Diego Garcia-Olano
Jon Lejarreta
Fall 2013, SMDE

EVALUATION OF G/G/1 SYSTEMS assignment

PART 1.

We are given a system with an Erlang arrival time distribution with parameters of rate of arrivals of 2, and k of 34, and a service time which comes from a Weibull distribution. The objective of Part 1 is to first provide a computer simulation of our given system with a sequence of different traffic factors (ρ =0.4, 0.7, 0.85, and 0.925) ranging from light to heavy loads and to carry out the calculation for the corresponding approximation using Allen-Cuneen's forumla.

First, we must set the proper mean value and the minimum possible variance to the assigned service time distribution for the traffic factor ρ under consideration.To find out the ϱ values, because we have that ϱ= E[x] / E[τ] and the arrival distribution is fully defined we can only modify the parameters from x, our service distribution.

We then have to choose the correct parameters that accomplish the best expected value, while also minimizing the variance value. Some distributions (like normal) are totally independent and hence we can choose 0, however in some others (like erlang) the variance does depends on the paramemters that also affect the mean value.

*NOTE: we incorrectly read the pdf which gave us our distributions parameters, and mistook E[staging] for k because we assumed the two parameters given were those that defined

the distribution itself. We were close to down when we caught the error, and considered changing our numbers to do that, but didn't think we would be able to turn it in on time if we did.

In our case we have that E[τ]=k / λ = 34/2 = 17 so we need to find out the parameters that best fit that E[x]=6.8 (ϱ=0.4), 11.9 (ϱ=0.7), 14.48(ϱ=0.85), 15.725 (ϱ=0.925).

We can find that out because we know in our case we have a Weibull as a service system so the mean value is E[x]=λ * Γ(1+1/k) ,   E[x] = λ * (1/k)!

while the variance is Var(X)=λ^2* (Γ(1+2/k) - (Γ(1+1/k)^2)).
We will use this information case by case.
Through out the first part of our paper, a=k=shape, and b=lambda=scale in reference to the Weibull distribution.

Case ϱ =0.4

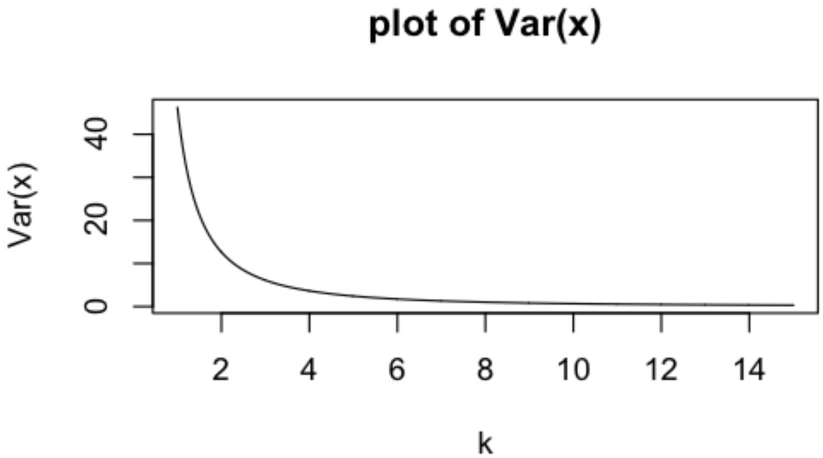
For E[x] we need that λ * (1/k)! = 6.8 while also minimizing the variance λ^2* ((2/k)! - (1/k)!^2) which is almost entirely determined by λ for k >= 1, and for k<1 dominated by k.

Because both lambda and k > 0, for E[x], (1/k)! < 6.8
and hence k > .3222 since for that value 1/k= 3.1, and (3.1)! = 6.8.
The variance will explode for any value of k < 1 due to this section of the variance equation ((2/k)! - (1/k)!^2) thus we want a k greater >=1 which will allow us to select the lowest lambda value.

For k=1, for E[x], λ = 6.8 and var(x) = 46.24.
For k=2, for E[x], λ=7.672978 and var(x) = 12.6346.
We can write λ = E[x] / (1/k)! , and substituting lambda in var(x),
we get var(x) = (E[x] / (1/k)!)^2 * ((2/k)! - (1/k)!^2).

We can plot the graph for k's in R as follows:

- tt = seq(1,15,by=.01)
- results = c(1:length(tt))
- for(i in 1:length(tt)){ a= tt[i]; results[i] = (6.8/factorial(1/a))^2 * (factorial(2/a) - factorial(1/a)^2)}
- plot(tt,results,type="l")



Thus any large k will help minimize the variance.

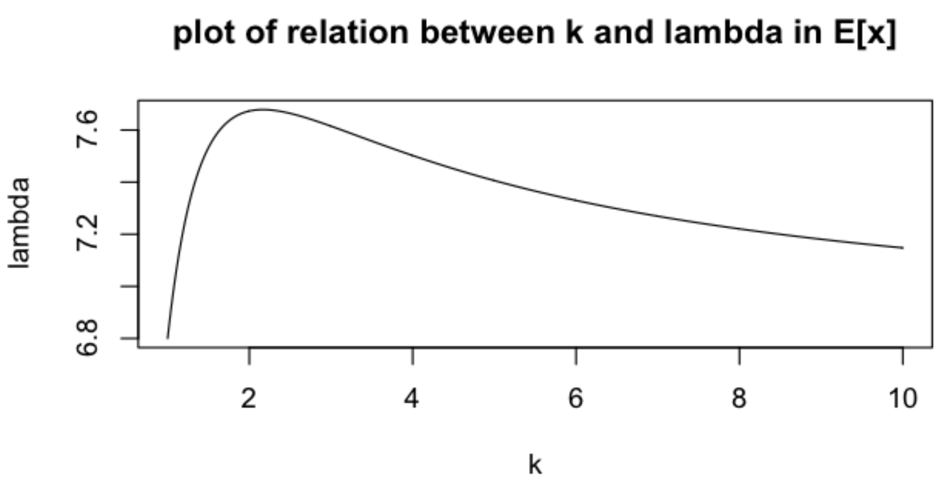
In practice, for Weibull distributions, and from our discussion in your office,

the k parameter affects the variance of the function and should be set to less than or equal 10,

and λ will manipulate E[x] and is expected to be in the ballpark of half of the desired E[x].

