networks

Multi-Programmed Server

STUDENTS:

Fernando De Nitto

Nicola Ferrante

Simone Pampaloni

Academic Year 2019/2020

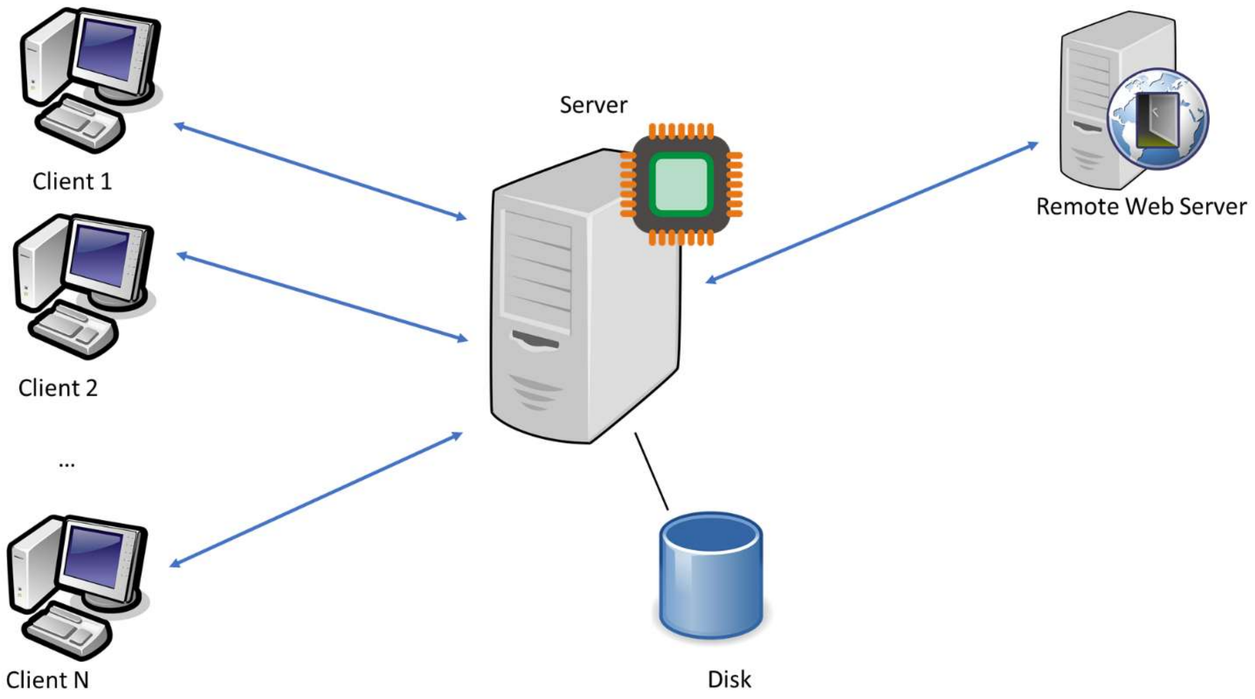
Contents

Overview

A multi-programmed server provides service to different concurrent clients. The server has access to a disk and can communicate with a remote web server. Clients send requests to the server and each request is processed one at a time by the server following a FIFO order. After an initial pre-processing phase, each request can be handled in the following ways:

- The request has finished processing and a response is sent to the client.
- The request requires a disk access and then a new processing is required.
- The request is sent to a remote web server and after its response a new processing is required.

Even disk and the remote web server handle one request at a time in a FIFO order.
A client that receives a reply immediately issues another request.



The aim of this study is to evaluate the performance of the system described above with particular emphasis on throughput. This study must be done considering various scenarios by varying the level of multiprogramming and making sure that the service demands at the three components (server processor, disk, and remote web server) have a considerably different mean.

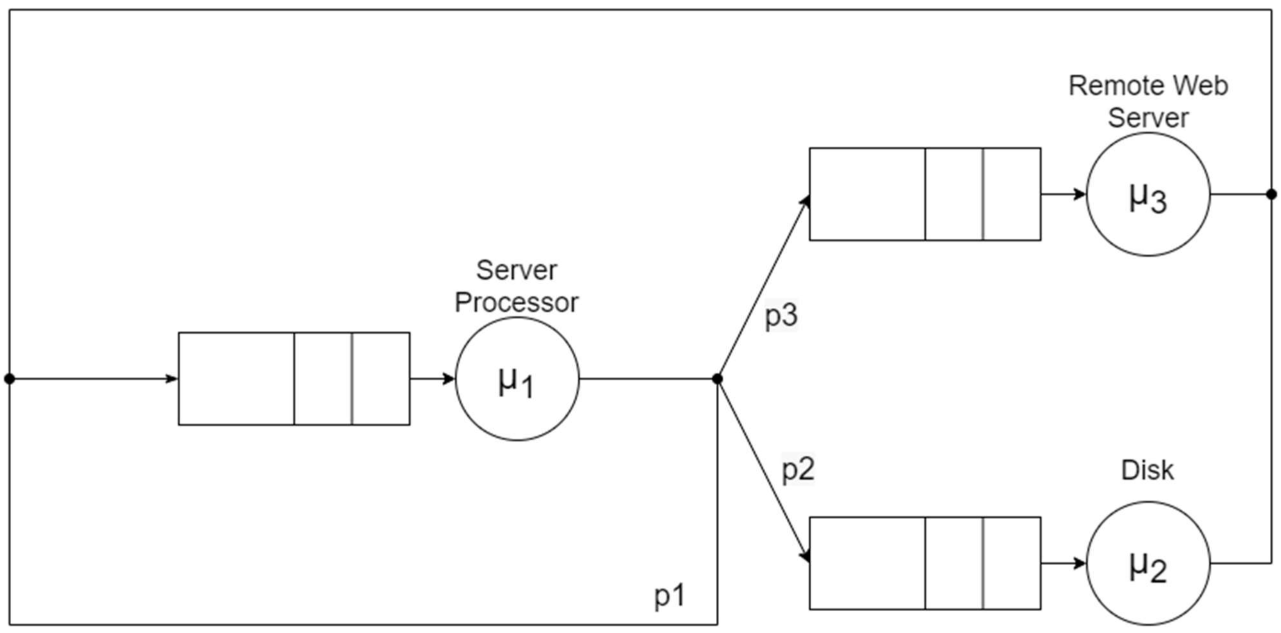
Performance Indexes

Since the objective of this study is to evaluate the performance of the system described above, the following performance indexes have been defined:

- Throughput: the number of served requests divided by the operating time.
- Mean response time: the mean time it takes between the departure of a request and the arrival of the response to the client
- Utilization: How long each system component (server processor, disk, and remote web server) processes requests during operating time.

Model

the following model represents an abstraction of the system to be studied:



The server has been modeled into two service centers (SCs) that represent the two main components that are needed for this specific study: the server processor, and the disk. The remote web server was modeled into a single SC.
Since clients send instantly a new request whenever they receive a response, there is always the same number of service requests in the system. For this reason, as can be seen from the diagram, the system is designed as a Closed Jackson’s Network.

So, transactions may involve processing in the main server, access to the disk, and remote queries to a remote web server. As mentioned above, a new transaction always requires some processing time as a first step. After the processing has occurred:

- With a probability $\boldsymbol{p_1}$ the transaction is terminated, and a reply is sent to the client that originated it.
- With a probability $\boldsymbol{p_2}$ a disk access is required. After that, a new processing is required.
- With a probability $\boldsymbol{p_3}$ = 1-p_1-p_2 a query to remote web server is required. After that, a new processing is required.

The server processor, the disk, and the remote web server handle one request at a time in a FIFO order. Processing, disk access and query to remote server service demands (with rates $\boldsymbol{\mu_1}$, $\boldsymbol{\mu_2}$ and $\boldsymbol{\mu_3}$) are exponential IID RVs, and they are different from one iteration to another, even for the same transaction.

In addition, a system model was created with the Queueing Theory. Using **Buzen's convolution algorithm**, it has been possible to create a model that returns the throughput, the mean response time and the utilization (of each SC) of the system inserting as input the probabilities $\mathbf{p_i}$, the rates $\mathbf{u_i}$, and the number of clients.
This model will be used to compare the data obtained from the simulations with the values of the theory to make evaluations on the goodness of our simulation.

Factors

For the system model designed, the following factors have been defined.

- Number of clients
- Mean service demands:
  - Mean service demand of server processor
  - Men service demand of disk
  - Mean service demand of remote web server
- Forwarding Probabilities

Tools

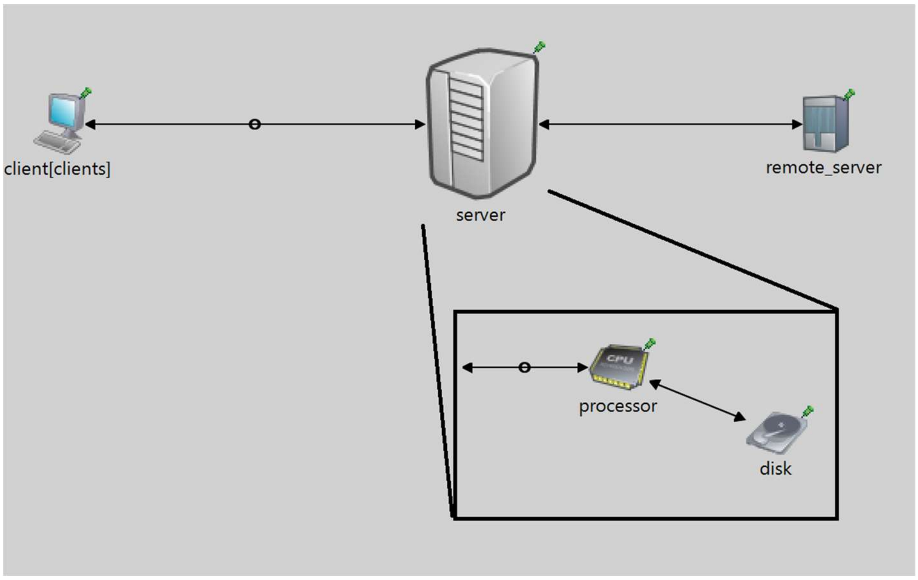
In the following list there are all the tools used during this study.

- <u>Simulation Tool:</u>
  - **Omnet++**: simulation tool used to implement the system simulator.
- <u>Data Analysis and Graph Creation:</u>
  - **MS Excel**

- **DB Browser**
- **Python Scripts** (Jupyter NoteBook + Pandas)

Implementation

The simulation tool *Omnet++* was used to implement the system simulator.
The system was implemented as a single network (as you can see in the following figure), consisting of N clients, the server, and the remote web server.



The connections between client and server have been set up with a delay of 40ms, the connection between server and remote web server has been set up with a delay of 80ms. Finally, connections within the server (between processor and disk) have been set up with a delay of 5ms.

Delays have been set based on the average delay within a LAN and the average delay on the Internet.

