Xi'an Jiaotong-Liverpool University

西交利物浦大学

EEE413 DATA COMMUNICATION AND COMMUNICATIONS NETWORKS

Coursework1: Simulation of the M/M/M queueing system

Student Name Xun Xu

Student ID 1926930

2019/11/24

1.Theoretical analysis

According to the queuing theory, when we simulate the M/M/M queueing system, the theoretical analysis part needs to use the following formula:

$$m * \mu = 100\%$$

For m=1,

$$\bar{T} = \frac{1}{\mu - \lambda} = \frac{1}{100 - \lambda}$$

For m>1,

$$P_0 = \frac{1}{\sum_{n=0}^{m-1} \frac{m\rho^n}{n!} + \frac{m\rho^n}{m!} \cdot \frac{1}{1-\rho}}$$

$$P_{m+} = \frac{m\rho^m}{m!\,(1-\rho)} p_0$$

$$\bar{T} = \frac{\rho}{\lambda(1-\rho)} p_{m+} + \frac{1}{\mu}$$

Among them m is number of servers, μ is the service rate, ρ is the average number of customers under service, λ is the steady-state arrival rate, $\bar{T}$ is the average waiting time.

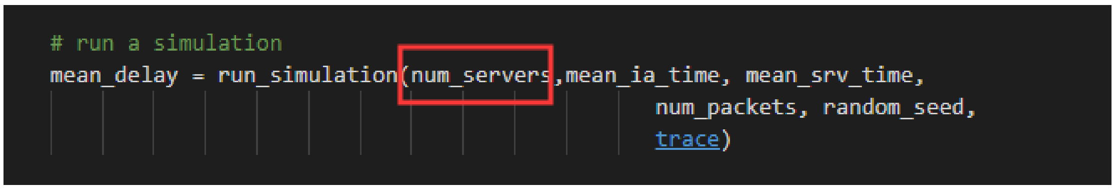
The main modification variables in Simpy needed in the simulation part are as follows:

num_servers means number of servers
mean_ia_time means packet arrival rate
mean_srv_time means packet service rate
num_packets means number of packets to generate
random_seed means seed for random number generation
trace = args.trace means M/M/1 or M/M/m(2 and 5) simulation

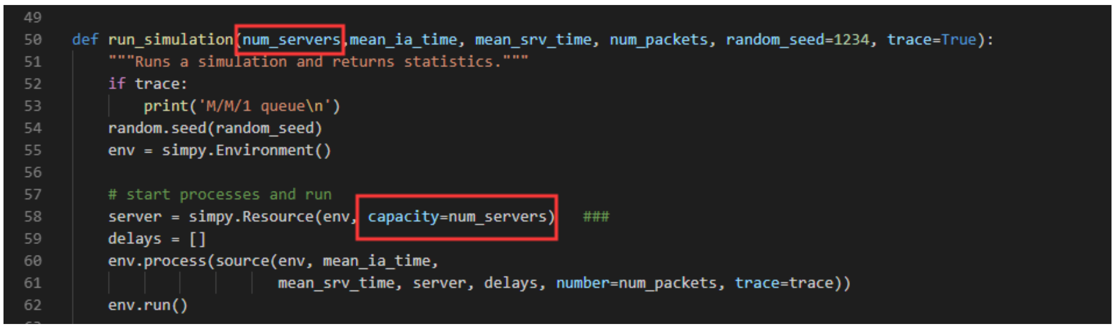
2. Code writing and modification

2.1 Modifications to new_mm1

For the simulation part, we mainly modified the parameters num_servers, and added variable variables to the main function and corresponding parts to facilitate modification, the main modification is as follows:



```
# run a simulation
mean_delay = run_simulation(num_servers,mean_ia_time, mean_srv_time,
                            num_packets, random_seed,
                            trace)
```



```
def run_simulation(num_servers,mean_ia_time, mean_srv_time, num_packets, random_seed=1234, trace=True):
    """Runs a simulation and returns statistics."""
    if trace:
        print('M/M/1 queue\n')
    random.seed(random_seed)
    env = simpy.Environment()

    # start processes and run
    server = simpy.Resource(env, capacity=num_servers)   ###
    delays = []
    env.process(source(env, mean_ia_time,
                       mean_srv_time, server, delays, number=num_packets, trace=trace))
    env.run()
```