We can similarly plot what lambda = E[x] / (1/k)! as follows:

```
>tt = seq(1,10,by=.01);
>results=c(1:length(tt));
>for(i in 1:length(tt)){ a = tt[i]; results[i] = 6.8 / factorial(1/a)}
>plot(tt,results,type="l")
```



We see a maximum at k=2.17 with lambda=7.67838.

This gives us a variance of 10.91164 which is lower than that for k=1 and k=2 previously calculated. However choosing the highest k acceptable at k=10 with lambda 7.147732

gives us a much lower variance of 0.6693 so we go with it.

As a sanity check in R, we could do the following to check if these values will give us the E[x] we suspect given a randomly generated distribution as follows:

- mean(rweibull(1000000,shape=10,scale=7.147732)

[1] 6.800929

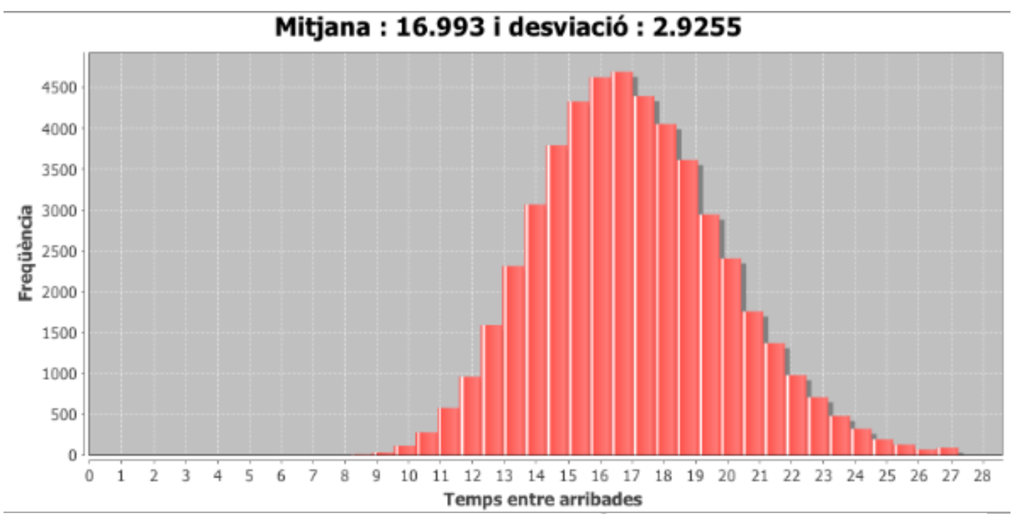
With the values for the server distribution set, we run a simulation in CUES with 50,000 clients to obtain Wq, the average waiting time in the queue for the system with an erlang arrival distribution with λ=2 and k=34, and weibull service distribution with shape = 10 and scale=7.147732

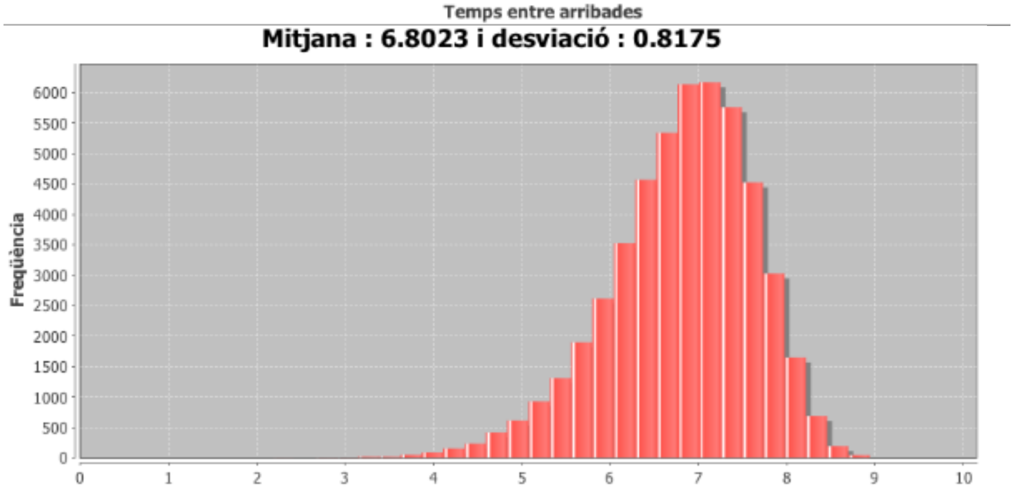
Doing so gives us the following table of magnitudes, and plots for our arrival and service distributions.

Magnituds fonamentals del Model : 1

Ocupació mitjana de la cua : 0.0 (clients)
Ocupació mitjana del S.E. 0.4002 (clients)
Clients perduts per capacitat finita de la cua : 0 (clien
Clients d'espera nul·la a la cua : 50000 (clients)
Temps mig entre arribades : 16.993 (temps)
Desviació estàndar entre arribades : 2.9255 (temps)
Temps mig de servei : 6.8023 (temps)
Desviació estàndar de servei : 0.8175 (temps)
Temps mig d'espera en cua : 0.0 (temps)
Desviació mitjana d'espera en cua : 0.0 (temps)
Temps mig de permanència en S.E. : 6.8023 (temps)
Desviació mitjana de permanència en S.E.: 0.8175 (te
Rho Real : 0.4002 (temps)

	Servidor1 :	0.4001
	Servidor2 :	********
Fracció (per u) d'ús del servidor :	Servidor3 :	********





This gives us an average service time of 6.8023 and Wq = 0.0 with a load factor of .4001