The system is composed as follows:

- **Client**: composed by a *cSimpleModule*. When initialized, it prepares and sends the first request (*cMessage* type) to the server. When the reply arrives, immediately prepare a new request to send to the server. Whenever a reply arrives, a signal is issued by the client to record the response time of that message.

- **Server**: represented by a *cCompoundModule* composed of two *cSimpleModule*, the processor and the disk.

  - **Processor**: each time it receives a request, if the it is busy, the request is queued, otherwise a pre-processing phase begins (the pre-processing time has an exponential distribution). Once this phase is finished, the processor determines one of the three possible forwarding alternatives using a uniform distribution (0,1) and comparing it with the forwarding probabilities p_1, p_2, p_3. The server emits a signal every time a transaction ends (forwarding probabilities equal to p_1), which allows to calculate throughput.

  - **Disk**: each time it receives a request, if it is busy, the request is queued, otherwise a processing phase begins (the processing time has an exponential distribution). Once this phase is finished, disk sends a response to the processor and if its queue is not empty, it starts a new processing with a new request.

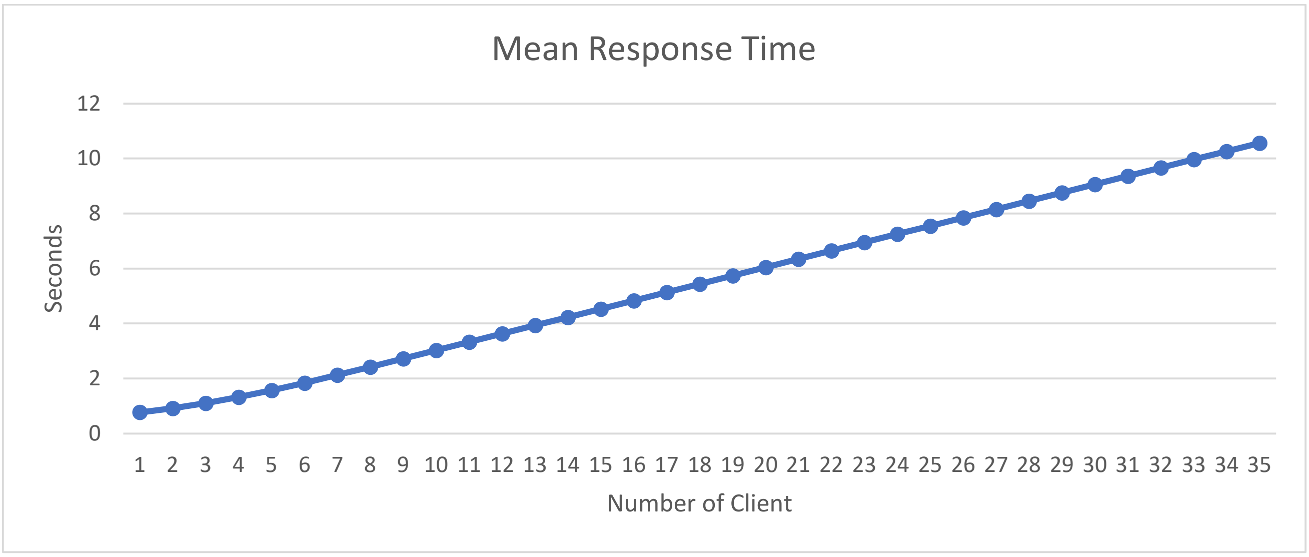
- **Remote Web Server**: composed by a *cSimpleModule*. Each time it receives a request, if the it is busy, the request is queued, otherwise a processing phase begins (the processing time has an exponential distribution). Once this phase is finished, remote web server sends a response to the processor and if its queue is not empty, it starts a new processing with a new request.

All the queues of the system are represented by a *cQueue*.
Each SC emits a signal every time it finishes processing a request and the queue is empty, indicating the time that has passed in processing. This signal is useful for calculating the utilization of each SC.

Verification

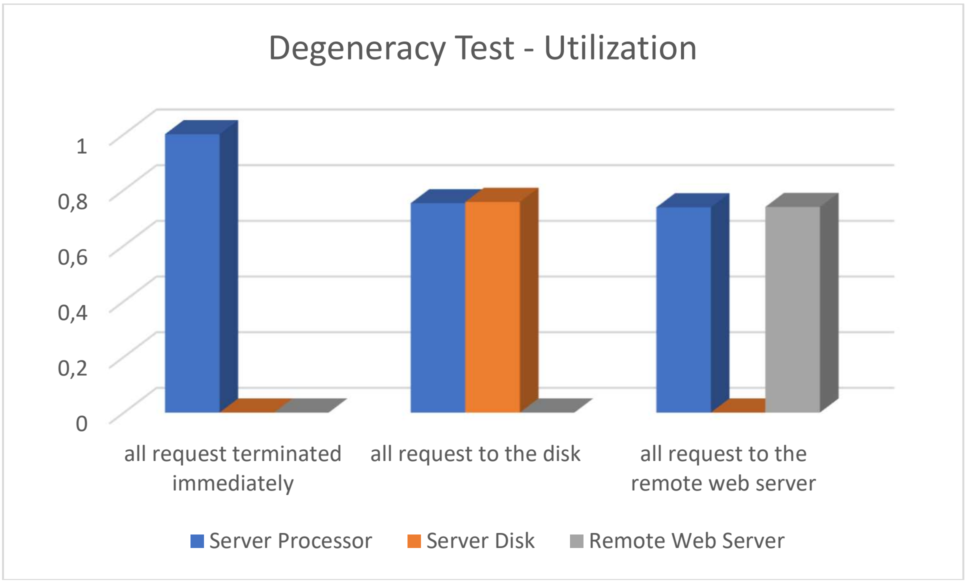
Various methods have been used to verify the correctness of the code and the correctness of the implementation of the model.
First, **Valgrind** was used to verify that there were no checking bad memory accesses or memory leaks.
Then, tests were made by running the simulation. All tests have been carried out considering the mean service demands of each SC equal to 1 second and with a simulation time of one hour (3600s). Each test has been repeated 10 times.

- **Continuity Test:** To verify the correctness of the code, we have seen how the mean response time behaves with small progressive changes in the number of clients.



As can be seen from the graph, the mean response time increases linearly with the number of clients.

- **Degeneracy Test:** We analyzed the cases where all the requests were forwarded to a single SC. We considered the case in which all requests ended immediately after the pre-processing phase ($p_1 = 1$, $p_2 = p_3 = 0$), all requests are sent to the disk ($p_1 = 0$, $p_2 =1$, $p_3 = 0$), and all the requests are sent to the remote web server($p_1=p_2= 0$, $p_3 = 1$).



As can be seen from the histogram, the degeneracy tests were correct. It is important to note that processor utilization is always high in all three cases because for each type of request there is a pre-processing phase.

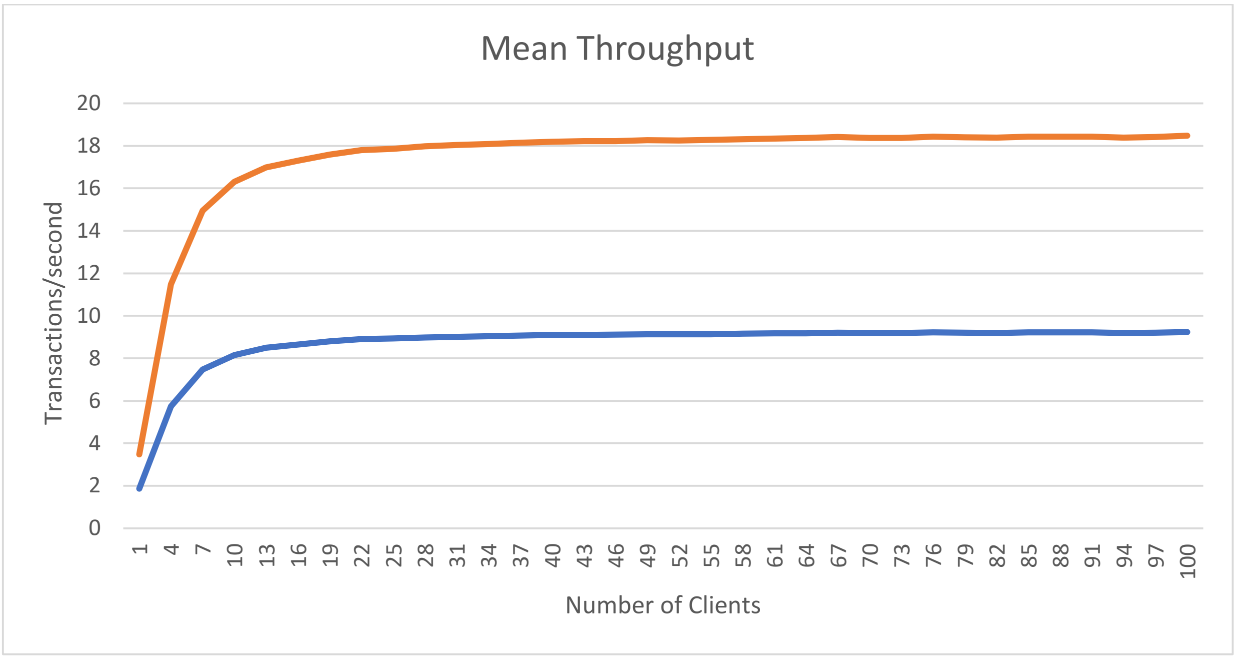
Calibration

Mean response times have been determined for processor, disk, and remote web server. For each SC, two different response time levels were chosen:

- **Processor**: 0.01 second or 0.1 second
- **Disk**: 0.1 second or 1 second
- **Remote Web Server**: 0.1 second or 1 second

For the disk and remote web server the means response time are the same so that the two types of requests do not differ by two orders of magnitude.

