University of Pisa
MSc in Computer Engineering
Performance Evaluation of Computer Systems and Networks
2017/2018

Fair Cellular Network

Leonardo BERNARDI
Gioele CARIGNANI
Federico ROSSI

Contents

1 Introduction

This study concerns the analysis of the behaviour of packets delivered by an antenna to mobile nodes connected to it. The antenna follows a very simplified approach of the one used in LTE; in fact all the clients send periodically a CQI that informs the antenna about the quality of the channel (i.e. that specifies how many bytes can be packed in one Resource Block). The antenna sends, for each timeslot, a frame composed of 25 Resource Blocks (each resource block belongs uniquely to a single user). Particular attention in this study has been put on how each client is served in terms of fairness w.r.t other nodes.

1.1 Early description

Each client in the antenna has his own queue, managed with a FIFO policy. We analyzed the behaviour in two different scenarios:

- Exponential interarrivals, uniform service demands and uniform CQIs
- Same as before but with binomial CQIs to point out particular behaviours when user report significantly different CQIs.

Packet scheduling among the different queues inside the antenna is managed with a least-served-first policy: on each timeslot, users are served by increasing order of received data. When an user queue it's considered for scheduling packets are extracted and inserted in frame's resource blocks until there's available space.

2 Model

2.1 Assumptions

For our needs we simplified the model as following:

1. We consider only the down-link side of the communication, thus packets are generated directly into the antenna with exponential inter-arrivals (rates are equal for each users)
2. The maximum size of each packet has been fixed to a value of 75 bytes. In this way each packet, even with the worst CQI (that is – RB of 3 bytes) can be sent in a single frame, without the need of any kind of fragmentation of a packet among two frames.
3. If a packet for the current user cannot be sent (e.g. it exceeds available size at the time of packing it) we skip to serve the next user. In this way we could avoid to waste useful space in the frame, if the next user has packets that can be inserted into the RB.
4. We have 3 different random number generators (RNGs) for interarrival times and size of packets and for CQIs. However we could not provide different RNGs for each user because Omnet has not included yet a way to instantiate number of RNGs as a function of another parameter. Anyway, the RNG used by Omnet is based on the state-of-art of RNGs, the Mersenne twister. This lets us to affirm with quite good confidence that we are not introducing a bias on our data samples due to inter-module interference in same RNG streams.
5. Communication channel between antenna and users can be considered ideal, i.e. no delay and no BER, for our purposes.

2.2 System parameters

These are the parameters that characterise the system:

- Network dimension
- Timeslot period
- Mean inter-arrival time of packets τ from now on
- CQI generation mode

2.3 Binomial CQI generation

We wanted to generate CQI so that in the system there were users with an high mean CQI and others with a low one. To accomplish this we classified users upon their user id thus:

- Odd user id: probability of success $p = 0.2$ in 14 repeated trials, $E[CQI] = 14 * 0.2 + 1 = 3.8$.
- Even user id: $p = 0.8$, $E[CQI] = 14 * 0.8 + 1 = 12.2$.

Note that we summed 1 to the mean value to have values in $[1, 15]$

2.4 Components

The core components for the model are Antenna, User, Frame, Resource Block, Packet and CQI.

Antenna It handles packet generation with exponential interarrival distribution, whose mean is defined by a parameter. At each timeslot it schedules packets as described before and inserts them in a Frame (more on this later), that is broadcasted to all users at the end of the scheduling. Also, at the beginning of each timeslot, it receives channel quality indicators (CQIs) from users. Each user is represented into the Antenna by a descriptor, holding information about current CQI, packet queue and bytes received so far by that user.

User Each user is in charge of reporting on each timeslot its own CQI, extracted from a specific distribution according to the scenario we are simulating. It will receive the Frame from the Antenna, considering only packets belonging to him.

Frame It is simply a collection of 25 resource blocks, composed by the Antenna at each timeslot and broadcasted to all users.

Resource Block A Resource Block (RB) models resource allocation for different users. It can belong only to one user and its size is set according to that user's CQI.

Packet A packet is the information unit we are considering in this work. It is created by the antenna and its size is extracted from an uniform distribution as described before, then it is inserted in one of the queues containd in user descriptors.

CQI Abstracts the concept of channel quality indicator as a message sent by users to the antenna.

2.5 Implementation

Antenna and User are implemented as `SimpleModule` (files: `Antenna.ned, Antenna.cc, Antenna.h` and `User.ned, User.cc, User.h`), Packet, Frame and CQI as `Message` (files: `Packet.msg, Frame.msg, Cqi.msg`), Resource Block and User Descriptor as plain `C++` classes (files: `ResourceBlock.h, ResourceBlock.cc` and `UserDescriptor.cc, UserDescriptor.h`).

Antenna Module declaration

```
simple Antenna {
    parameters:
        int n;
        double timeSlotPeriod @unit(s);
        double packetMeanIntTime @unit(s);
        int exponentialIntArrRNGID = default(0);
        int exponentialSizeRNGID = default(1);
        [...] //statistic declarations
    gates:
        input in[];
        output out[];
}
```

User Module declaration

```
simple User{
    parameters:
        int n;
        double timeSlotPeriod @unit(s);
        int uniformServ; //0 Uniform 1 and 2 Binomial
        int CqiRNGID = default(2);
        [...] //statistic declarations
    gates:
        input in;
        output out;
}
```

Frame Message declaration

```
cplusplus {{
#include "ResourceBlock.h"
typedef ResourceBlock* ResourceBlockPtr;
}}
class noncobject ResourceBlockPtr;

message Frame {
    ResourceBlockPtr _rbs[25];
}
```

Cqi Message declaration

```
cplusplus {{
message Cqi {
    int userID;
    int cqiValue;
}
```

Packet Message declaration

```
packet Packet {
    int size;
    int packedSize=0;
    simtime_t creation;
    bool fragment=false;
}
```

2.6 Network

The overall network for the simulation is composed thus (resulting in Figure 1):

```
network Network
{
    parameters:
        int n;
    submodules:
        antenna: Antenna;
        user[n]: User;
    connections:
        for i=0..n-1 {
            antenna.out++ --> user[i].in;
            user[i].out --> antenna.in++;
        }
}
```

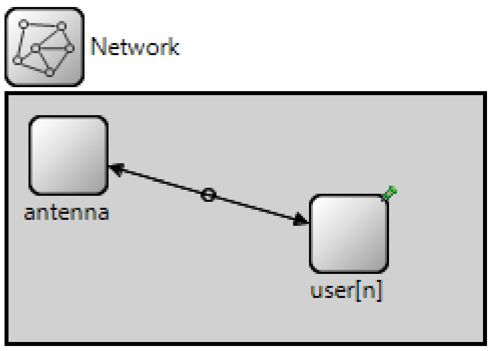


Figure 1: Network illusration from Omnet++

3 Model validation

In order to validate the model, we performed some tests.

No memory leak We run several simulations checking for *undisposed object* warnings from Omnet++.

Packet loss test We run a simulation measuring the number of packet sent and the number of packet received, expecting them to be equal. From this we derive also next test.

Rate in/out Given the mean inter-arrival time of packets for each user λ, we expect that with n users, we will have a mean packet throughput of $\gamma = n * \lambda$.

Little's law test We compare E[R] obtained from measurements in the simulation against the value computed by Little's law $E[R] = E[N]/\gamma$.

Continuity test We vary the simulation parameters slightly, expecting the output not to behave wildly.

Consistency test We scale up of factor k the number of users and down the arrival rate, expecting the same results we obtained without the scaling.

Degeneracy test We tried a configuration with 0 users and checked if the system worked as expected.

3.1 Preliminary parameter tuning

Stability In order to find a stability condition we performed several simulations with different mean inter-arrival times. For the uniform CQIs, we found that stability condition imposes that $\tau > 0.8ms$ as we can see from Figure 2

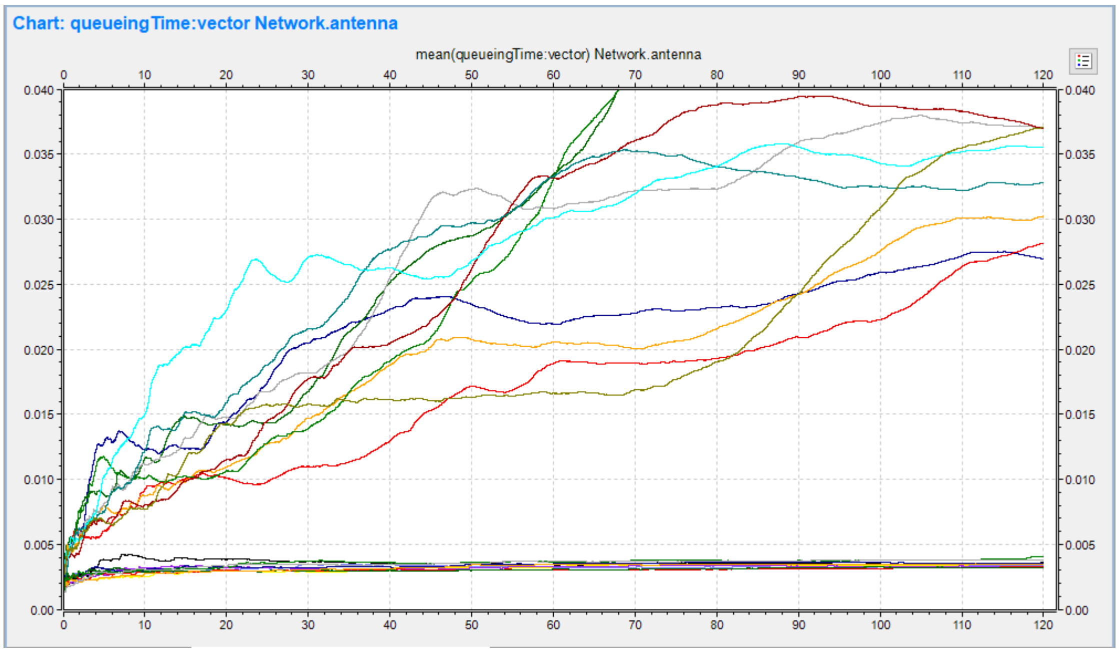


Figure 2: Response time around stability point, the curve above is $\tau \approx 0.7$, the other is $\tau \approx 0.8$

For binomial CQIs, with success probabilities described as before, performing several simulations we obtained a $\tau > 1.2ms$. Note that since we managed to reach stability for users with lower average CQI, the stability is hence granted for the the other class of users. as we can see from Figure 3