We then need to calculate a corresponding approximation using Allen-Cunneen's formula, which is as follows:

$$\mathrm{E}[w_q] = W_q \approx \frac{C(s,\theta)(\lambda^2\sigma_\tau^2 + \mu^2\sigma_x^2)}{2s\mu(1-\rho)}$$

where

$$C(s,\theta) = P_{M/M/s}(N \geq s) = \frac{\frac{\theta^s}{s!(1-\rho)}}{\sum_{\ell=0}^{s-1}\frac{\theta^\ell}{\ell!} + \frac{\theta^s}{s!(1-\rho)}} \quad , \quad \theta = \frac{\lambda}{\mu}$$

s = λ / (μ * ρ) ,
λ = 1 / E[τ] ,
μ = 1 / E[x].
In our case E[τ] = 17 so λ=1/17 ≈ .0588.
Similarly, E[x] = 6.8, so μ=1/6.8 ≈ .147 ,
and then, θ = λ/μ = .4
and s = λ / (μ * ρ) = .0588 / (.147 * .4) = 1

so C(s,θ) = (.4^1) / (1! * (1-.4)) / ((.4^0 / 0!) + ((.4^1) / (1!(1-.4))))
= (.4 / .6) / (1 + .4/.6) = 2/3 / 5 /3 = 2/5 = .4

$$\sigma_\tau^2 = \mathrm{Var}[\tau], \;\; \sigma_x^2 = \mathrm{Var}[x]$$

for Erlang, Var[τ] = 1 / (k * (1/E[τ])^2) // k / λ^2 so 34 / 2^2
= 1 / (34 * (1/17)^2)
= 8.5

for Weibull, Var[x] = $b^2\left\{\Gamma\left(\frac{a+2}{a}\right) - \Gamma^2\left(\frac{a+1}{a}\right)\right\}$ where a is shape parameter (10) and b is the scale parameter (7.147732).

hence, Var[x] = (7.147732^2) * ((2/10)! - (1/10)!^2))
= 0.6693032

and then, Wq = (C(s,θ) * (λ^2 **Var[τ] + μ^2** Var[x])) / (2μ(1-p))

= (.4 * (((.0588^2) * 8.5) + ((.147^2) * 0.6693032))) / (2*1*(1/6.8)*(1 - .4)) ≈ **0.09947565**

Similarly, we developed an R script to calculate the approximation for other traffic cases, and it is as follows:

```
p = .4                    #input load factor
```

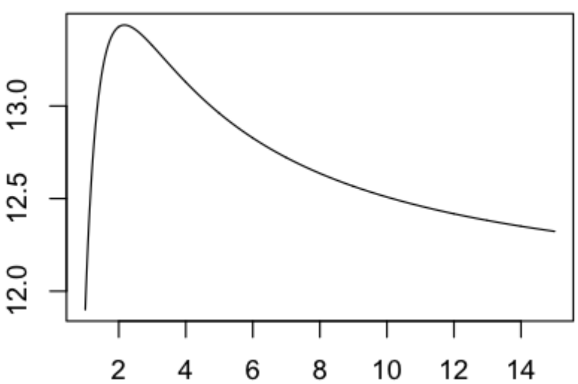
```
service_shape = 10          #input  service scale parameter
service_scale = 7.147732    #input  service shape parameter

k = 34
a = 2
Et = k / a
Ex = p * Et
lambda = 1 / Et
u = 1 / Ex
theta = lambda / u ;
cstheta = (theta /(1 - p))  / (1 + ( theta/(1-p)) )
vart = 1 / ( k * (1 / Et)^2)
a = service_shape
b = service_scale
varx =  (b^2) * (factorial(2/a) - factorial(1/a)^2)
wq = (cstheta * (((lambda^2)*vart) + ((u^2)*varx))) / (2 * u * (1-p))
```

Case ϱ =0.7

For E[x] we need that λ * (1/k)! = 11.9 while also minimizing the variance. As in the prior case, we plot lambda = E[x] / (1/k)! as follows:

```
tt = seq(1,15,by=.01);
results=c(1:length(tt));
for(i in 1:length(tt)){ a = tt[i]; results[i] = 11.9 / factorial(1/a)}
plot(tt,results,type="l")
```



We find the maximum at lambda=13.43717 and k=2.17. Which gives us a variance of 33.41692 and our expected mean of 11.9. But again if we allow for k=10, which gives us a lambda of 12.50853, and then a variance of 2.049741

With the values for the server distribution set, we run a simulation in CUES with 50,000 clients to obtain Wq,   the average waiting time in the queue for the system with an erlang arrival distribution with λ=2 and k=34, and weibull service distribution with shape = 10 and scale=12.50853 . Doing so gives us the following table of magnitudes.

Magnituds fonamentals del Model : 1

Ocupació mitjana de la cua : 0.0034 (clients)
Ocupació mitjana del S.E. 0.7039 (clients)
Clients perduts per capacitat finita de la cua : 0 (clien
Clients d'espera nul·la a la cua : 47403 (clients)
Temps mig entre arribades : 16.993 (temps)
Desviació estàndar entre arribades : 2.9255 (temps)
Temps mig de servei : 11.904 (temps)
Desviació estàndar de servei : 1.4307 (temps)
Temps mig d'espera en cua : 0.0583 (temps)
Desviació mitjana d'espera en cua : 0.3336 (temps)
Temps mig de permanència en S.E. : 11.962 (temps)
Desviació mitjana de permanència en S.E.: 1.4696 (te
Rho Real : 0.7005 (temps)

	Servidor1 :	0.7002
	Servidor2 :	********
Fracció (per u) d'ús del servidor :	Servidor3 :	********

This gives us an average service time of 11.904 and Wq = 0.0583 with a load factor of .7002

We then need to calculate a corresponding approximation using Allen-Cunneen's formula with our R code above using p=.7, service_shape=10, and service_scale=12.50853

and get a Wq = 0.6092883

Case ϱ =0.85

For E[x] we need that λ * (1/k)! = 14.48 while also minimizing the variance.

We set k to 10, and get lambda 15.22046 which gives us a variance of 3.034883 and E[x] of 14.48

With the values for the server distribution set, we run a simulation in CUES with 50,000 clients to obtain Wq, the average waiting time in the queue for the system with an erlang arrival distribution with λ=2 and k=34, and weibull service distribution with shape = 10 and scale=15.22046

Doing so gives us the following table of magnitudes.

Magnituds fonamentals del Model : 1

Ocupació mitjana de la cua : 0.0459 (clients)
Ocupació mitjana del S.E. 0.8983 (clients)
Clients perduts per capacitat finita de la cua : 0 (clien
Clients d'espera nul·la a la cua : 33935 (clients)
Temps mig entre arribades : 16.993 (temps)
Desviació estàndar entre arribades : 2.9255 (temps)
Temps mig de servei : 14.485 (temps)
Desviació estàndar de servei : 1.7408 (temps)
Temps mig d'espera en cua : 0.7805 (temps)
Desviació mitjana d'espera en cua : 1.6549 (temps)
Temps mig de permanència en S.E. : 15.265 (temps)
Desviació mitjana de permanència en S.E.: 2.4067 (te
Rho Real : 0.8523 (temps)

	Servidor1 :	0.8520
	Servidor2 :	********
Fracció (per u) d'ús del servidor :	Servidor3 :	********

This gives us an average service time of 14.485 and Wq = 0.7805 with a load factor of .8520

We then need to calculate a corresponding approximation using Allen-Cunneen's formula with our R code using p=.85, service_shape=10, and service_scale=15.22046

and get a Wq = 1.799242

Case ϱ =0.925

For E[x] we need that λ * (1/k)! = 15.725 while also minimizing the variance.