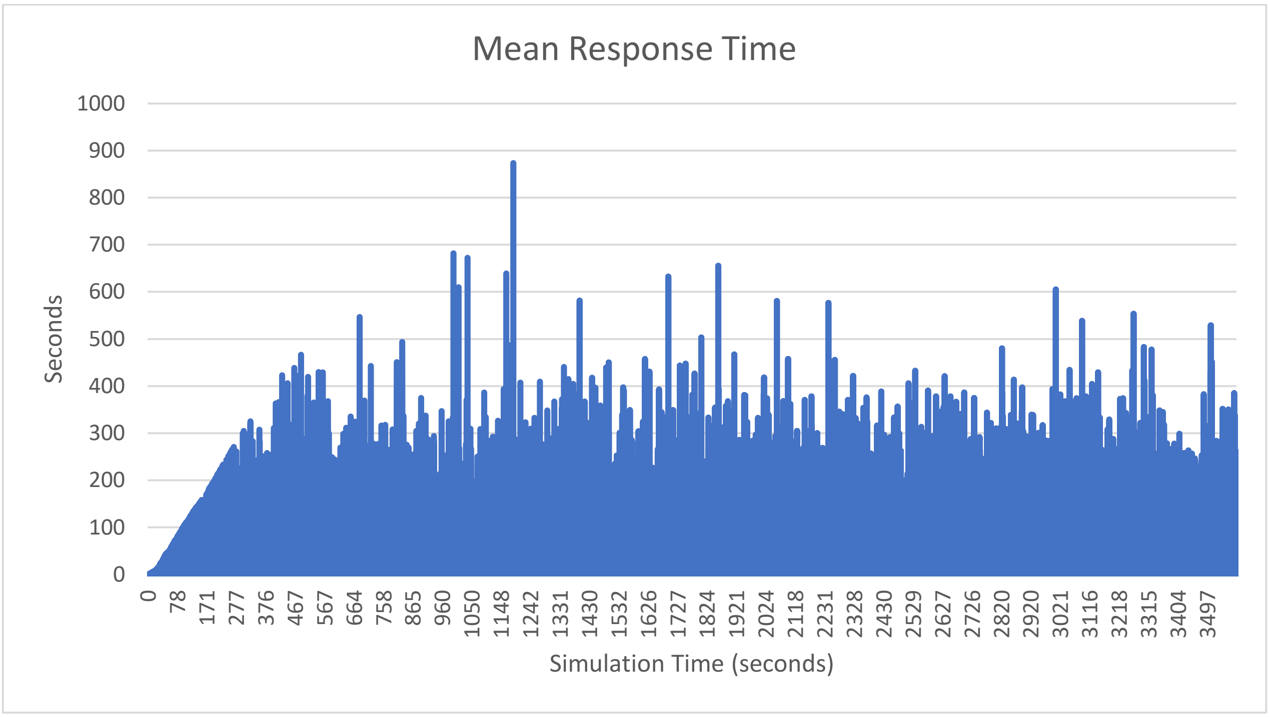
To understand the maximum number of clients to be set in order to have values of interest in the simulations. The mean throughput has been calculated for an increasing number of clients (from 1 to 100 with a step of 3). Ten tests were carried out for each number of clients. Two tests have been made: one with the lowest mean response times of each SC, and one with the highest mean response times of each SC. The graph below shows the average throughput trend as the number of clients increases.



Throughput becomes almost constant as the number of clients increases. For this reason, we have chosen to set the maximum value of the clients to 49.

Warm-up period and Simulation Time

In this phase is important to identify the warm-up period of the system. To do this, a simulation has been made with the maximum number of clients previously selected. Ten tests have been carried out. The mean response time has been analyzed to see when it would stabilize. The maximum number of clients was chosen because it is the worst case, as with increasing clients the average response time increases.



As can be seen from the graph, the mean response time begins to stabilize after 600 seconds. To ensure that the system is stable, a 1200 second warm-up time has been chosen.

Consequently, a simulation time of 1200+ 3600 ≅ 4800 seconds was chosen, time necessary to have the data for one hour of simulation time.

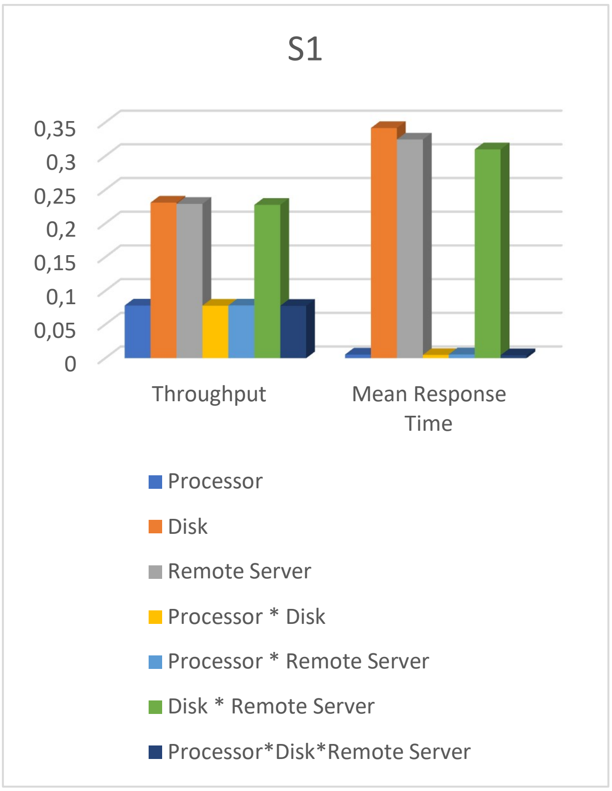
Experiment Design

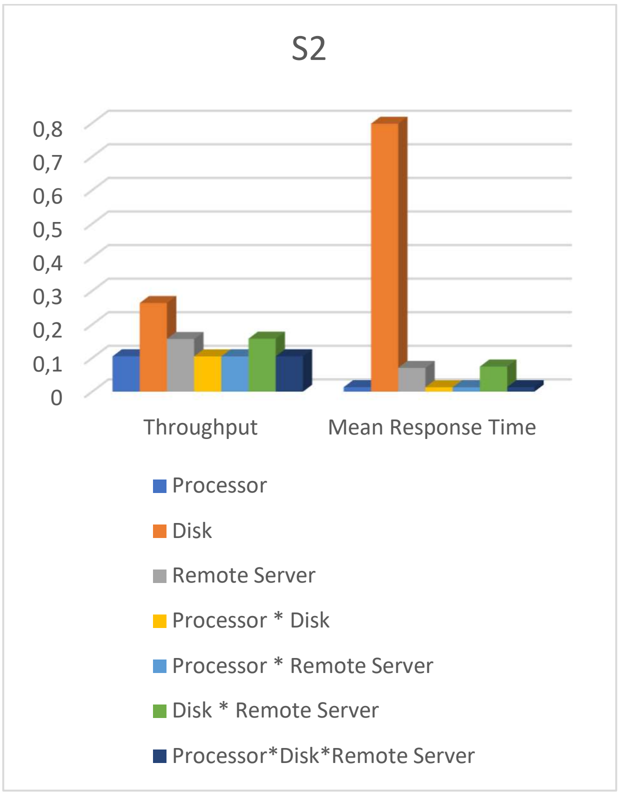
The experiments were designed by combining forward probability values trying to create different types of scenarios. In particular, the following 4 scenarios have been chosen:

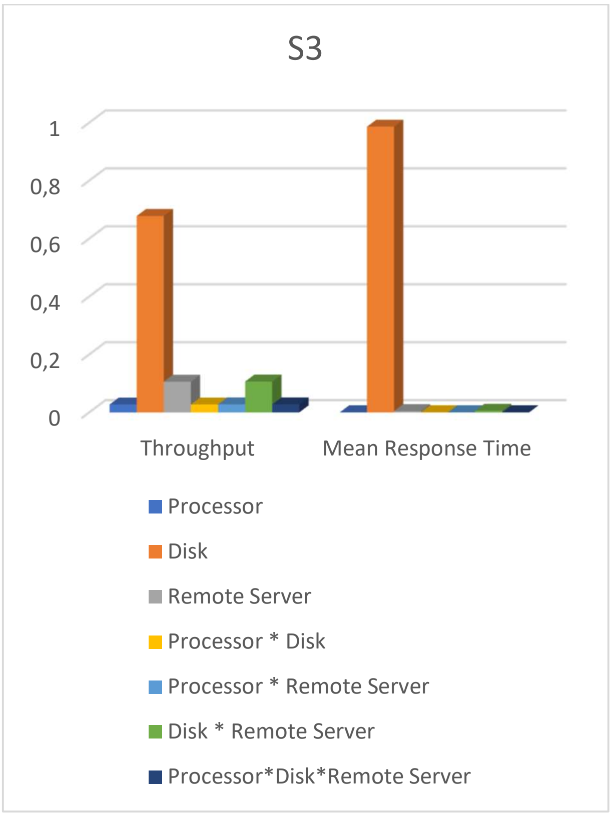
- **S1**: p_1=0.34, p_2=0.33, p_3=0.33
- **S2**: p_1=0.60, p_2=0.25, p_3=0.15
- **S3**: p_1=0.15, p_2=0.60, p_3=0.25
- **S4**: p_1=0.15, p_2=0.25, p_3=0.60

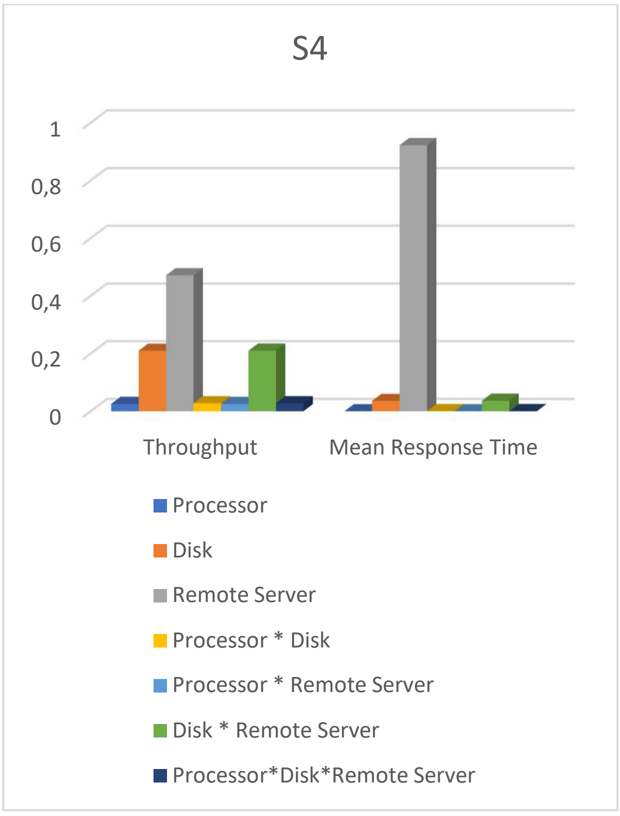
The S1 scenario has been chosen to have a balanced distribution of the various types of requests. The S2, S3 and S4 scenarios have been chosen because they represent respectively a scenario in which a large number of one of the requests is required. These four scenarios allow to analyze various types of applications, from those more interactive, to those more oriented towards downloading or uploading data from disk, to those that require remote queries.

Various configurations were applied to each scenario by selecting a combination of different service time means of processor, disk, and remote web server. A *2^k r factorial analysis* has been made on the factors regarding the service time means to understand which of these influences mostly the performances. Based on this information, the most relevant configurations have been selected for the scenarios. Each analysis has been made considering *r* = *10* and considering the case of 49 clients. The following bar charts represent the results obtained:









The analysis shows that the disk and the remote web server have the greatest impact on the performance of all four selected scenarios. So, considering the analysis previously made, the following configurations were chosen:

- **C1**: Processor=0.01, Disk=0.1, Remote Web Server=0.1
- **C2**: Processor=0.01, Disk=0.1, Remote Web Server=1
- **C3**: Processor=0.1, Disk=0.1, Remote Web Server=0.1
- **C4**: Processor=0.1, Disk=1, Remote Web Server=0.1

These configurations allow to understand how the system behaves when:

- C1: The SCs have their minimum response time mean.
- C2: the SCs have different mean response times of an order of magnitude (as required by the study requirements).
- C3: the SCs have all the same mean response time.
- C4: One of the two SCs of the system that have the greatest impact on performance (in this case the disk) has higher mean response time and the processor has lower mean response time.

