Quantitative Methods & Simulation

Final Project

11 - 5 - 2016

Gustavo Ferrufino - A00812572
Manuel Sañudo - A01192241
Abraham Rodriguez - A01195653

Introduction

In this project we will study different movie video files to better understand what would be needed in terms of servers and bandwidth to be able to create and run a service for video on demand. A video on demand service is a service like Netflix, where a user can select a movie or video that they want to watch, and the service will then have to serve, via the internet, said video to the client. Because the service is run via the internet, it is extremely important to optimize the amount of servers needed to run the service to be able to serve all videos that clients request without any lag or drop in framerate, but also to not have too many servers so that you will be wasting money paying for them to be idle most of the time because there isn't enough demand. First we'll do a statistical analysis of one movie randomly chosen, followed by its simulation and modeling. Finally, we'll evaluate the results and show our conclusions.

In our simulations we used data from the Technical University of Berlin, where they publish the frame size of severals movies and videos. We choose to make use of the high quality video trace files, this because we want our video on demand service to have the highest quality that we can offer.

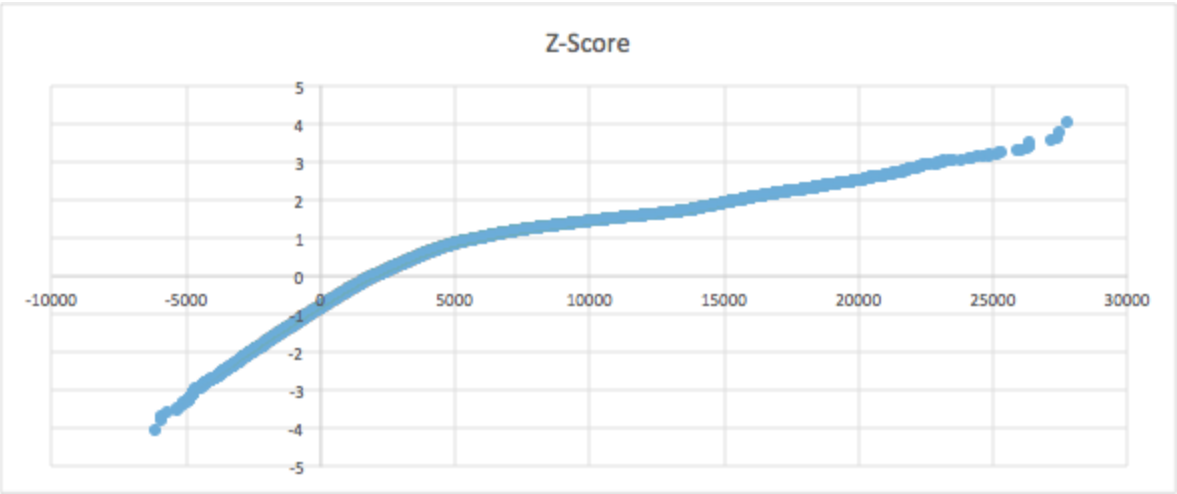
Statistical Analysis

a) Principal Components analysis

Excel named phase 1

b) Determine Normality of data

Probability Diagram:



It is a heavy tail distribution, since the value continue to increase, furthering away from x-axis.

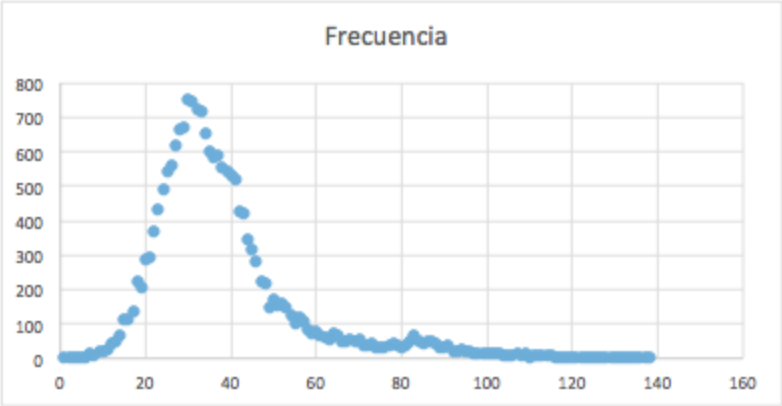
a) Descriptive statistical analysis

i)

Mean	2903.41336
Variance	18022834.08
Std. Deviation	4245.330856
Symmetry	1.763618696
Kurtosis	4.04554318

ii) Since the Kurtosis is greater than 3, it is identified as a leptokurtic. This could be easily identified as a heavy tail.

iii) Histogram



Light-tail, both tails are very small/close to x-axis.

iv) Lognormal distribution, since we can see an immediate increase at a start of the graph and then a continuous decrease once it reaches a lower value in y-axis.

b) Adjustment of probability distribution

- Normal

Distribution:	Normal
Log likelihood:	-185655
Domain:	-Inf < y < Inf
Mean:	9229.94
Variance:	1.80001e+07

Parameter	Estimate	Std. Err.
μ	9229.94	30.7803
σ	4242.65	21.7658

Estimated covariance of parameter estimates:

	μ	σ
μ	947.424	-9.57112e-13
σ	-9.57112e-13	473.749

- Gamma

Distribution:	Gamma
Log likelihood:	-182623
Domain:	0 < y < Inf
Mean:	9229.94
Variance:	1.47572e+07

Parameter	Estimate	Std. Err.
a	5.77288	0.0575999
b	1598.84	16.6673

Estimated covariance of parameter estimates:

	a	b
a	0.00331775	-0.918878
b	-0.918878	277.798

- Weibull

Distribution:	Weibull
Log likelihood:	-184336
Domain:	0 < y < Inf
Mean:	9245.07
Variance:	1.85927e+07

Parameter	Estimate	Std. Err.
a	10437	35.4339
b	2.27075	0.0114173

Estimated covariance of parameter estimates:

	A	B
A	1255.56	0.136826
B	0.136826	0.000130354

- Lognormal

Distribution:	Lognormal
Log likelihood:	-182279
Domain:	-Inf < y < Inf
Mean:	9224.24
Variance:	1.64728e+07

Parameter	Estimate	Std. Err.
μ	9.0411	0.00305204
σ	0.420683	0.0021582

Estimated covariance of parameter estimates:

	μ	σ
μ	9.31493e-06	-2.02651e-19
σ	-2.02651e-19	4.65783e-06

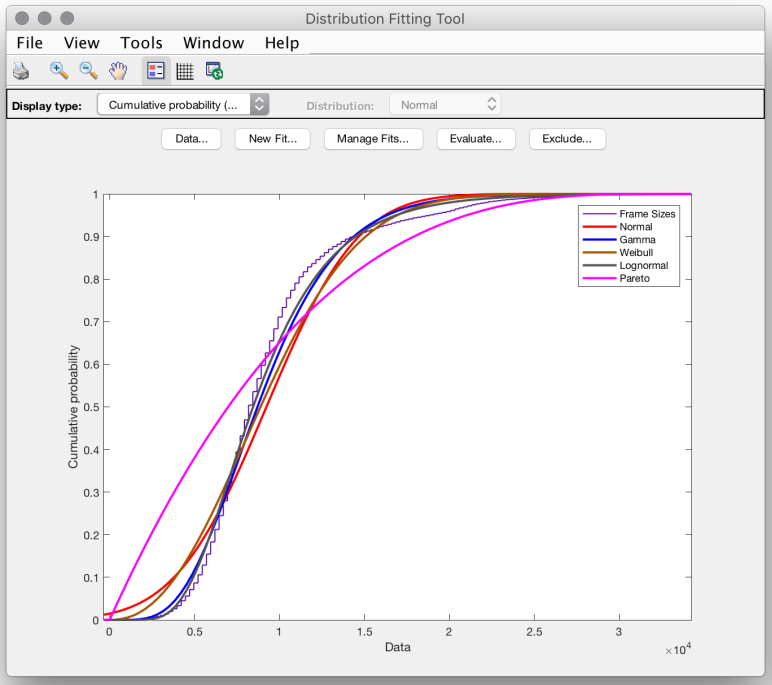
- Pareto [θ = 0]

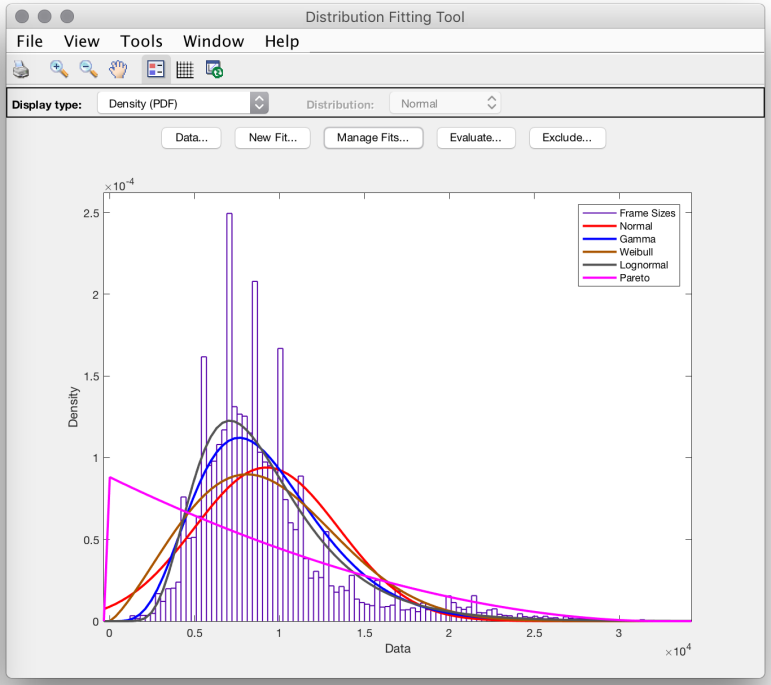
Distribution:	Generalized Pareto
Log likelihood:	-189708
Domain:	-Inf < y < Inf
Mean:	8394.9
Variance:	4.13952e+07

Parameter	Estimate	Std. Err.
k	-0.351238	0.0026371
σ	11343.5	82.6561
θ	0	0

Estimated covariance of parameter estimates:

	k	σ	θ
k	6.95431e-06	-0.214938	0
σ	-0.214938	6832.03	0
θ	0	0	0


Distribution Fitting Tool
File View Tools Window Help
Display type: Cumulative probability (...
Distribution: Normal
Data... New Fit... Manage Fits... Evaluate... Exclude...
Frame Sizes
Normal
Gamma
Weibull
Lognormal
Pareto
Cumulative probability
Data
×10^4


Distribution Fitting Tool
File View Tools Window Help
Display type: Density (PDF)
Distribution: Normal
Data... New Fit... Manage Fits... Evaluate... Exclude...
×10^-4
Frame Sizes
Normal
Gamma
Weibull
Lognormal
Pareto
Density
Data
×10^4

