UNIVERSITÀ DI PISA

Deficit Scheduler

PECSN project

Francesco Mione, Andrea Lelli, Leonardo Lossi

July 21, 2019

Contents

1 Introduction

The system we are going to analyze is composed by a vacation queue and a server which occasionally takes vacations, its service discipline is a *non-gated time-limited policy* with *multiple vacations*. The term *time-limited* refers to the fact that the server works on the queue only up to a limited amount of time during each turn. This time, referred as *turn time*, is initialized to the `Q` parameter (which is a fixed constant) and it is equal to `Q+D` in all the other turns. The quantity `D` is called *deficit* and it is the residual of the previous *turn time* not used by the service times `ST` of the Jobs served in the previous turn. The term *non-gated* refers to the fact that jobs that arrive while the server is serving the queue are candidates for service during the current turn as long as the *turn time* has not yet expired.
The jobs are queued and served following the FCFS policy. The server takes a vacation if:

- all the jobs in the system have been served;
- the *turn time* expires;
- the `ST` of the job which must be served according the FCFS policy exceed the residual of the *turn time*.

If the queue is empty when the server returns from a vacation, the server immediately takes another vacation, due to this fact we talk about a *multiple vacation* model.

2 Model

2.1 System Factors

- Turn length (`Q`);
- Inter arrival time (`IT`);
- Service time (`ST`);
- Vacation (`V`).

2.2 Components

JobProducer: this component produces Jobs and provides them a `ST` constant or exponentially distributed, then it sends them to the Server component;

Server: this component models a service center composed by a FIFO queue and a Server. It performs cyclically the following operations:

1. if the queue is empty it takes a vacation and return to this point otherwise it goes to point 2;
2. it picks the job at the head of the queue, if its `ST` is greater than the *deficit* it goes to point 3, otherwise the server serves it and returns to point 1;
3. if the `ST` of the picked Job is greater than the *deficit* it sums the *deficit* to the *turn time*, it takes a vacation and it goes to point 4;
4. if the `ST` is lower than the new *turn time* length the server serves the Job and then returns to point 1, otherwise it sums the new *deficit* to the *turn time*, it takes a vacation and it returns to this point.

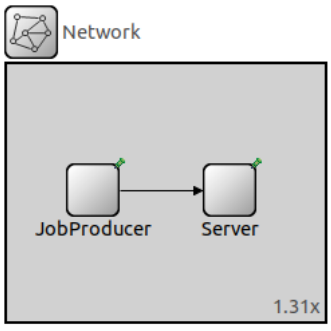


Figure 1: Model components.

2.3 Assumptions

In order to simplify the model of the system we made the following assumptions:

- the delay between the `JobProducer` and the `Server` is negligible;
- the FIFO queue on the `Server` has no losses (infinite buffer);
- the delay of the `Server` entering/leaving vacations is negligible;
- the costs in enqueueing/dequeueing and updating of the deficit counter are negligible.

These assumptions do not affect the final results of our analysis but are needed to let us focus on our purposes.
Under these assumptions we can model our system as follows (for the sick of semplicity from now on we consider the *constant scenario*): if the `ST` of the Job which must be served according the FIFO policy of the queue is greater than the residual of the *turn time* (or if the `ST` of the previous Job ended right at the end of the *turn time*), we consider the eventually multiple vacations that preced the service of this job as part of its `ST`.

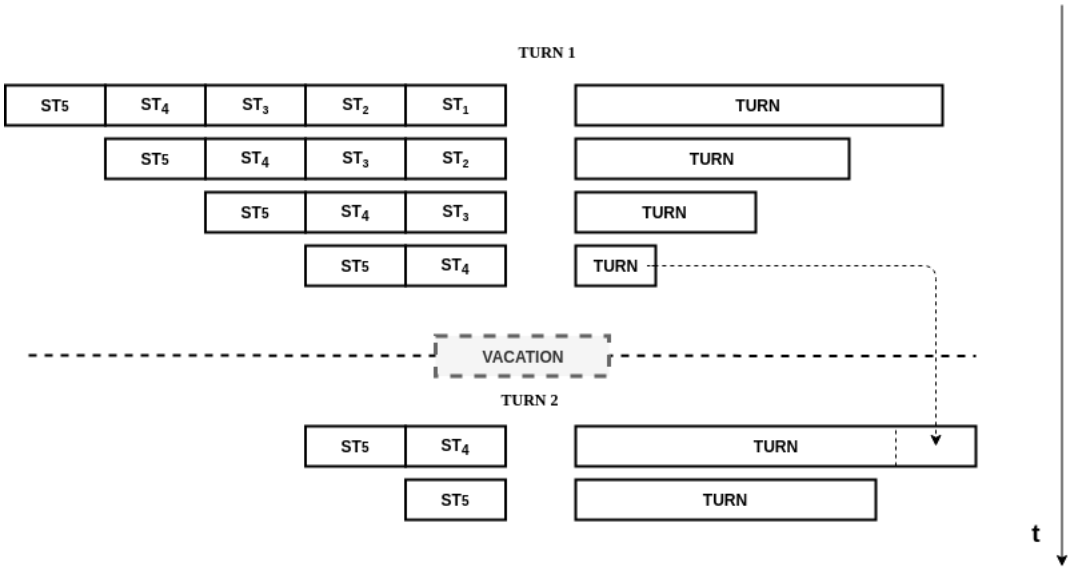


Figure 2: Constant Scenario: example of execution.

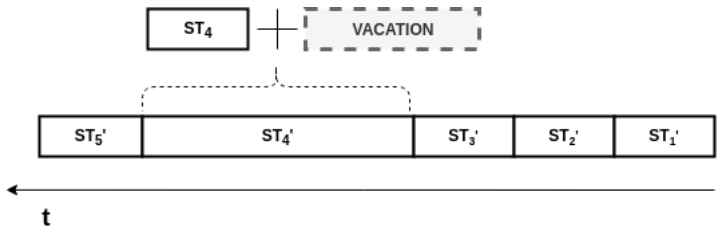


Figure 3: Constant Scenario: ST' definition scheme.

In other words, as we can see from figure 3, we have considered $ST'_4 = ST_4 + V$ as the real `ST` of the Job under service if it is executed after one or more vacations. According to this model we could remove the concept of *turn*.

