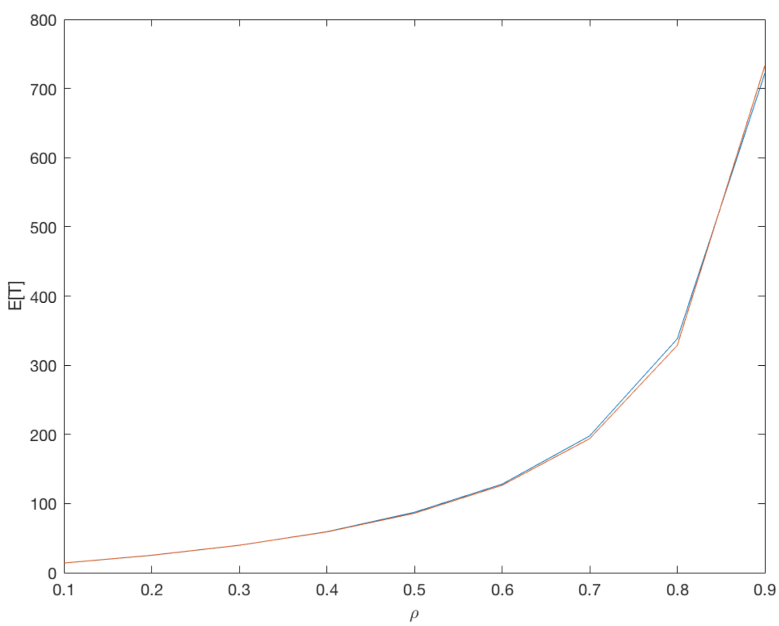
Monday, February 20, 2017

Fabio Ellena

Homework 3, Laboratory

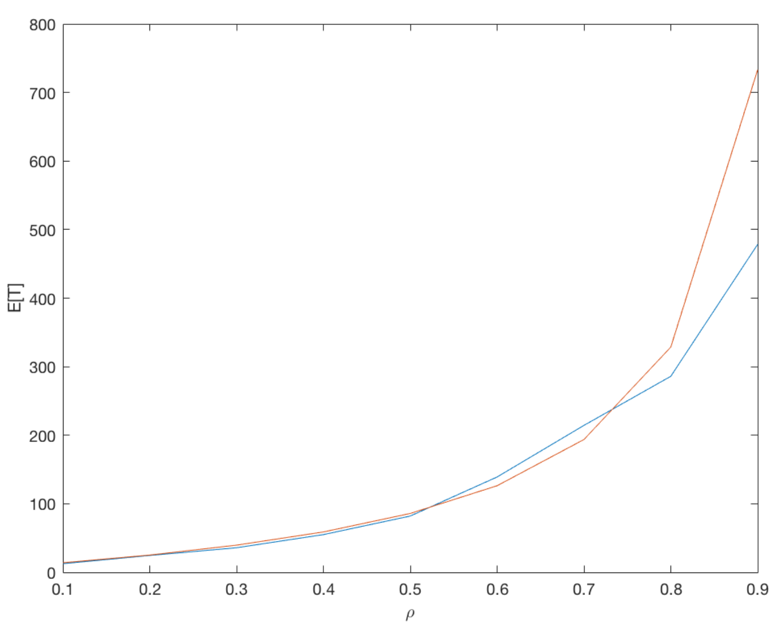
Comments of code and algorithm are in the matlab scripts.

1) Plot also the theoretical values for each simulation point.

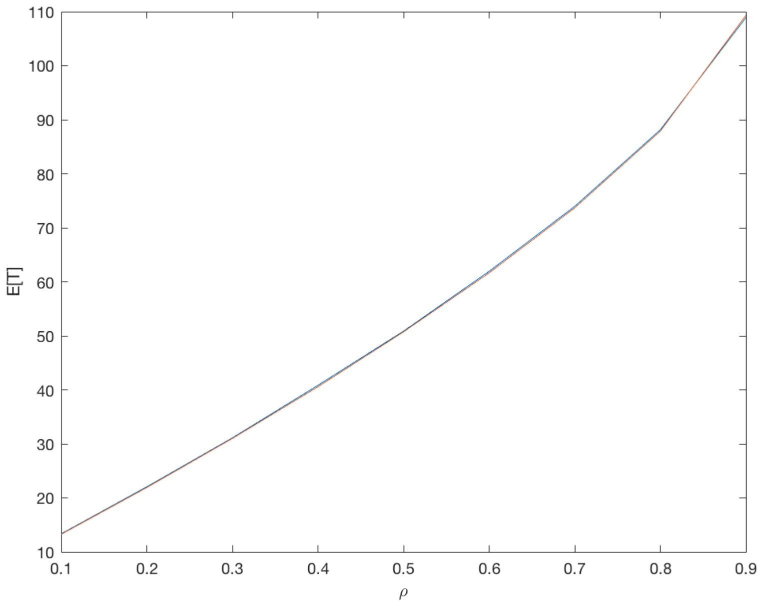


Q: Do the simulations agree with the theoretical predictions? (make sure your simulations have converged)

A: Yes, the simulation, when done with a big number of jobs, agree perfectly with the theoretical prediction.