Thus we can modify the number of servers. Next we call the batch file to enter different parameters and execute the program. And save the results to their respective out files.

```
for /L %%A in (5, 5, 95) do python new_mm1.py -M 1 -A %%A -S 100  --no-trace>> mm1.out
for /L %%A in (5, 5, 95) do python new_mm1.py -M 2 -A %%A -S 50  --no-trace>> mm2.out
for /L %%A in (5, 5, 95) do python new_mm1.py -M 5 -A %%A -S 20  --no-trace>> mm3.out
```

	M(number of servers)	S(service rate)
M/M/1	1	100%
M/M/2	2	50%
M/M/5	5	20%

Table1:Parameter modification corresponding table

2.1 Modifications of plotmm1

For M/M/2 and M/M/5 queueing system, the average delay format of theoretical analysis is different from M/M/1, so we need to write new functional formulas to express their results. This is a rewrite of the previous formula.The main code is as follows:

```
import math
m=5
x1 = np.arange(1, 100)
rou=x1/100
n=0
miu=100/m
xigema=np.zeros(99)

for n in range(m):
   xigema = xigema+((m*rou)**n)/math.factorial(n)
xigema_right=(1/(1-rou))*((m*rou)**m)/math.factorial(m)
p0=1/(xigema+xigema_right)
pm_plus=p0* ((m*rou)**m)/math.factorial(m)/(1-rou)
y1=(pm_plus*(rou/(1-rou)/x1))+1/miu
```

3. Results analysis

By default, the result when the number of packets equals 1000 is as follows:

The contents of mm1.out are as follows:

```
    5.0000E+00   1.0640E-02
    1.0000E+01   1.1237E-02
    1.5000E+01   1.1913E-02
    2.0000E+01   1.2733E-02
    2.5000E+01   1.3536E-02
    3.0000E+01   1.4300E-02
    3.5000E+01   1.5230E-02
    4.0000E+01   1.6420E-02
    4.5000E+01   1.7755E-02
    5.0000E+01   1.9363E-02
    5.5000E+01   2.1401E-02
    6.0000E+01   2.3922E-02
    6.5000E+01   2.6818E-02
    7.0000E+01   3.0910E-02
    7.5000E+01   3.6378E-02
    8.0000E+01   4.3777E-02
    8.5000E+01   5.3441E-02
    9.0000E+01   6.8108E-02
    9.5000E+01   8.9153E-02
```

The contents of mm2.out are as follows:

```
    5.0000E+00   2.0351E-02
    1.0000E+01   2.0574E-02
    1.5000E+01   2.0835E-02
    2.0000E+01   2.1179E-02
    2.5000E+01   2.1600E-02
    3.0000E+01   2.2131E-02
    3.5000E+01   2.2856E-02
    4.0000E+01   2.3734E-02
    4.5000E+01   2.4853E-02
    5.0000E+01   2.6258E-02
    5.5000E+01   2.8106E-02
    6.0000E+01   3.0305E-02
    6.5000E+01   3.3196E-02
    7.0000E+01   3.7364E-02
    7.5000E+01   4.2834E-02
    8.0000E+01   4.9561E-02
    8.5000E+01   5.8766E-02
```

```
9.0000E+01    7.2822E-02
9.5000E+01    9.3728E-02
```

The contents of mm3.out are as follows:

```
5.0000E+00    5.0737E-02
1.0000E+01    5.0737E-02
1.5000E+01    5.0744E-02
2.0000E+01    5.0757E-02
2.5000E+01    5.0770E-02
3.0000E+01    5.0896E-02
3.5000E+01    5.1210E-02
4.0000E+01    5.1586E-02
4.5000E+01    5.2115E-02
5.0000E+01    5.3036E-02
5.5000E+01    5.4405E-02
6.0000E+01    5.6188E-02
6.5000E+01    5.8921E-02
7.0000E+01    6.2987E-02
7.5000E+01    6.8601E-02
8.0000E+01    7.5776E-02
8.5000E+01    8.4008E-02
9.0000E+01    9.8545E-02
9.5000E+01    1.1819E-01
```