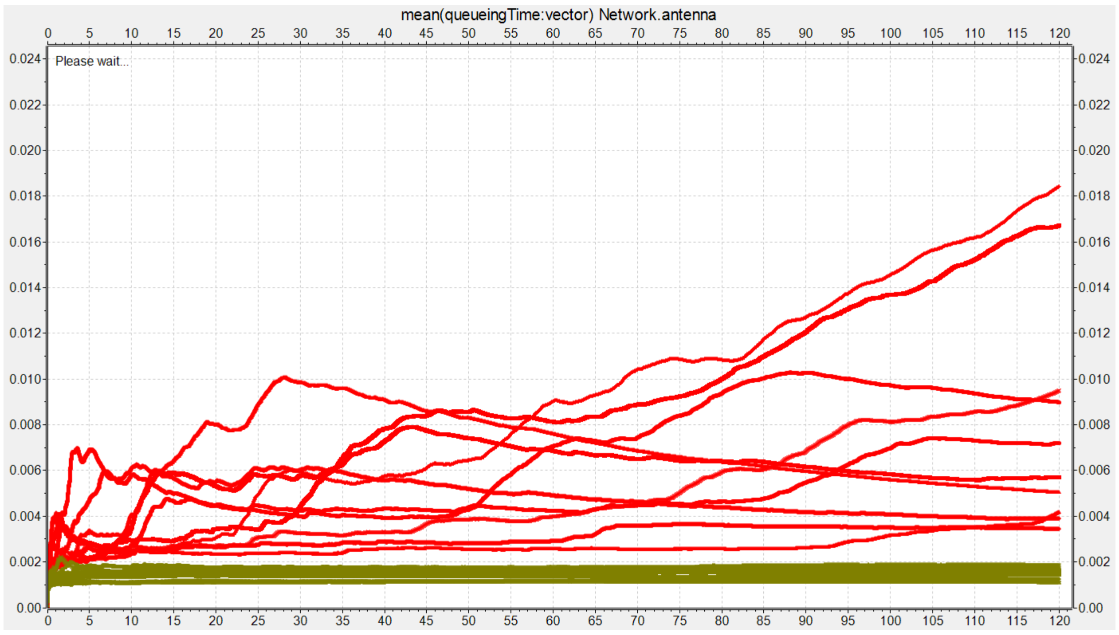


Figure 3: Response time around stability point, red curve is $\tau = 1.1ms$, green curves are $1.2ms$ and $1.3ms$

Although the system being stable at $\tau \approx 1.2ms$ it behaves too wild among different repetitions (Figure 4), so we picked $\tau \approx 1.3ms$ as worst stable case.

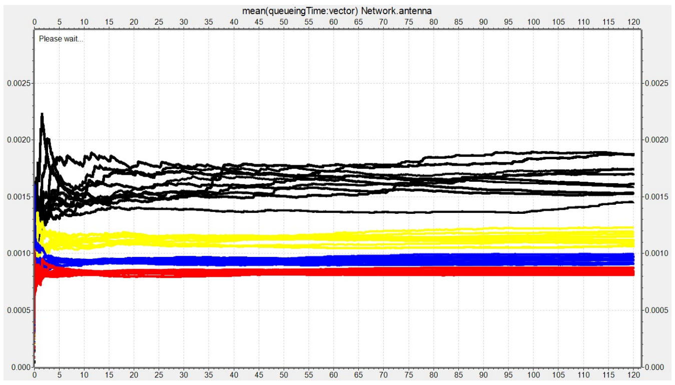


Figure 4: Various stable simulation, note the wild behavior of black curve

To perform early test on our system we choose the parameters of the simulation in order to have meaningful yet "debuggable" results. Therefore we used uniform distributed CQIs, 10 repetition combined with $120s$ of simulation time in order to have good coverage of various trajectories for our system.

We have fixed timeslot at $1ms$ in order to keep close to reality (at least in this) and we picked $\tau = 5ms$ to stay very far away from limit condition, that will be considered later on performance analysis.

3.2 Packet loss and rate in/out test

Test scenario:

# Users	10
Timeslot	1 ms
Mean int-time	5 ms
Duration	120 s
Repetitions	10

Table 1: Packet loss and rate in/out test scenario

Ending up as expected, without packet loss and with $\gamma \approx n * \lambda = 2pkt/ms$

Rep	Sent	Received	$n * \lambda$ (pkt/ms)	γ (pkt/ms)	E[N]	E[R]	Little's E[R]
0	240405	240405	2	2.003375	1.418021	0.000542	0.000708
1	239717	239717	2	1.997641667	1.414815	0.000541	0.000708
2	240735	240735	2	2.006125	1.41984	0.000540	0.000708
3	239961	239961	2	1.999675	1.415739	0.000541	0.000708
4	239877	239877	2	1.998975	1.417684	0.000543	0.000709
5	240072	240072	2	2.0006	1.41615	0.000542	0.000708
6	238656	238656	2	1.9888	1.40756	0.000541	0.000708
7	239631	239631	2	1.996925	1.415612	0.000542	0.000709
8	239845	239845	2	1.998708333	1.416666	0.000541	0.000709
9	239811	239811	2	1.998425	1.411586	0.000541	0.000706

Table 2: Packet loss and rate in/out test results

3.3 Continuity test

Test scenario:

# Users	10
Timeslot	1ms
Mean int-time	${4.8, 4.9, 5, 5.1, 5.2} ms
Duration	120s
Repetitions	10

Table 3: Continuity test scenario

We then plotted the response time (Figure 5) showing that the system does not behave too differently varying the inter-arrival time.

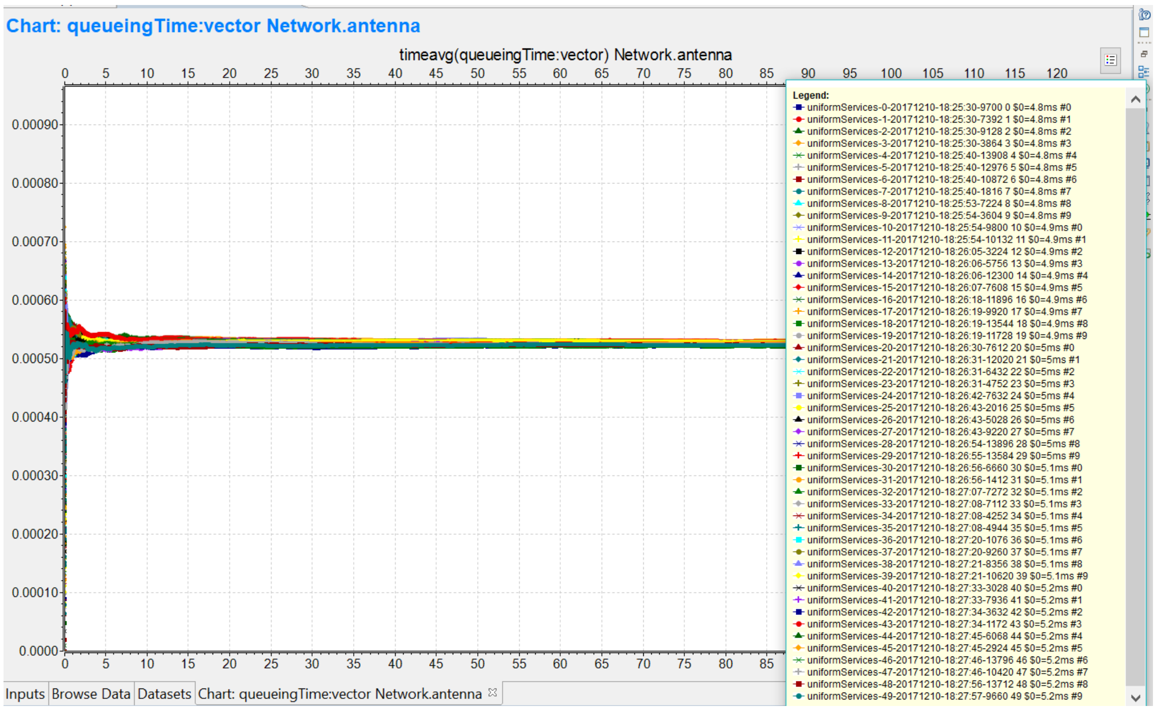


Figure 5: Continuity test

3.4 Consistency test

Test scenario:

# Users	20
Timeslot	1ms
Mean int-time	10 ms
Duration	120s
Repetitions	10

Table 4: Consistency test scenario

Again, we obtain no packet loss and $n * \lambda = \gamma$ and comparable results in Table 5 w.r.t Table 2

Repetition	Packets sent	Packets received	γ (pkt/ms)
0	239840	239840	1.998667
1	239880	239880	1.999
2	240971	240971	2.008092
3	240205	240205	2.001708
4	239955	239955	1.999625
5	240247	240247	2.002058
6	240143	240143	2.001192
7	240378	240378	2.00315
8	240141	240141	2.001175
9	239371	239371	1.994758

Table 5: Consistency test results

3.5 Degenaracy test

Test scenario:

# Users	0
Timeslot	1ms
Mean int-time	5 ms
Duration	120s
Repetitions	10

Table 6: Degeneracy test scenario

Experiment	Module	Name	Mean	StdDev
uniformCQI	Network.antenna	throughput:vector	0.0	0.0
uniformCQI	Network.antenna	packetsPerSlot:vector	0.0	0.0
uniformCQI	Network.antenna	queuedPacketsPerSlot:vector	0.0	0.0

Table 7: Degeneracy test result

As expected the system didn't send or receive any packet, hence we assured the absence of strange behaviours under these conditions.