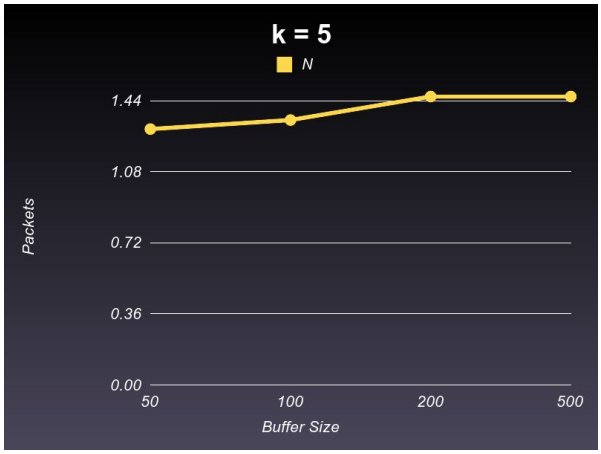
Simulation

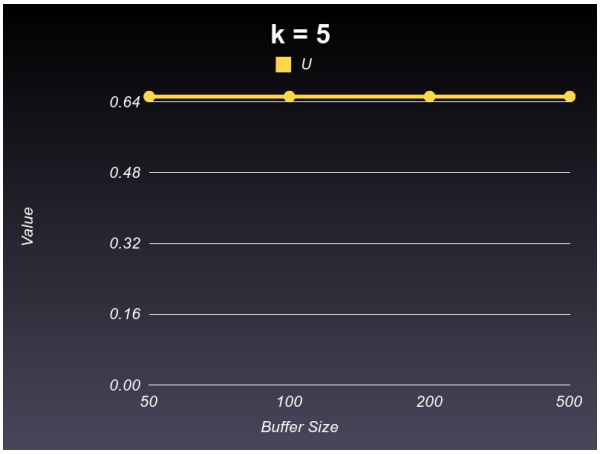
For the saturation probability be equal or less than 0.05, 0.01 or 0.001 n can have any of the following values tested: 50, 100, 250, 500. This implies that the maximum capacity of the buffer irrelevant of the limit chosen between the four cases, it'll have a low probability of saturation. Besides that we've found 3 scenarios which the probability of saturation is completely 0. Which is in the case of n=250 & k=5, n= 500 & k=5 , n= 500 & k=10.

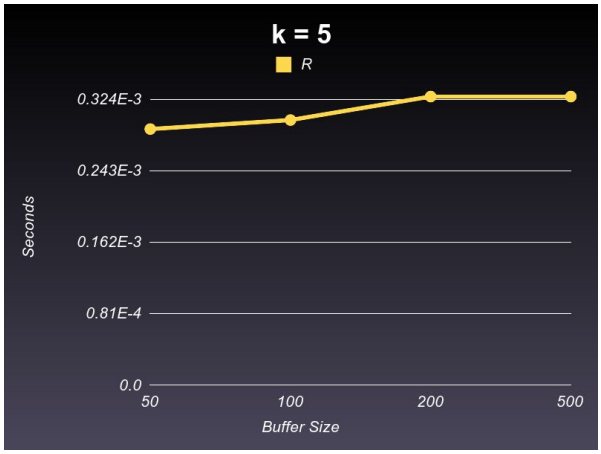
Simulation brief results:

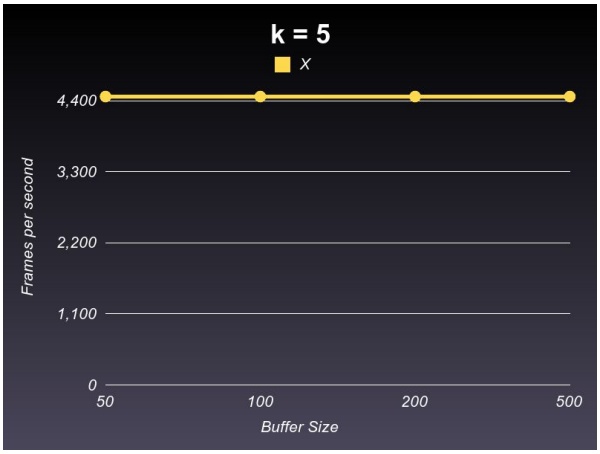
k	n	N	U	R	X	probability buffer full
5	50	1.29612716	0.65180841	0.00029035	4463.980911	0.000038
10	50	1.37630067	0.65186835	0.00031147	4418.773876	0.000086
15	50	1.39535197	0.65189716	0.000319	4374.094519	0.000141
20	50	1.58424883	0.65190715	0.00036605	4327.951524	0.000254
5	100	1.34188436	0.65181457	0.0003006	4464.086509	0.000026
10	100	1.43795684	0.6518754	0.00032541	4418.924514	0.000074
15	100	1.4815092	0.65191092	0.00033866	4374.599519	0.000121
20	100	1.69299513	0.65195691	0.00039108	4329.078758	0.000168
5	250	1.46088417	0.65182967	0.00032726	4464.031509	0
10	250	1.64724501	0.65189363	0.00037279	4418.737673	0.000038
15	250	1.7494217	0.65193245	0.00039989	4374.788519	0.000086
20	250	1.99388367	0.65197663	0.00046052	4329.618524	0.000133
5	500	1.46088417	0.65182967	0.00032725	4464.119509	0
10	500	1.72291072	0.65191669	0.00038991	4418.686514	0
15	500	2.16020144	0.65197019	0.00049379	4374.779519	0.000026
20	500	2.49521448	0.65201215	0.00057636	4329.250524	0.000074

