UNIVERSITÀ DI PISA

MSc degree in Computer Engineering – A.A. 2016/2017

Performance Evaluation of Computer Systems and Networks

Project 11 – Multiprogrammed Server

Mauro Abozzi

Stefano Cicero

Matteo Luciani Brunini

Simone Tavoletta

Table Of Contents

1. Introduction

This report aims to show the results obtained from the study of a multiprogrammed system represented by a server that manages different clients. The server is composed of a central processing unit that processes the requests, from a disk and an interface through which it makes query requests to a remote web server. The requests are processed one at a time by each service center through a first-come, first-served policy. With a certain probability, a job gets out of the server, accesses the disc or makes a request for remote query. Finally, when the request has been served, the server will send the response to the relative client, which will subsequently forward a new request.

In order to collect statistical data for the performance analysis, it has been simulated the behavior of the system using the simulation tool OMNeT++. MS Excel has been used to create graphs and for the computation of the confidence intervals.

Below is reviewed the structure and implementation of the system with the simulation software. Then, it will be presented an analysis of system performance and the results obtained will be summarized.

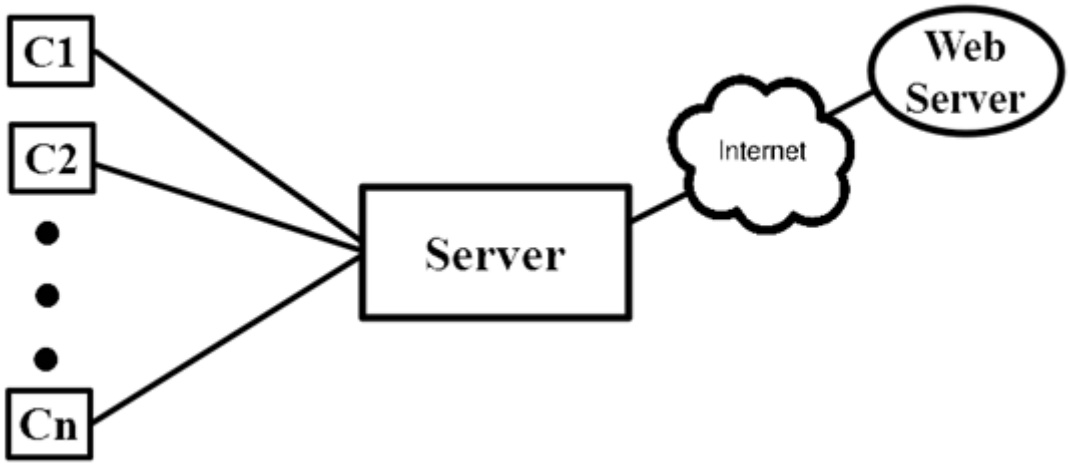

Fig.1 - System Design

2. Model

2.1 Theoretical Description

The server provides service to N concurrent clients, whose requests will always require some processing time as a first step. After the processing has occurred:

- ➔ with a probability **p1** the transaction is terminated and a reply is sent to the client;
- ➔ with a probability **p2** a disk access is required;
- ➔ with a probability **p3** (1-p1-p2), a remote query is issued;

In both cases p2 and p3, a new processing will then be required.

The processor, the disk and the remote web server will handle one request at a time, in a FIFO order. Processing, disk access and remote query service rates μ_1, μ_2, μ_3 are exponential IID RVs, and they will be different from one iteration to another, even for the same transaction.

The system is made of a constant number of clients which instantly require a new service whenever they receive a response, hence there is always a fixed number of service requests. For this reason the system could be represented like a Closed Jackson's Network (Fig. 2).

