REPORT

Discrete event traffic simulation - Part 2

Daniela Faria | up201604500 | Miguel Pires | up201503463

1. A - Loss system: Erlang-B distribution

For this exercise, you can test the code doing this:

```
/*compile:
 *
 * gcc -Wall func.c LIsta_ligada.c ErlangB-a.c -o ErlangB-a -lm
 * ./ErlangB-a.c <lambda><dm><N><sim_time>
 * ex. ./ErlangB-a 200 0.008 1 2000
 *
*/
```

Figure 1: ErlangB-a: How to compile

With this we were able to determinate the probability of packet or call loss, because the ones that encountered all the N channels busy, didn't get accepted by the system. As you can see in the Figure 2, for the given N=1 (one channel available).

```
*** Simulation starting ***
*** Simulation ending ***
Lambda: 200.000000
Number of channels: 1
Simulation time: 2000.000000
Number of calls: 399788
Number of blocked calls: 245950
Probability of a call to be blocked: 61.520106
```

Figure 2: Simulation results for N=1

As you can see in the Figure 3, the probabilities are very close to the theoretical ones.

N	Probability obtained	Online Calculator
1	61.520106	61.54
2	32.869711	32.99
3	14.839371	13.96
4	5.727836	5.65
5	1.787496	1.77

Figure 3: Blocking Probabilities

2. B - Waiting system with infinite length queue: ErlangC distribution

At this exercise, we are now tacking those packages or calls that are blocked and we introduce them into a queue simulation a waiting list. This has impact in the actual time that the package will be processed: introducing the delay into to the system analysis.

We are assuming that the length of this waiting list is very close to infinity, so the only thing that interferes is the number of channels available.

In order to test the code, you can compile it in the following way.

So the value of A is obtained by the quotient between the total of calls that were in the queue and the total of the calls in the all simulation. Depending

```
/*compile:
 *
 * gcc -Wall func.c LIsta_ligada.c ErlangC-b.c -o ErlangC-b -lm
 * ./ErlangC-b.c <lambda><dm><N><sim time><probability>
 * ex. ./Erlang-C-b 200 0.008 1 2000 0.01
 *
*/
```

Figure 4: ErlangC-b: How to compile

on the resources available this probability will getting lower, because you can have more calls/packages being attended at the same time, as we can see happening in the figure bellow. You can also observe in the output from the terminal, the probability of the delay being bigger than Ax. We established Ax = 0.001 on Figure 11.

N	Probability obtained	Online Calculator
1	99.999499	100
2	70.821998	71.11
3	27.254936	27.38
4	9.119288	9.07
5	2.598701	2.59
6	2.598701	0.64

Figure 5: Probability of a packet being delayed with different N values

N	Am obtained	Online Calculator
1	117.327558	NaN
2	0.013737	0.01
3	0.001519	0
4	0.000311	0
5	5.9E-05	0
6	1E-05	0

Figure 6: Average of the delay

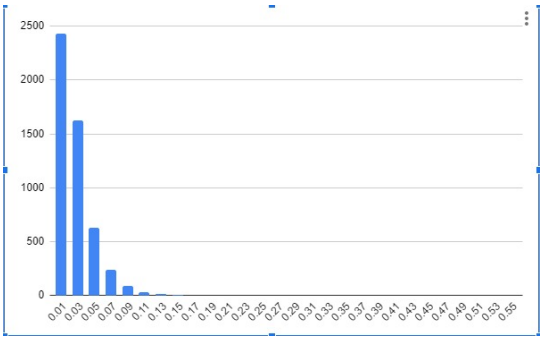

Figure 7: Delay histogram for N = 1

You can also observe in the output from the terminal, the probability of the delay being bigger than Ax. We established Ax = 0.001 on Figure 11.

