Report

project idea

First considering the pdf already given, we try to set a master clock which remembers all the time point , in this time point may be is the server finish one task, or the set_up phase completed or the delay off completed need to shut down the server or there are a new arrival. Therefore master_clock is a very important list in my whole project.

The project is combined by two class(trace and random), and the handle arrive and handle departure methods are same in this two model, the only difference is in the main function. The reason is the different between trace and random is the method of get arrive and service. Trace got them from the txt , however random got them by randomly built and also has time end to control the amount of the arrive as well .

The most difficult part for me is that I didn't realize the important to set a master_clock in the whole project at first, which made me feel confused for a long time.

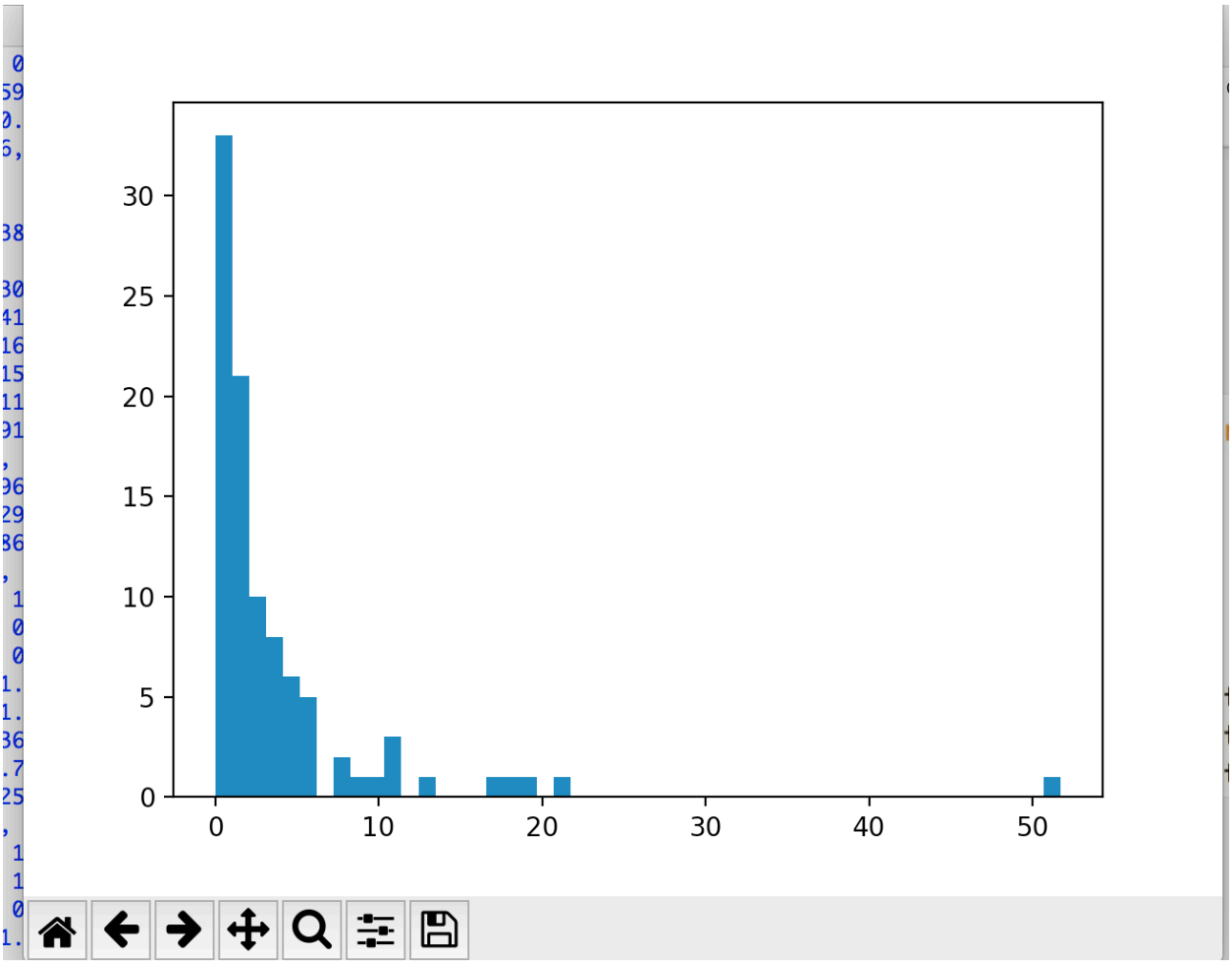
1.1 Look for evidence in your report that you have verified the correctness of your simulation code.