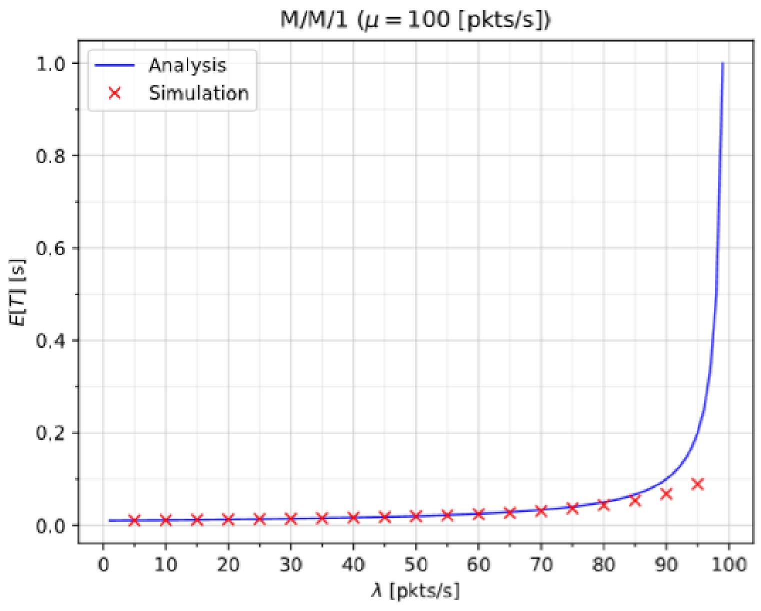


Figure 1: Simulation of M/M/1(packets=1000)

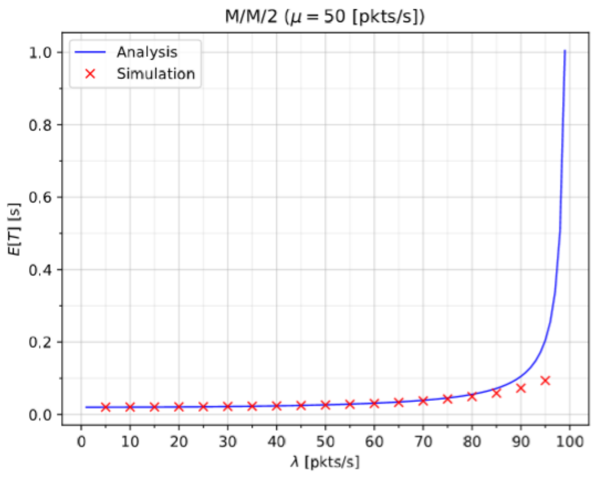


Figure 2: Simulation of M/M/2(packets=1000)

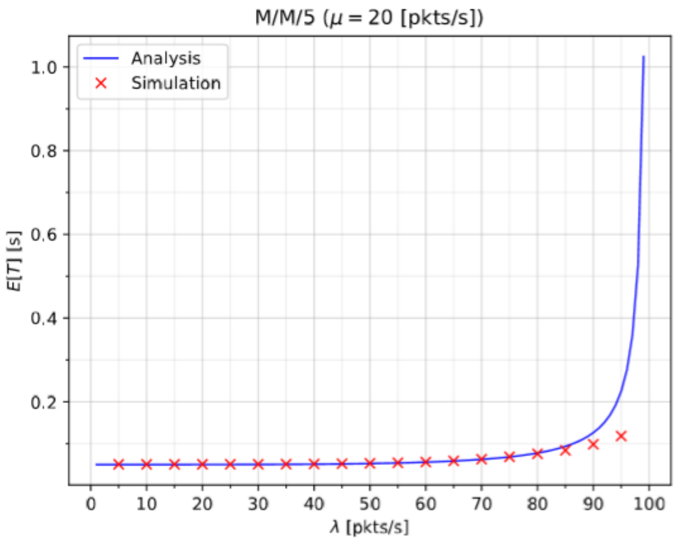


Figure 3: Simulation of M/M/5(packets=1000)