When there is a smaller number of jobs, the result of the simulation follows the theoretical one, but there is a greater uncertainty



2) Plot again the mean response as a function of E[T] as a function of ρ.

Q: Do that values agree with the theoretical prediction?

A: Yes, they do

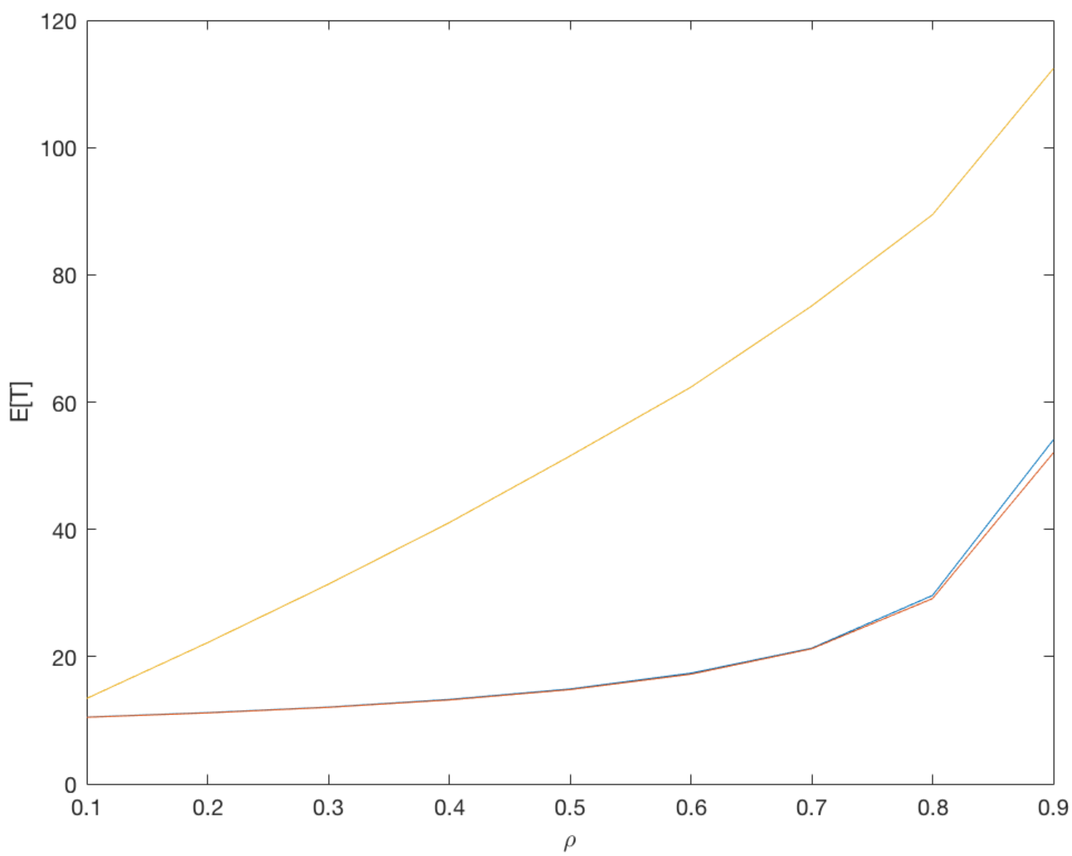
Q: What do you observe in comparison to FCFS?

A: E[T] is way lower than the one in FCFS, this is particularly true when rho is near 1. It looks that in this scenario, E[T] grows linearly with rho when 0.1 < rho < 0.8, while in the FCFS system, E[T] grows like a quadratic function.

3) Q: Can you do any better than SJF, without using a preemptive policy?

A: Yes, I can split the total capacity in two server(M/D/1), each with its queue, and send the short jobs to server 1 and large jobs to server 2. In this way the variability in each server is zero because in the queue there are only equal jobs and the expected time in the system is lower than with SJF, which still depends from the second moment of the job size distribution.

Q: If so, describe your proposed policy, implement it, and plot E[T] again.



A: Description: The idea is to split the capacity of the server in two, in this way we create two 'virtual server' that serve only jobs with a specific size.

Short jobs are served by server1 and large jobs are served by server 2.

The two server put together have the same capacity of the original one, but each job will be served at a lower speed.

This is not important, because in this way jobs in each queue will have the same size, hence variability equal to zero, which means low queuing time. This is because in this case short jobs are not stuck behind large ones, while in SJF this can happen.

Each queue is an M/D/1, FCFS system, thus is quite easy to calculate the expected time in the system. The main problem is how to split the server capacity:

The only hard constraint is that each queue should be stable, and I chose the values that meet this constraint when rho is high. Knowing the size distribution and the arrival rate of jobs, I can calculate the first derivative and find the optimal capacity split.

As we can see, the expected time is lower than the SJF policy (yellow line), this is always true, also with high load. Another positive thing is that the expected time grows slowly respected to SJF when the load is low (rho < 0.8)