The above-described scenarios and configurations have been analyzed on varying the number of clients (from 1 to 50 with step of 2) and for each of them ten repetitions were made.

Data and Simulation Analysis

The following paragraphs will list the results of the scenario simulations. The values for the utilization are related to the data obtained with the maximum number of clients (49). The graphs will show the mean value between the ten repetitions.

Scenario 1

This scenario model the case of an application where the three type of transactions (to disk, to remote server and return to the client) are equally likely. In this scenario we have considered, as said previously, 4 mean service times configurations that are more representative from the point of view of the outcomes.

- Throughput

  Looking at the simulations result the throughput reach a stable maximum value for each configuration, as expected. The higher throughput has been measured with the configuration c1 ($\cong$ 10 transactions/s) that is the configuration with lower service time for each SC. The other three configurations have significantly lower values then c1. In particular, c2 and c4 have the lowest throughput, and c3 has a throughput lower than c1 but higher than c2 and c4.

  Looking at the details of the configurations, we can see that the maximum throughput decreases as the mean service time of SC increase. An interesting result is that the two lowest throughputs are achieved in correspondence of the two configurations where the one between disk and remote web server have their maximum service times. In these configurations the processor's mean service time is different, but as confirmed from the *$2^k r$ analysis,* the impact of its service time on the throughput is negligible. In fact, as shown in Figure 1, the two configurations have a very similar throughput. The c2 configuration, the one where all SCs have the same service time, has an intermediate throughput, but significantly lower than the c1's one.

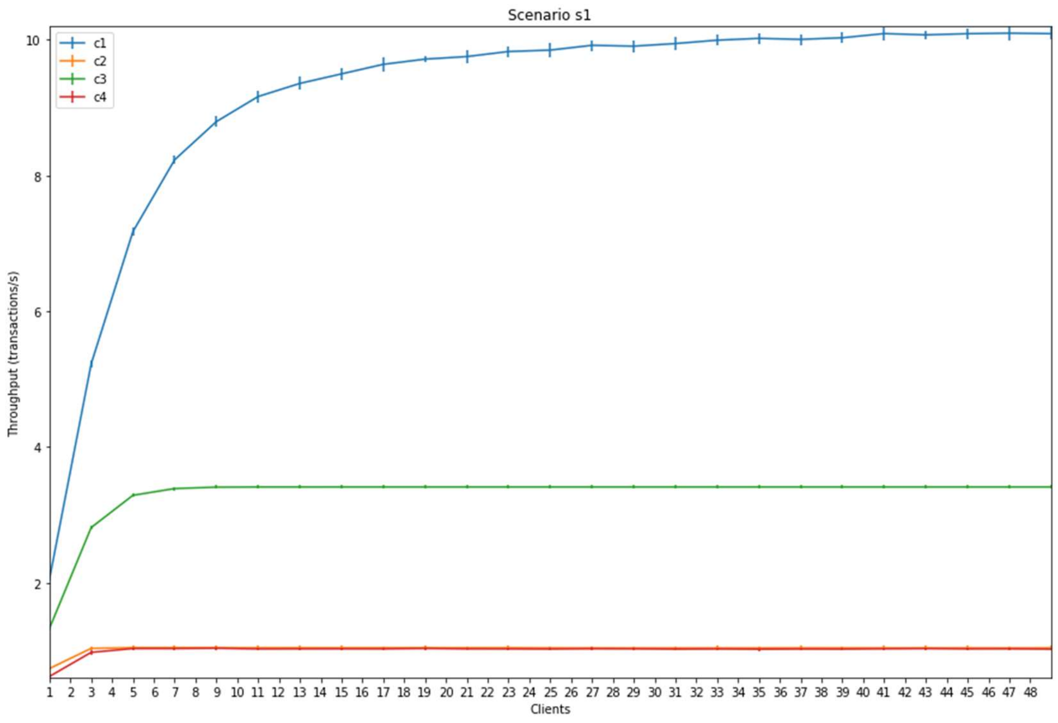


Figure 1 – Throughput

- Mean Response Time

The simulation data confirm that the most important factors that influence the throughput are the same that influence the response time of the system (even if in different measure). As we can see from the Figure 2, the configurations with higher throughput are the one with lower response time. As expected, the response time increases with the number of clients, thus not reaching a stable value.

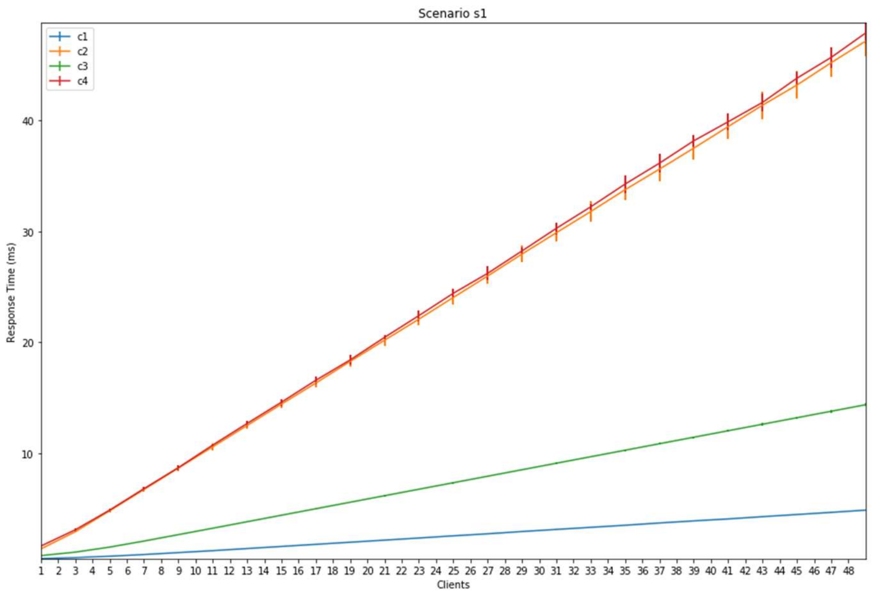


Figure 2 - Mean Response Time

- <u>Utilization</u>

Looking at the utilization of each SC, we can see that when the service centers have different mean service times, the utilization of the slowest is the higher. Instead when the three SCs have the same mean service time, the processor has the highest utilization.

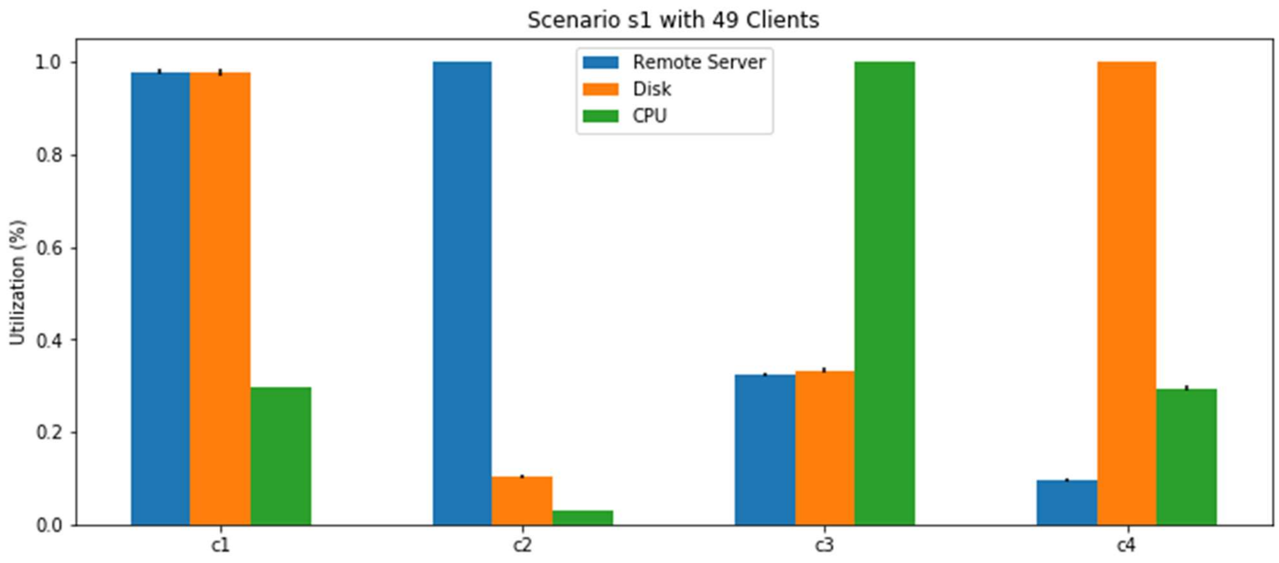


Figure 3 – Utilization

Scenario 2

This scenario model the case of an application where the transaction that immediately sends a response to the client has a higher probability (p_1=0.60, p_2=0.25, p_3=0.15).

- <u>Throughput</u>

  As in the previous scenario, the c1 configuration achieves a higher throughput value than the other configurations. However, the value on which the throughput stabilizes is 25 transactions/s, a higher value than in the previous scenario. This is due to the fact that transactions are more likely to be immediately re-sent to clients. In particular, each scenario-configuration pair will have a greater throughput improvement with larger forwarding probabilities and less mean response time. For the same reason, all the other configurations obtained a higher throughput than in scenario 1.

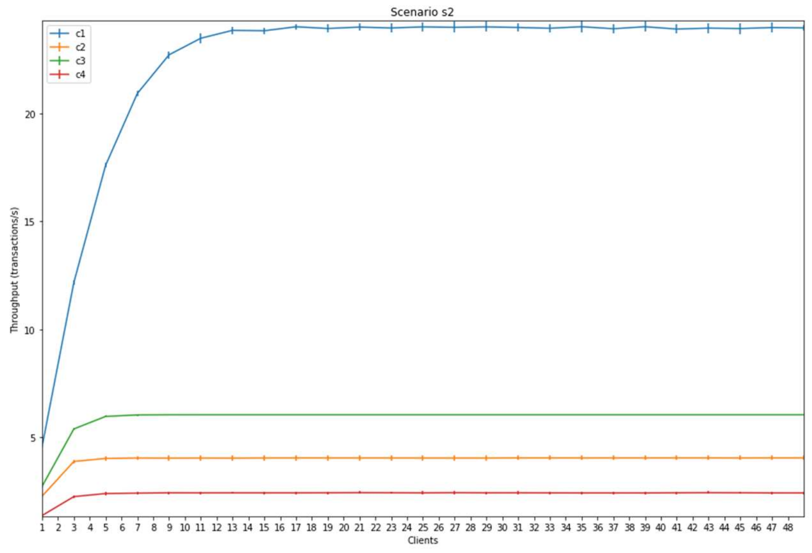


Figure 4 – Throughput

- <u>Mean Response Time</u>

As in the first scenario, the results obtained with response time are consistent with those obtained with throughput. Also, in this case it is noted that the configurations with a high throughput have low response times and vice versa.