Graphs for k = 5

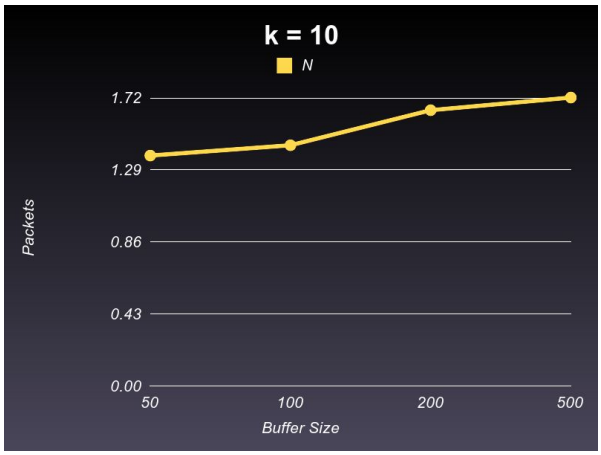


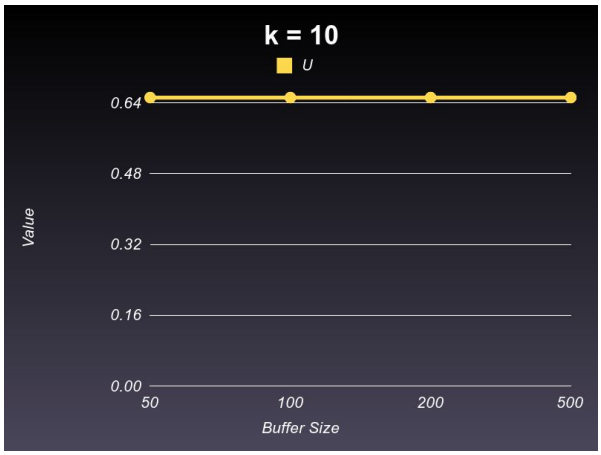


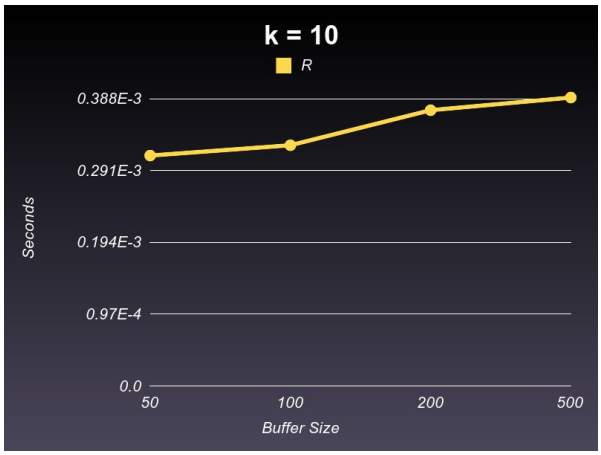


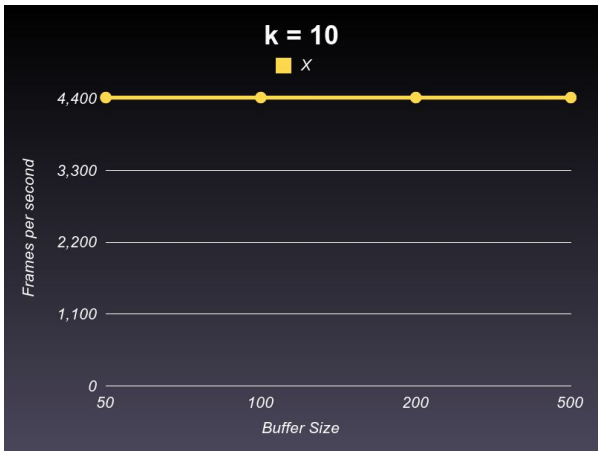


Graphs for k = 10

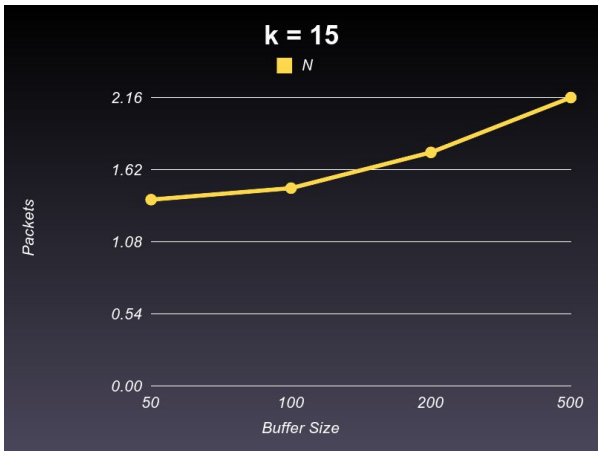


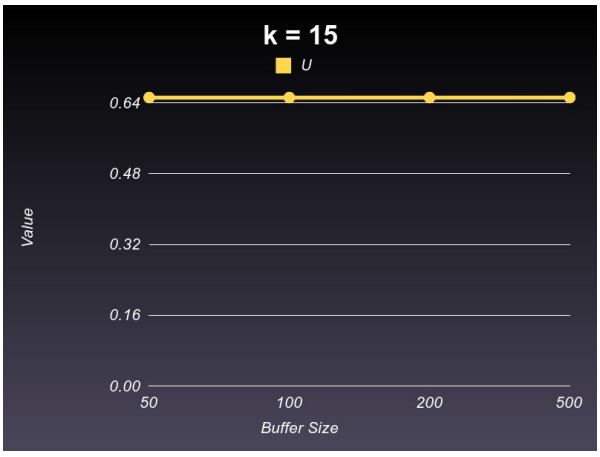