4 Warmup analysis

4.1 Uniform CQIs

In order to perform warmup analysis we considered the two main performance indexes of our work, throughput and queueing time. While the throughput stability is pretty load-independent, converging quite the same at any workload, queueing time stability is very load-dependent, becoming totally unstable if we don't match stability condition. Under this considerations, we extimated the warmup time for the system at the heaviest workload possible, but within stability condition - that is $\tau = 0.9ms$. Since we didn't notice any wild variation with different seeds we stopped at 10 repetitions in simulation, obtaining the following plots for queueing time (Figure 6) and throughput (Figure 7):

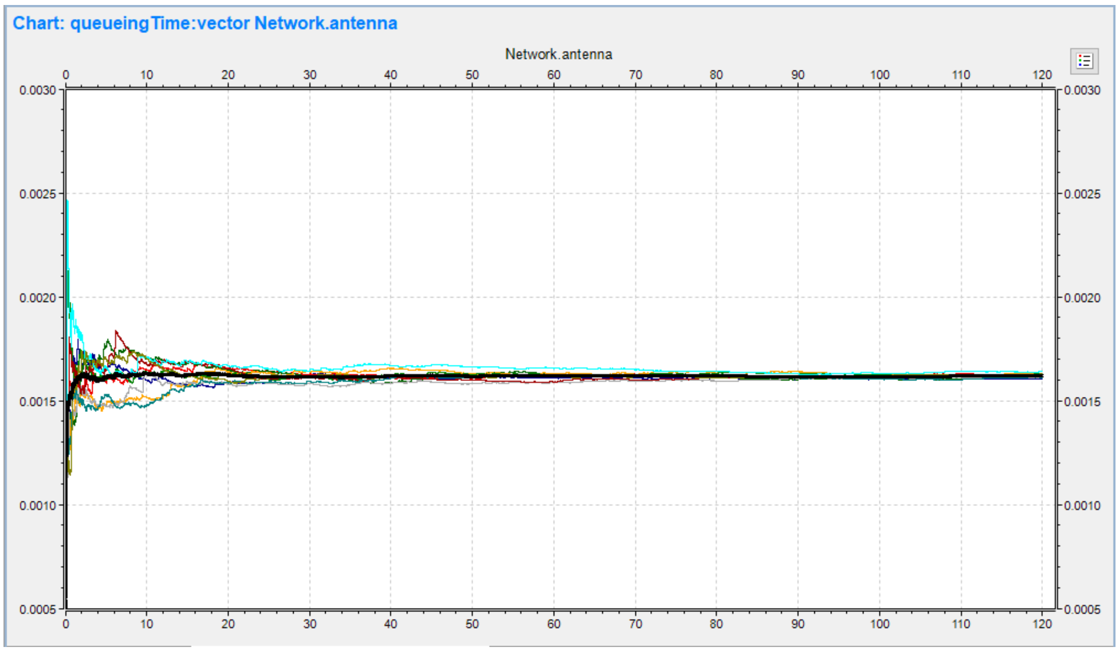


Figure 6: Warmup of response time

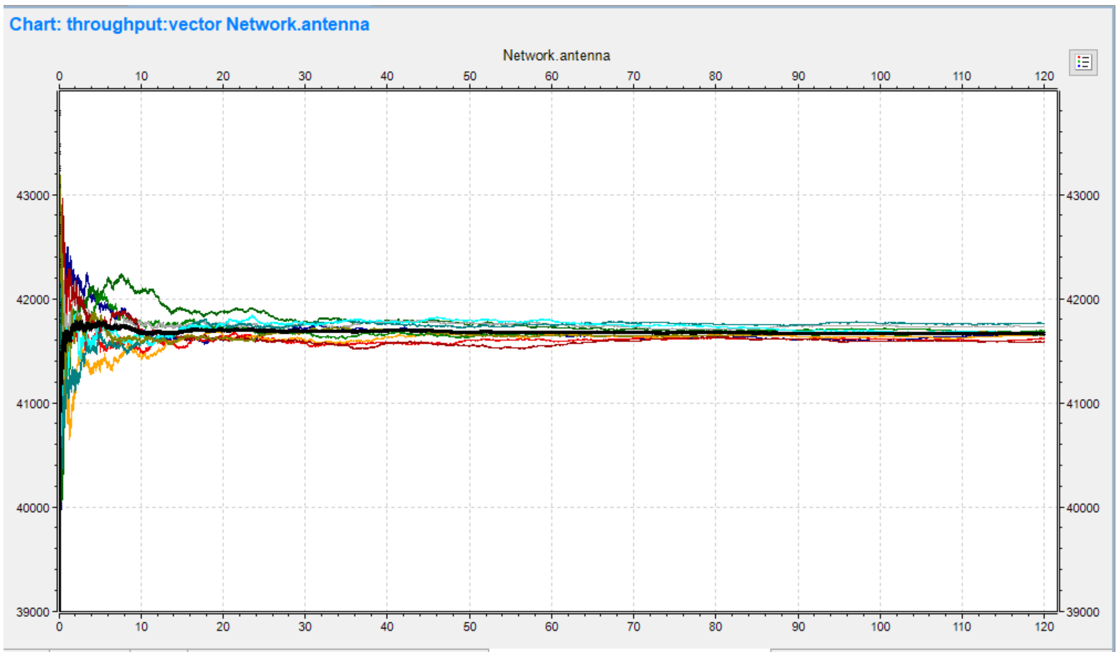


Figure 7: Warmup of throughput

As we can see from above plots, warmup period can be safely extimated to 30s for the queueing time, and this holds also for throughput.

4.2 Binomial CQIs

With binomial CQIs stability condition is more strict,due to the users that have a low average CQI, thus having a lower bound of $\tau \approx 1.3ms$, so the scenario considered for this warmup analysis uses this value for inter-arrival time. Also we obtained a good confidence for this evaluation again with 10 repetitions. We obtained following plots for queueing time (Figure 8) and throughput (Figure 9).

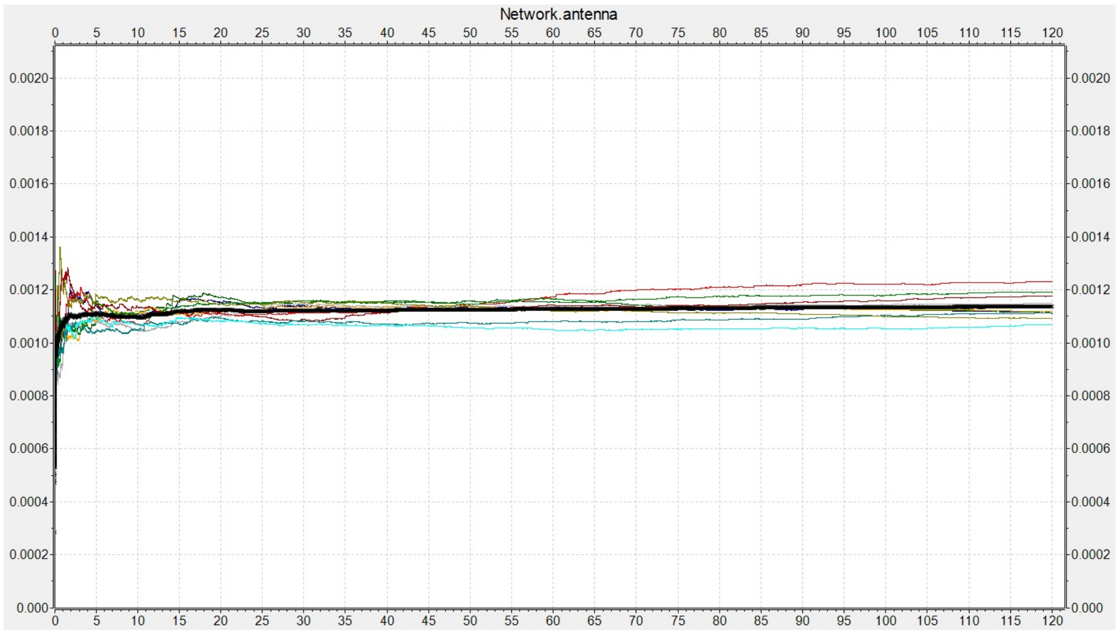


Figure 8: Warmup of response time

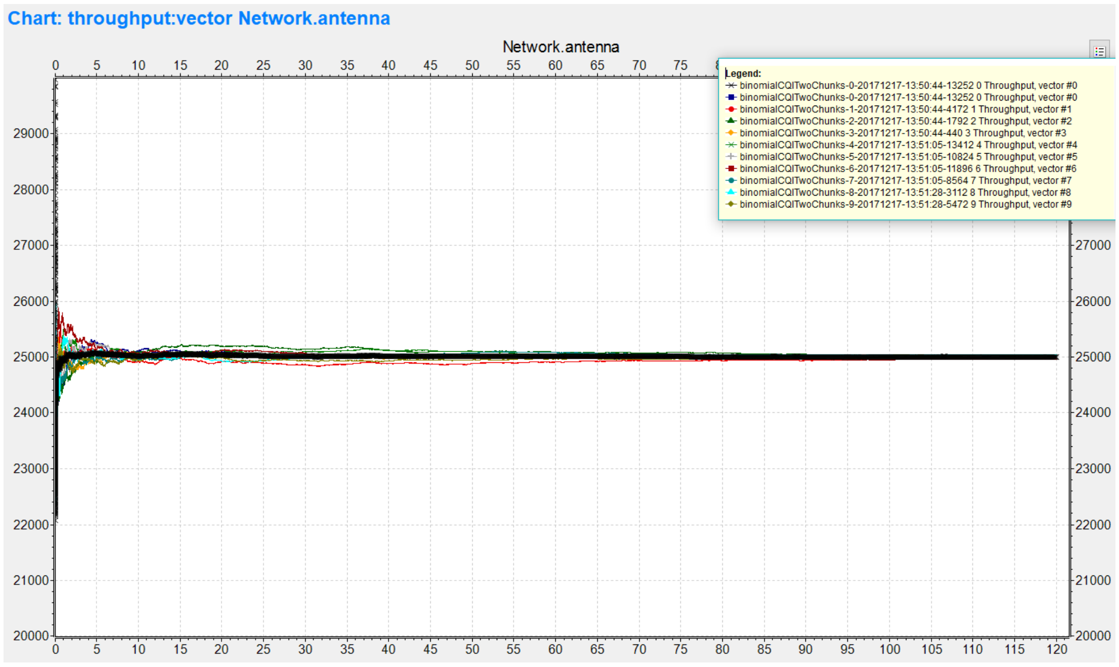


Figure 9: Warmup of throughput

Throughput reaches stability after few seconds $\approx 5s$, while response time is more jittery, thus reaching stability only at $\approx 30s$

5 Scenario analysis

From now on, we will use more repetitions w.r.t to warmup analysis, since we want to asses a 95-th percentile of response time with 99% confidence level, thus we need more samples.