We set k to 10, and get lambda 16.52913 which gives us a variance of 3.579203and E[x] of 15.725

With the values for the server distribution set, we run a simulation in CUES with 50,000 clients to obtain Wq, the average waiting time in the queue for the system with an erlang arrival distribution with λ=2 and k=34, and weibull service distribution with shape = 10 and scale=16.52913

Doing so gives us the following table of magnitudes.

Magnituds fonamentals del Model : 1

Ocupació mitjana de la cua : 0.1777 (clients)
Ocupació mitjana del S.E. 1.1034 (clients)
Clients perduts per capacitat finita de la cua : 0 (clien
Clients d'espera nul·la a la cua : 20207 (clients)
Temps mig entre arribades : 16.994 (temps)
Desviació estàndar entre arribades : 2.9254 (temps)
Temps mig de servei : 15.733 (temps)
Desviació estàndar de servei : 1.8895 (temps)
Temps mig d'espera en cua : 3.0201 (temps)
Desviació mitjana d'espera en cua : 4.4020 (temps)
Temps mig de permanència en S.E. : 18.753 (temps)
Desviació mitjana de permanència en S.E.: 4.8029 (te
Rho Real : 0.9258 (temps)

	Servidor1 :	0.9254
	Servidor2 :	********
Fracció (per u) d'ús del servidor :	Servidor3 :	********

This gives us an average service time of 15.733 and Wq = 3.0201 with a load factor of .9254

We then need to calculate a corresponding approximation using Allen-Cunneen's formula with our R code above using p=.925, service_shape=10, and service_scale=16.52913

and get a Wq = 4.255692

As the Allen Cunneen's approximation was less precise than expected in approximating the simulators values, we additionally calculate the Kramer Langenbach-Belz correction factor.

To do so we calculate the coeffecient of variation for the arrival distribution as follows: Cτ = sqrt(Var[τ]) / E[τ] which is sqrt(8.5) / 17 = 0.1714986.

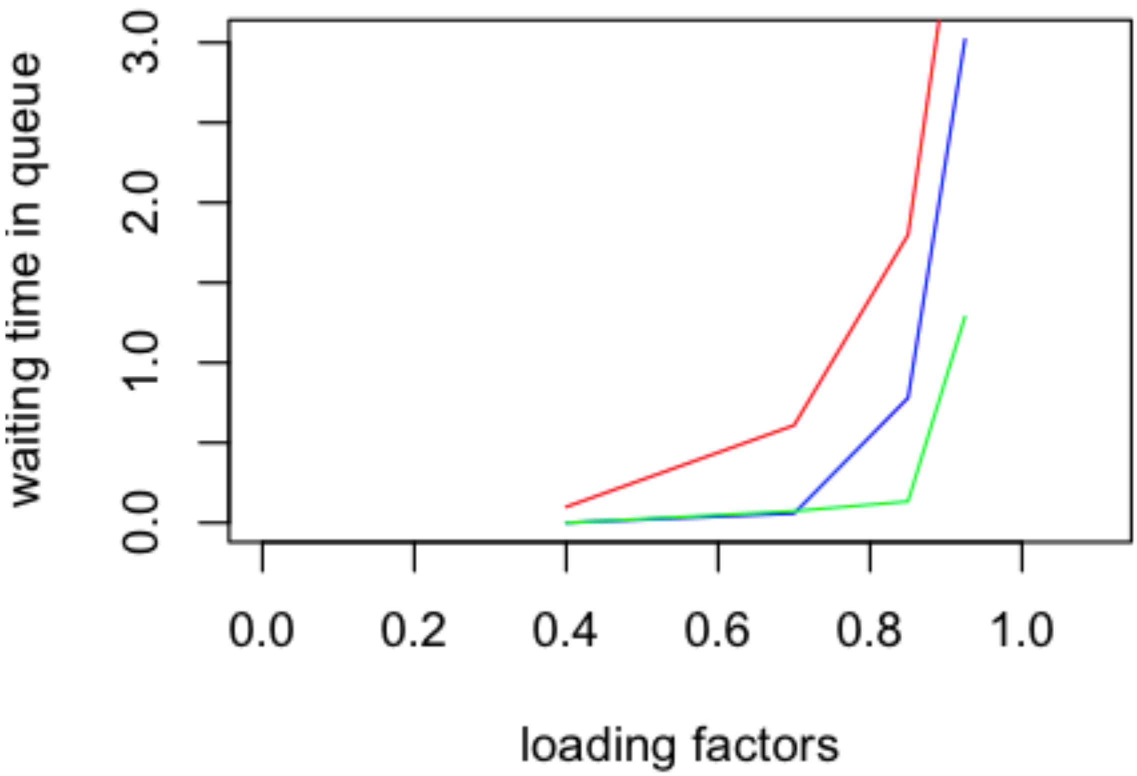
Thus we calculate the correction factor Gklb for each loading factor,
and then multiple it to the Wq approximated by Allen Cunneen's method.

$$\exp\left(-\frac{2}{3}\cdot\frac{1-\rho}{\rho}\cdot\frac{(1-c_A^2)^2}{c_A^2+c_B^2}\right), \quad 0 \le c_A \le 1,$$

for ρ=.4, the coeffecient of variation of our service distribution (Cb), is

then Gklb = e ^ (-2/3 * ((1 - p)/p) * ((1 - ca^2) / (ca^2 + cb^2))) = 0.001802019,
so our new value Wq after the correction factor is Wq * Gklb = 0.09947565 * 0.001802019 = 0
for ρ=.7, the coeffecient of variation of our service distribution (Cb), is
Cb = sqrt(Var[x]) / E[x] = sqrt(2.049741)/ 11.9 = 0.1203102, and Ca = Cτ
then Gklb = 0.1203102, and so our new value Wq after the correction factor is 0.6092883 * 0.1203102 = 0.0733036
for ρ=.85, the coeffecient of variation of our service distribution (Cb), is
Cb = sqrt(Var[x]) / E[x] = sqrt(3.034883)/ 14.48 = 0.1203102, and Ca = Cτ
then Gklb = 0.0741342, and so our new value Wq after the correction factor is 1.799242 * 0.0741342 = 0.102949
for ρ=.925, the coeffecient of variation of our service distribution (Cb), is
Cb = sqrt(Var[x]) / E[x] = sqrt(3.579203)/ 15.725 = 0.1203102, and Ca = Cτ
then Gklb = 0.3025655, and so our new value Wq after the correction factor is 4.255692 * 0.3025655 = 0.102949