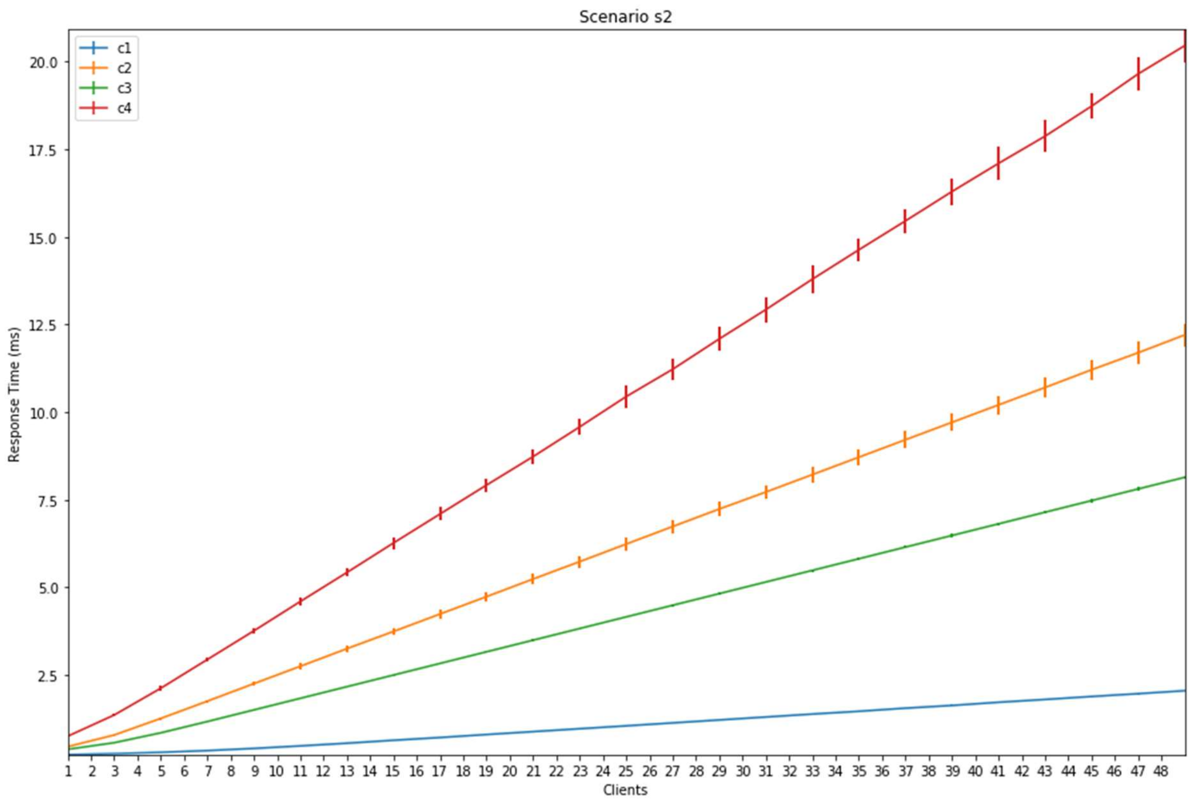


Figure 5 - Mean Response Time

- Utilization
  As in the previous scenario, it is noted that when all SCs have the same mean service time, the utilization is higher in the processor. On the other hand, when the processor has a lower mean response time than the others, the utilization is concentrated more in the SC with higher mean service time. In fact, in c2 and c4, the SC with the greatest utilization is the remote web server and the disk, respectively. In case c1, where the disk and the remote web server have the same mean response time, the utilization is greater on the disk, consistently with what obtained from the *2^k r analysis.*

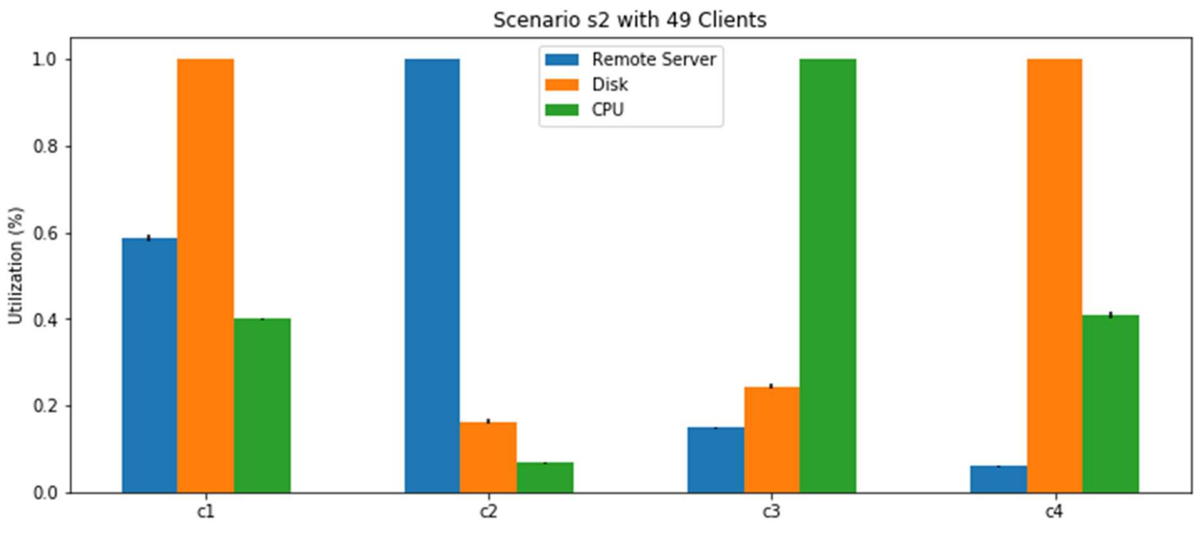


Figure 6 – Utilization

Scenario 3

This scenario model the case of an application where the transaction that sends a request to the disk has a higher probability (p_1=0.15, p_2=0.60, p_3=0.25).

- Throughput

The plot below shows that in this scenario, where the disk is used more than the other two SCs, the difference between c2 and c4 becomes larger. In fact, in c2 the disk has its lowest service time, in c4, instead, it has its highest service time. The maximum throughput achievable is again the one of c1, but in this scenario it is considerably lower than in the previous two. This can be explained from the different forwarding probabilities, in this scenario transactions need a disk access the 60% of the time, and only 15% of the times are sent back to the client once arrived to the processor.

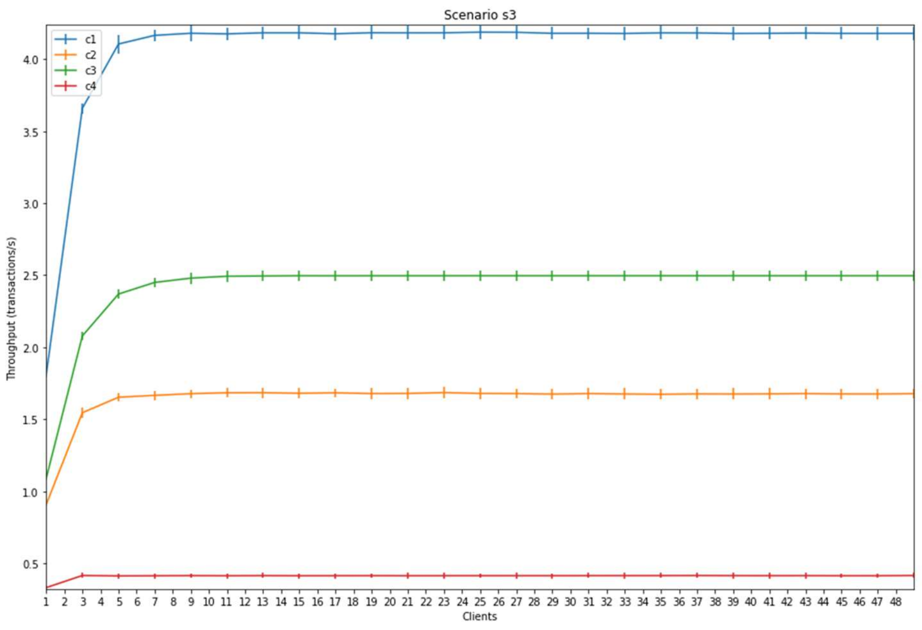


Figure 7 – Throughput

- Mean Response Time

As in the other scenarios the response time and the throughput show coherent outcomes, the configurations with lower response time are the one with the higher throughput.