5.1 Uniform CQIs

Test scenario :

# Users	10
Timeslot	1 ms
Mean int-time	${0.9 ms, 2 ms,5 ms}
Duration	120 s
Repetitions	100
Warmup Period	30 s

Table 8: Scenario analysed with varying workloads

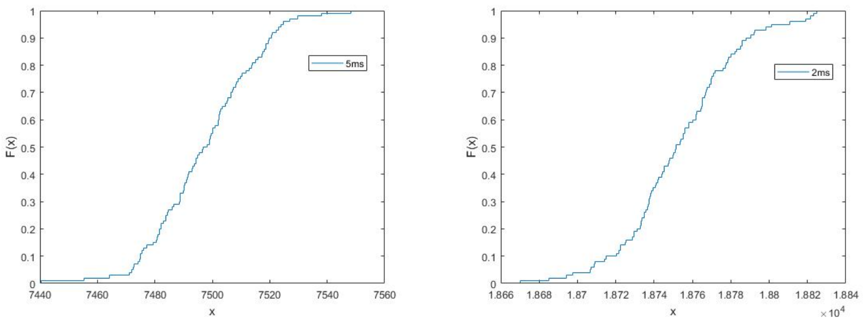
5.1.1 Throughput

Throughput analysis in the considered scenario gave us these results:

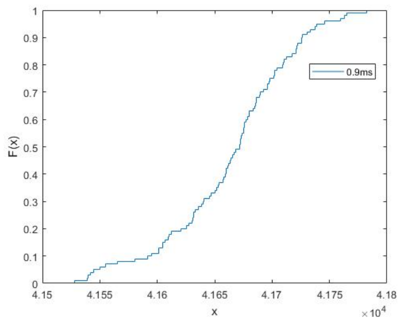
Mean inter-arrival time	**0.9ms**	**2ms**	**5ms**
Mean [b/s]	41663	18753	7497
Mean CI 99% **[b/s]**	14.0	7.6	4.7
StDev [b/ss]	54.47	29.65	18.05

Table 9: Response time results from simulation

Distribution of throughputs ECDF for throughput has the following shape at various workloads:



(a) 5ms (b) 2ms



(c) 0.9ms

Figure 10: Throughput ECDF at different workloads

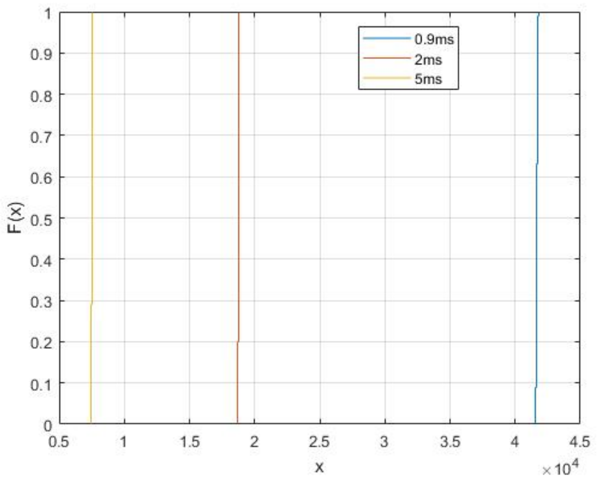


Figure 11: Comparative of throughput ECDFs under varying workloads

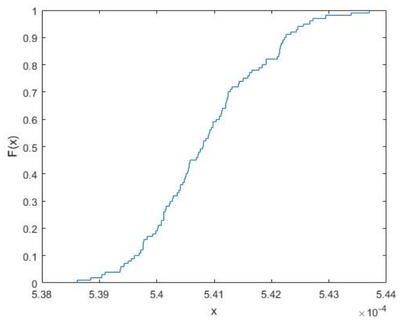
5.1.2 Response time

Response time analysis in the considered scenario gave us these results:

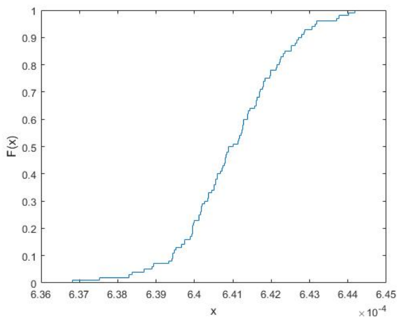
Mean inter-arrival time	**0.9ms**	**2ms**	**5ms**
Mean [s]	0.0016	0.0006	0.0005
Mean CI 99%	$4.9 * 10^{-6}$	$3.5 * 10^{-7}$	$2.7 * 10^{-7}$
StDev [s]	$1.89 * 10^{-5}$	$1.36 * 10^{-6}$	$1.03 * 10^{-6}$
Avg LCG [out of $\approx$ 1]	0.2022	0.0722	0.0226
CI for LCG 99%	0.0156	0.0042	0.0012

Table 10: Response time results from simulation

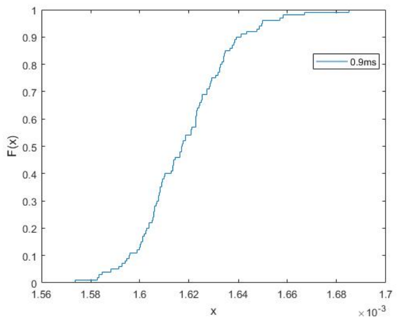
Distribution of response times ECDF for response times has the following shape at various workloads:



(a) 5ms



(b) 2ms



(c) 0.9ms

Figure 12: Response time ecdf at different workloads

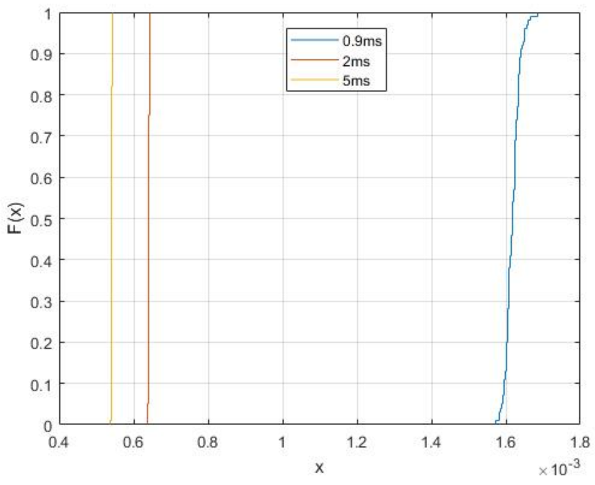
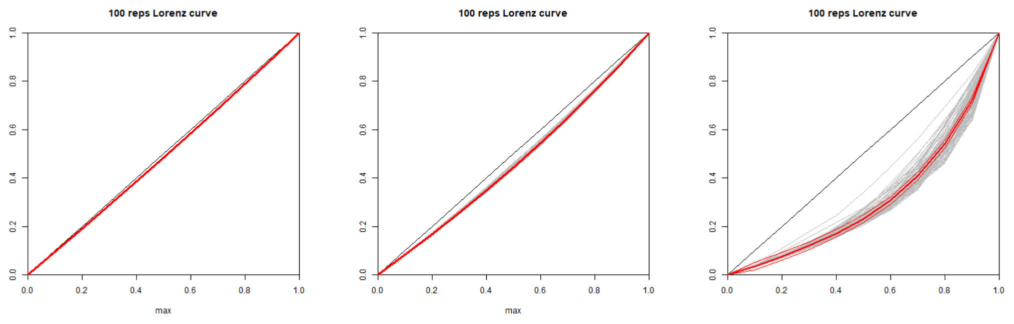


Figure 13: Comparative of ECDFs under varying workloads

From Figure 13 we can see how the system behaves at different workloads. Since response time is a lower-is-better indicator, higher curve mean better response time, thus having pretty the same response time distribution for loads after 2ms, while having a less steep curve for intense loads at 0.9ms

From computation we obtained an important result on response times distribution; the 95-th percentile for the response time, in the worst scenario, is $t_{0.95} \in [0.0016, 0.0017)s$ with a confidence interval of 99%. This means that altough the system is under an heavy workload, near to the stability limit, it behaves pretty well with a response time $\approx$ 1.5 timeslots.

Response time fairness As we can see from Table 10 and Figure 14, the Lorenz curve gap, thus the system unfairness w.r.t response times, increases with workload, thus having a quite perfect fairness level at $2ms$ and $5ms$. At $0.9ms$ system fairness decreases due to the more jittery behaviour of the system when we approach the unstable region.



(a) 5ms (b) 2ms (c) 0.9ms

Figure 14: Lorenz curves at different workloads

5.2 Binomial CQIs

# Users	10
Timeslot	1 ms
Mean int-time	${1.3 ms, 2 ms,5 ms}
Duration	120 s
Repetitions	100
Success probability (odd user IDs)	0.2
Success probability (even user IDs)	0.8
Warmup Period	30 s

Table 11: Scenario analysed with varying workloads

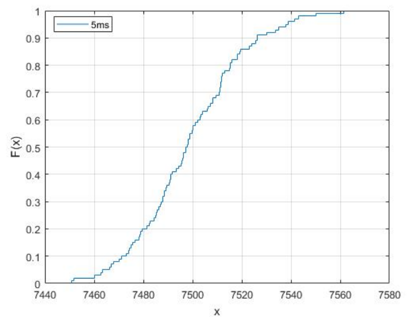
5.2.1 Throughput

Throughput analysis in the considered scenario gave us these results:

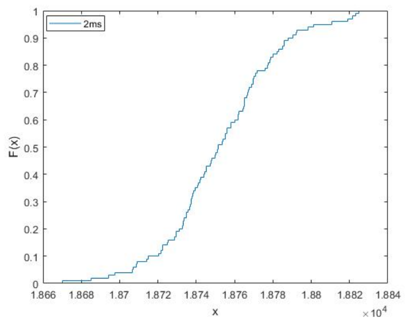
Mean inter-arrival time	**1.3ms**	**2ms**	**5ms**
Mean [b/s]	28843	18743	7498
Mean CI 99% **[b/s]**	11.0	7.6	5.5
StDev [b/ss]	42.84	32.24	21.68

Table 12: Response time results from simulation

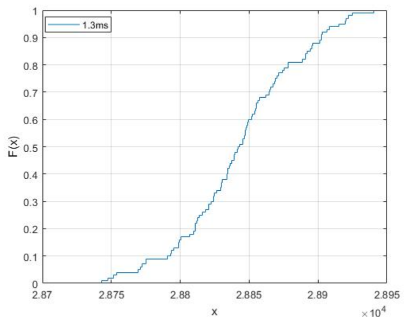
Distribution of throughputs ECDF for throughput has the following shape at various workloads:



(a) 5ms



(b) 2ms



(c) 1.3ms

Figure 15: Throughput ECDF at different workloads

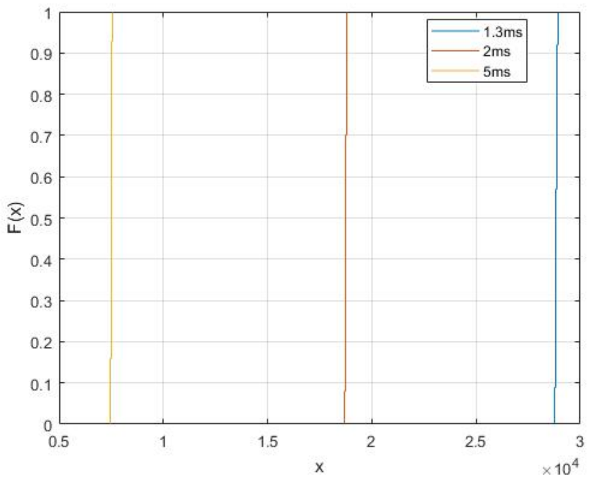


Figure 16: Comparative of throughput ECDFs under varying workloads

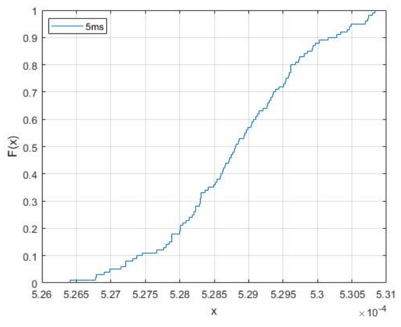
5.2.2 Response time

Response time analysis in the considered scenario gave us these results:

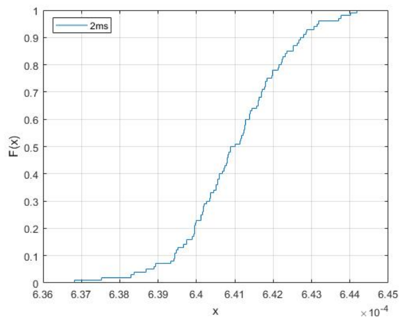
Mean inter-arrival time	**1.3ms**	**2ms**	**5ms**
Mean [s]	0.0010	0.0006	0.0005
Mean CI 99%	$1.02 * 10^{-5}$	$1.61 * 10^{-6}$	$2.59 * 10^{-7}$
StDev [s]	$3.97 * 10^{-5}$	$6.26 * 10^{-6}$	$1.00 * 10^{-6}$
Average LCG [out of $\approx$ 1]	0.1201	0.0592	0.0165
CI for LCG 99%	0.0157	0.0053	0.0013

Table 13: Response time results from simulation

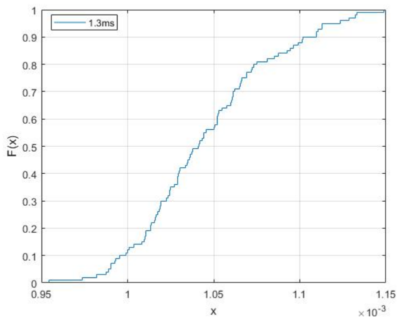
Distribution of response times ECDF for response times has the following shape at various workloads:



(a) 5ms



(b) 2ms



(c) 0.9ms

Figure 17: Response time ecdf at different workloads

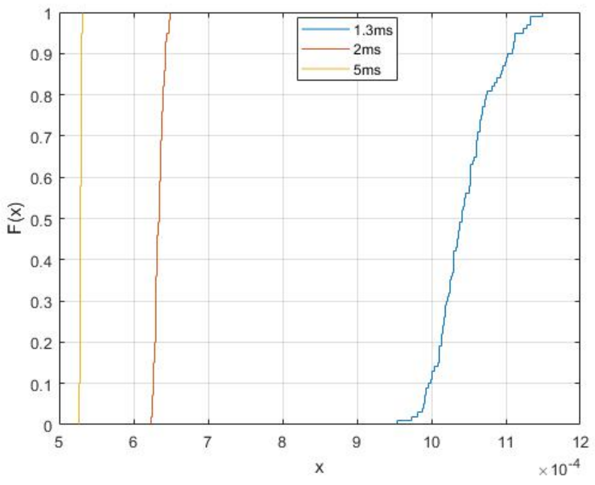
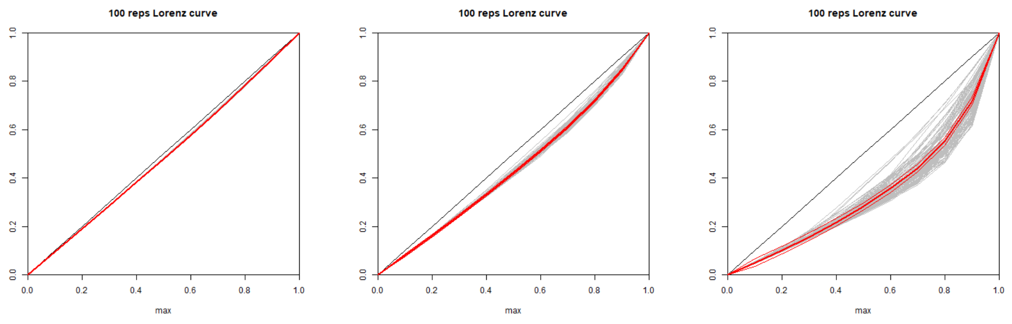


Figure 18: Comparative of ECDFs under varying workloads

From Figure 18 we can see how the system behaves at different workloads. Since response time is a lower-is-better indicator, higher curve mean better response time, thus having pretty the same response time distribution for loads after 2ms, while having a less steep curve for intense loads at 1.3ms. This behaviour is very similar to the one in the previous scenario.

From computation we obtained an important result on response times distribution; the 95-th percentile for the response time, in the worst scenario, is $t_{0.95} \in [0.0011, 0.0011)s$ with a confidence interval of 99%. This means that altough the system is under an heavy workload, near to the stability limit, it behaves pretty well with a response time $\approx$ 1 timeslots.

Response time fairness As we can see from Table 13 and Figure 19, the Lorenz curve gap, thus the system unfairness w.r.t response times, increases with workload, thus having a quite perfect fairness level at $2ms$ and $5ms$. At $1.3ms$ system fairness decreases due to the more jittery behaviour of the system when we approach the unstable region.



(a) 5ms (b) 2ms (c) 1.3ms

Figure 19: Lorenz curves at different workloads

5.2.3 Peculiarity

By looking at the queueing time of each user we noticed that when the network was under stress (heavy workload) there was a quite clear distinction between users with an even userID and the ones with an odd userID. In order to point out in a more evident way the differences we increased the gap between the two success probabilities by setting in the .ini file the success probability at 0.05, thus we have p=0.05 (even userID) and p=0.95 (odd userID). With the new probabilities we obtain the following mean CQIs:

userID	Success Probabilty	mean CQI
odd	0.05	1.75
even	0.95	14.30
odd	0.20	3.80
even	0.80	12.20

Table 14: mean CQIs with different success probabilities

From Table 14 we can see how by setting a very low success probability gets us to have a wider gap between the mean CQI of the two classes of users. But if we observe the response times of each user we can see graphically how this influences our results.

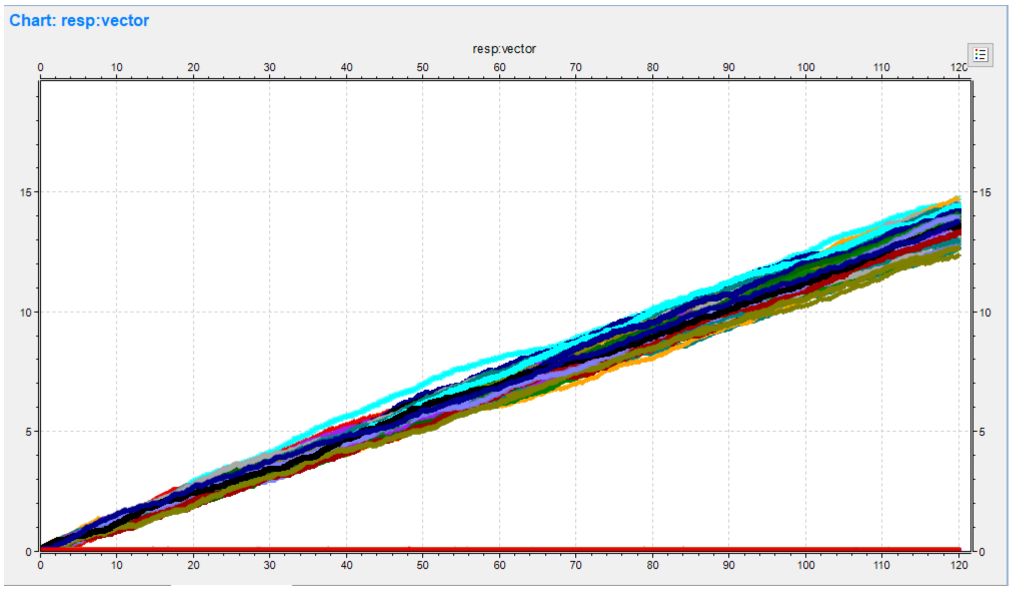


Figure 20: Response time of each user (success probability=0,05 and τ = 2ms)

In Figure 20 the upper lines are of users with a mean CQI ≈ 2 (odd ones) and the lower curves refer to users with a mean CQI near to 14. The problem of users with a very low CQI is given by the fact that this type of users are able to send very less packets (at most around 1 per timeslot), viceversa who has a high CQI even if he disposes only of one resource block is able to fit at least one packet in it. This makes the queues of odd users to grow indefinitely and as a consequence to have very high response times. In order to have a stability condition that verifies a stable behaviour for each user we consider the users with the worse CQI for this evaluation.

6 Binomial CQIs compared to Uniform CQIs

The two scenarios are significantly different in fact in the uniform one all the users at the end will have the same mean CQI instead in the Binomial case the mean values are different depending on the probability of success, that as required has been chosen not in the same way for each user. Anyhow the two scenarios are supported pretty much in the same way from our network but with some differences. The first difference is in the point of stability: $\tau \approx 0.8$ms for the Uniform scenario and $\tau \approx 1.2$ms for the Binomial one.

6.1 Response Time

By comparing Table 10 and Table 13 we can point out some differences. With same τ at 2 ms and 5 ms we can see that the mean values are the same, at least if we stop at the fifth decimal place. But if we look at the Confidence Intervals we can see that they are tighter for the Uniform scenario this is caused from the more jittery behaviour of the response times in the Binomial case.
We can anyhow compare the two response times:

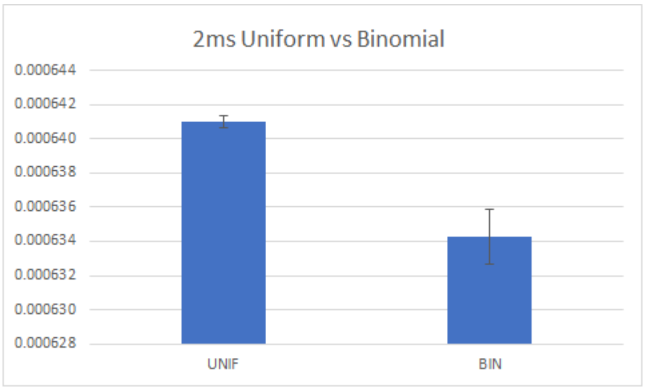


Figure 21: Comparative of Response times in the two scenarios

6.1.1 Fairness

In terms of fairness for each user in our it is important to analyse how equally our network deploys the jobs requested by the clients. For this we used Lorenz Curve and from Table 10 and Table 13 we can see that our network is more fair if interarrivals are Binomial with the parameters that we have decided.