Comparing the waiting time in queue values obtained via simulation and Allen-Cunneen's formula for the different loading factors calculated above is plotted below.



loading factors

ρ	Wq (Simulation - blue)	Wq (Allen-Cunneen's Approximation - red)	Allen Cunneen's with Kramer-Langenbach-Belz correction - green
.4	0	0.09947565	0
.7	0.0583	0.6092883	0.0733036
.85	0.7805	1.799242	0.1333854
.925	3.0201	4.255692	1.287626

Then, for the traffic factors ρ = 0,4 and ρ = 0,925 we repeat 10 times the simulation using different initial seed values.

Then, we determine the average value of our sample $W_q^{(1}, \ldots, W_q^{(10}$ and a confidence interval (95 %).

For ρ = 0,4, we run 10 simulations in CUES with 50,000 clients to obtain Wq, the average waiting time in the queue for the system with an erlang arrival distribution with λ=2 and k=34, and weibull service distribution with shape = 10 and scale=7.147732

We also do the same for ρ = 0.925 to the average waiting time in the queue for the system with an erlang arrival distribution with λ=2 and k=34, and weibull service distribution with shape = 10 and scale=16.52913.

We modify the seed ("les llavors") for each run.

The following is a table of the results found:

initial seed:	Wq for ρ = 0,4 a= 10 , b=7.147732	Wq for ρ = 0.925 a= 10 , b=16.52913
193945	1 -- 0.0	1 -- 3.0201
591394	2 -- 0.0000085377	2 -- 2.9302
13929	3 -- 0.0	3 -- 2.7438
382547	4 -- 0.0000134050	4 -- 2.8926
658787	5 -- 0.0000087447	5 -- 2.9819
780950	6 -- 0.0000025022	6 -- 2.8570
662592	7 -- 0.0	7 -- 2.7577
528289	8 -- 0.0	8 -- 2.9149
939174	9 -- 0.0000099645	9 -- 2.7687
517520	10 - 0.0	10 - 2.8261

We then calculate the average wait time in the queue for each loading factor in this simulation,
E[Wq] = (∑ Wq) / 10
for ρ = 0.4, E[Wq] = 0, and with standard deviation, σ= 0
and for ρ = 0.925, E[Wq] = 2.8693. with standard deviation, σ= 0.09550635.
Plotted on the same graph we see the following:

We then calculate the confidence interval of 95% for Wq, CI = E[Wq] ± SE

where SE is the t-test distribution score with 9 degrees of freedom for 97.5% confidence for a two sided test multiplied by the standard deviation of the simluation divided by the square root of number of iterations run.

For ρ = 0.4, SE = qt(.975,df=9) * (σ / √n) = 2.262157 * (0 / √10) = 0 and **CI = 0 ± 0**.
For ρ = 0.9.25, SE = qt(.975,df=9) * (0.09550635 / sqrt(10)) = 0.06832113
and **our Confidence Interval = 2.8693 ± 0.06832113**

PART 2

For part2, we maintain the same arrival distribution as before and change our service distribution to be a long-tailed Pareto distribution to evaluate the effects of service

times following a long-tailed distribution. We write a small program in R evaluating a G/G/1 system implementing the recurrence relationships provided.

We select the traffic factor ρ=.4, and

our arrival distribution from before, Erlang with k=34 and rate=2, will give us E[T] = 34/2 = 17.

We then solve ϱ= E[x] / E[T] , .4 = E[x]/ 17 and get E[x] = 6.8

The Pareto distrubtion has following probability density distribution function, constraints on variable a, and Expected value equations:

$$F_X(x) = 1 - \left(\frac{a}{x}\right)^{\alpha}, \;\; x \geq a\,(>0), \;\; E[x] = \frac{a\alpha}{(\alpha - 1)}$$

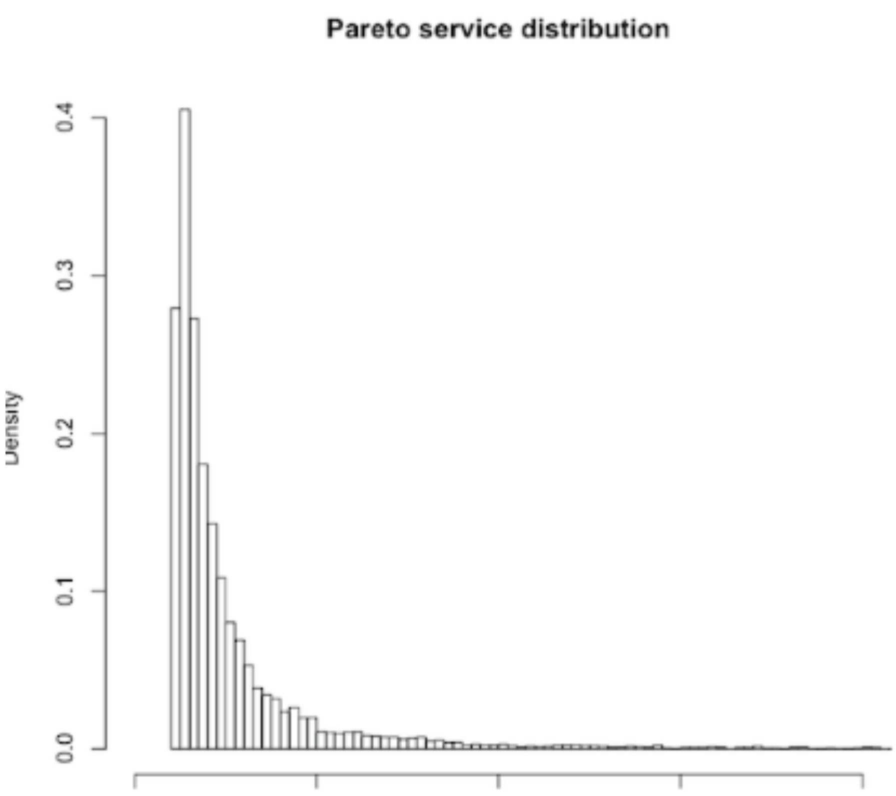
Using E[x] = (a*alpha) / (alpha-1) and the additional information that $1 < \alpha \leq 2$, we set alpha to 1.5 and thus only need to solve for the location parameter a as follows:

6.8 = (a * 1.5) / (1.5 - 1) which gives us, a = 2.26.