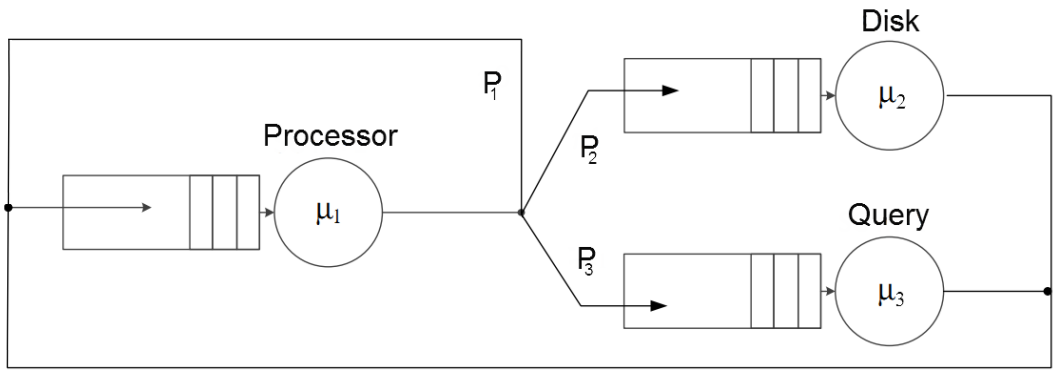

Fig.2 - Theoretical Model Design

2.2 Software Implementation

In order to implement the system described above, a single network called "Queuing Network" has been used. This network is made of different components which are implemented through classes of type *cModules*. These components are the following:

Client

Each client is represented by a *cSimpleModule*. It creates a new request message (type *Message*) to be sent to the server and waits a response from the latter before creating a new one. The RTT is 80ms.

Server

The main server is represented through a *cCompoundModule* made of two *cSimpleModule*:

- **Processor**: each time it receives a message, the processor will process the request and will determine one of the three forwards possible through an uniform distribution with an interval 1-100;
- **Disk**: the disk receives access requests from the processor and a response is returned after a certain processing time.

Query

The remote web server is represented by a *cSimpleModule* called *Query* in order to identify the query requests issued from the server. When the web server has finished to process the request, a response is sent to the main server. The RTT is 80ms.

The messages are defined through a class implemented in a ".msg" file. They contain information such as the sender and the timestamp of their arrival into the server. Furthermore, each SC has a processing time represented by Exponential RVs with different means. The queues are managed using the *cQueue* class.

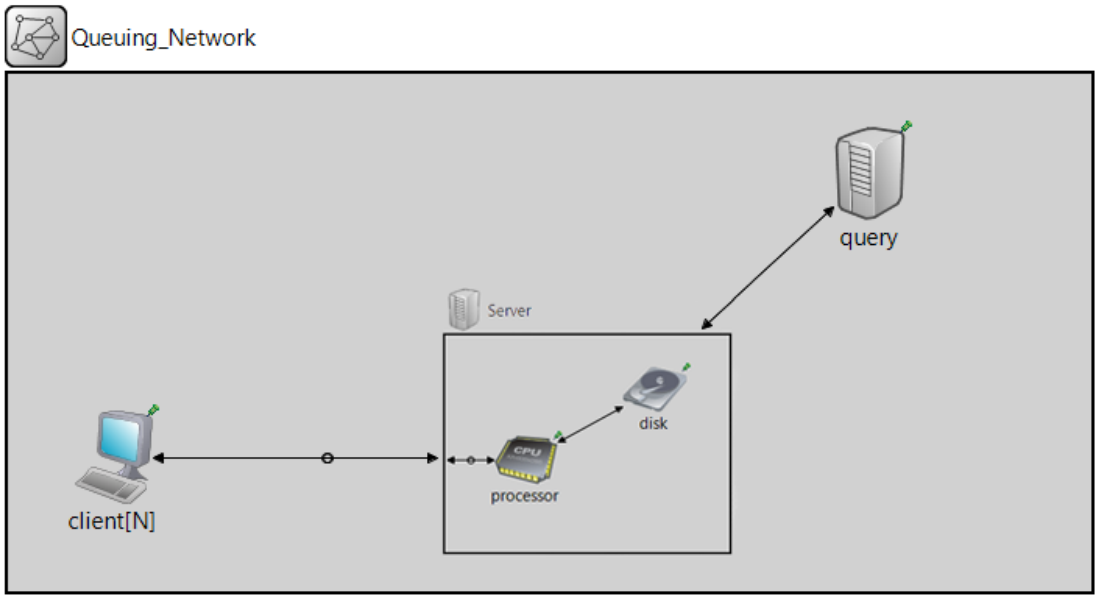

Fig.3 - Omnet++ Model

2.2.1 Configuration File

The ".ini" configuration file contains the parameters used during the simulation in OMNeT++:

- Service Time Mean for each SC
- Forwarding probabilities
- Number of clients
- Number of repetitions
- Simulation time
- Warm-up period

The values of the first three parameters are different for each scenario.