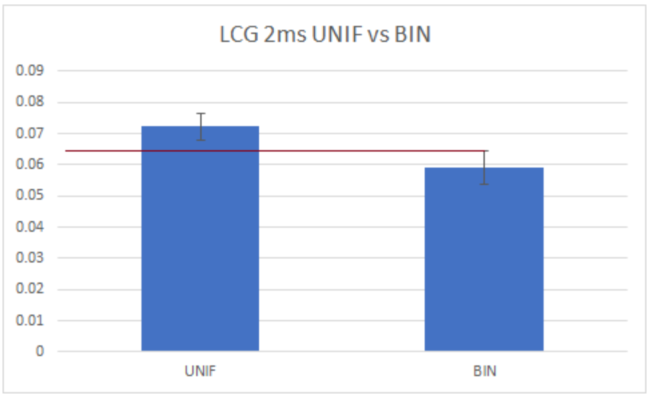


Figure 22: Comparative of LCG in the two scenarios

6.2 Throughput

By comparing Table 9 and Table 12 we can see that at the same workload there is not any meaningful difference between the two scenarios. But in the Uniform scenario the network is able to supply a higher maximum throughput since it is stable with heavier workloads.

7 A possible optimisation

7.1 The received bytes counting algorithm

So far we have considered a simple algorithm to record the number of bytes received by users (i.e. sums of bytes stored in a counter inside the user descriptor). This may lead to two problems.

The first problem is a numerical one. Since the sum is recorded into a int32 member, it is upper-bounded to $2^{31} - 1$. While we didn't encounter any numerical problem thanks to the short simulation time, in a longer simulation this will be a problem

The other issue is related to the behaviour of the least-served-first policy with this simple sum. We analysed the system under this scenario (uniform CQIs), recording the number of timeslots in which an user is being served:

# Users	10
Timeslot	1 ms
Mean int-time	0.9 ms
Duration	120 s
Repetitions	100
Warmup Period	30 s

In Figure 23 we can see the distribution of the number of timeslots among the users (for a given repetition):

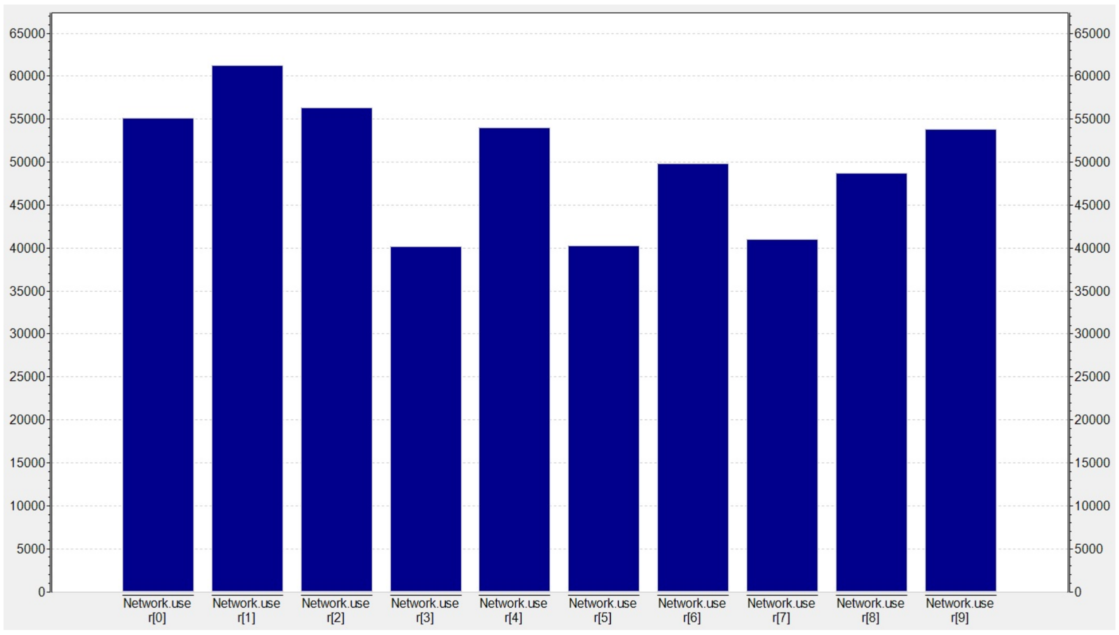


Figure 23: Number of "useful" timeslots for each user

Since users have IID uniform distributed CQIs and their service demands are IID as well, this difference in service is a strange behavior.

However this makes perfect sense; in fact at each timeslot we are summing the number of received bytes to a given user counter, thus weighting this number like all the other. So it's very likely to have some users that have received some big packets in the past starving due to some other users that received a burst of small packets recently. This can be also seen in the distribution of response times in figure Figure 24

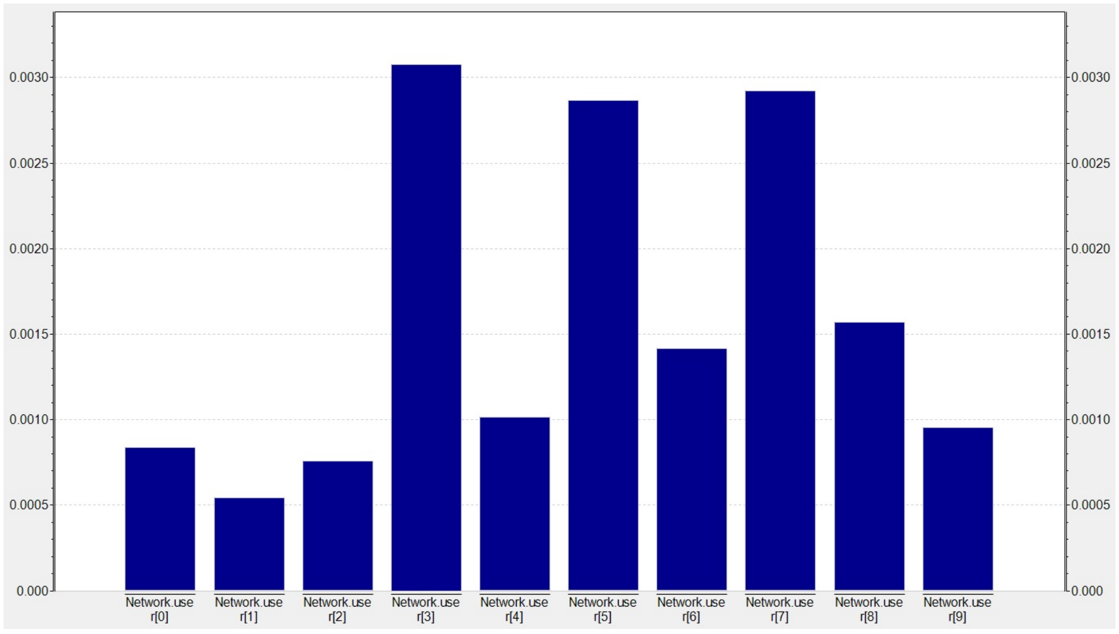


Figure 24: Average response time for each user

This doesn't hold only for this particular repetition, but for every repetition we analysed we saw this pattern, obviously with different users involved in the mechanism, due to the different seed. To prove this more analytically we tried to find a correlation between the number of useful timeslots and the response time. In particular we tried to find fit a non-linear function between response time and number of timeslot, finding a negative correlation as expected (we also tested residual IID-ness and homoskedasticity).

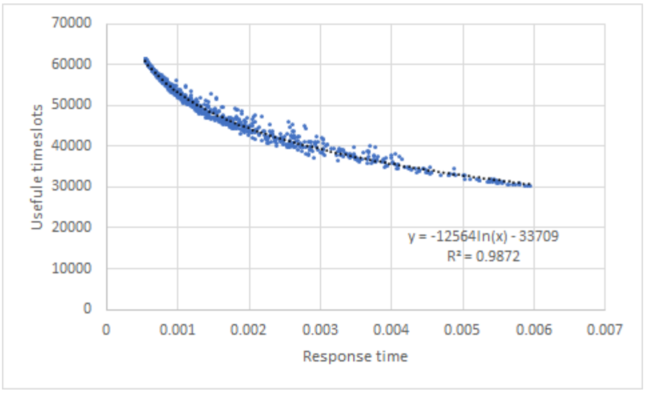


Figure 25: Non-linear fit

7.2 The optimisation

In order to weight in a more accurate way the number of received bytes we introduced a weighted average:

$$w_s^i = \alpha * r_i + (1-\alpha) * w_s^{i-1} \tag{1}$$

Where w_s^{i-1} is the (weighted) sum of bytes received so far and r_i is the number of bytes received at timeslot i.

Value for α To choose a value for α we performed several simulations seeking for the best possible uniformity between response times. We got a good behavior for $\alpha = 0.4$. This makes sense with previous consideration, since we are weighting the current sample less than the overall sum so far, being more immune to changes that lasts a short time (e.g burst of small packets between timeslots).

7.3 Results on Uniform CQIs

We performed a simulation adding the optimisation we thought about, and we obtained the results we expected, as we can see in figure Figure 26 and Figure 27