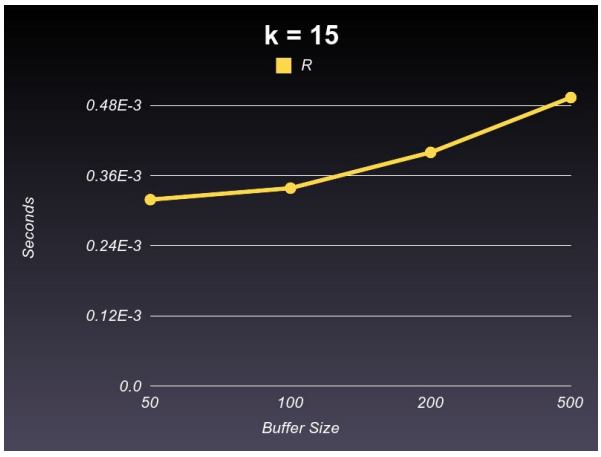


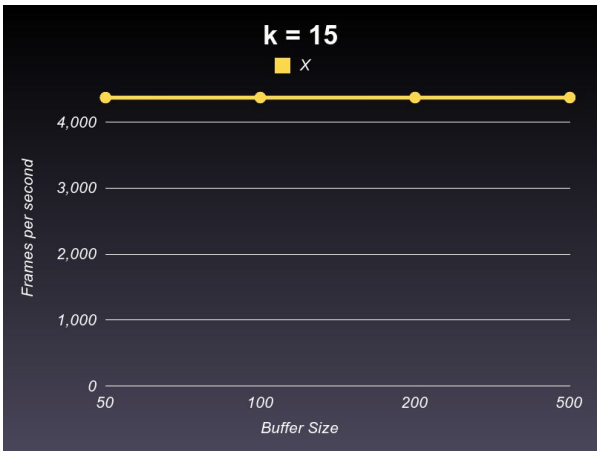


Graphs for k = 15

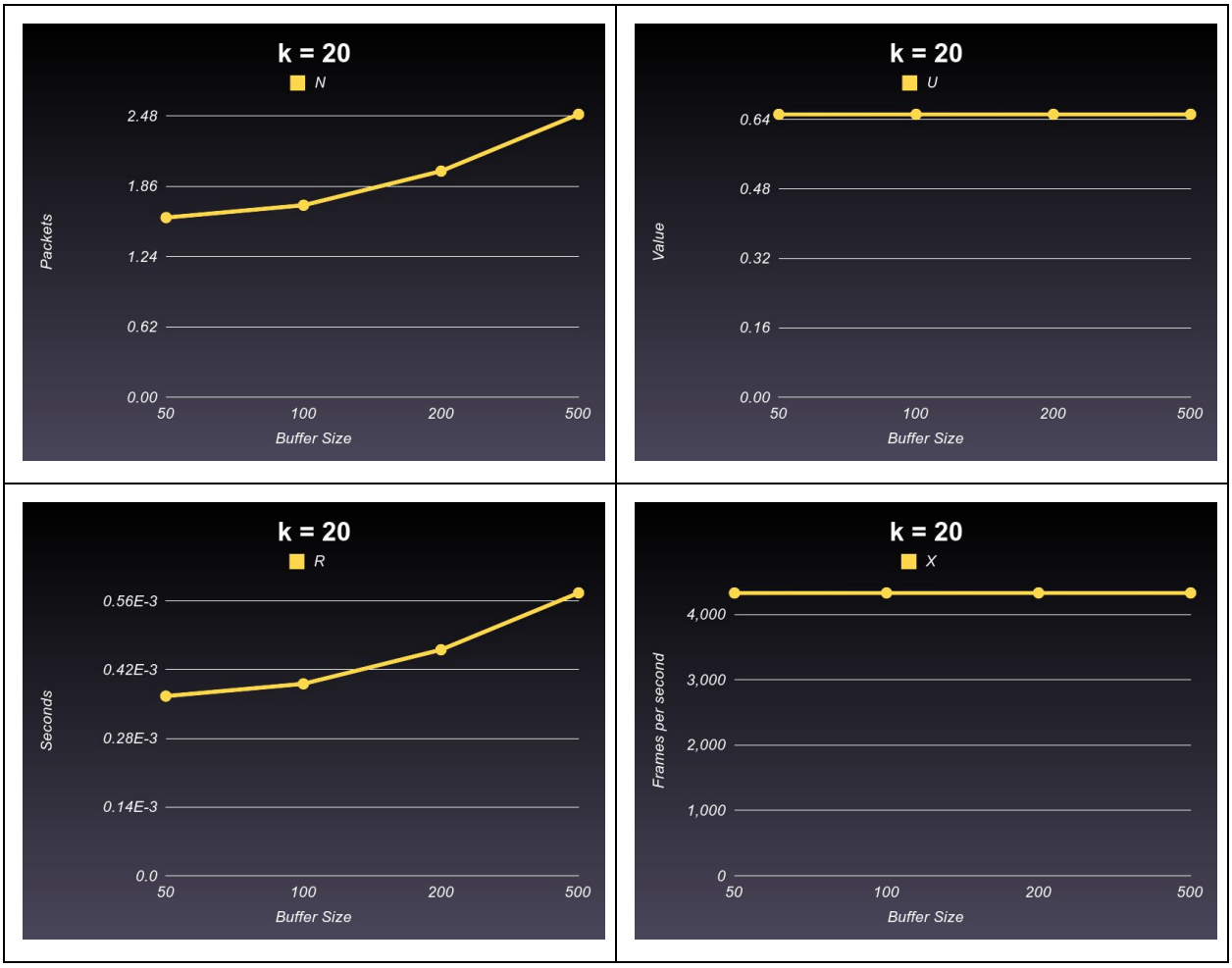






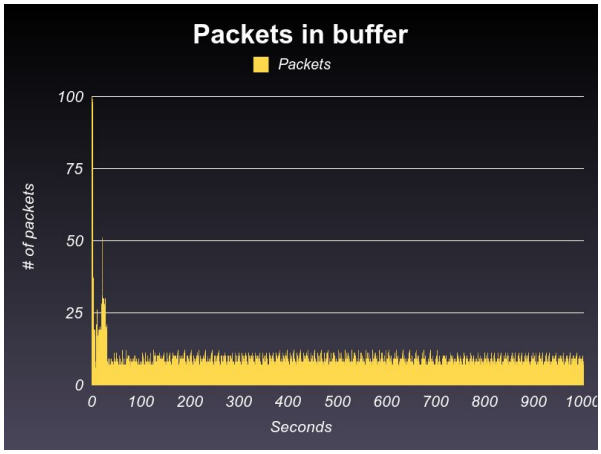
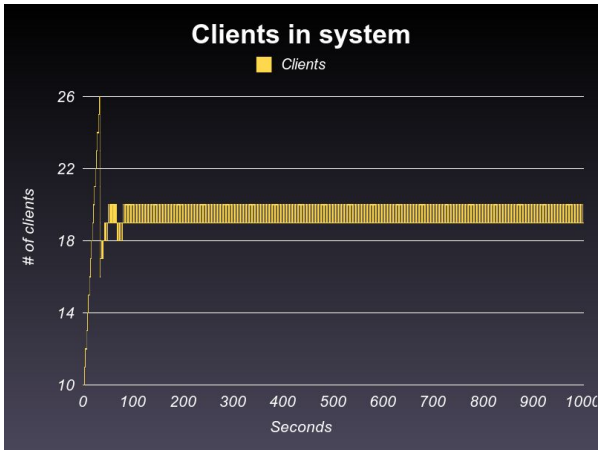
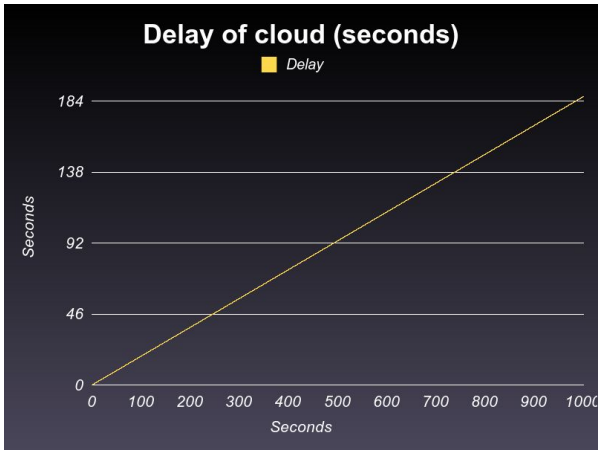