Using some algebraic manipulations we obtained the following expression for the *constant scenario*:

$$E[ST'] = ST + \frac{ST}{Q} * V \tag{1}$$

where the ratio `ST/Q` is the mean number of vacation that precedes the service of a Job. Thanks to the *additivity* property of the *Expectation* operator E[ ] and the independence between `ST` and `V` this consideration can be extended to the exponential scenario obtaining:

$$E[ST'] = E[ST] + \frac{E[ST]}{Q} * E[V] \tag{2}$$

According to this consideration we could formulate a simple stability condition which states a lower bound for the `IT`, that is:

$$IT \geq E[ST']$$

The same result could be obtained simply thinking that the number of Jobs that arrive in a `Q+V` period must be lower than the one of Jobs served in a turn, i.e.:

$$\frac{(V+Q)}{IT} \leq \frac{Q}{ST}$$

$$(V+Q) * ST \leq IT * Q$$

$$IT \geq ST + \frac{ST}{Q} * V = E[ST']$$

2.3.1 Model Validation

In order to validate the model we performed some tests:

- **No memory leak test**: we ran several simulations and we verified that no undisposed objects were present;
- **Average Service Time test**: we ran several simulations to test if the average `ST` computed from the data collected by the simulator is equal to the one computed according the formula above;
- **Stability test**: we ran several simulations in order to test the stability formula above, either in the constant and in the exponential scenario;
- **Job Loss and Throughput Test**: we ran several simulations to check if the number of the packets sent by the JobProducer is equal to the number of received packet by the server;
- **Continuity test**: we ran several simulations varying slightly our parameters between consecutive simulations in order to verify if there were anomalous changing between the related two consecutive outputs.

2.3.2 Average Service Time Test

Parameter	Value
ST	1s
V	${ 0.1s - 10s }
Q	${ 0.1s - 10s }
Limit sim_time	604800s

Table 1: Parameter Tuning for the Average Service Time Test

Note: this test has been performed also setting other values for the `ST` parameter, each one has shown results that were equal to the ones computed using the above formula. The infinitesimal differences are due to the fact that in case of Job arrival with Server in vacation and empty queue part of the vacation get lost.
Constant Scenario:

V	Q	IT	$E[ST']_{sim}$	$E[ST']_{formula}$
0.1	0.1	${ 0.5 ; 1 ; 2 }	1.999999	2
0.1	1	${ 0.5 ; 1 ; 2 }	1.09999	1.1
0.1	10	${ 0.5 ; 1 ; 2 }	1.009999	1.01
1	0.1	${ 0.5 ; 1 ; 2 }	10.999982	11
1	1	${ 0.5 ; 1 ; 2 }	1.999996	2
1	10	${ 0.5 ; 1 ; 2 }	1.099999	1.1
10	0.1	${ 0.5 ; 1 ; 2 }	100.999842	101
10	1	${ 0.5 ; 1 ; 2 }	10.999828	11
10	10	${ 0.5 ; 1 ; 2 }	1.999968	2

Table 2: Simulations' results Average Service Time Test in the constant scenario.

Exponential Scenario:

E[V]	Q	E[IT]	$E[ST']_{sim}$ 99% CI	$E[ST']_{formula}$
0.1	0.1	${ 0.5 ; 1 ; 2 }	[1.9944 ; 2.0026]	2
0.1	1	${ 0.5 ; 1 ; 2 }	[1.0975 ; 1.1018]	1.1
0.1	10	${ 0.5 ; 1 ; 2 }	[1.0077 ; 1.0118]	1.01
1	0.1	${ 0.5 ; 1 ; 2 }	[10.897 ; 11.035]	11
1	1	${ 0.5 ; 1 ; 2 }	[1.9951 ; 2.0043]	2
1	10	${ 0.5 ; 1 ; 2 }	[1.0982 ; 1.1027]	1.1
10	0.1	${ 0.5 ; 1 ; 2 }	[99.1430 ; 103.3845]	101
10	1	${ 0.5 ; 1 ; 2 }	[10.9021 ; 11.0724]	11
10	10	${ 0.5 ; 1 ; 2 }	[1.9928 ; 2.0119]	2

Table 3: Simulations' results Average Service Time Test in the exponential scenario.

Note: The 99% CI in the case `V`=10 and `Q`=0.1 is larger than the other ones, this is due to the fact that a smaller number of jobs has been served, hence the sample width is smaller and the CI width increases according to the definition.

2.3.3 Stability test

After the confirmation of our model equation 2 we can verify our stability condition, i.e.

$$IT \geq ST'$$

Constant Scenario:

In the constant scenario we have verified the stability configuring `IT=ST'` and `IT-0.001s=ST'`.

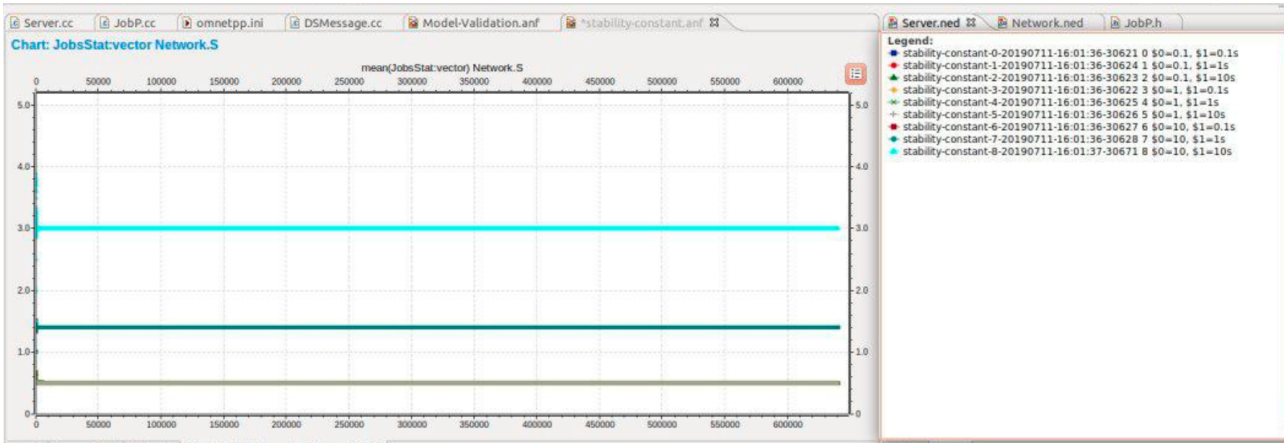


Figure 4: Constant Scenario: stability condition $IT = ST'$.