2.2.2 Statistics Collection

The performance indexes are:

- **Throughput:** calculated through the number of served requests divided by the simulation time;
- **Mean Response Time:** is the mean time it takes between the arrival and the departure of the same request;

- **Utilization Rate:** itis interesting to find out how long each SC is processing requests during the simulation time. The utilization rate, for each SC, is calculated summing up the processing time of the single requests and dividing it by the simulation time.

In order to collect the statistics data for these performance indexes, specific signals are inserted and managed within each component, excluding the ones performed during the warm-up period.

3. Experiment Design

3.1 Verification

In order to perform system verification, the limit values of the various parameters have been considered. For instance, the probability p1 has been set to 100%, p2 and p3 to 0% and viceversa. The obtained results have been compared with the expected values regarding the utilization of the different components, the system throughput and the mean response time, with everything resulting correct.

3.2 Calibration

In order to understand the maximum number of clients with which it is interesting to perform the simulation, the values that the service times and the probabilities assume in the different selected scenarios have been fixed in a heuristic manner. Using the recursive bisection method on different trials, it has been concluded that the system can be studied with a maximum number of clients equal to **20**.

3.3 Scenarios

The various scenarios have been chosen combining the different service time means of disk, processor and remote query. The values that these service time means can assume are 0.01, 0.1 and 1.0 (seconds) because it is preferred to have at least one order of magnitude of difference, as requested. Moreover, it has been chosen a scenario where the SCs have the same service time mean (0.1).
Given the high number of possible scenarios, the most meaningful ones have been chosen and they are reported below (SCs ordered by service time mean):

1. Processor = Disk = Query → (0.1 = 0.1 = 0.1)
2. Processor << Disk << Query → (0.01 << 0.1 << 1.0)
3. Query << Processor << Disk → (0.01 << 0.1 << 1.0)
4. Disk << Query << Processor → (0.01 << 0.1 << 1.0)

Each scenario includes a series of variants with different configurations of probabilities p1, p2 and p3, selected considering borderline cases that characterize the traffic within the system. It has been selected as extreme case the value of 70%, that is the probability of carrying out a particular operation, then distributing the 30% in the remaining.
The chosen configurations of probabilities are the following:

A. p1=70%, p2=20%, p3=10%
B. p1=10%, p2=20%, p3=70%
C. p1=10%, p2=70%, p3=20%
D. p1=33%, p2=33%, p3=34%

Intuitively it can be noted that, in cases where p1 is large there is a greater flow of jobs in output from the server. Conversely in the case where p1 is small, the requests tend to remain inside the server increasing the client waiting time. The above-described configurations, with the related scenarios will be analyzed on varying the degree of multiprogramming.

3.4 Warm-up and Simulation Time estimation

As first thing in the study of a queuing network, it is necessary to identify the warm-up period of the system. To do this, they have been simulated the chosen scenarios with the maximum possible number of clients (it has been seen that the warm-up period increases with the increase of the number of clients). From these simulations, the graphs, representing the response time vector (one point for each served message), have been obtained, carrying out a careful analysis on them. It has been taken into account the worst case, which has a warm-up time of 800s (Fig. 4). Considering this value, it has been chosen a simulation time of 4800s.