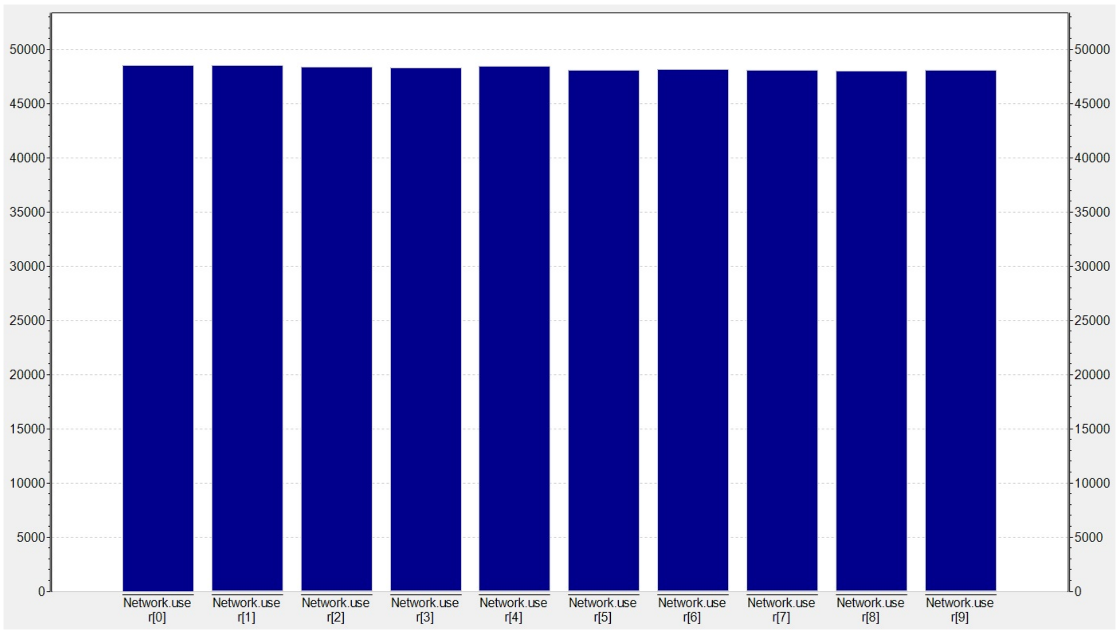


Figure 26: Number of useful timeslot for each user with new algorithm

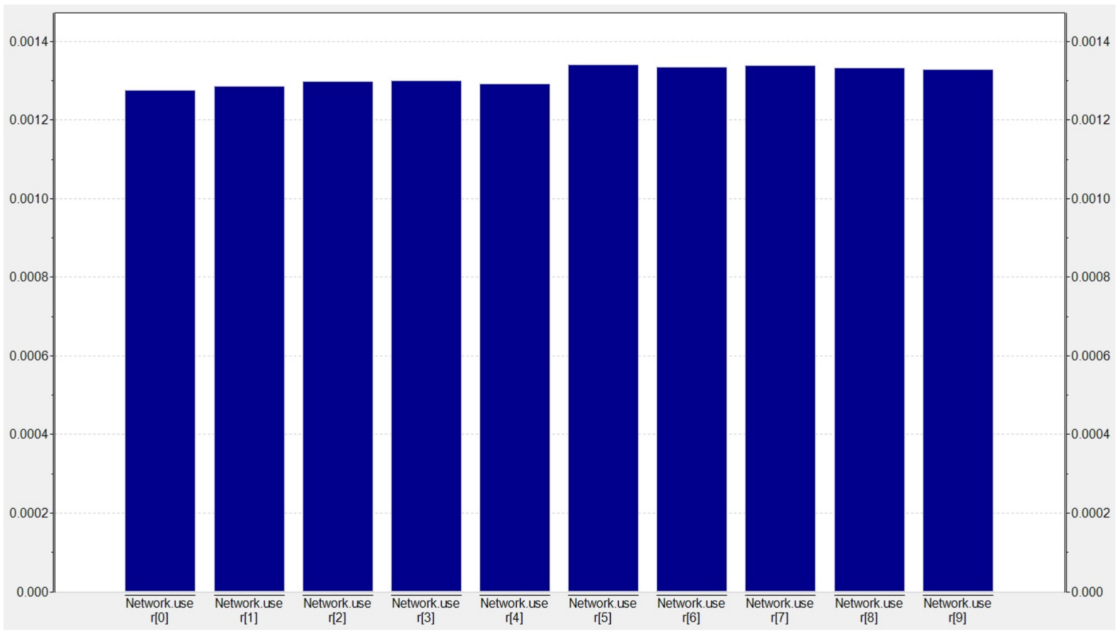


Figure 27: Average response times for each user

To asses in a statistically sound way the better performances of the new system we performed the same analysis as before on the results from simulation. While the throughput stays the same, the response time decreases a little, thanks to the absence of delay peaks that were caused by starvation:

Mean inter-arrival time	**0.9ms NEW**	**0.9ms OLD**
Mean [s]	0.0013	0.0016
Mean CI 99%	$2.15 * 10^{-6}$	$4.87 * 10^{-6}$
StDev [s]	$8.35 * 10^{-6}$	$1.89 * 10^{-5}$
Average LCG [out of ≈ 1]	0.0071	0.2023
CI for LCG 99%	0.0006	0.0156

Table 15: Response time results from simulation

We do not report Lorenz curves since they are quite equal to the perfect fairness line.

Response time comparative including CI is shown in Figure 28

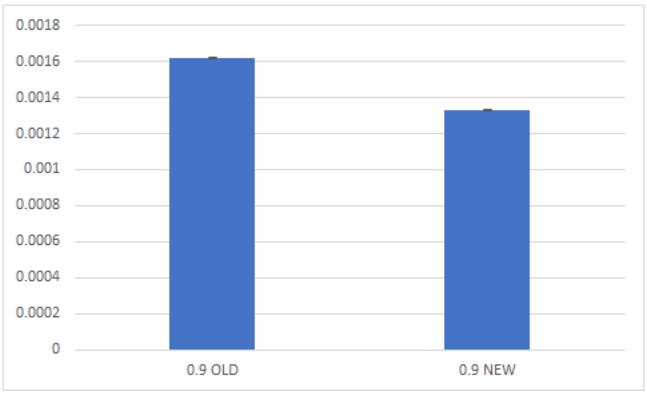


Figure 28: Response time comparative between the two solutions

Lorenz curve gap comparative including CI is shown in Figure 29

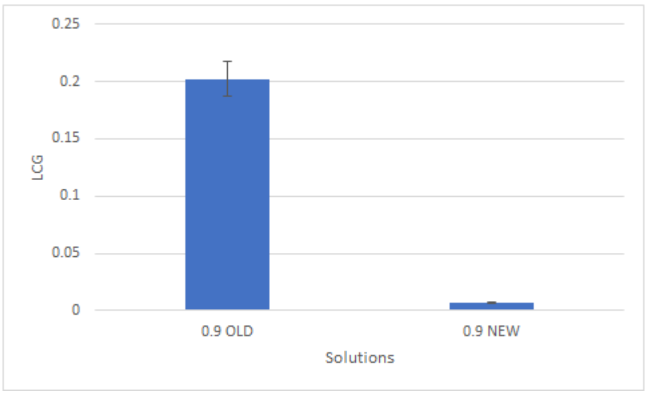


Figure 29: Lorenz curve gap comparison between the two solutions

7.4 Results on Binomial CQIs

We performed a simulation, with Binomial CQIs, with and without the optimisation under the following scenario:

# Users	10
Timeslot	1 ms
Mean int-time	1.3 ms
Duration	120 s
Repetitions	100
Warmup Period	30 s

In Figure 30 we can see the distribution of the number of timeslots among the users (for a given repetition):

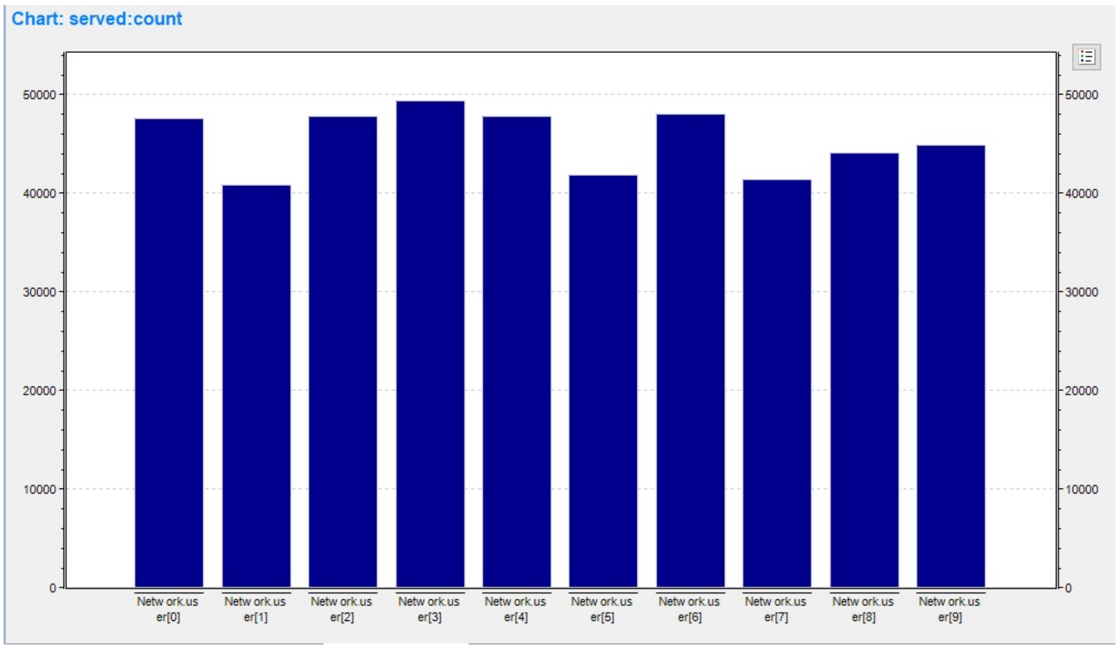


Figure 30: Number of "useful" timeslots for each user

Also in this scenario we can see the strange difference between the useful number of slots for each user. The same reasoning about starvation can be done in this case. And as before it can be seen in the distribution of response times in figure Figure 31

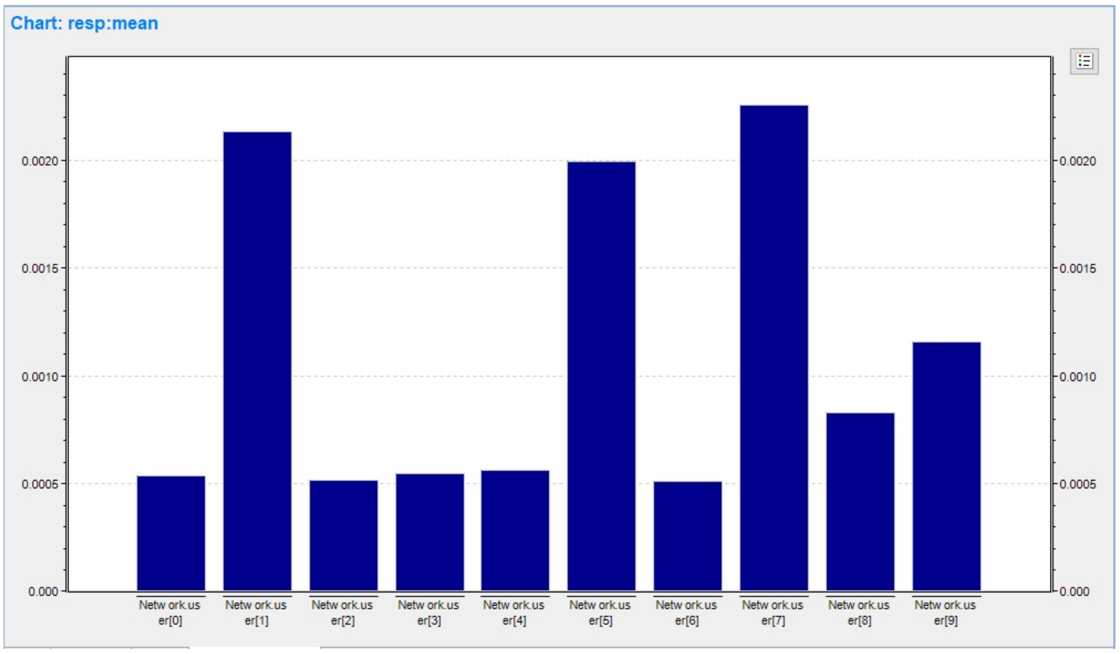


Figure 31: Average response time for each user

So we decided to apply the optimisation with Binomial CQIs and we observed the behaviour we expected, as we can see in figure Figure 32 and Figure 33