Now using the probability density function for Pareto, we can generate sample service times. For our simulation, to simulate the arrival Erlang distribution we used the rgamma function available within the VGAM library to generate time arrivals. To check if it was safe to do so we generated a random gamma distribution using our parameters, rgamma(n,shape=34, rate=2) , with a very large value for n (100000000) and then measured that the standard deviation divided by mean of the distribution was equal to 1 over the square root of k.

2.915471 / 17.00002 = 0.1714981 vs 1 / sqrt(34) = 0.1714986. To generate samples for the Pareto distribution, we create a random uniform distribution of size n, and then for each value in it, we calculate Fx(x) according to the probability density function with the alpha and a parameters we derived early. So Fx(x) = 1 - (2.26/x)^(1.5) in our case.

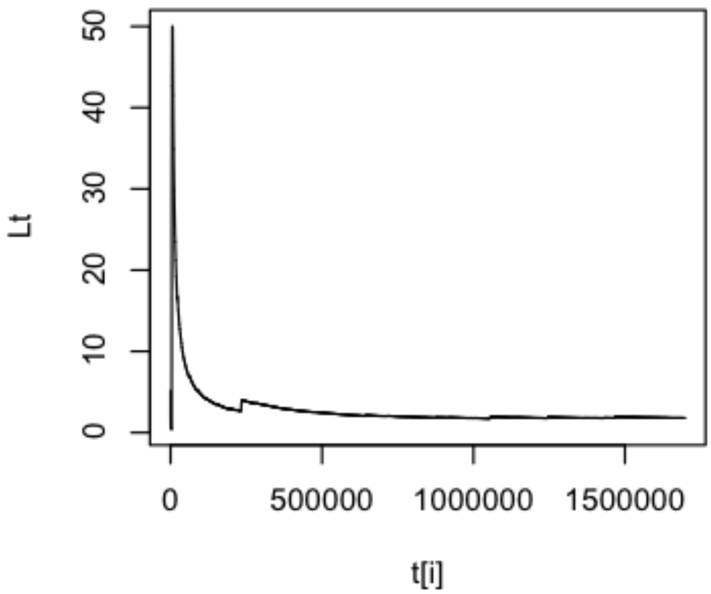
Running our code for the simulation, for 10,000 clients with the service distribution generated, we get a mean of 7.10291, a standard deviation of 42.37371, and a coefficient of variance of, sd / E[x] = 5.965683. The following is a histogram of the left hand side of our Pareto distribution,



We then run the simulation with 100,000 clients, ten times and plot some statistics and the relation between Lt and ti over time.

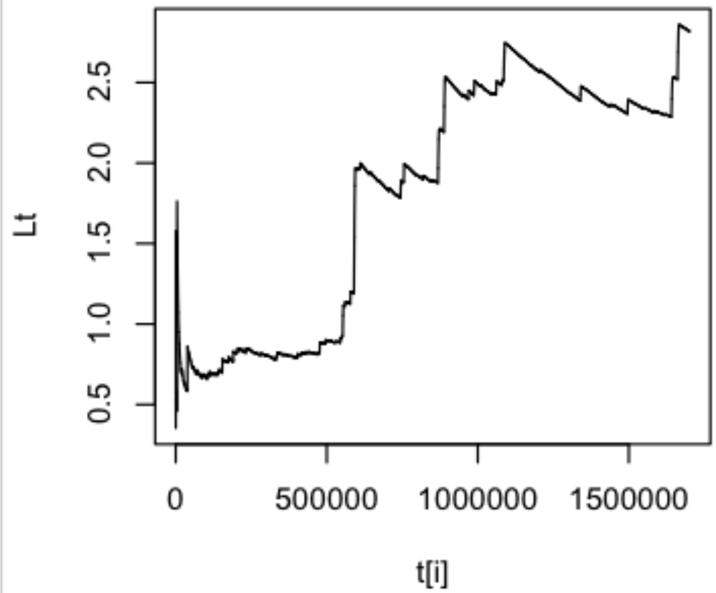
Simulation 1

```
After simulation is run:
W  =  30.5317383803844
Wq =  23.9439118281823
L  =  1.79789823075704
Lq =  1.40996612040109
```



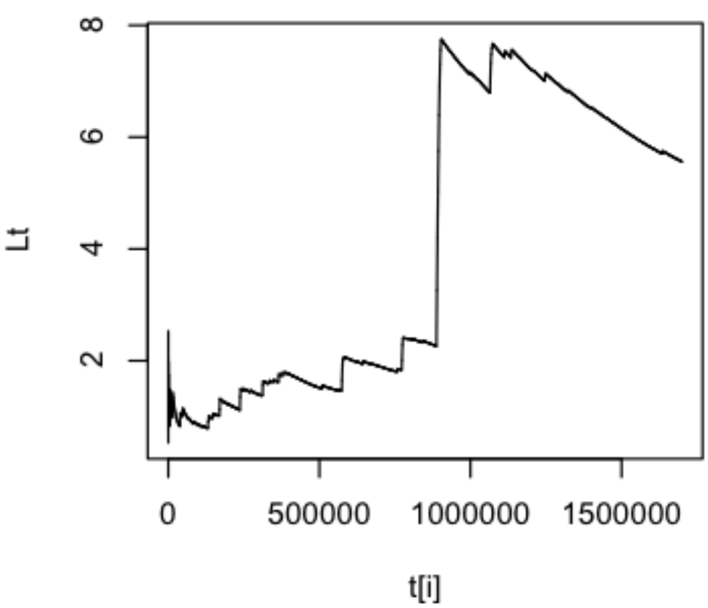
Simulation 2

```
After simulation is run:
W  =  47.9644861541299
Wq =  41.2353130983739
L  =  2.81825772691316
Lq =  2.42287052524108
```