The result of different kind of simulation results
Which arrived from the arrive and service and the process is followed one

Master clock	Dispatcher	Server 1	Server 2	Server 3	Notes
T = 15.0	(10.0, 2.0, 'Marked')	setup(completed in t = 15.0)	Off	Off	An arrival at time 15. Server 1 is turned on and is in SETUP state. The jobs in the dispatcher are modeled with 3-tuples: first element is the arrival time, second element is the service time and the last element indicates whether it is MARKED or UNMARKED. This job is MARKED be-cause it is waiting for the setting up of a server
T = 17.0	-	busy(completed in t = 17.0)	Off	Off	At time 17.0 the first arrive is already finished in server1
T = 17.1	-	delay-off(completed in t = 17.1)	Off	Off	At time 17.1 no things in dispatcher then busy server enter the state of the delay-off

T =20.0	[(18.0, 4.0, 'Marked')]	setup(completed in t = 23.0)	Off	Off	At time 20.0 we find there're already a arrival in t = 18.0 which made the server enter into setup state
T = 23.0	(18.0, 4.0, 'Marked'), (20.0, 14.0, 'Marked'), (23.0, 5.0, 'Marked')]	setup(completed in t =23.0)	setup(completed in t = 25.0)	setup(completed in t = 28.0)	The server1 setup right at finished and the server 2 is setup due to the arrival so as the server 3
T = 25.0	[(20.0, 14.0, 'Marked'), (23.0, 5.0, 'Marked')]	busy(completed in t = 27.0)	setup(completed in t = 25.0)	setup(completed in t = 28.0)	The server1 started to start the job.
T = 27.0	[(23.0, 5.0, 'Marked')]	busy(completed in t = 32)	Busy(completed in t=39.0)	setup(completed in t=28.0)	have changed in new arrival in two servers
T = 28.0	-	busy(completed in t = 32)	Busy(completed in t=39.0)	Off	Server 3 offed by the marked go to other sever and dispatcher is empty
T = 32.0	[(28.0, 6.0, 'Marked'), (32.0, 21.0,'Unmarked')]	busy(completed in t = 32)	Busy(completed in t = 39.0)	Setup(completed in t = 33.0)	New arrival set up the server3
T = 33.0	[(32.0, 21.0, 'Marked'), (33.0, 2.0, 'Unmarked')]	busy(completed in t = 38.0)	Busy(completed in t = 39.0)	Setup(completed in t = 33.0)	new arrival in server1 (unmarked)
T = 34.0	[(33.0, 2.0,'Unmarked')]	busy(completed in t = 38.0)	Busy(completed in t = 39.0)	busy(completed in t = 54.0)	The changed state of server 3 , because the setup time finished
T = 39.0	[(34.0,16.0,'Unmarked'), (35.0, 9.0, 'Unmarked')]	busy(completed in t = 40.0)	Busy(completed in t = 39.0)	busy(completed in t = 54.0)	The change state of server3 of unmarked arrival
T = 40.0	[(35.0, 9.0,'Unmarked')]	busy(completed in t = 40.0)	Busy(completed in t = 55.0)	busy(completed in t = 54.0)	The state of server2 changed by unmarked arrival
T = 49.0	-	busy(completed in t = 49.0)	Busy(completed in t = 55.0)	busy(completed in t = 54.0)	The state of server1 changed by unmarked arrival
T = 49.1	-	delay-off (Completed in t = 49.1)	Busy(completed in t = 55.0)	busy(completed in t = 54.0)	The queue is empty therefore server1 started delay-off
T = 54.0	-	off	Busy(completed in t = 55.0)	busy(completed in t = 54.0)	the delay-off of server1 finished

T = 54.1	-	Off	Busy(completed in t = 55.0)	delay-off(completed in t = 54.1)	The dispatcher is empty
T =55.1	-	Off	delay-off(completed in t = 55.1)	Off	The dispatcher is empty
T = 57.0		Off	Off	Off	the delay-off of server2 finished
T = 62.0	[(57.0, 4.0, 'Marked')]	setup(completed in t = 62.0)	Off	Off	new arrival in server1 (marked) set up state
T = 66.0	-	busy(completed in t = 66.0)	Off	Off	new arrival in server1 (marked)
T = 66.1	-	delay-off (Completed in t = 66.1)	Off	Off	The dispatcher is empty
T = 91.0	[(86.0, 15.0, 'Marked')]	setup(completed in t = 91.0)	Off	Off	new arrival in server1 (marked) set up state
T = 92.0	-	busy(completed in t = 106.0)	Off	Off	new arrival in server1 (marked)
T = 97.0	[(92.0, 9.0, 'Marked')]	busy(completed in t = 106.0)	setup(completed in t = 97.0)	Off	new arrival in server2 (marked) set up state
T = 106	-	busy(completed in t = 106.0)	Busy(completed in t = 106.0)	Off	All enter in busy state
T = 106.1	-	delay-off (Completed in t = 106.1)	delay-off(completed in t = 106.1)	Off	The dispatcher is empty
T = 106.1	-	Off	Off	Off	No arrival enter into off state.