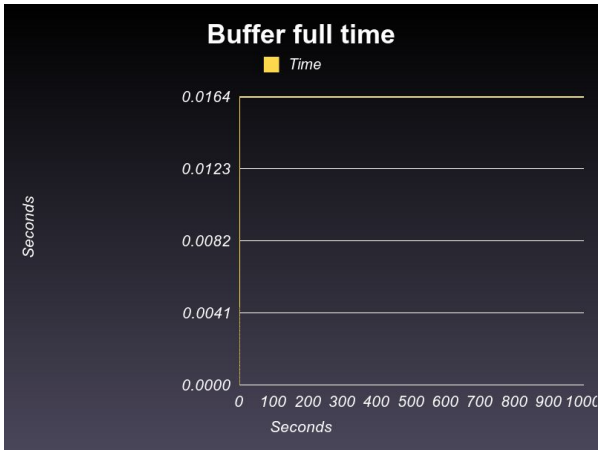
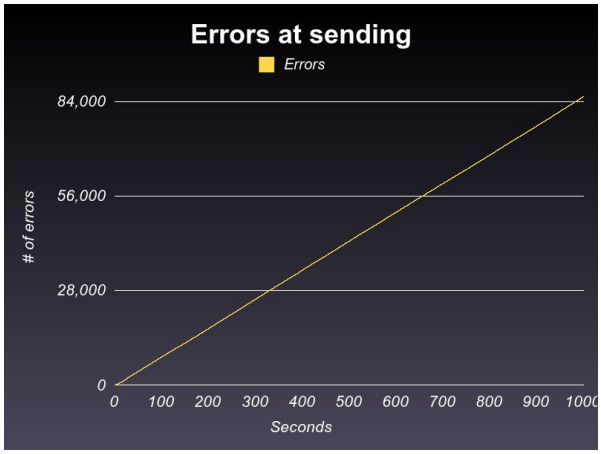
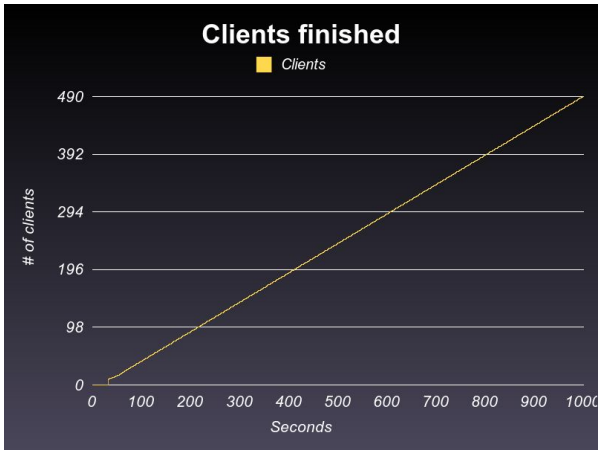
Graphs for k = 20



For all the simulations there is a maximum quantity of clients of 1000, that is because of the way that the error of transmission is calculated (error = 0.001 × number of clients), if the quantity of clients is greater than 1000, the error will be 100% and the system will crash. Although there is a limit of clients, in our simulations we never reached this limit.

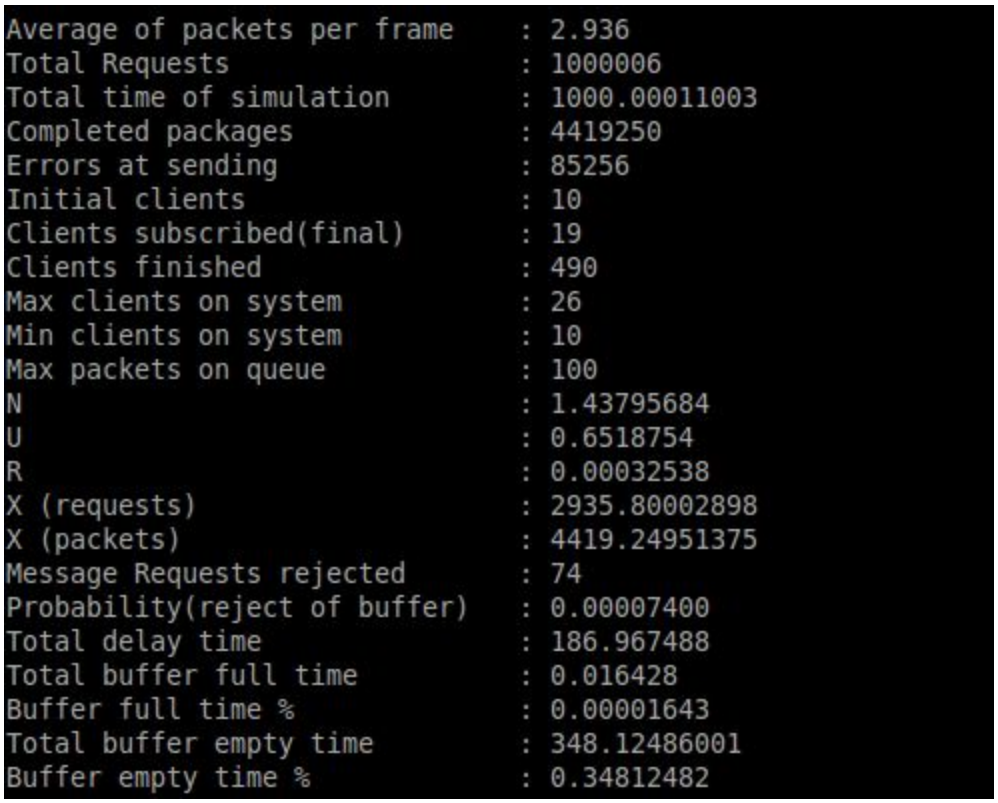
We chose to run the simulation with a value of 10 for k and buffer size of 100, we thought that this would showcase a good sample to analyze the data. From this simulation we created some graphs:

	With this we can see that, at first, the buffer will tend to full, but then the buffered packets never exceed something like 15 packets. This is great because with a buffer of 100 is more than enough to the system (well in our simulation with Jurassic Park, it could exist a case where the frames of the video are very large and the buffer saturate)
	With the help of this graph, we can see that the minimum clients on the system is at the start, then, there is a high value, and in that moment the clients start to leave the system. After a few seconds the system stabilizes and the number of clients is always in the range of 19-20
	We can see with this, that the number of clients in the system doesn't matter, the delay of the cloud will always be constant. This is of great help because with this we can create strategies to counter this delay. If the delay was exponential, there will be problems for the clients because they will experience a slow system if there were a lot of clients in it.

	With the help of this graph, we can see that the buffer is just full at the very beginning and then the buffer is never full again. This because the time of doesn't increase
	This is similar to what happened in the delay time, the errors increase constantly, and this help us to counter it. This graph is very helpful because this mean that the system will not saturate and crash because of the errors (at least with Jurassic Park)
	This graph wasn't expected, because we think that, with the time, the clients will take longer to leave the system, but, because of the clients in the system always vary in a little range, the clients finishing will also increase with little variation

Modelation

Based on this specific scenario from the simulation:



```
Average of packets per frame    : 2.936
Total Requests                  : 1000006
Total time of simulation        : 1000.00011003
Completed packages              : 4419250
Errors at sending               : 85256
Initial clients                 : 10
Clients subscribed(final)       : 19
Clients finished                : 490
Max clients on system           : 26
Min clients on system           : 10
Max packets on queue            : 100
N                               : 1.43795684
U                               : 0.6518754
R                               : 0.00032538
X (requests)                    : 2935.80002898
X (packets)                     : 4419.24951375
Message Requests rejected       : 74
Probability(reject of buffer)   : 0.00007400
Total delay time                : 186.967488
Total buffer full time          : 0.016428
Buffer full time %              : 0.00001643
Total buffer empty time         : 348.12486001
Buffer empty time %             : 0.34812482
```

Operative Laws

Calculate NURX
For this calculation, we defined a utilization of 0.6518754 and a service time of 0.000222 seconds. The comparison within the throughput will be based on the requests.

a) Number of mean jobs in buffer

N = U / (1 - U)
N = 0.6518754 / (1 - 0.6518754) = 1.87

b) Utilization of the system

U = 0.6518754

c) Average residence time

$R = S / (1 - U)$

$R = 0.000222 / (1 - 0.6518754) = \underline{0.0006377}$

d) Average performance of the system

$X = U / S$

$X = 0.6518754 / 0.000222 = \underline{2236.375}$

e) Are these results consistent with those obtained in the simulation?

We used the same utilization value that we got from running the simulation with objective of comparing the results with those obtained in the simulation. The differences we notice on the results we obtained here are caused because the simulation uses a markovian system which involves errors and queues and little's law doesn't take any of those into account.

Modelation by Queue theory

Using the formulas of the various models and establishes which one best fits the system you modeled.

Mean frames / request = 2.936

M/M/1

Mean frames / request = 2.936

$\lambda = 2.936 \times 1 / 0.001 = 2936$ frames / second

$\mu = 1/0.000222 = 4504.504$ frames per second

a) Number of mean jobs in buffer

$L = \rho / (1 - \rho)$

$L = 0.6516 / (1 - 0.6516) = \underline{1.87}$

b) Utilization of the system

$\rho = \lambda / \mu$

$\rho = 2936 / 4504.504 = \underline{0.6516}$

c) Average residence time

$W = 1 / (\mu - \lambda)$

$W = 1 / (4504.504 - 2936) = \underline{0.0006375}$

d) Average performance of the system

$\lambda = 2936$ requests per second

e) Are these results consistent with those obtained in the simulation?

The results from this modeling are similar to our simulation, the only thing that differs is the residence time, we think that this is because we are calculating it with an average of frames per second, but this is not the real number of packets that each client request generates.

M/M/c

C = 2

λ = 2936 frames / second

μ = 4504.504

a) Number of mean jobs in buffer

π = $\left(1+\frac{(c\rho)^c}{c!(1-\rho)}+\sum_{n=1}^{c-1}\frac{(c\rho)^n}{n!}\right)^{-1}$

π = (1 + (2 × 0.3257)^2 / 2!(1 - 0.3257) + ((2 × 0.3257) / 1!)) ^ -1

π = 0.5572

B = $\frac{(c\rho)^c}{c!(1-\rho)}\pi_0$

B = ((2 × 0.3257)^2 / (2!(1 - 0.3257))) × 0.5572

B = 0.1753

L = c × ρ + ρ × B / (1 - ρ)

L = 2 × 0.3257 + 0.3257 × (0.1753 / (1 - 0.3257))

L = 0.73607

b) Utilization of the system

ρ = λ / (μ × c)

ρ = 2936 / (4504.504 × 2) = 0.3257

c) Average residence time

W = $\frac{1}{\mu}\left(1+\frac{B}{c(1-\rho)}\right)$

W = (1 / 4504.504) × (1 + 0.1753 / (2(1 - 0.3257)))

W = 0.0002508

d) Average performance of the system

λ = 2936

e) Are these results consistent with those obtained in the simulation?