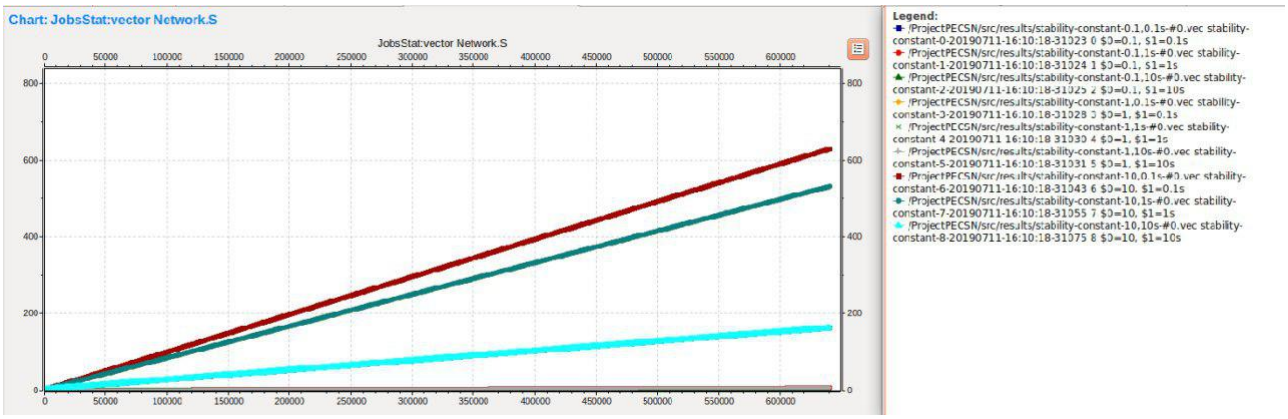


Figure 5: Constant Scenario: stability condition $IT < ST'$ (i.e. $IT - 0.001s = ST'$).

Exponential Scenario:

In order to verify the stability condition we ran several simulations with the parameters configuration which represents the worst case for the enqueueing.

Parameter	Value
E[ST]	1s
E[V]	${ 10s }
Q	${ 10s }
Repetitions	10
Limit sim_time	640800s

Table 4: Parameter Tuning for the Stability Test in the Exponential Scenario

For the sick of simplicity from now on we will call ρ the ratio

$$\rho = \frac{E[ST']}{E[IT]}$$

As in the previous case we have firstly considered the case E[IT]=E[ST'], so in the following plot we consider the case $\rho = 1$.

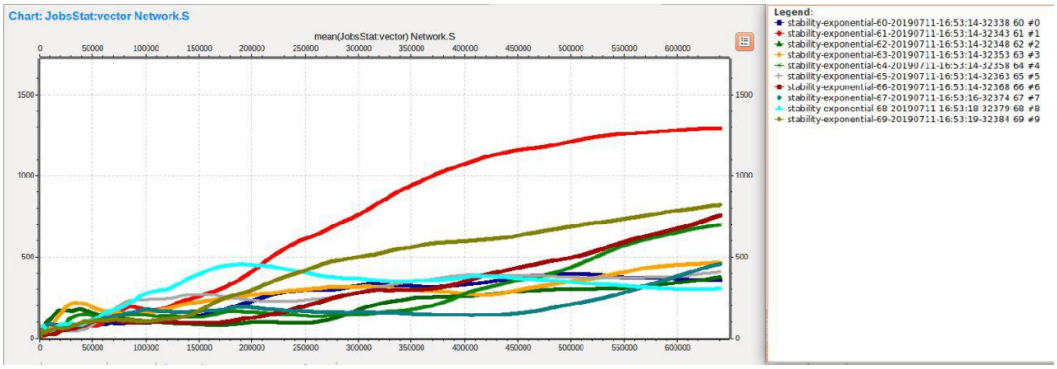


Figure 6: Stability Condition E[IT]=E[ST']

As we can see from the figure 6 the system is unstable and, as we can find in the literature, this is due to the fact that the system is *null recurrent*.
So we have concluded that in the exponential case we needed to force the following condition

$$E[IT] > E[ST']$$

Note that now the inequality has a strictly greater as logical operator.
The following graph represents the results of the simulations that have been performed forcing $\rho = 0.8$ (even if we found that the system remains stable even with eventually greater values in the range $\rho \in [0.8, 0.9]$).

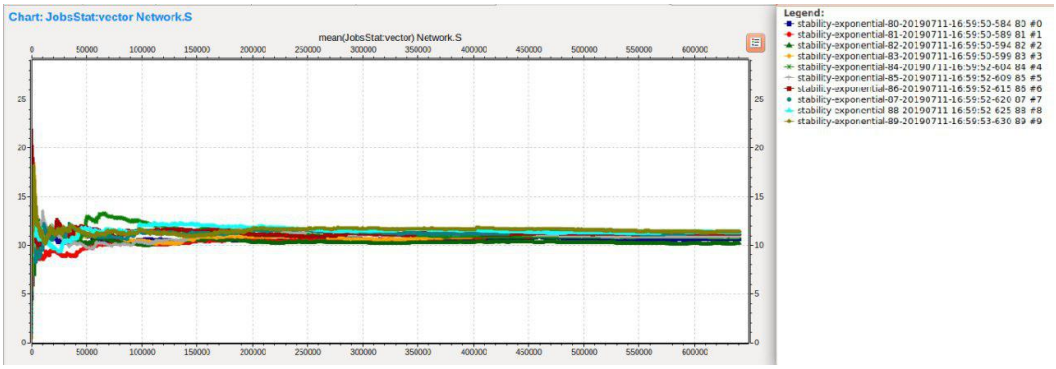


Figure 7: Stability Condition E[IT] > E[ST'].

2.3.4 Job Loss and Throughput Test

Parameter	Value
E[ST]	1s
E[V]	10s
Q	10s
ρ	0.8
Limit sim_time	604800s

Table 5: Parameters configuration for the Job loss and throughput test.

From the literature we know that if the system does not create or destroy jobs internally, the throughput of the system must be equal to the arrival rate λ.