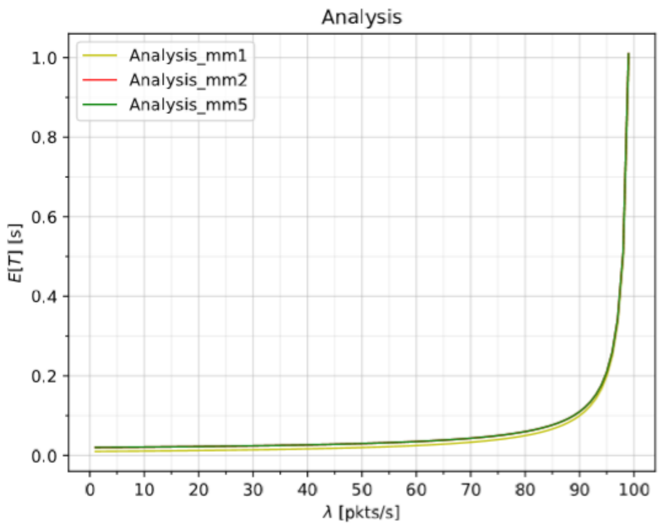


Figure 4: Comparison of analysis values(packets=1000)

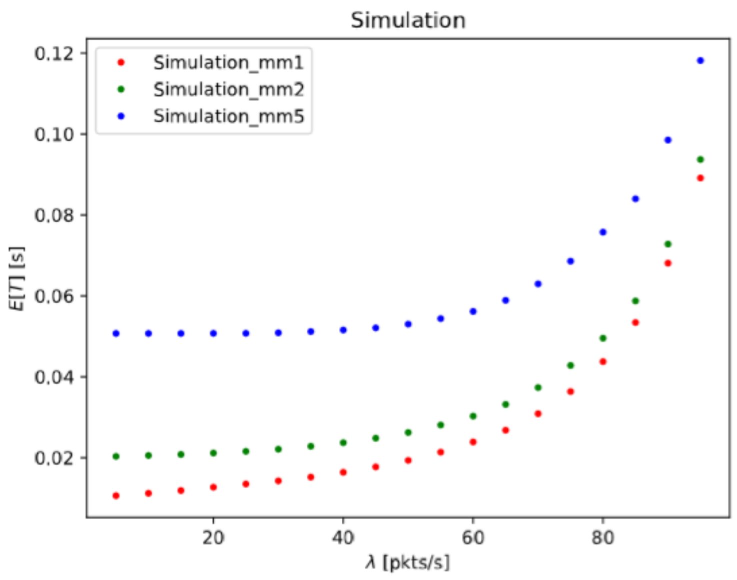


Figure 5: Comparison of simulation values(packets=1000)

From the figure 1, figure 2 and figure 3, the simulation results of three situations generally accord with the theoretical results. However, the average delay obtained by the simulation starts to shift the original curve after more than 80 when λ>80, and the delay is much lower than the theoretical value. This indicates that the queuing system did not enter the congestion situation. We speculate that the insufficient number of packages caused this problem.

As can be from the figure 4, the average delay of M/M/2 and M/M/5 is similar, both a bit higher than M/M/1. But the values of delay are all small lower than 0.05s. However, according to figure5, the result of M/M/2 and M/M/1 are similar, and the result of M/M/5 is larger. Because the coordinate range is different for convenient observation.

When under the condition of invariable total service ability, a server is divided into multiple servers, the average time delay, this may be because when you need the number of customer service reaches a certain degree, the system into a stable state, the state of Markov cycle, customers need to wait for a while to receive the service, intuitively, we may feel parallel may improve efficiency, reduce the time delay, in fact, due to a single server processing ability, in theory, the best situation is close to a single server, combined with multiple servers into the busy state, It will take some time for the customers who need to receive the service to choose the free server. In this case, the more servers there are, the more time it will take to select the server. Therefore, the average delay of M/M/2 is higher than that of M/M/2.