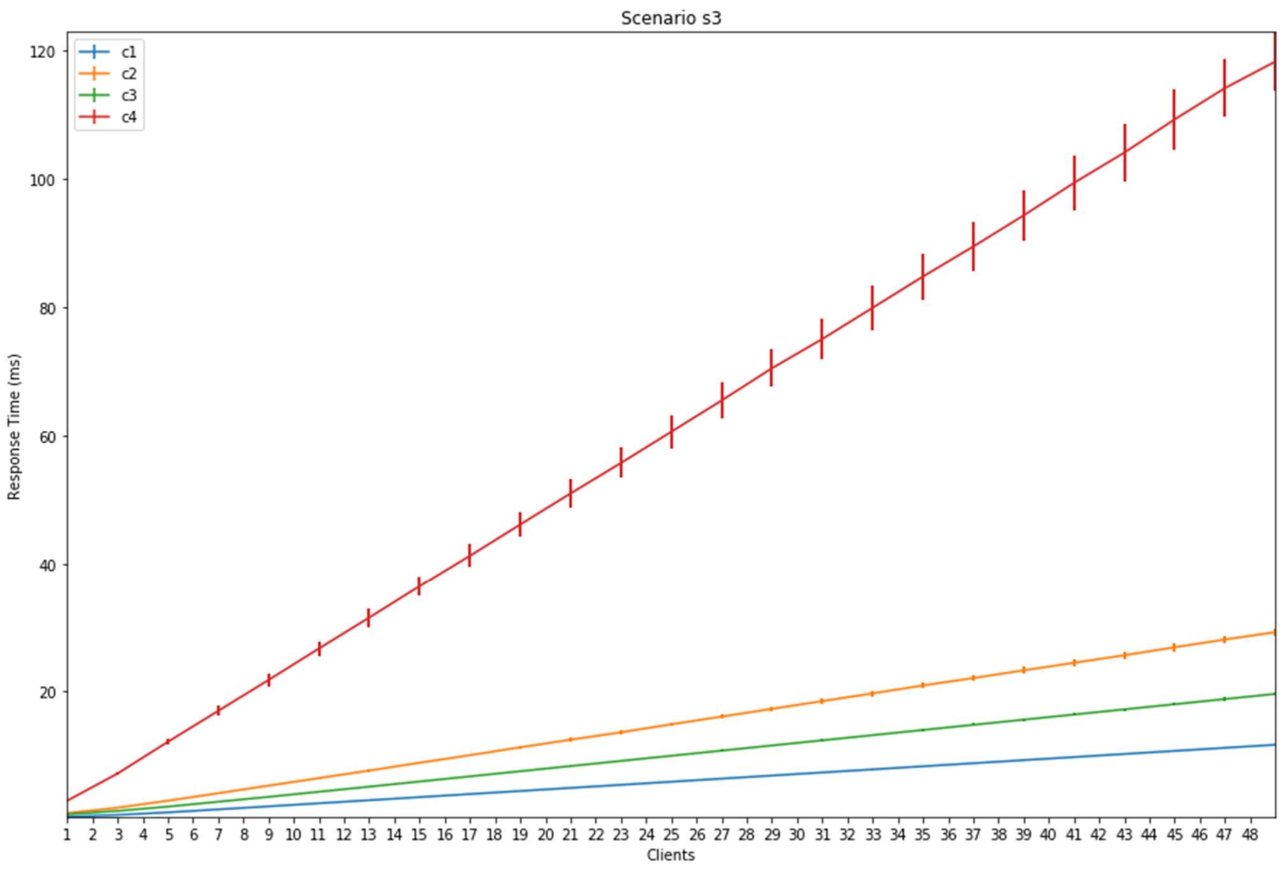


Figure 8 - Mean Response Time

- <u>Utilization:</u>

As can be expected, the utilization in c1 shows that, since the probability that a transaction is sent to the disk is higher, the disk's utilization is considerably higher than the other two SCs. In c2 we can see that, since the slowest SC is the remote server, its utilization is higher. Configuration c2 shows a similar trend to the previous scenarios, in the case that all the SC have same mean service time the CPU has the higher utilization, in this case the disk has an higher utilization than the remote server (due to different forwarding probabilities). The last configuration shows that the utilization of the disk approaches one, since it is the slowest component and also the most likely SC towards transaction are sent.

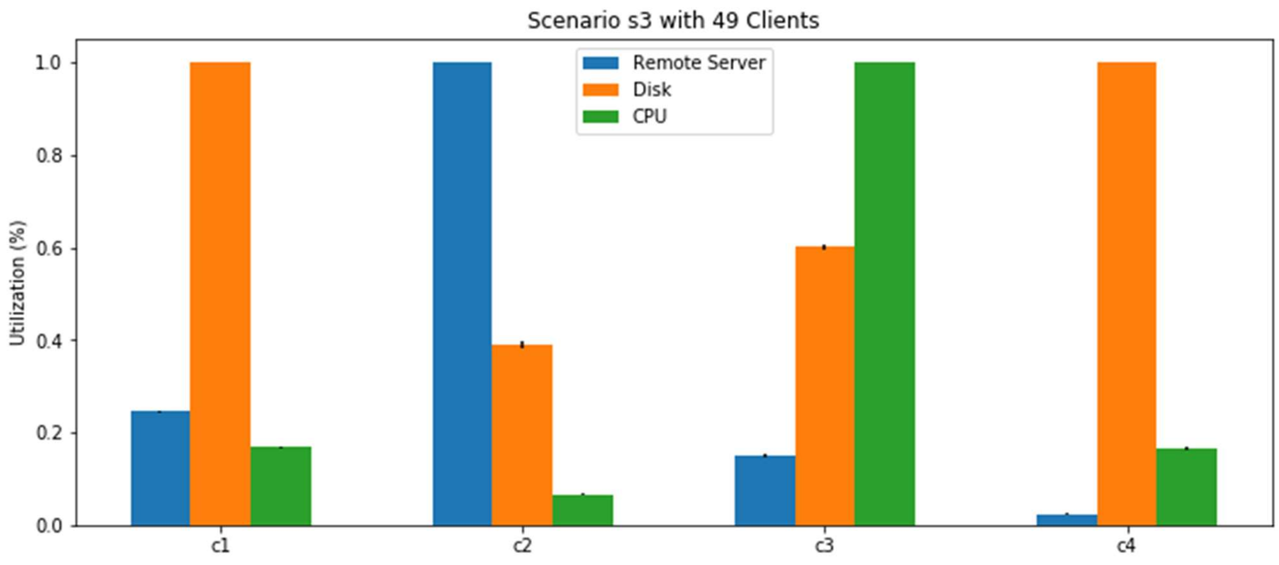


Figure 9 - Utilization

Scenario 4

This scenario model the case of an application where the transaction that sends a request to the remote web server has a higher probability (p_1=0.15, p_2=0.25, p_3=0.60).

- <u>Throughput</u>

The most remarkable difference from the other scenarios is the inversion of the two lowest throughput configurations. In this case c4 has a higher throughput than c2. This can be explained looking at the forwarding probabilities (higher probability that a transaction is sent to the remote server) and at the service times in c2 and c4. In c2 the disk has a lower service time than the remote server, and the inverse happens in c4. Again, as seen from *2kr analysis*, even if the processor in c2 has a lower service time then in c4, we can achieve an higher throughput in the latter, since the SC with the highest forwarding probability has a lower service time. In this scenario the maximum throughput achievable is lower the (2.5 transactions/s in c1). The explanation can be that we have a larger delay in the communication with remote server. And a low probability that a transaction is sent back to the client (15%).

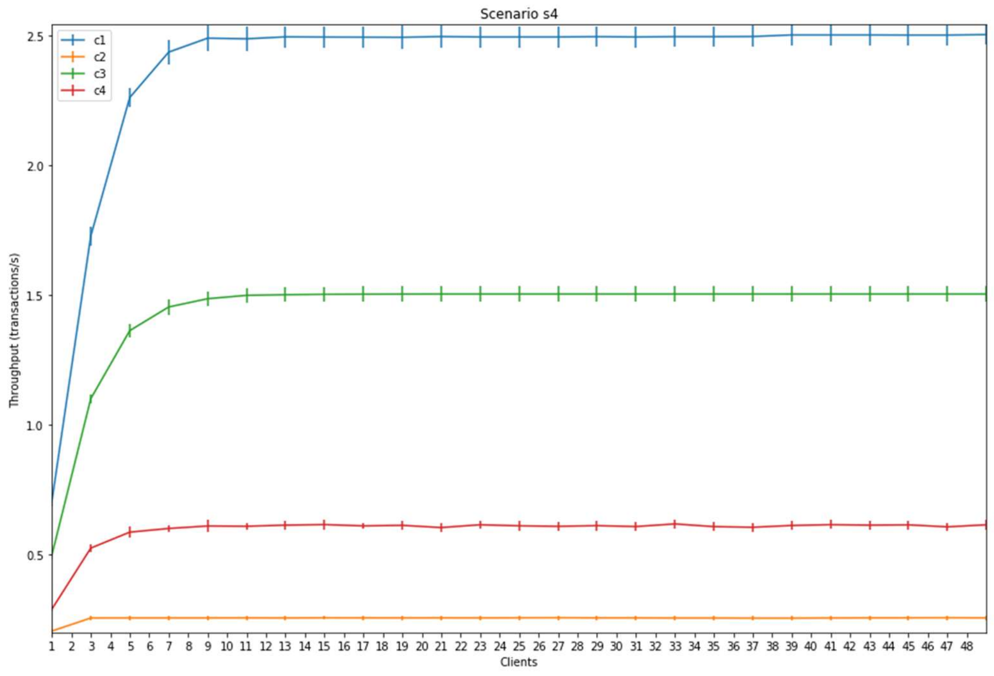


Figure 10 – Throughput

- <u>Mean Response Time</u>

The response time, coherently with what seen in the other scenarios, increase linearly with the number of clients, and lowest throughput calls for higher response time.

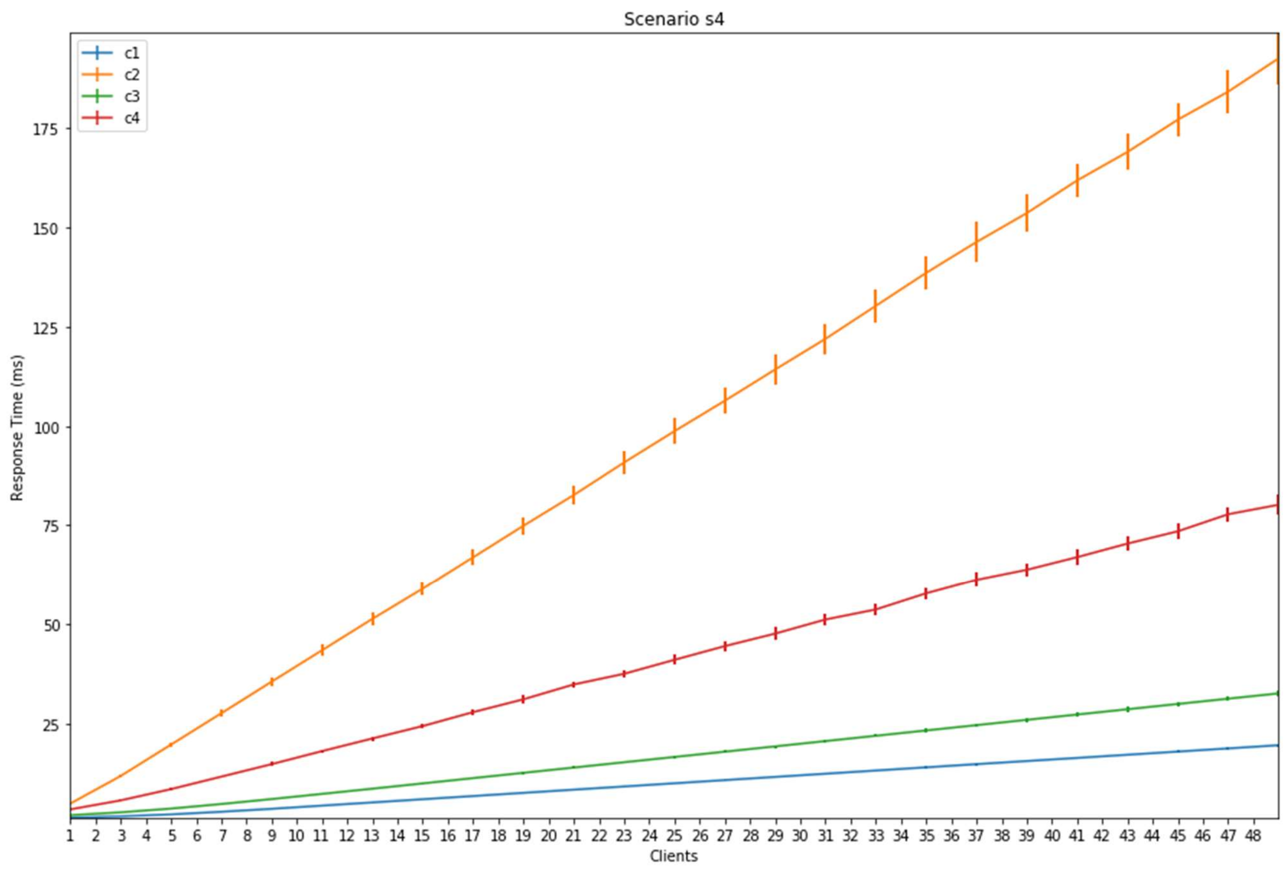


Figure 11 - Mean Response Time

- Utilization

The utilization, similarly to the other scenarios, shows that the remote server (which has the highest forwarding probability) has an higher utilization in the first two configurations. Instead, when all the SCs have same mean response time, the processor has the highest utilization. In c4, where the disk has a service time an order of magnitude higher than the remote server one, comes out that its utilization is the higher of the three. Even if the remote server has a higher forwarding probability, since its mean response time is equal to the processor one, the latter has higher utilization then the former.