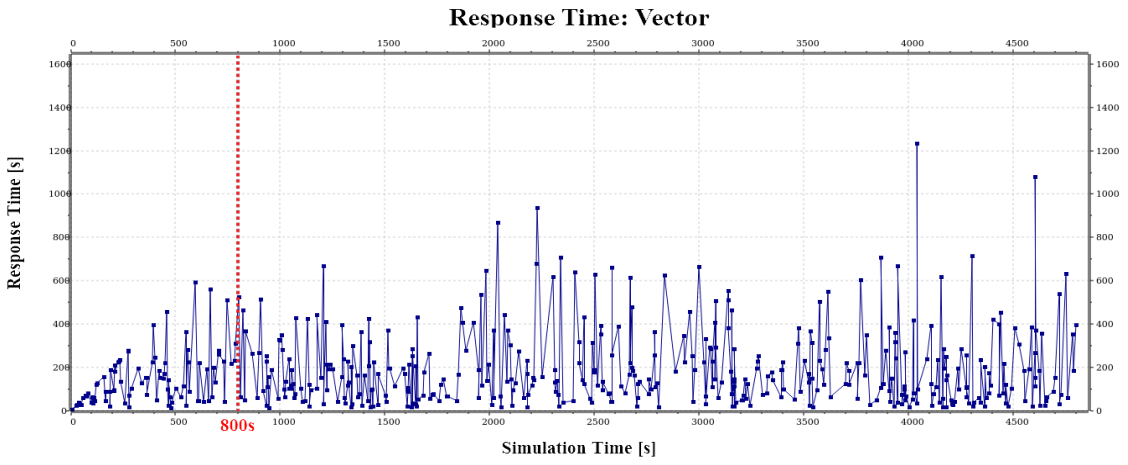

Fig.4 - Response Time Vector

4. Performance Analysis

The analysis is carried out on the various scenarios compared between each other, with the selected probability configurations (as defined in the chapter 3.3). In the analysis, all the performance indexes have been computed by considering 10 simulation repetitions. The considered values for the utilization rates are related to those obtained with the maximum number of clients (20). The graphs will report the mean value between the 10 repetitions made and will show all selected scenarios.

4.1 Case A

In this case the probabilities are p1=70%, p2=20% and p3=10%.

Throughput

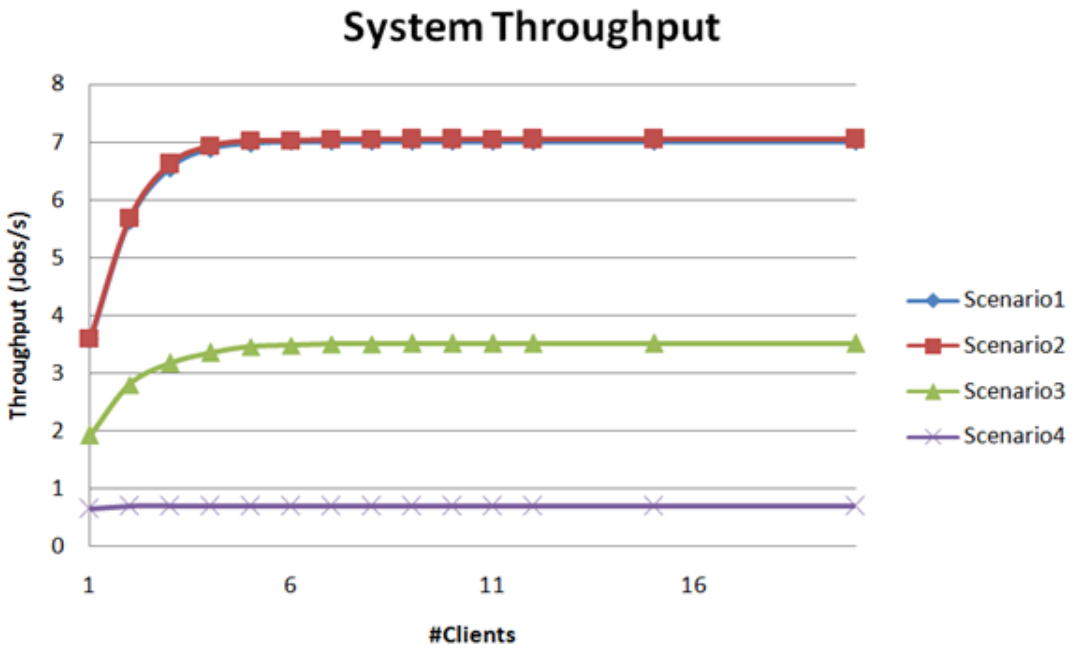

Fig.5 - Case A - System Throughput

From the graph, it is clear that the maximum throughput values are obtained in both the scenarios 1 and 2. This happens because the elaboration speed of the processor is the highest or equal with respect to the other components. For this reason, the first scenario is in a balanced situation because of the same service time mean between all the devices, while in the second one the higher processing speed of the CPU will be compensated by the slower pace in the elaboration of the remote query. The fourth scenario owns an extremely low throughput value because the biggest part of all the requests will not be served, instead it will stay in the processor's queue due to its very low speed.