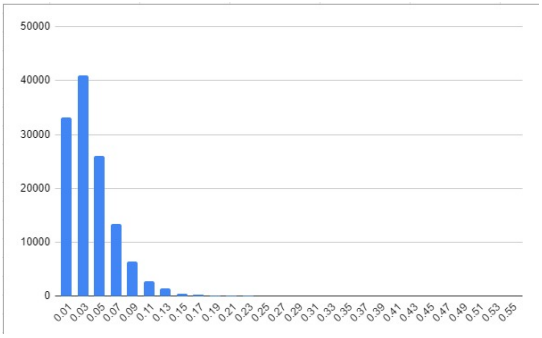

Figure 8: Delay histogram for N = 3

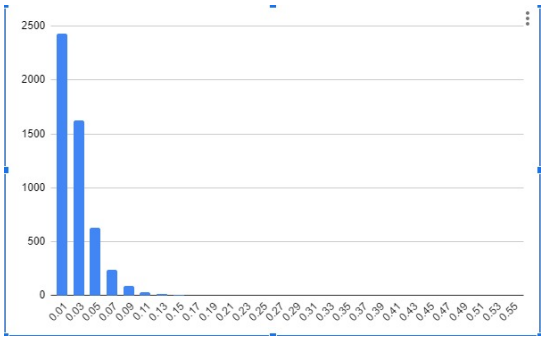

Figure 9: Delay histogram for N = 5

Figure 10: The waiting times evolution with the number of channels available

```
*** Simulation starting ***!
*** Simulation ending ***!
Total delayed: 200455
Number of calls: 200456
Total delays: 23506700.656009
Delay avarage: 117.266136
Delay probability: 99.999501
Probability of the delay be greater than 0.001000 is: 62.236601
Init the confi of file histogram.csv
```

Figure 11: Probability of delay being bigger than Ax = 0.001, when N=1

3. C - General case - waiting system with finite length queue

The Figure 12 show the way you should compile and execute the code provided to this exercise resolution.

First, we can analyse the particularities of the previous exercises if for the A we say that the length of the waiting list is L = 0 (Figure 13) and for the B we can say that the length take a very big value (Figure 14).

For the determination of the length of the package loss to be one percent, we made by tentative fail of successive queue lengths. We used N = 2.

```
/*compile:
 *
 * gcc -Wall func.c LIsta_ligada.c ErlangC-c.c -o ErlangC-c -lm
 * ./ErlangC-c <lambda><dm><N><sim_time><probability><Queue_size>
 * ex. ./ErlangC-c 200 0.008 1 2000 0.01 1
 *
*/
```

Figure 12: ErlanC-c: How to compile

```
*** Simulation starting ***!
*** Simulation ending ***!
Total delayed: 0
Number of calls: 200291
Total delays: 0.000000
Delay avarage: 0.000000
Delay probability: 0.000000
Probability of the delay be greater than 0.001000 is: 0.000000
Queu size:0
Probability of losing a package: 32.974522
Total of calls blocked: 66045
Init the confi of file histogramc.txt
```

Figure 13: Probability of the packets to be blocked when N=2 and L=0

```
*** Simulation starting ***!
*** Simulation ending ***!
Total delayed: 142008
Number of calls: 199762
Total delays: 2890.121740
Delay avarage: 0.014468
Delay probability: 71.088595
Probability of the delay be greater than 0.001000 is: 67.639992
Queu size:1215752192
Probability of losing a package: 0.000000
Total of calls blocked: 0
Init the confi of file histogramc.txt
```

Figure 14: Probability of the packets to be blocked when N=2 and L=100000000

For L=10 we obtained a probability of loosing a package 1.57 percent

```
*** Simulation starting ***!
*** Simulation ending ***!
Total delayed: 135899
Number of calls: 199880
Total delays: 2070.958421
Delay avarage: 0.010361
Delay probability: 67.990294
Probability of the delay be greater than 0.001000 is: 64.302081
Queu size:10
Probability of losing a package: 1.574445
Total of calls blocked: 3147
Init the confi of file histogramc.txt
```

Figure 15: Probability of the packet losing when N=2 and L=10

For L=12 we obtained a probability of loosing a package 1.03 percent and for L=13 probability gets 0.77.

So to have a probability of loosing a packet the best size for the queue should be L = 12.

4. References

Virtual Calculator ://erlang.chwyean.com/erlang/erlangB.html