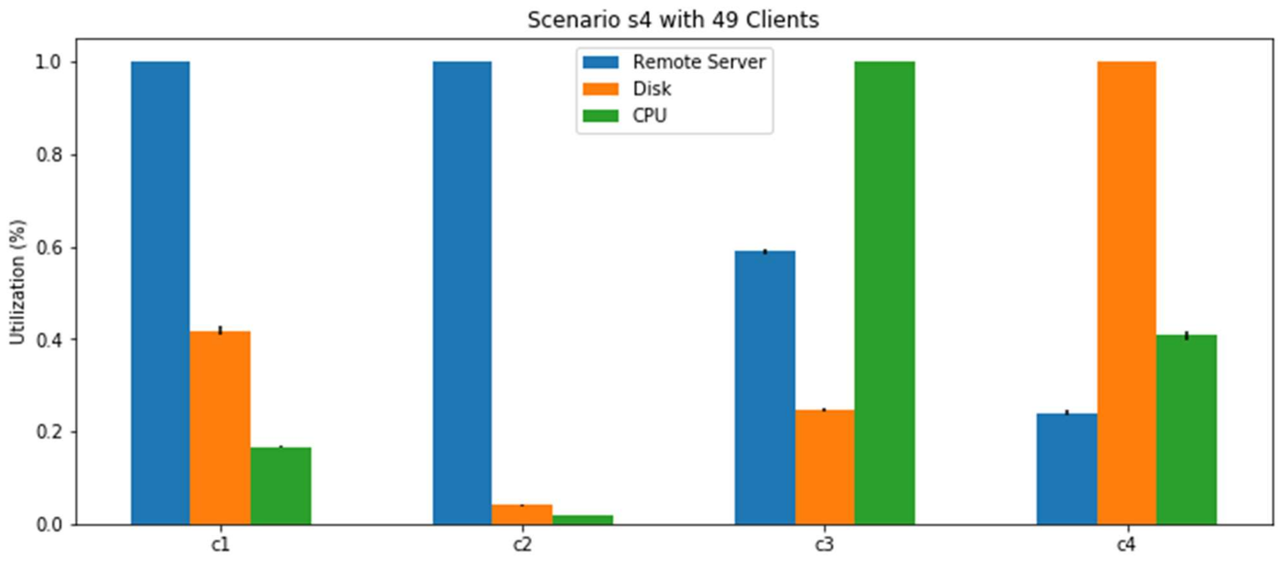


Figure 12 – Utilization

Comparison with the model

The simulations have been compared with the model that had been built with the Buzen's convolution algorithm. The comparison shows a consistency between the results obtained and the model.

Confidence Intervals

To compute confidence intervals, showed in previous plots, we have to test some hypothesis, depending on the statistic we're computing.

Performance Indexes

To compute the confidence interval of the performance indexes analyzed, the hypothesis of Independence and Normality, must hold. Since the simulation has been configured with different seeds for each repetition, and different RNGs for each random stream needed, the hypothesis of independence holds. The hypothesis that the sample is composed of Normal RVs has been verified with a visual test, using a QQ plot for each scenario and configuration. The confidence level used is 99%.

The following are respectively the best and the worst results from which we can infer that the normality hypothesis cannot be rejected.

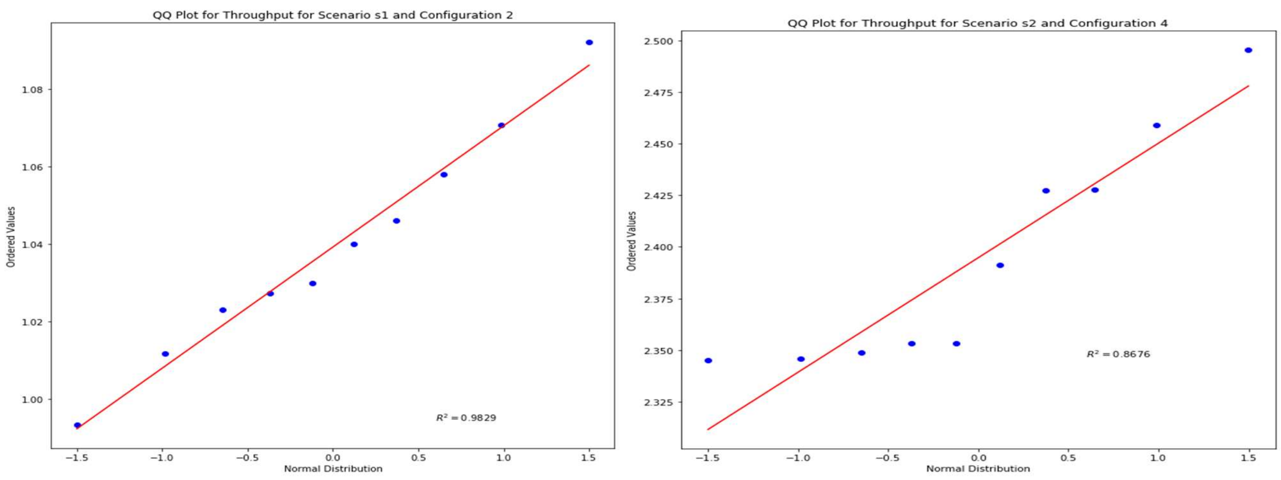


Figure 13 – Best/Worst Troughput

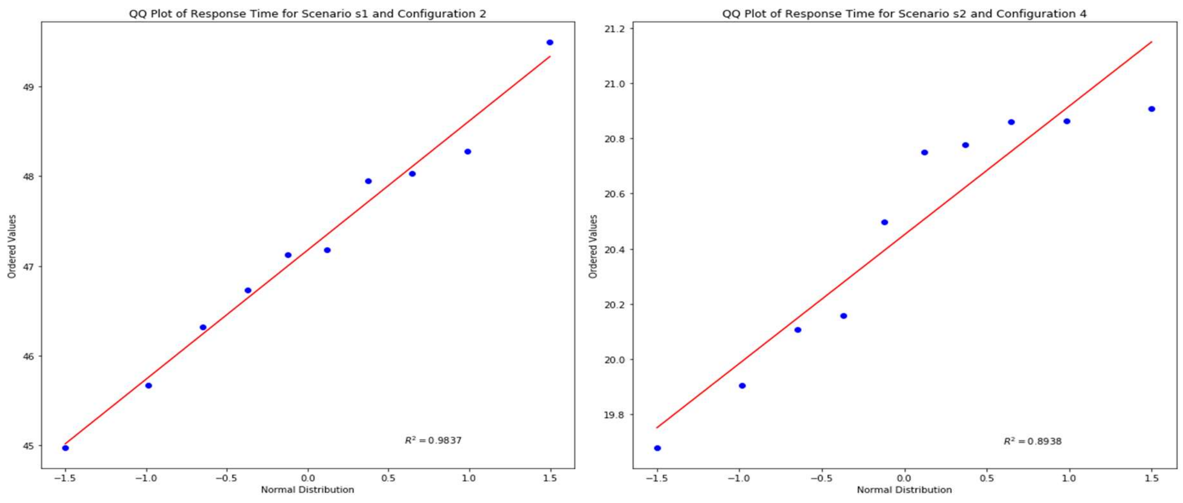


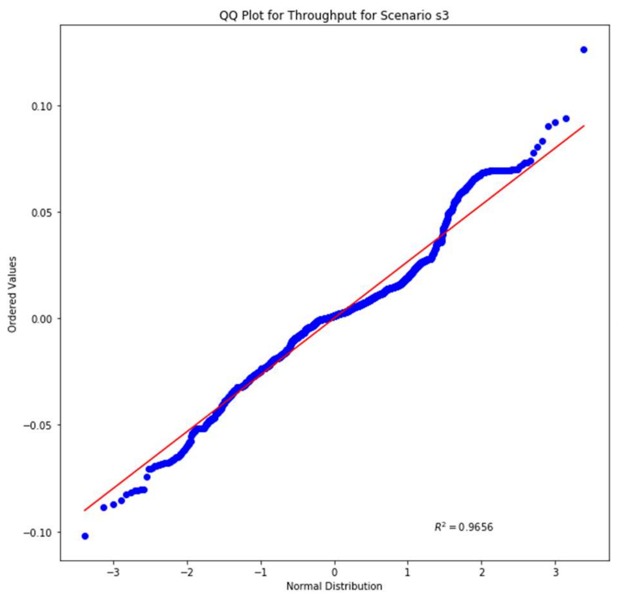
Figure 14 – Best/Worst Mean Response Time

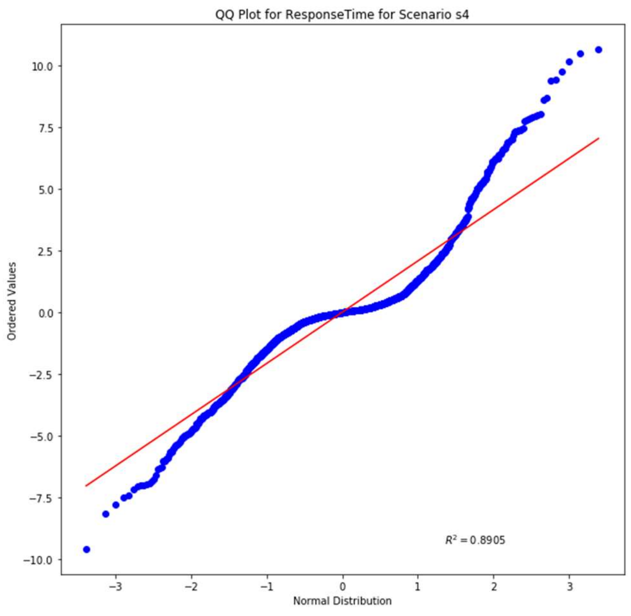
$2^k r$ Analysis

To assess the CI of the nonlinear regression model found with the *$2^k r$ analysis*, for the influence of the three SCs' mean service times on the throughput and the response time of the system we have to test three hypothesis on the residuals.