1.2 verified the correctness of the inter-arrival probability distribution and service time distribution correctly

I use the “-log(1-random.random())/lamda" as inter-arrival
And “-log(1-random.random())/miu-log(1-random.random())/miu-log(1-random.random())/miu “ as service time

The service time and arrival are both use the same sequence “-log(1-U_k)/l “

The picture is shown as belows. And the code is also saved in test.py

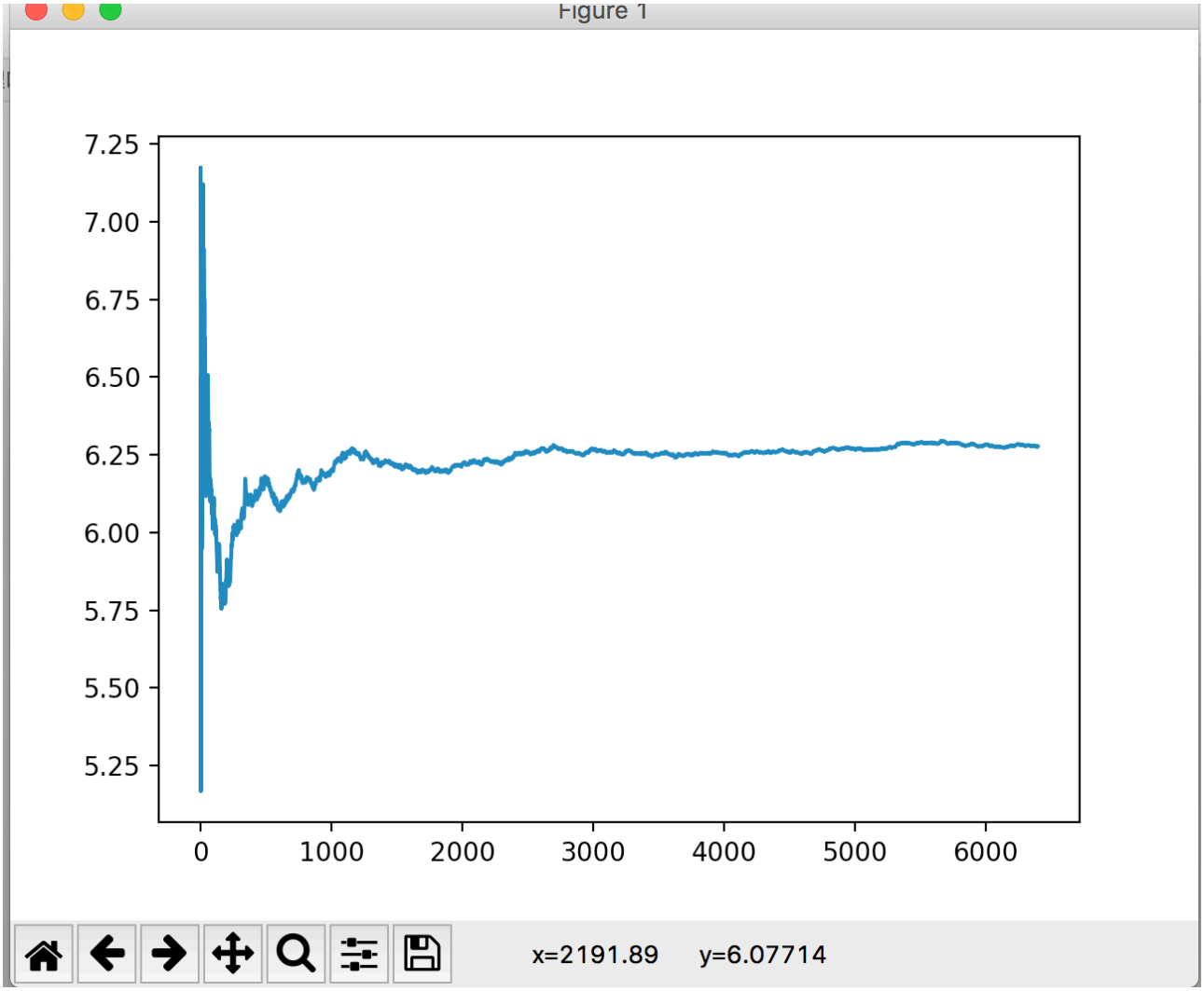
2.1 You will need to demonstrate that your results are reproducible.