Repetition	Jobs Received	Jobs In Queue	Jobs Served	λ	Throughput	Last Dep Time
0	256483	11	256472	0.4	0.4002	640799.88
1	256216	8	256208	0.4	0.3998	640799.16
2	255831	15	255816	0.4	0.3992	640798.3
3	256400	0	256400	0.4	0.4001	640793.68
4	255453	13	255440	0.4	0.3986	640793.83
5	256500	4	256496	0.4	0.4002	640784.31
6	256035	9	256026	0.4	0.3995	640793.08
7	255666	21	255645	0.4	0.3989	640799.57
8	255981	5	255976	0.4	0.3994	640798.3
9	256487	7	256480	0.4	0.4002	640797.48

Table 6: Job loss and throughput test results.

As we can see from the table 6 the system behaves correctly for both the Job loss and the throughput. The latter observation is confirmed by the fact that the throughput is almost equal to the arrival rate λ.

2.3.5 Continuity Test

Parameter	Values
E[ST]	1s
E[V]	${ 0.4s ; 0.5s ; 0.6s }
Q	${ 4.9s ; 5s ; 5.1s }
ρ	0.8
Repetitions	5
Limit sim_time	604800s

Table 7: Parameters configuration for the continuity test.

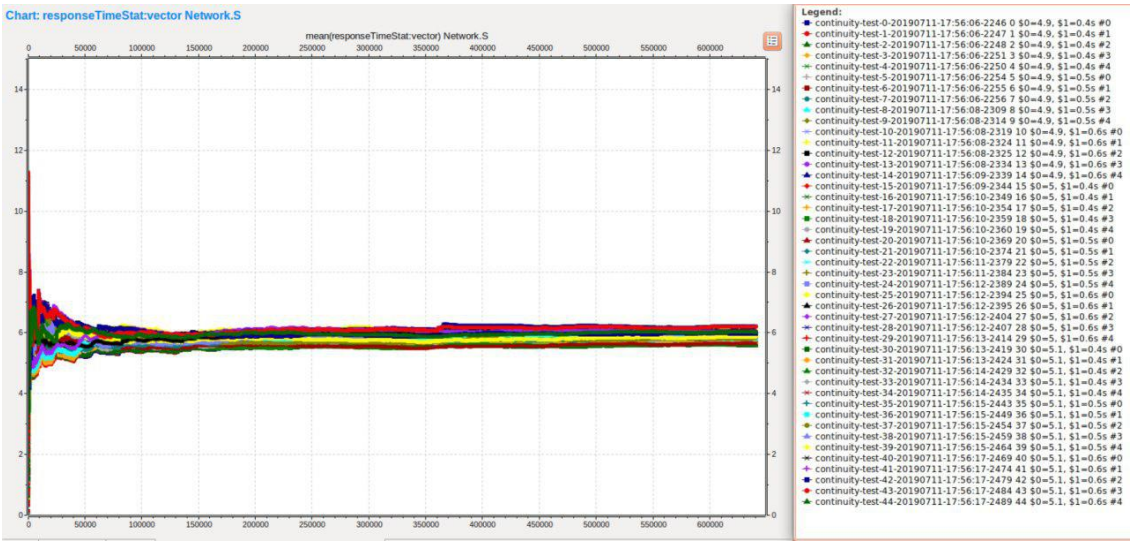


Figure 8: Stability Condition E[IT] = E[ST'].

As we can see from figure 8 the output of the system does not show big variation changing slightly the input parameters.

3 Factor tuning

Now we have to define and tune the parameters that can affect the results of the simulations, i.e. IT, ST, Q, V.

For the ST factor we have simply chosen a value arbitrarily in such a way that results of simulations would be easily computable: in our analysis we have set it to 1 second.

From the system requirements we know that the Q factor can assume values between 0.1 and 10 times ST, hence in our simulations it will assume values between 0.1s and 10s.

The V factor, i.e. the vacation duration, is strictly related to the type of real system we want to model, without any additional information we can only try to arbitrarily vary the V factor in the range [0,1*ST ÷ 10*ST] so, according to our choice to impose ST=1s, we have varied V in the range [0.1s, 10s].

For the IT factor we have seen from the model validation that the system is stable if:

- *constant scenario*: IT≤ST'
- *exponential scenario*: IT>ST'

So we have tuned IT as follows:

- *constant scenario*: IT=ST'
- *exponential scenario*: $\mathtt{E[ST]} = \frac{\mathrm{E[IT]}}{0.8}$

4 Warmup Analysis

In the constant scenario, the warmup period analysis is not needed beacause the steady state is immediately reached, the same does not hold for the exponential scenario.
The warmup analysis in the exponential scenario has been performed applying the stability condition found previously with $\rho = 0,8$ in the worst case, i.e. the one in which E[IT] assumes the maximum value, so the minimum number of jobs are served and the steady state is reached after the longest time.

Parameter	Values
E[ST]	1s
E[V]	10s
Q	0,1s
ρ	0.8
Repetitions	10
Limit sim_time	640800s

Table 8: Parameters configuration for the warmup analysis.

The response time plot for the 10 repetitions is the one represented in the following figure:

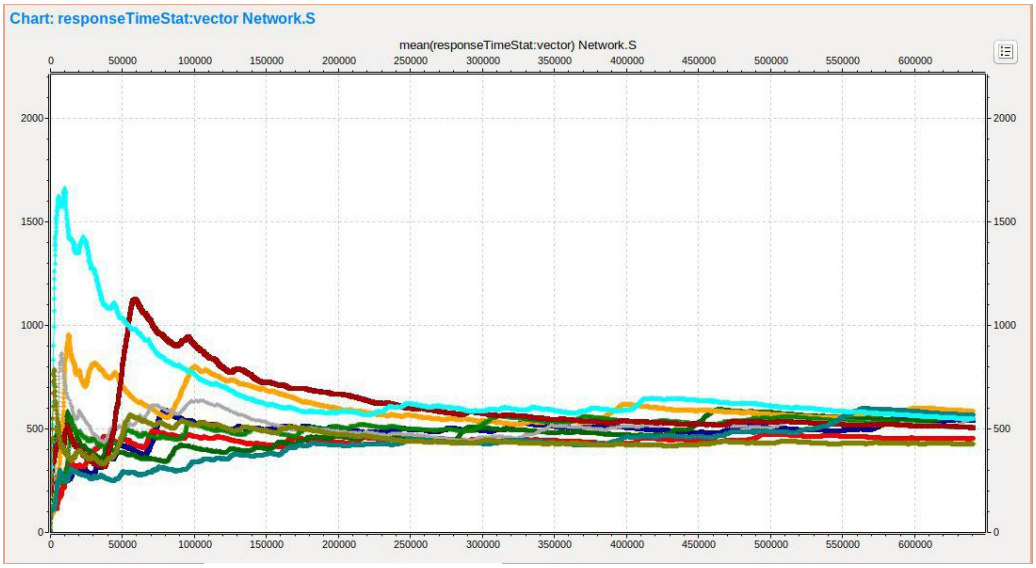


Figure 9: Warmup period with Q=0.1s and V=10s.