Mean Response Time

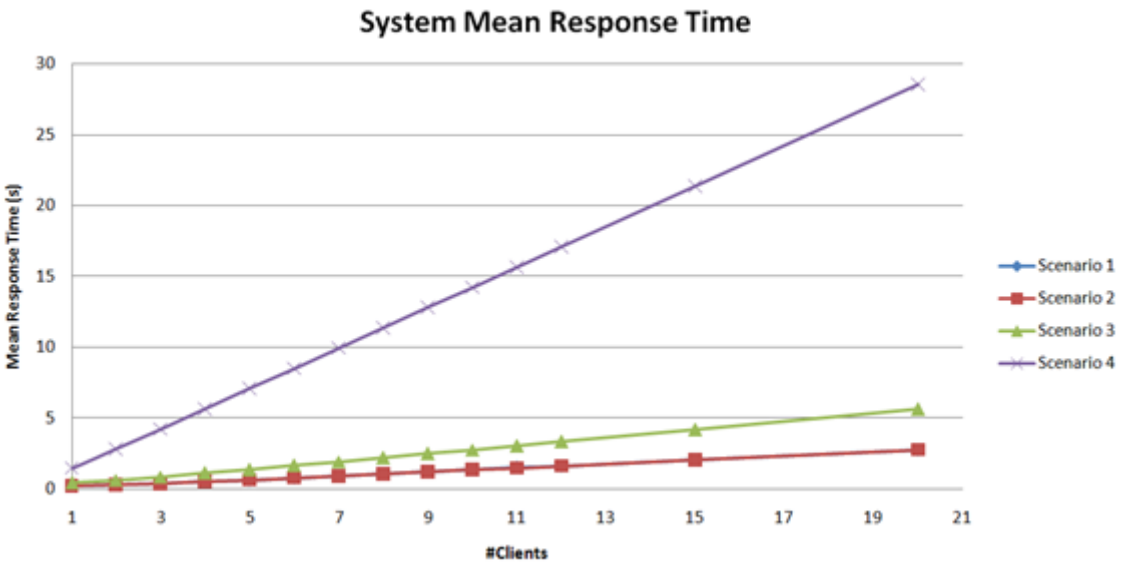

Fig.5 - Case A - System Mean Response Time

The system mean response time, as expected, increases linearly with the increasing of the clients. Coherently with what seen for the throughput, it is possible to notice how the fourth scenario proves to be the worst because of the low speed of the processor, while the first and second one are also here very similar, precisely because in that case the system is able to serve the requests quickly thanks to the higher CPU speed.

Utilization Rate

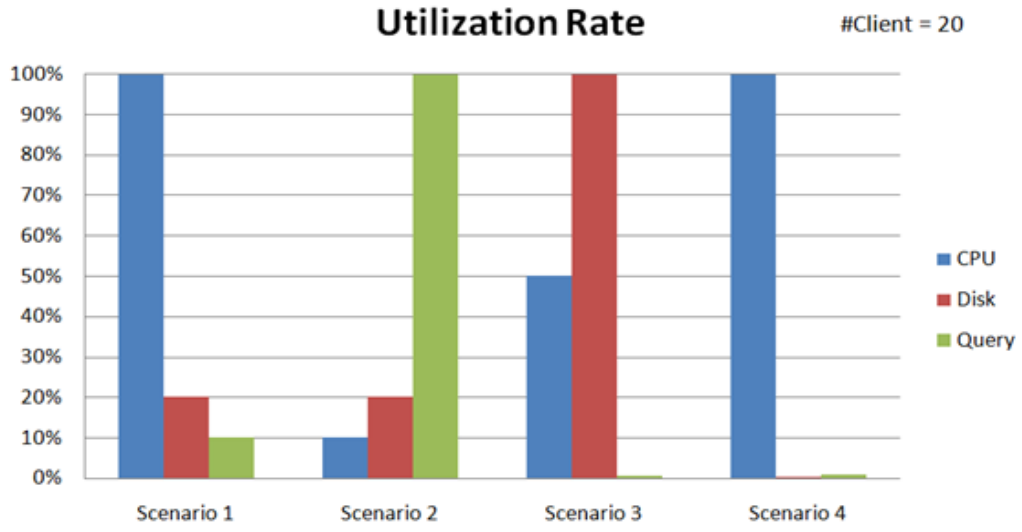

Fig.6 - Case A - Device utilization rate

By observing the graph, it can be seen how the scenarios 2, 3 and 4 will present a bottleneck in the component which has the lower speed. In the first case, instead, the processor depicts the bottleneck since each request, at its arrival, has to make at least one access to it (as it is in fact the only access and exit point of the system) and the service time mean is the same for all components.