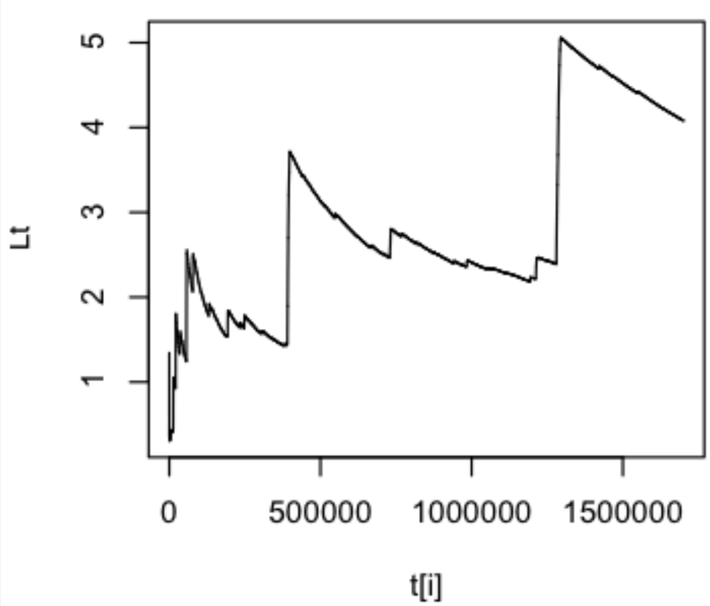
Simulation 3

```
After simulation is run:
W  =  94.5168253309329
Wq =  87.7076077724873
L  =  5.55950712223621
Lq =  5.1589869674332
```



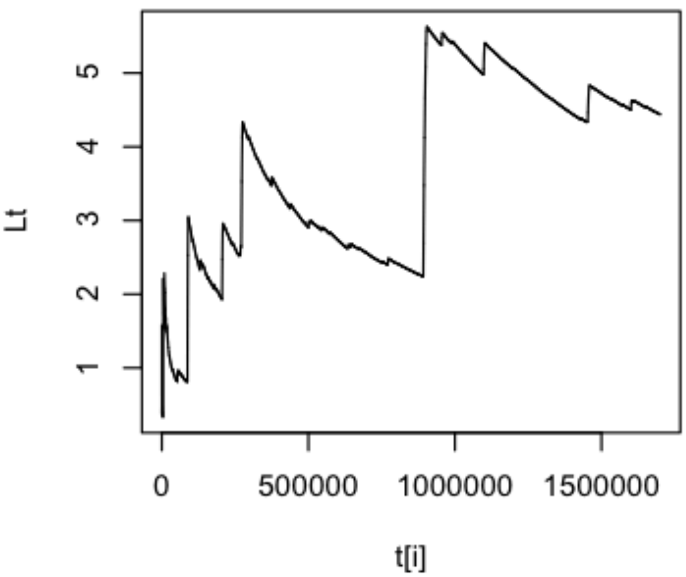
Simulation 4

```
After simulation is run:
W  =  69.4153677646815
Wq =  62.7506911371332
L  =  4.07993051292063
Lq =  3.68821008548354
```



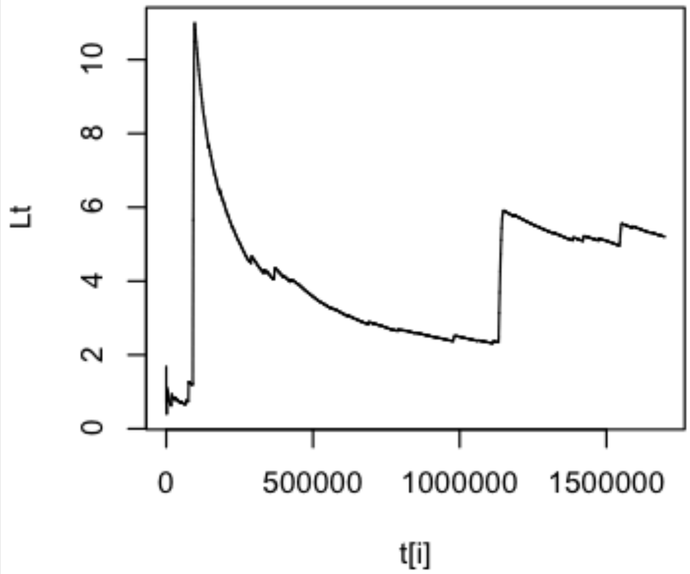
Simulation 5

```
After simulation is run:
W  =  75.4909312693974
Wq =  68.7163935809613
L  =  4.43828650091426
Lq =  4.0399957570205
```



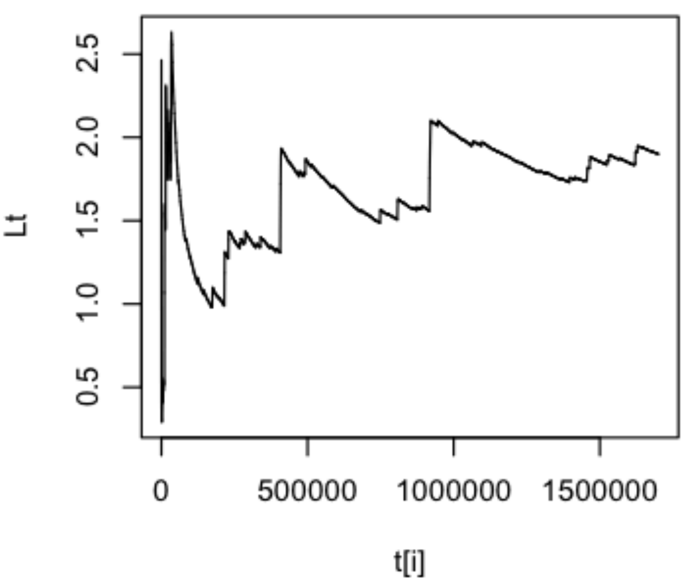
Simulation 6

```
After simulation is run:
W  =  88.3398214033416
Wq =  81.5601542773568
L  =  5.20020875279786
Lq =  4.80111711134398
```