As we can see from figure 9 in the worst case we reach the steady state of the system starting from around 200000s, so we can use this warmup period value for any other parameters configuration.

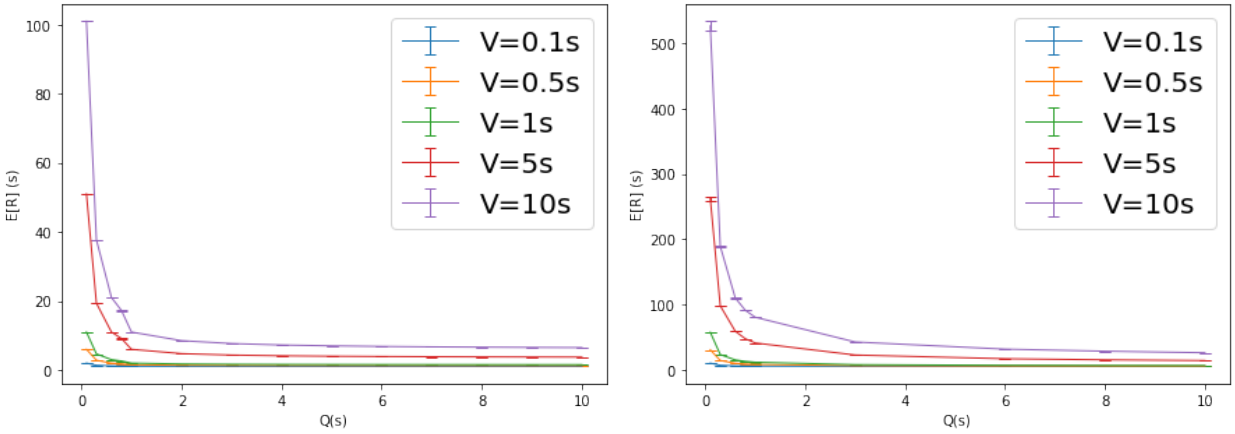
5 Scenario Analysis

5.1 Full Factorial Analysis

In order to evaluate the relative importance of our factors `Q` and `V` w.r.t. the response time we have performed a Full Factorial Analysis to see how the system behaves with different parameters configurations.

Parameter	Value
E[ST]	1s
E[V]	${ 0.1s ; 0.5s ; 1s ; 5s ; 10s }
Q	${ 0.1s ÷ 10s }
Limit sim_time	640800s
Exponential Scenario Parameters	
CI	99%
ρ	0.8
Repetitions	10

Table 9: Parameters Configuration Full Factorial Analysis in the exponential scenario.



(a) Constant Scenario

(b) Exponential Scenario.

Figure 10: E[R] Full Factorial Analysis.

As we can see from figure 10 starting from a particular `Q` value, the `RT` does not show tangible benefits as `Q` increases . Due to this consideration our analysis focused on the reason of this `RT` behavior and on the study of the `Q` value starting from which this `RT` behavior begins.

5.2 Constant Scenario

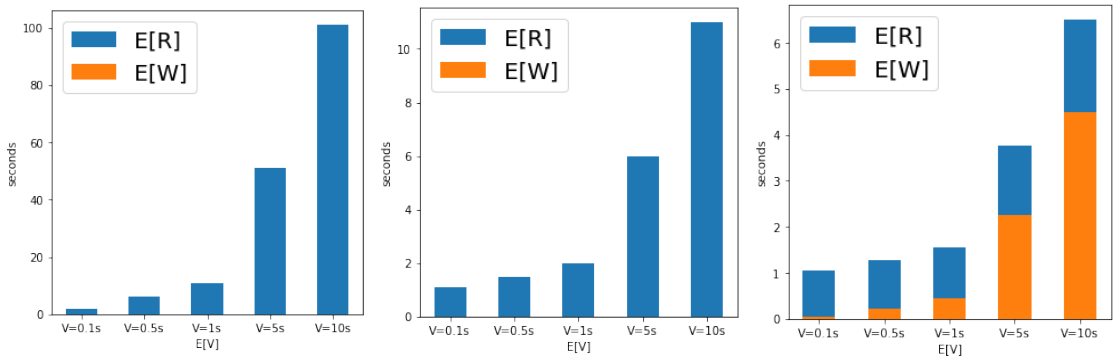
From the theory we know that:

$$RT = t_{departure} - t_{arrival}$$

We decided to analyze the composition of the response time as the sum of the waiting time in the queue and the service time as we have previously modeled (i.e. `ST'`):

$$E[RT] = E[W] + E[ST']$$

In order to establish the contributes of the two components E[W] and E[ST'] on the E[RT] we performed several simulations obtaining the following results:



(a) Q=0,1s (b) Q=1s. (c) Q=10s.

Figure 11: Constant scenario: E[R] Splitting.

Case Q≤ST:

Firstly we noticed that, since we have forced `IT=ST'` in order to make the system stable, in case `Q≤ST` there is no Job enqueueing, so the E[`RT`] is composed by only E[`ST'`]. Considering the cases in figures 11a and 11b we can state that:

$$E[RT] = E[ST'] = ST + \frac{ST}{Q} * V$$

We recall that the Job `ST'` is composed by `ST` and the vacations that eventually precedes its service.
While `ST>Q` each job will be preceded by a mean number of vacations equal to `ST/Q` before beeing served; this means that if `V` assumes high values a little increase of `Q` determines a smaller number of vacations before each Job and so large improvements in the `RT`.
This behavior can be seen looking at the second addend in the `ST'` definition: this addend constitutes the variable part of the `ST'` (so, in this scenario, of the E[`RT`]) and, as we can see from the formula, it will be larger as `V` increases and as the `ST/Q` ratio increases.