4.2 Case B

In this case, the probabilities are p1=10%, p2=20% e p3=70%.

Throughput

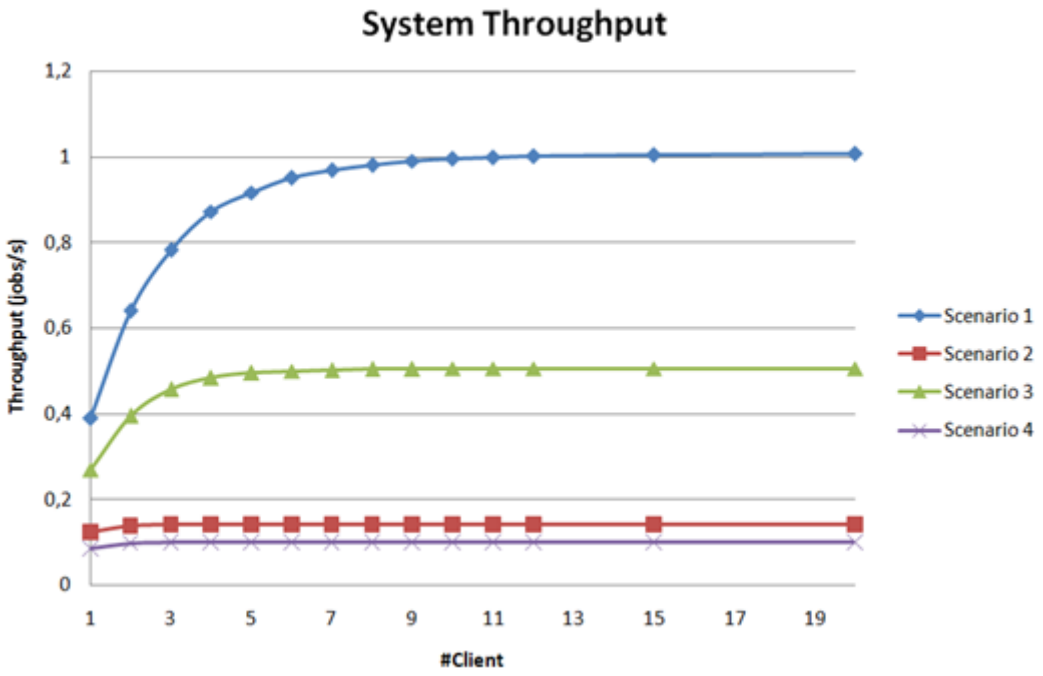

Fig.7 - Case B - System Throughput

The graph shows that the highest throughput value can be obtained in the first scenario, in which the service time means are equal for each service center. It is indeed possible to notice how the second scenario presents a lower throughput because the query, which has an higher service time mean with respect to the first scenario, will result as the component which has to serve a lot of requests because of the high value of p3. As regards for the other scenarios, it is possible to notice how the throughput in the third scenario is in the average because of the higher speed granted to the remote query, which is thus able to support the load assigned to it, while in the fourth scenario the processor results to be too slow to dispose of the received requests, despite having a medium-fast query.

Mean Response Time

From the analysis of the system mean response time, it was noticed that also here it results linked to the throughput and, consequently, the graph still shows times that increase linearly with the increasing of the clients, with a smaller response time in the cases where the throughput is bigger. For this reason, it does not introduce significant information for performance analysis and thus will no longer be reported, but they are presented in the delivered files.

Utilization Rate

As regards the analysis of the bottleneck, it was noticed that also in this case it was identified by the device with the lower speed with respect to the others. The first scenario presents the bottleneck in the processor for all the already mentioned reasons in the previous case. Therefore, also this analysis will no longer be reported in the next cases.

4.3 Case C

In this case the assigned probabilities are p1 = 10%, p2 = 70% and p3 = 20%. From the analysis it has in fact been noticed an entirely similar behavior to the previously seen case. The only noticed difference is that here the scenarios 2 and 3 are exchanged from the point of view of the throughput due to the inversion of probability between disk access and request for remote query.

4.4 Case D

In this case, the probabilities are p1=33%, p2=33% and p3=34%.