This modeling has a lot of differences with our simulation due that our simulation not using severals servers, whereas in the modeling we used two. This difference in the definition of the system brings a lot of differences in the results of NURX, the mean jobs in buffer is less, as the utilization of the system and the residence time.

M/M/1/K

K = 100

λ = 2936 frames / second

μ = 1/0.000222 = 4504.504 frames per second

a) Number of mean jobs in buffer

$$L = \frac{\rho}{1-\rho} - \frac{(K+1)\rho^{K+1}}{1-\rho^{K+1}}$$

L = (0.6516 / (1 - 0.6516)) - ((101 × (0.6516^101)) / (1 - (0.6516^101)))

L = 1.87

b) Utilization of the system

ρ = λ / μ

ρ = 2936 / 4504.504 = 0.6516

$$U = \rho\left(1 - \frac{(1-\rho)\rho^{K}}{1-\rho^{K+1}}\right)$$

U = 0.6516 (1 - (((1 - 0.6516) × 0.6516^100) / (1 - 0.6516^(101))))

U = 0.6516

c) Average residence time

$$W = \frac{N}{\lambda\left(1 - \frac{(1-\rho)\rho^{K}}{1-\rho^{K+1}}\right)}$$

W = 1.87 / (2936 (1 - ((1-0.6516)0.6516^100) / (1-0.6516^101)))

W = 0.0006369

d) Average performance of the system

λ = 2936

e) Are these results consistent with those obtained in the simulation?

This modeling seems a lot like our simulation, this because all the results but the residence time are similar, we think that this is because we are using an average of packets generated by a client request, we're doing this because each frame can generate different number of packets. Also we observed that these results are similar to those generated with M/M/1 modeling, this is because the size of the buffer used (100) is very large, and the variations that generate this

queue are not significant. In this model, the queue is never full (we used a static arrival time of packets), but in our simulation the rate varies because of the size of the frames, so our queue is sometimes saturated. This, and the probability of error that we have in the system, are the reasons that makes the results vary, but not a lot.

M/G/1
Lq = 18
σ = 0.00001
S = servers = 1

λ = 2936 frames / second
μ = 4504.504 frames per second

a) Number of mean jobs in buffer

$L = ((\lambda \times \sigma)^2 + (\lambda/\mu)^2)/(2\times(1-(\lambda/\mu))) + (\lambda/\mu)$
L = 1.2630

b) Utilization of the system

$\rho = \lambda / (s \times \mu)$
$\rho = 2936 / (1 \times 4504.504) = 0.6517$

c) Average residence time

$W = L / \lambda$
$W = 1.2630 / 2936 = 0.0004301$

d) Average performance of the system

λ = 2936 frames / second

e) Are these results consistent with those obtained in the simulation?

Results from M/G/1 were the ones that were the least consistent with the rest, but this was expected since the way this model works is widely different from all the others. While all others had a static time between requests, this model doesn't work like that, and instead has requests served at random intervals, so this causes the results to be very different to those obtained previously with the simulation and other modeling systems.

Conclusions

This project helped us understand the benefits of using simulation in certain scenarios. This mainly goes along when you want to predict an outcome in a unpredictable analytic scenario. Saving costs and identifying possible problems in the process, the assessment of some outcomes from the simulation, must be regarded by modeling them in an analytic form. This validates the consistency of the outcome you're trying to predict. The essence of this project made us realize that we can't simply design a project and start building it right away. Some form of validations of predictions, such as statistical analysis, simulation or modeling, are very useful when trying to build a project.

When it comes to planning and developing a project it is our responsibility as engineers to be able to find the best way to develop it, resulting in an efficient program, that satisfies all requirements from the stakeholders of the project. We must satisfy the requirements from the client and provide the best quality product for the final users, all this responding in an ethical and moral manner of what it is required from us to produce. In this case we would aim to develop the most efficient program for our client, fulfilling all his requirements, also taking into account the cost that each possible way of making the final deliverable would affect the client and how it aligns with what they need.

If the client wants to reduce implementation costs, we would need to adjust to the requirements of the client, but aiming to optimize the project in a different way for better quality and performance. Now we could always discuss with the lead developer or project manager, that by investing more, we could create a better return of investment. But if capital is a problem, we must adjust the system resources to improve its performance.

For instance in the simulation we realized that we use a huge amount of memory. We should actually be aiming for an 80% of utilization, having only 60% of utilization is low use of our resources, which results in a waste of money. If we instead used less memory for the buffer, achieving an utilization of 80%, we could reallocate the residual money in improving other parts of the project.

The adequate modeling of this project allows to identify an efficient use of the resources needed for its development, having a positive impact in the sustainability of the place where the data centers are located and satisfying the objectives of the client. We are also ensuring the best user experience for our final users. This encourages a form of development that would allow a smooth and simple future incremental development by colleagues. If there was an increase of requirements in the future, the simplest solution of adding another server would be the most inadequate answer towards cost and sustainability of the environment. Instead we should evaluate the use of resources, by modeling, and conclude that an increase in memory buffer (which is cheaper and requires less components) would give a better outcome.

REFERENCES

Ben, A. M., Kroener, T., Kandil, M., Salih, A., Schafer, R., Dr., & Panchanathan, S., Prof. (2000). MPEG-4 and H.263 Video Traces for Network Performance Evaluation. Retrieved May 11, 2016, from http://www-tkn.ee.tu-berlin.de/research/trace/trace.html