Case Q>ST:

When `ST<Q` we have to take into account also the Jobs enqueueing, so:

$$E[RT] = E[ST'] + E[W]$$

E[`ST'`] will continue to decrease according to the definition but as we can see from figure 10a, the values of E[`RT`] related to different `Q` values tend to decrease slower for any value of `V`.

The mean waiting time E[W] will be logically dependent from E[N_q] and E[`ST'`]. According to the graph, as `Q` increases, if E[ST'] decreases E[N_q] must increase. We have verified this fact performing simulations with different values of `Q` and we have plotted the related E[N_q] (*note:* this is the mean number of queued jobs according to our model, so when the system is in vacation the first Job is considered under service and not in the queue):

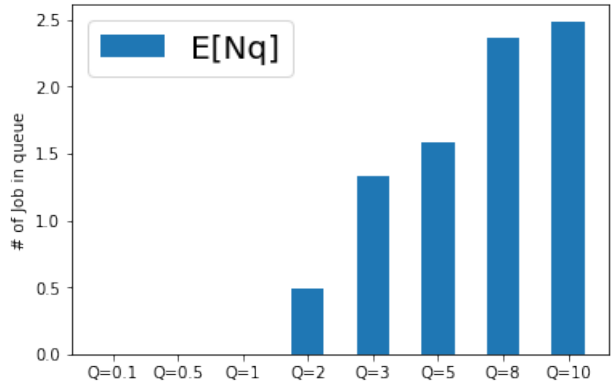


Figure 12: E[W] with V=10s.

Thanks to all these considerations we can state that, if the system maintain the same stability condition, i.e. the `IT` assumes the minimum value in order to keep the system stable, the `RT` will not show tangible improvements as `Q` increases when

$$Q > ST$$

This contraint will imply that in each turn at least one job will be served, or equivalently, any job will be preeceded by at most one vacation before being served.

5.3 Exponential Scenario

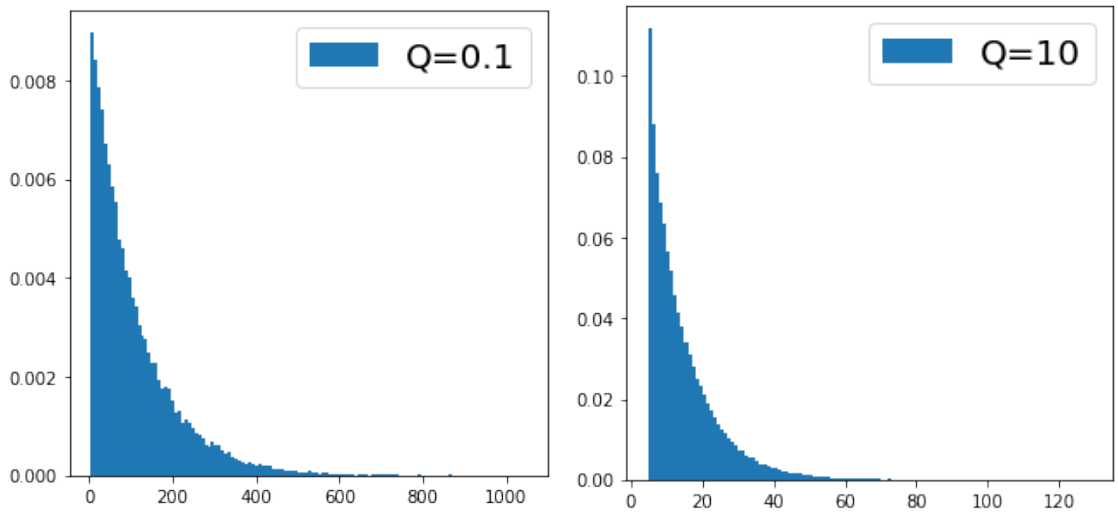
5.3.1 ST' Distribution study

If the distribution of the `ST'` had been an exponential one we could model our system as an M/M/1 with service rate equal to 1/E[`ST'`]. According to this consideration we decided to study the `ST'` distribution performing several simulations with the following parameters configurations in order to collect statistics and to plot the histograms to have some insights starting from its EPDF.

Parameter	Value
Repetitions	10
ST	1s
Q	${ 0,1s ÷ 10s }
V	10s
ρ	0,8

Table 10: Parameters configuration for the Memorylessness study of ST'.

We have chosen the maximum value for the `V` factor according our parameters tuning, i.e. `V`=10*`ST`, and we varied the `Q` factor exploiting all its range.
For all the simulations we have performed we obtained EPDFs with trends similar to the exponential distribution's one. In the following figures are represented the histograms obtained from the simulations in which the `Q` factor assumes the extreme values of its range.

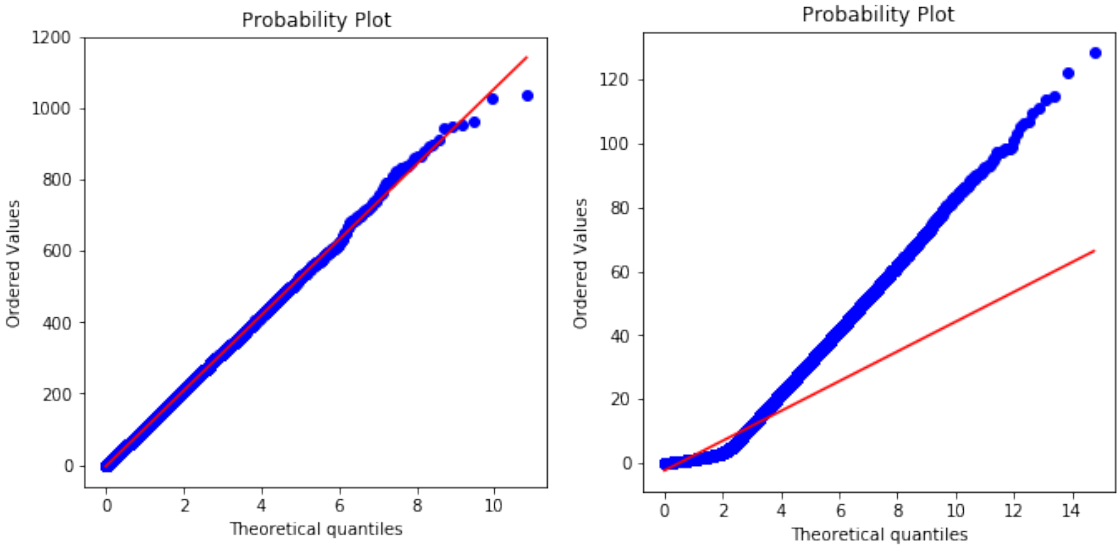


(a) Case Q=0,1*ST.