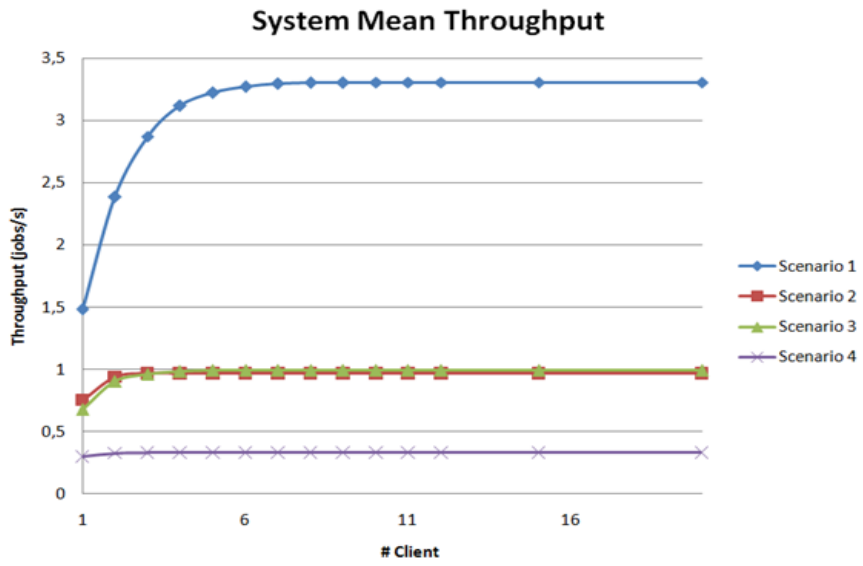

Fig.8 - Case D - System Mean Throughput

Even in this latter case, the highest throughput value can be seen in the first scenario. It is possible to notice that the scenarios 2 and 3 are very similar because the disk and query's service time means are de facto exchanged. Indeed, in the third scenario even if the query processing is faster, it will be limited by the CPU. The fourth scenario continues to be the worst due to the slowness of the cpu.

4.5 Final Overview

The analysis shows how the first scenario is the one that performs better in all considered cases, by selecting intermediate service time mean values between those presented.

This happens because, given the structure of the system, it is convenient to select a situation in which each component has similar service times (considering the fastest one), being so sure that there is a better balance in the use of the devices, which therefore will affect very less on the performance of those linked to them. Indeed, all cases in which the service time means are of the same order of magnitude will converge to a behavior similar to that shown for this scenario.

Considering a scenery in which a component has higher speed than the others does not make substantial improvements compared to the first scenario. However, if one knows for sure that the majority of the received requests do not need to make disk accesses or remote queries, then it will be possible to increase the performance of the CPU in order to have an optimal throughput (and, consequently, a lower response time). On the contrary, a value of probability of p1 smaller than the others does not allow to improve the performance of the system even by introducing components that are faster than the processor, being the latter anyway the system access point. In general, it will be still appropriate to ensure that the performance of the processor are not lower than those of the other devices (an example is given by the scenario 4, in which a very slow CPU limits the system performance drastically).

5. Confidence Intervals

Now it must be proved the correctness of the measurements made. To do this, it is necessary to compute the confidence intervals for these values. The formula to use is based on the assumption that the sample population is normally distributed.

$$\left[\overline{X} - t_{n-1,a/2} \cdot S/\sqrt{n},\ \overline{X} + t_{n-1,a/2}\, S/\sqrt{n}\right]$$

In order to apply this approximation, it would be necessary to perform a number of repetitions of the experiment of at least 30. In this case, however, only 10 independent repetitions of the same experiment are made. For this reason, it is necessary to check by QQ plots if the sample population can really be brought back to a normal distribution. As shown in the figures below, this assumption is valid because the points are aligned in a particularly evident way.