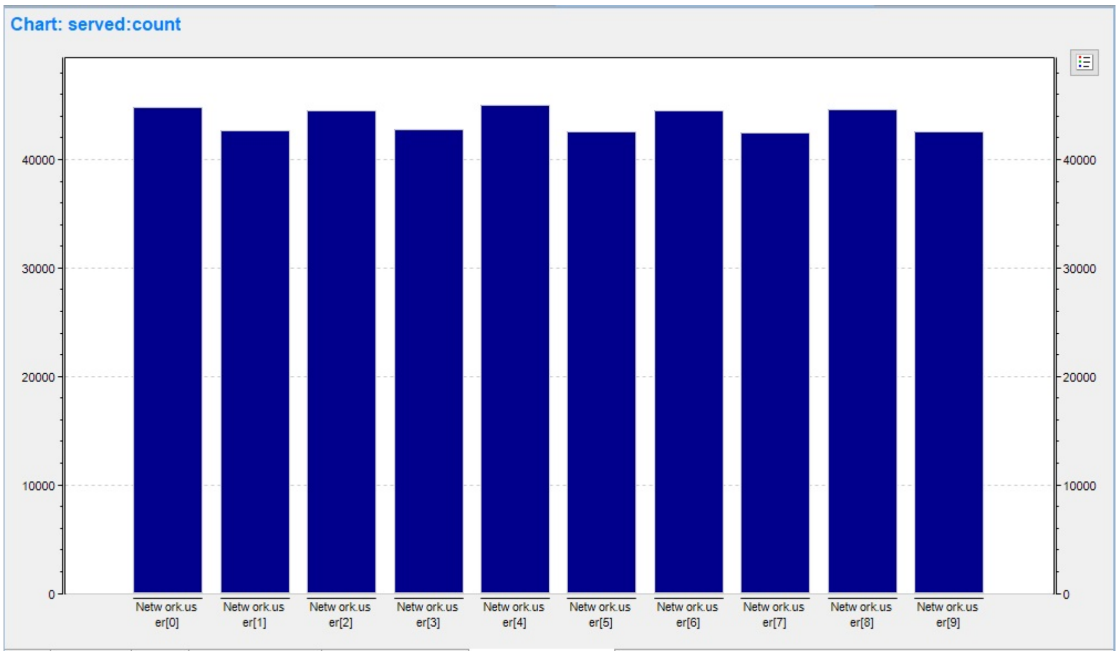


Figure 32: Number of useful timeslot for each user with new algorithm

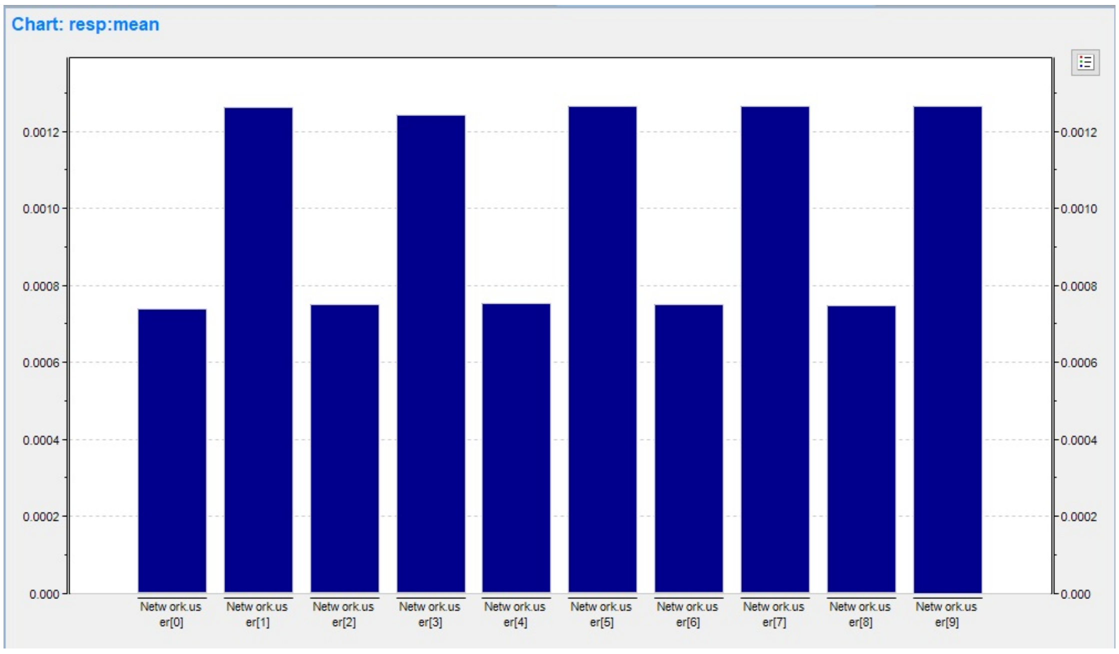


Figure 33: Average response times for each user

We performed the same analysis as before on the results from simulation. While the throughput stays the same, the response time decreases a little, thanks to the absence of delay peaks that were caused by starvation. The optimisation tends to equalize the behaviour of the two chunks of users this brings an increase of the LCG given by the difference of the two groups, however this reduces the standard deviation and as a consequence decreases the confidence intervals.

Mean inter-arrival time	**1.3ms NEW**	**1.3ms OLD**
Mean [s]	0.0010	0.0010
Mean CI 99%	$1.63 * 10^{-6}$	$1.02 * 10^{-5}$
StDev [s]	$6.34 * 10^{-6}$	$3.97 * 10^{-5}$
Average LCG [out of $\approx$ 1]	0.1905	0.1201
CI for LCG 99%	0.0010	0.0157

Table 16: Response time results from simulation

From the following Figure 34 we can easily spot out how it is influenced by the two separate behaviours:

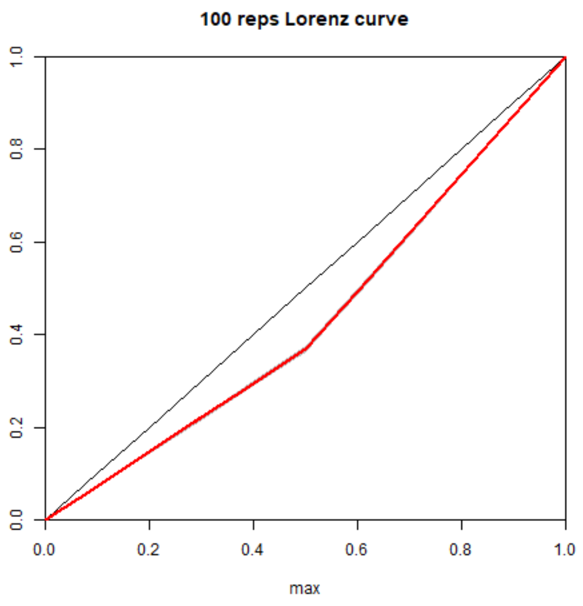


Figure 34: Lorenz Curve

Response time comparative including CI is shown in Figure 35

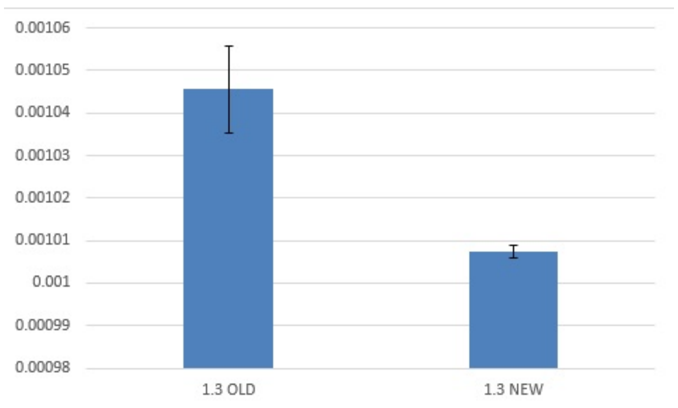


Figure 35: Response time comparative between the two solutions

Lorenz curve gap comparative including CI is shown in Figure 36

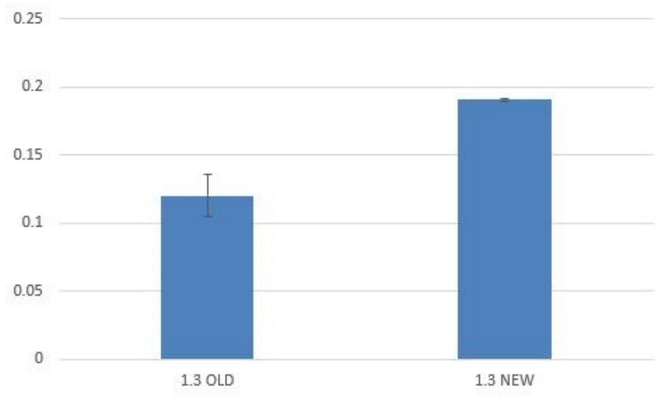


Figure 36: Lorenz curve gap comparative between the two solutions

8 Conclusions

From this work, we obtained these important insights about the system we analysed:

- While stability for uniform CQIs only depends on the mean arrival rate, due to the fact that users report, on average, the same CQI, the stability for binomial CQIs depends also on the success probability of users, in particular on the lowest one.
- Uniform CQIs can undergo an heavier workload w.r.t binomial CQIs, that are highly limited by the users with lower CQIs, thus having a lower maximum throughput. However, given a workload in which both uniform and binomial scenarios match stability condition, the latter's users experience,on average, a lower end-to-end delay, due to the fact that stability is shaped upon the less performing users, while having other users that perform far better than unlucky users and uniform ones, where stability is shaped at the same way for all the users.
- Using a weighted historical average for received bytes counting, we managed to increase the - already good - system fairness both in uniform, having all users being served equally, and in binomial, having the two classes of users being internally served as uniform ones. However in the latter case, due to this optimisation, the differences between the two class of users - response time and useful timeslots - is more clean-cut, leading to a slightly lower fairness of the overall system.