For packets=10000, we can set the N=10000, as follows:

```
for /L %%A in (5, 5, 95) do python new_mm1.py -M 1 -A %%A -S 100 -N 10000  --no-
trace>> mm1.out
for /L %%A in (5, 5, 95) do python new_mm1.py -M 2 -A %%A -S 50  -N 10000  --no-
trace>> mm2.out
for /L %%A in (5, 5, 95) do python new_mm1.py -M 5 -A %%A -S 20  -N 10000  --no-
trace>> mm3.out
```

The result when the number of packets equals 10000 is as follows:

The contents of mm1.out are as follows:

```
5.0000E+00    1.0511E-02
1.0000E+01    1.1095E-02
1.5000E+01    1.1725E-02
2.0000E+01    1.2452E-02
2.5000E+01    1.3263E-02
3.0000E+01    1.4141E-02
3.5000E+01    1.5148E-02
4.0000E+01    1.6342E-02
4.5000E+01    1.7738E-02
5.0000E+01    1.9411E-02
5.5000E+01    2.1465E-02
6.0000E+01    2.4243E-02
6.5000E+01    2.7676E-02
7.0000E+01    3.2535E-02
7.5000E+01    3.9691E-02
8.0000E+01    4.8997E-02
8.5000E+01    6.4200E-02
9.0000E+01    9.7699E-02
9.5000E+01    2.0845E-01
```

The contents of mm2.out are as follows:

```
5.0000E+00    2.0092E-02
1.0000E+01    2.0258E-02
1.5000E+01    2.0511E-02
2.0000E+01    2.0871E-02
2.5000E+01    2.1329E-02
3.0000E+01    2.1893E-02
3.5000E+01    2.2615E-02
4.0000E+01    2.3504E-02
4.5000E+01    2.4626E-02
5.0000E+01    2.6058E-02
5.5000E+01    2.7842E-02
6.0000E+01    3.0358E-02
```

```
6.5000E+01 3.3583E-02
7.0000E+01 3.8385E-02
7.5000E+01 4.5473E-02
8.0000E+01 5.4823E-02
8.5000E+01 7.0142E-02
9.0000E+01 1.0392E-01
9.5000E+01 2.1546E-01
```

The contents of mm3.out are as follows:

```
5.0000E+00     5.0127E-02
1.0000E+01     5.0127E-02
1.5000E+01     5.0132E-02
2.0000E+01     5.0149E-02
2.5000E+01     5.0197E-02
3.0000E+01     5.0312E-02
3.5000E+01     5.0520E-02
4.0000E+01     5.0868E-02
4.5000E+01     5.1402E-02
5.0000E+01     5.2227E-02
5.5000E+01     5.3460E-02
6.0000E+01     5.5348E-02
6.5000E+01     5.8098E-02
7.0000E+01     6.2226E-02
7.5000E+01     6.8487E-02
8.0000E+01     7.7166E-02
8.5000E+01     9.2385E-02
9.0000E+01     1.2569E-01
9.5000E+01     2.3718E-01
```

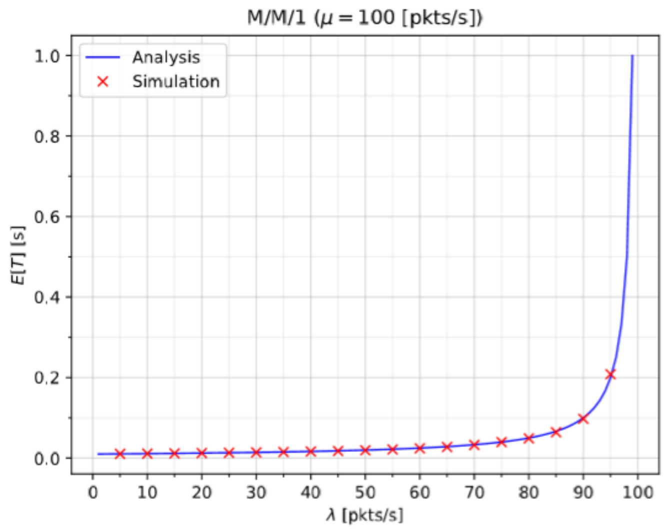


Figure 6: Simulation of M/M/1(packets=10000)