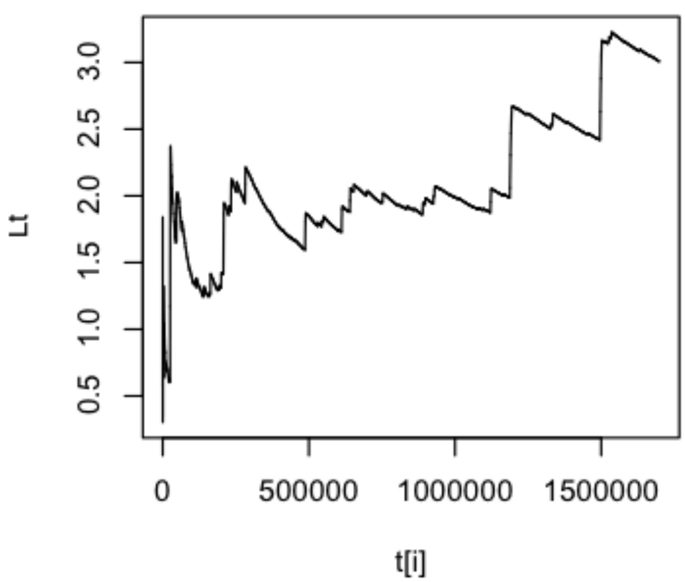
Simulation 7

```
After simulation is run:
W  =  32.3101516323302
Wq =  25.6936125673036
L  =  1.8997907800346
Lq =  1.51074773082469
```



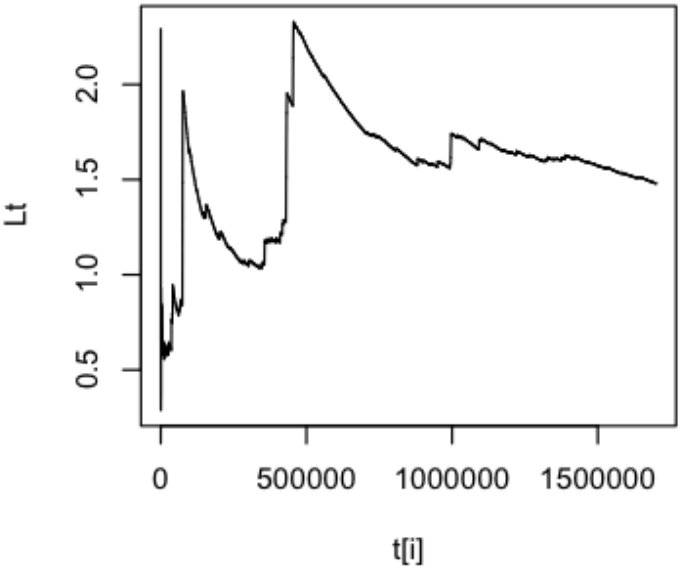
Simulation 8

```
After simulation is run:
W  =  51.1196058799188
Wq =  44.364714249337
L  =  3.00830396944089
Lq =  2.61078980720035
```



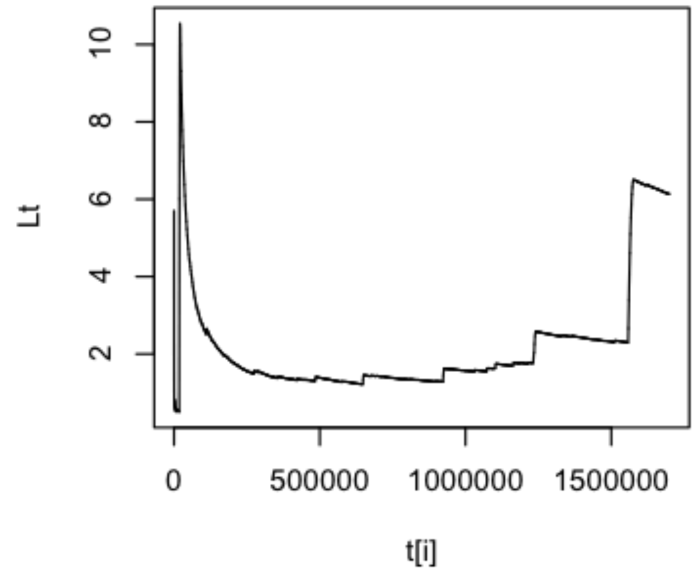
Simulation 9

```
After simulation is run:
W  =  25.1910971406935
Wq =  18.6563066049079
L  =  1.48113505201453
Lq =  1.09691568808339
```



Simulation 10

```
After simulation is run:
W  =  104.339609272041
Wq =  97.6817660364476
L  =  6.13847950036045
Lq =  5.74678707882043
```



E[Wq] = sum(Wq) / 10 = 55.23104
E[Lq] = sum(Lq) / 10 = 3.248634
σ[Wq] = 28.46328
σ[Lq] = 1.674627

We then calculate the confidence interval of 95% for Wq, CI = E[Wq] ± SE

where SE is the t-test distribution score with 9 degrees of freedom for 97.5% confidence for a two sided test multiplied by the standard deviation of the simluation divided by the square root of number of iterations run.

SE = t-test(97.5,9) * σ[Wq] / sqrt(n)
For ρ = 0.4, SE = qt(.975,df=9) * (σ[Wq] / √n) = 2.262157 * (28.46328/ √10) = 20.3614
and Confidence for Wq = 55.23104 ± 20.3614.

We then similarly calculate the confidence interval of 95% for Lq, CI = E[Lq] ± SE

SE = t-test(97.5,9) * σ[Lq] / sqrt(n)

For ρ = 0.4, SE = qt(.975,df=9) * (σ[Lq] / √n) = 2.262157 * (1.674627/ √10) = 1.197956
and **Confidence for Lq = 3.248634 ± 1.197956**.

From our findings it is clear that Pareto distribution does not stabilize over time and that it produces a large standard error for both the Wq, the average waiting time in the queue, and Lq, the average occupancy of the queue in our system. The "long tail" of the distribution causes spikes to occur at unknown times at random, causing the service time to shoot up and cause large waits in the system.

NOTES ON CODE PROVIDED

Provided in part2_r_code.R, is the code for evaluating a G/G/1 system and it implements the recurrence relationships given in the assignment document. It has a dependency on the VGAM library for use of the rgamma function for part 2 and also on the gplots library for text output on plots. Both can be installed if not already on the system, via the following commands:

```
install.packages("VGAM")
install.packages("gplots")
```