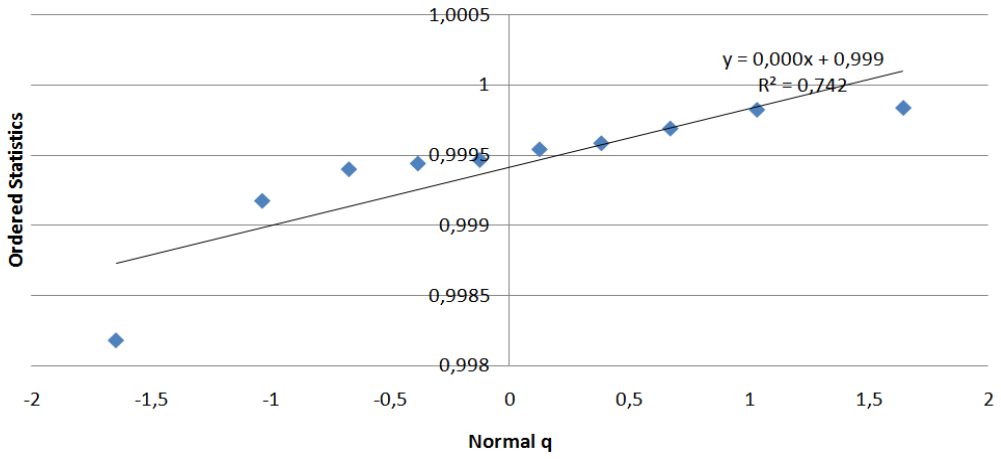

Fig.9 - One of the worst QQ Plots

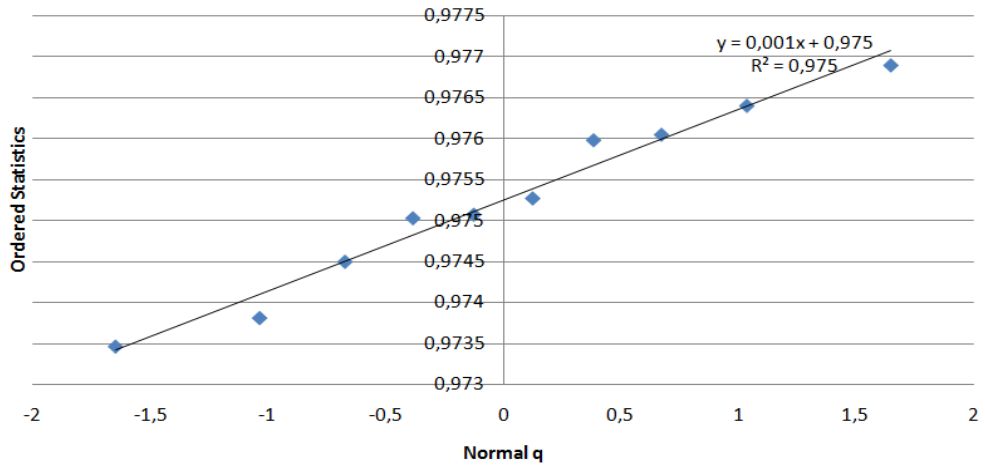

Fig.10 - One of the best QQ Plots

With these data, it has been selected a confidence interval with accuracy (1-α) = 0.99, from which it results $t_{\alpha/2,(n-1)} = 3.25$, with n = 10.

6. Conclusion

This project involved the analysis of a multiprogrammed system, made of 3 components, which serves requests generated by a certain number of clients. More specifically, the throughput trend has been studied through the various scenarios within a growing number of clients. It has been noticed that the throughput tends to a maximum value, which is reached with a different number of clients in different scenarios. This depends on the system's ability to balance the workload among the various components.

To better understand the system's behavior, the utilization rates of the single components and the mean response times were in fact analyzed. In general, the mean response time grows linearly with an increasing number of clients. For this reason the charts of the mean response time are made of straight lines with slopes influenced by the throughput values. Higher throughput values implies lower slopes.

The charts of the utilization rates allows to identify system's bottleneck which is the component with the highest utilization rate. In the case analyzed, the bottleneck is usually the slowest component. However, the processor is the only entry/exit point of the server, so it should be considered when you want to upgrade the system to obtain a better throughput. So, in order for the system to perform well, it is important that the CPU is not the slowest component of the system.

For example, one could imagine this system like a treatment plant with a single entry/exit point represented by a quality check for the products and with 2 possible type of treatments. At the beginning a quality check is issued and if a treatment is needed, the product is sent to one of the two responsible departments; otherwise it will be packaged for sale. After a treatment, a quality check is performed again. Obviously, the quality check speed will always affect the system performance, but also the treatment operations will do so if they are too slow compared to the quality control speed. So, to obtain better performance it is needed to upgrade the bottleneck component in order to have a speed similar to the quality check, while to get an optimal behavior an upgrade to the control mechanism will be also convenient.