- Normality: Verified with the QQ plot against corresponding percentiles of a Standard Normal.
- Independence: Verified with the plot residuals vs the repetition number.
- Finite Variance: Verified with the plot residuals vs the predicted response.

The normality hypothesis holds, as we can see from the below graphs.





Residuals' independence can be tested from the visual inspection of a plot with residuals vs the observation number. The hypothesis holds since there is no visible trend.

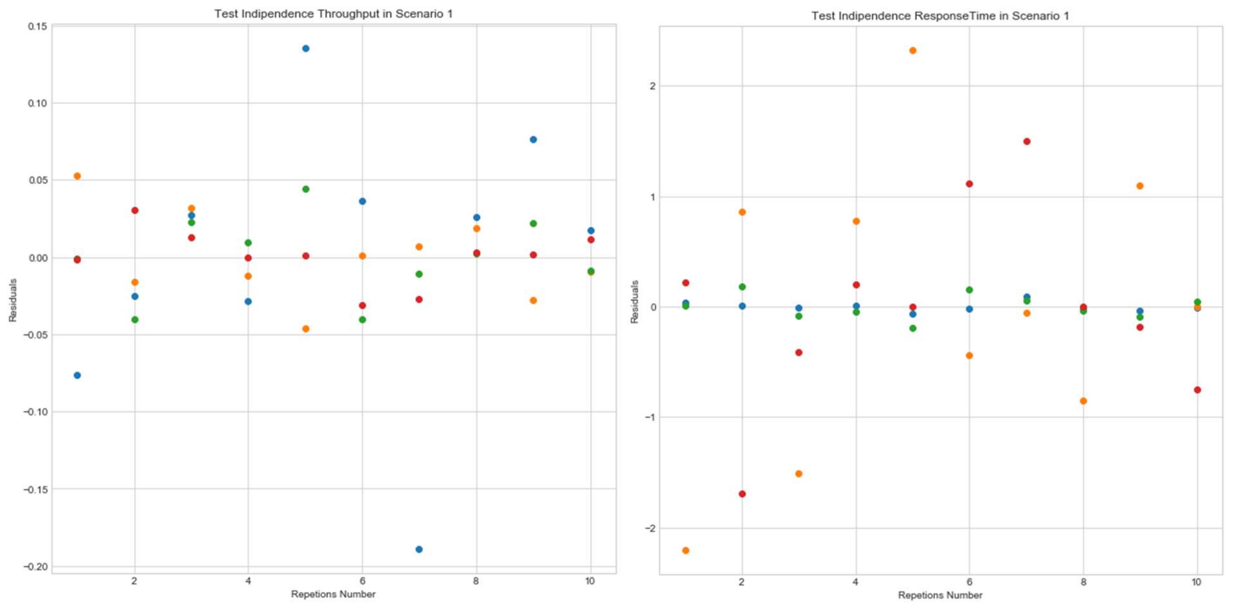


Figure 15 Testing independence

The hypothesis of finite variance has been tested via a visual test on the plot of residuals vs the predicted response. We cannot reject the hypothesis since:

- For what concern the throughput there is a constant spread of values, even if a slightly increase can be experimented for higher responses. But since the residuals are (at least) one order of magnitude smaller than the corresponding predicted response the hypothesis cannot be rejected.

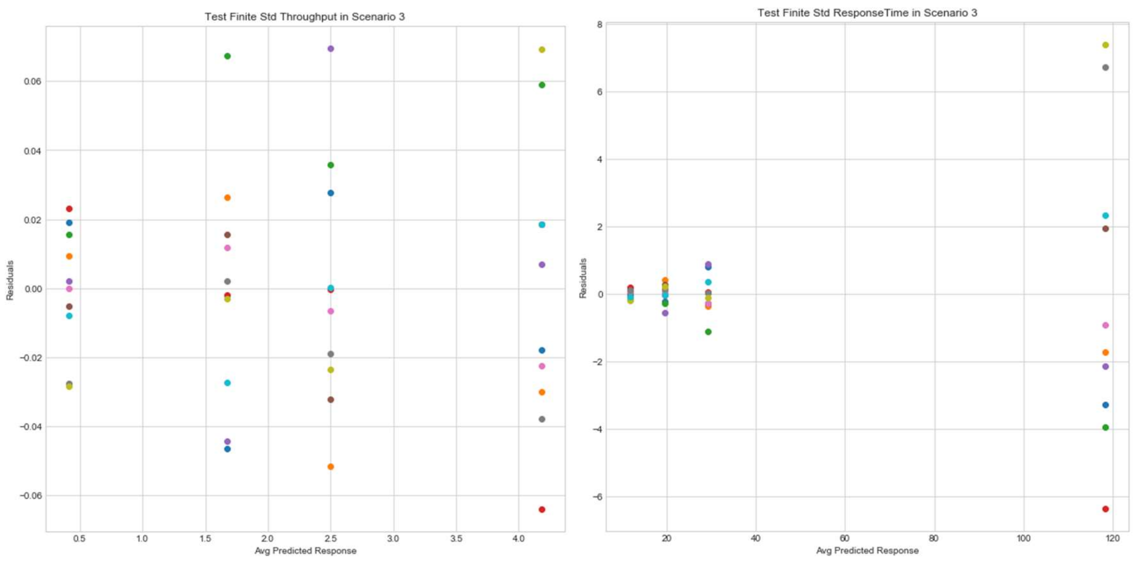


Figure 16 Testing Finite Variance

- With respect to the response time, there is a visible increasing tendency but also in this case the difference between the residuals and the corresponding predicted response is at least one order of magnitude, so the hypothesis cannot be rejected.

Final Considerations

From our analysis we can infer the following results:

From the point of view of throughput and response time, system’s performances are better when the processor has the lower service time and the others SCs have at most a service time one order of magnitude higher than the processor one. When remote server and disk have considerably different mean service time (one order of magnitude or two) the performance are worse.

From the point of view of the utilization, the SC with the higher response time has the higher utilization. In the case that there are more than one SC with highest mean service time, the application scenario makes the difference, in the sense that the highest utilization occurs at the SC with higher forwarding probability among those with highest mean service time.

Another thing that we can see from the utilization graphs showed before is that with 49 clients, that we have chosen as maximum number of clients possible in our system, there is always a SC that reach an utilization near to 1. Thus this (or these) SCs are the SCs where queueing occurs more likely, and as shown from the model, where there is the maximum mean number of clients. Thus, the throughput curve has a decreasing slope, with an increasing number of clients, and finally stabilize around a maximum value that is reached when the utilization of almost a SC approaches 1.

In the following graph we can see how the throughput is related to the utilization of the “bottleneck” SC.

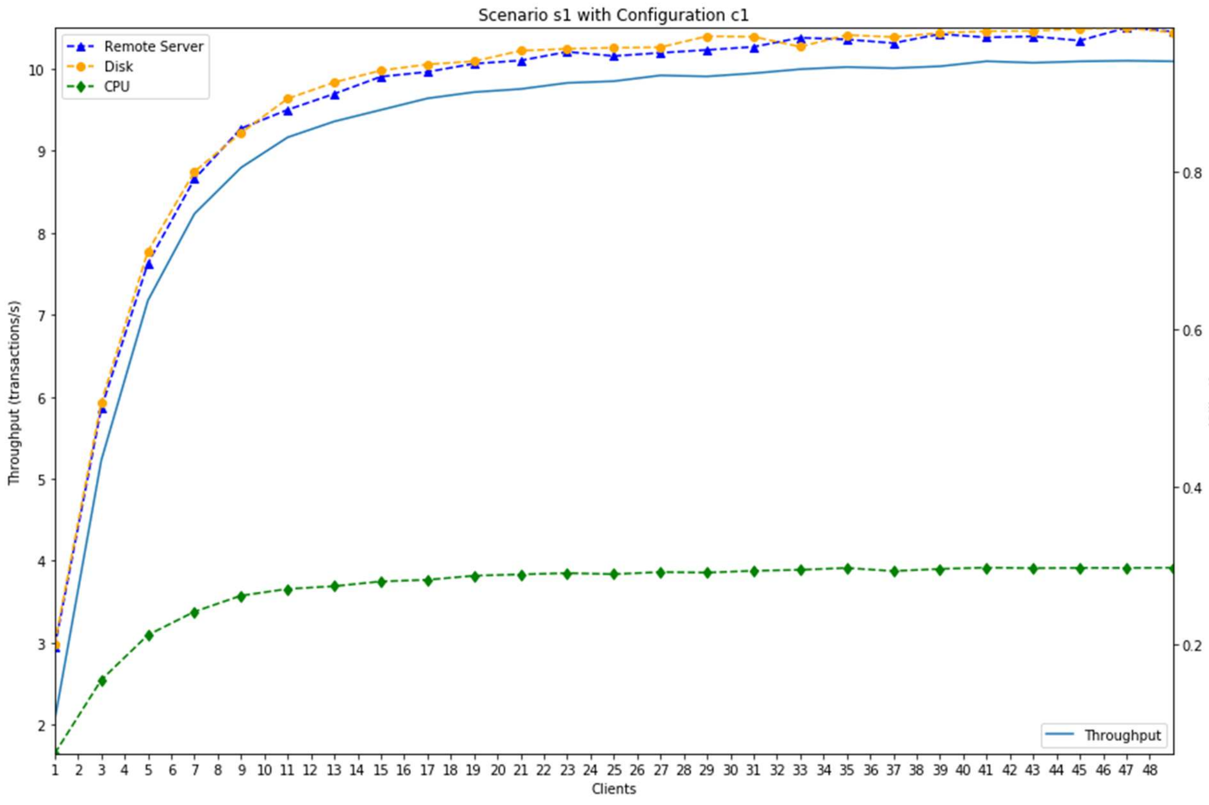


Figure 17 - Relationship between Utilization and Throughput in one experiment

To have a performance upgrade from the system we must first of all improve the bottleneck SC (the one with higher mean service time, or the one with the higher utilization), then when all SCs have more or less the same mean service time, the best thing to do is to improve the processor, as we can see from the utilization plot, the processor's utilization approaches one when the three SCs have similar mean service time, so queuing occurs mostly at the processor and we can't have a valuable performance return improving the others.

Considering the application scenario (the forwarding probabilities does change, with the application scenario), we can have a better performance return when improving the SC with higher forwarding probability, i.e. an application with an high number of transaction involving the disk will have a better performance return when upgrading the disk.

Another point that must be considered is that the maximum throughput and minimum service time are strongly application dependent, clearly an interactive application will have an higher throughput then a batch one, and the same stands for the minimum service time, but what is application independent is the fact that the system must be composed of SCs with balanced mean service time in order to achieve the best performance possible.