(b) Case Q=10*ST.

Figure 13: ST'Histograms.

In order to verify that the ST' distribution effectively was an exponential one we plotted the QQ-plots comparing it with the exponential distribution and we have computed the CoV for all the parameters configurations.



(a) Case Q=0,1*ST.

(b) Case Q=10*ST.

Figure 14: QQ-plot(ST', exponential distribution).

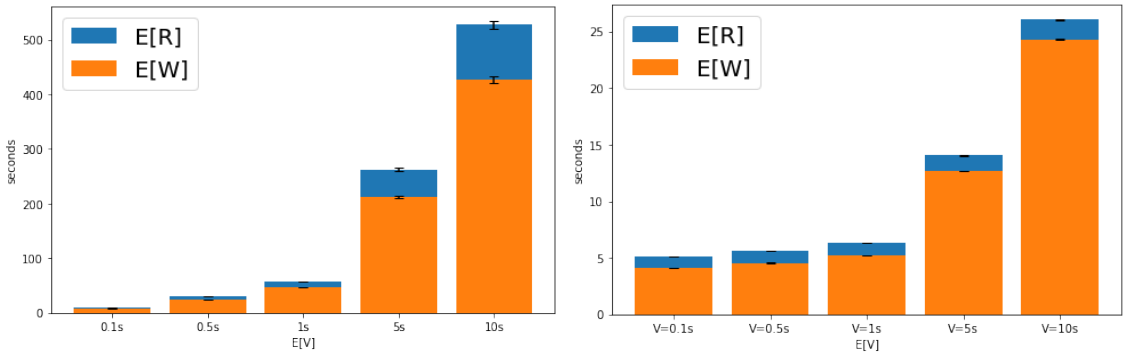
Q/ST	CoV
0,1	1,0375
0,3	1,1125
0,6	1,2087
1	1,333
3	1,765
6	2,084
10	2,2413

Table 11: CoV ST'.

As we can see from the results in figure 14 and in table 11 we can definitely exclude that the ST' distribution is not an exponential one, so we can state that our system, using the Kendall notation, is an M/G/1 one.

5.3.2 Response Time analysis

In the constant case we have seen that the tends of the mean `RT`s related to different `V` values tend to be similar when we guarantee that at least one Job is served within each turn. This last warranty (i.e. `ST`<`IT`) cannot be guaranteed in the exponential scenario because of the nature of the exponential distribution, in fact it can provide big numbers with non-null probability.
Also in this scenario we decided to analyze the composition of the mean response time, computed with 99% CI:



(a) Case Q=0,1s.

(b) Case Q=10s.

Figure 15: Exponential Scenario: E[R] Splitting.

As we can see from figure 15a due to the uncertainty of the exponential distribution we had to take into account the Job enqueueing even in the case `Q`≤`E[ST]`, so the response time is always equal to:

$$E[RT] = E[ST'] + E[W]$$

Recalling our E[`ST'`] definition, we know that `V` will affect more E[`ST'`] as the E[`ST`]/`Q` ratio increases.
Furthermore also in this case E[W] will be logically dependent on both E[`ST'`] and E[N_q]. For the last two consideration we can conclude that in the exponential case we obtained an E[`RT`] greater and more sensible to the `V` variations as `Q` decreases w.r.t. the constant case.

As further validation of the model we have checked that the values of E[RT]-E[W] (the blue rectangles heights) were equal to E[ST'] with positive results.
In order to quantify the `Q` value such that its increasing does not provide tangible improvements from the `RT` point of view, we need to perform a different study w.r.t. the constant scenario. We have seen that in the constant scenario this `Q` value is equal to `ST`, in other words the value which guarantees that each turn will serve at least one Job.
This last warranty cannot be guaranteed in the exponential scenario because we cannot guarantee that the `ST` produced by the exponential distribution is always lower than `Q`.
So firstly we have estimated using the relative frequency formula the probability that any turn serves at least one job through the simulator results: we called k the number of turns in which at least one Job has been served and with $N = 1'000'000$ the total number of turn elapsed in each of the simulations (we have retained 1000000 a sufficiently large number):

$$\hat{p} = k/N$$

Then we have computed the 99% CI for this probability using the CI for the success probability formula and we have checked if this probability corresponds to the probability that the ST is lower (recall that is produced randomly from an exponential distribution) than the `Q` considered:

Q	K	N	$\hat{p}$	CI 99% for $\hat{p}$	P$\{ST \leq Q\}$
0,1s	94544	1000000	0,09544	[0.0949, 0.0962]	0,09516
0,3s	257328	1000000	0,257328	[0.2562, 0.25845]	0,2591
0,6s	445717	1000000	0,445717	[0.444, 0.447]	0,4511
1s	622907	1000000	0,622907	[0.6217, 0.62415]	0,6321
2s	852784	1000000	0,852784	[0.8519 ,0.8537]	0,8646
3s	937981	1000000	0,937981	[0.9374, 0.9386]	0,9502
4s	969223	1000000	0,969223	[0.9688, 0.9697]	0,9816
5s	980842	1000000	0,980842	[0.9805, 0.9812]	0,9932
6s	985125	1000000	0,985125	[0.9848, 0.9854]	0,9975

Table 12: Probability that a turn serves at least one job.