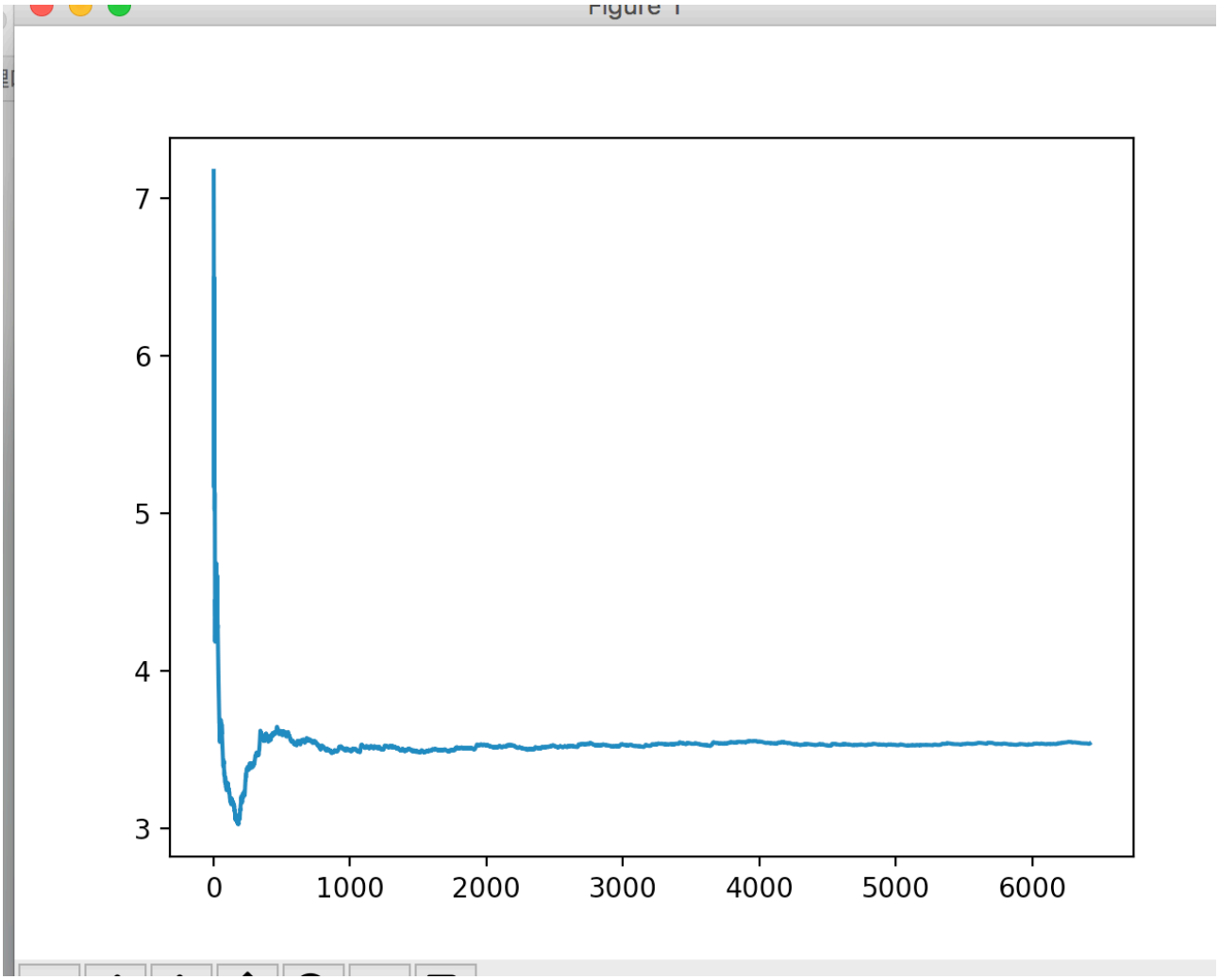
I used seed in random which controls the output of the inter- arrival and service time which ensures that my results is reproducible. Furthermore, the seed also can be changed from the wrapper file for further test as well.

3.1 Evidence of using statistically sound methods to analyze simulation results

Use confidence interval to solve the problem
Several replication
According to t-distribution n =29 and alpha = 0.25 as we got the 2.045
We can found around. 40-140 the confidence interval is not highly changed therefore I determined tc = 30 which has higher confidence interval
Therefore the mean response time is shorter which means the capability of the system soundly improved as well.

3.2 Explanation on how you choose your simulation and data processing parameters, e.g lengths of your simulation, number of replications, end of transient etc.





The first fig is that of tc = 0.1 and the second is tc = 30 .
According to the picture, we will do the transient removal to make the final result would not be affected by the useless figure.

Then we can use the new mean response time to get confidence interval of this two tc.

Furthermore, there are also several replication are applied which used random. And the variables in this process is seed. Which means several tests the arrival service case of both tc is the same which ensures we can compare them confidence interval reliably.