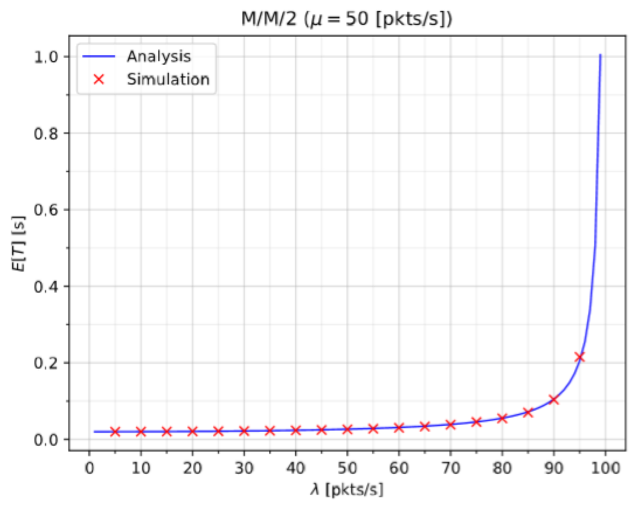


Figure 7: Simulation of M/M/2(packets=10000)

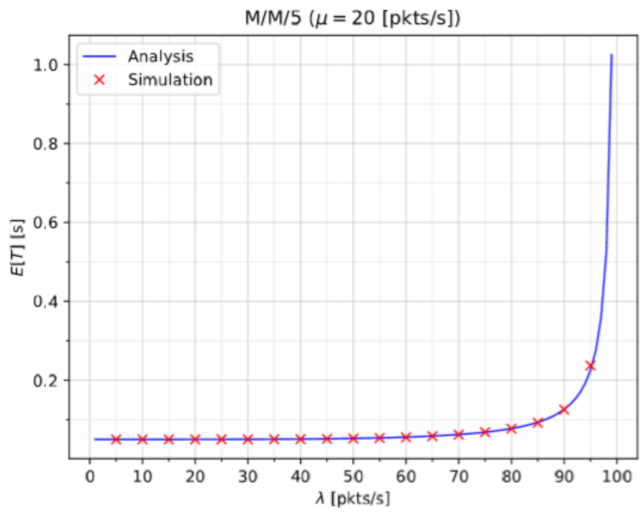


Figure 8: Simulation of M/M/5(packets=10000)

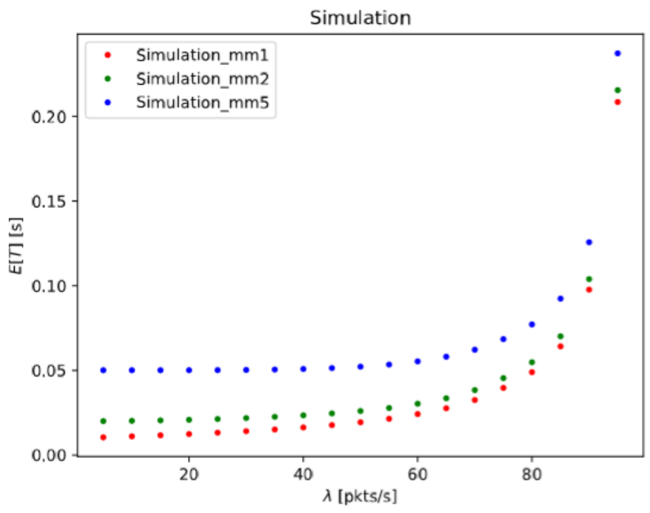


Figure 9: Comparison of simulation values(packets=10000)

As we can see, when packets=1000, the simulation results is exactly the same as the analysis values. When the package is 1000, the load threshold of the server is not reached, so there is no congestion. When the package reaches 10000, the load threshold of the server in simpy is reached, so it is consistent with the analysis results.

4.Appendix: code

Two folders in compressed package
For packets=1000:
Packet name: packets=1000
new_mm1.py: simulation code
case1.bat: code for different parameters generate out file
plotmm1.py: plot simulation for M/M/1
plotmm2.py: plot simulation for M/M/2
plotmm3.py: plot simulation for M/M/5
plot.py: plot simulation for analysis and simulation comparison diagram

For packets=10000:
Packet name: packets=10000
new_mm1.py: simulation code
case1.bat: code for different parameters generate out file
plotmm1.py: plot simulation for M/M/1
plotmm2.py: plot simulation for M/M/2
plotmm3.py: plot simulation for M/M/5
plot.py: plot simulation for analysis and simulation comparison diagram