The small differences between the theorical probability and the one computed from the simulator data are due to the limited number of turns N taken into account.
According to these results we can state that as `Q` value increases as the difference between probabilities related to consecutive `Q` values decreases. The latter means that as much as `Q` increases the `RT` will show negligible improvements, this is due to the fact that the probability to produce a Job with a `ST` greater than the Q stays approximately constant. This behavior can be seen from both the `RT` ECDFs and the mean values with 99% CI:

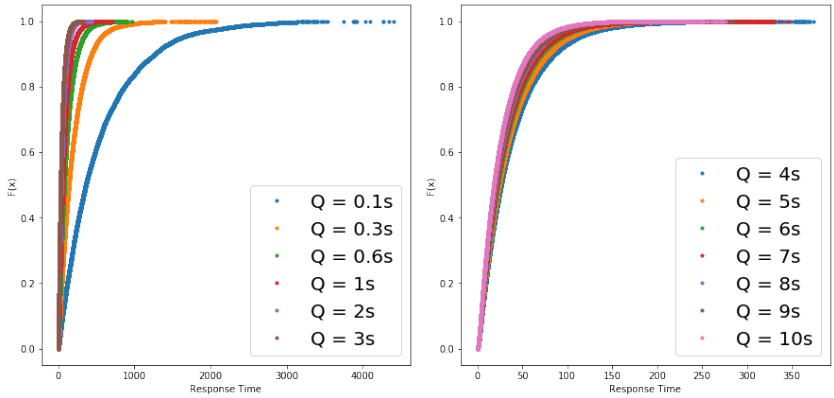


Figure 16: `RT` ECDFs.

Since `RT` is a "lower-is-better" metrics an higher curve means better `RT` performances. As we can see in figure 16 while the `Q` factor is low an increment causes non-negligible differences between trajectories related to consecutive values of `Q`.

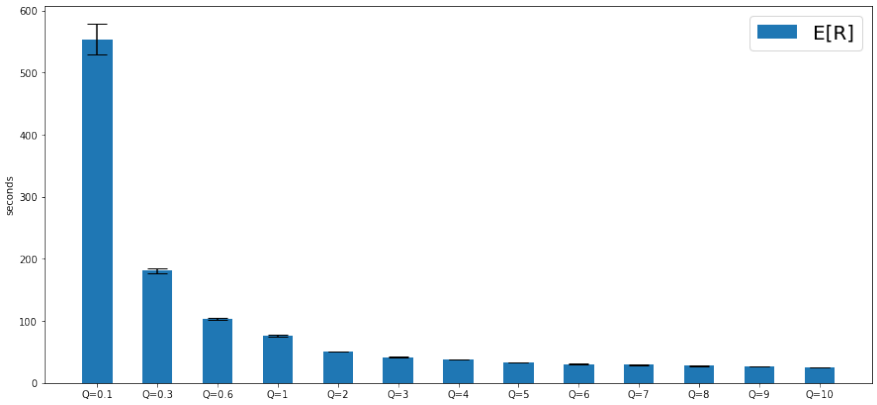


Figure 17: E[`RT`] with 99% CI varying `Q`.

Also in the exponential case the mean number of queued jobs, maintaining the same stability condition (i.e. ρ=0.8), increases with `Q`.

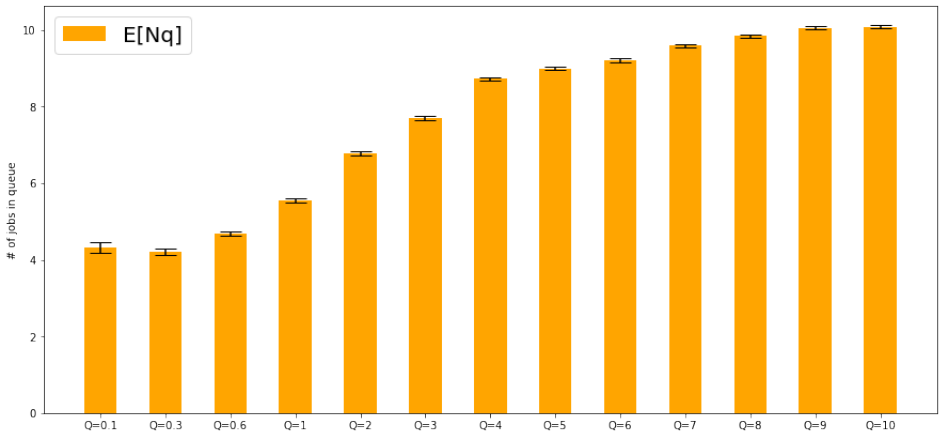


Figure 18: Exponential Scenario: E[N_q] with 99% CI.

6 Conclusions

In this project we have studied the RT of an M/G/1 system with *non-gated time-limited serving policy* with *multiple vacations*.
In both the constant and the exponential scenario we have seen that a the necessary condition which must be satisfied in order to reach good response time performances from the RT point of view is that each turn must serve at least one Job.
Forcing a Q parameter smaller than the ST means that the server will take a larger number of vacations so the V value will have a higher affection on the average RT. Moreover in this analysis assumptions we have explicity stated that the system has no costs in entering/leaving vacations and in the comparison between the Job ST and the deficit of the turn time. The latter consideration means that in a real scenario the costs in RT terms are even larger. The fact that the system takes two or more consecutive vacations among the service of two consecutive jobs can be logically wrong.
On the other hand it can be reasonable from the real systems point of view to assume that the Job ST cannot exceed a predetermined maximum value; for this reason it would be easy to find the value of the Q factor such that each turn serves at least one job.
Anyway increasing Q over than a certain value does not guarantee tangible benefits in terms of RT. This happens because the system will takes less and less vacations as Q increases, in fact the ST/Q ratio decreases and so the number of Jobs that will be preceded by a vacation will be smaller w.r.t. the total number of Jobs served. As a consequence of this last obaservation the mean value of ST' will tend to the mean value of the ST as Q increases. While ST' decreases the mean waiting time E[W] increases; at first sight it seems contradictory since E[W] depends itself on E[ST'], but this increasing is due to the fact that the supported IT in order to keep the system stable can be higher, this means that the mean number of job enqueued E[N_q] increases because of the Jobs accumulation during the vacation and so E[W] increases.

In conclusion from our analysis we can state that, in a real system with this features, increasing Q w.r.t. the ST provides an improvements from the number of served jobs point of view, in other words, increasing Q means increasing the number of served Jobs. Indeed it provides also two drowbacks:

- The RT does not reach tangible benefits as Q increases;
- The number of queued Jobs increases as Q increases so, in real systems in which the queue buffer is always finite, this could imply Job losses.