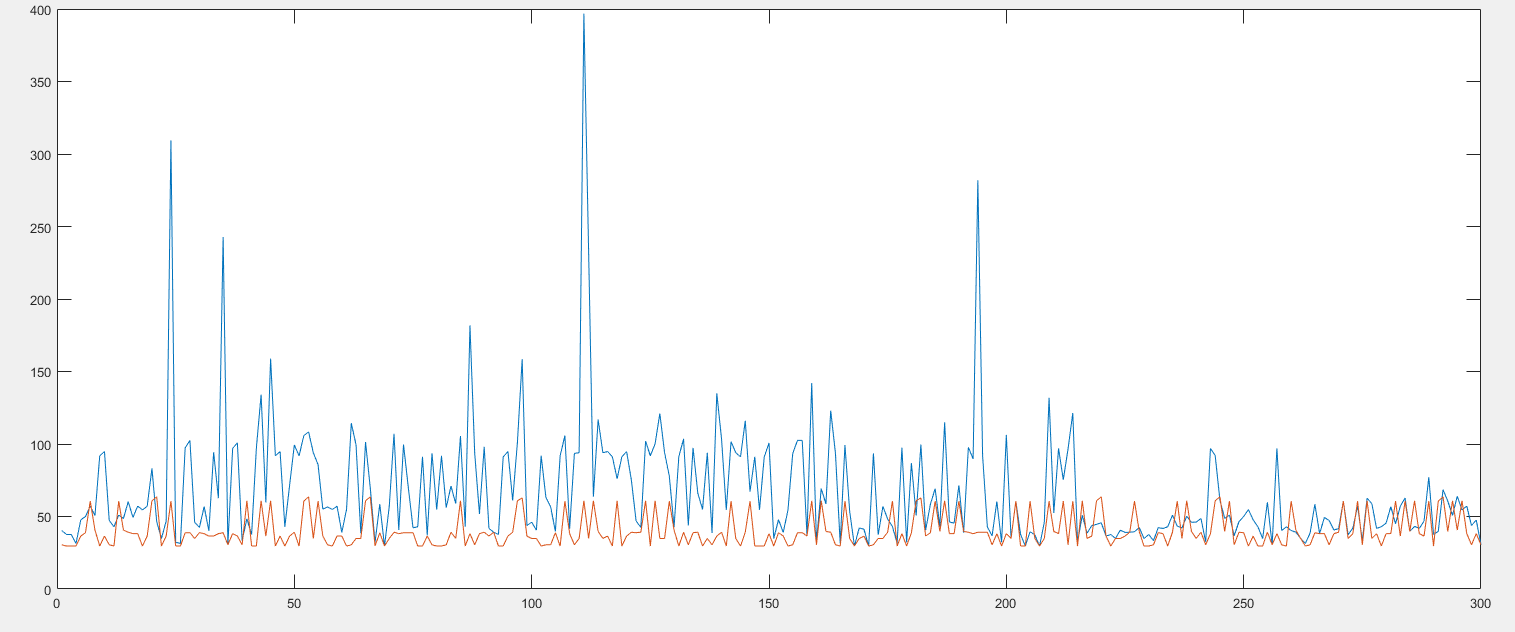
I perform the analysis by assuming we have 4 machines. Therefore there are a total of 20 configurations(action). I have seen that RL algorithm converges to the optimum configuration after some time. [But, sometimes its inconsistent: sometimes if the number of time frames for training is increased, RL algorithm converges to some sub-optimal configuration. For eg: When I repeated requests for 200 times, for arrival rate 70 it converged to the optimal configuration, but when I repeated requests for 300 times, for arrival rate 70, it converged to suboptimal configuration. This is not intuitive and I am looking at it. ].

I have assumed inter-arrival rates of { 50,60,70,80,100,150,200, 400,1000,2000 } ie. each adjacent requests are generated with inter-arrival rates exponentially distributed in above values. [Values below these such as 20,25 fail the system, as in these conditions, requests come at a high rate].

This figure plots the graph of average latencies at the end of each Time frame (for now, I have just given fixed number of requests in each time frame i.e size of time frame is not constant for different arrival rates, rather each time frame has some fixed number of requests).

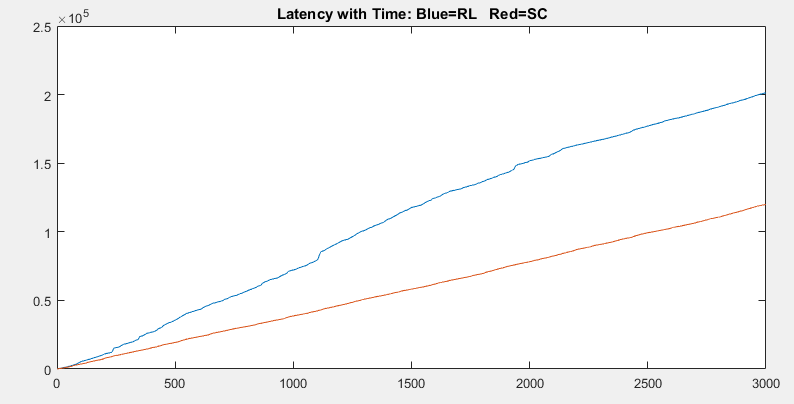
Blue: Reinforcement Learning

Red: SC Baseline algorithm

(Setting: each arrival rate is repeated 30 times. Each time frame consists of 10 requests ( a total of 3000 requests is processed); the arrival rates are randomly generated and are same for both algorithms)

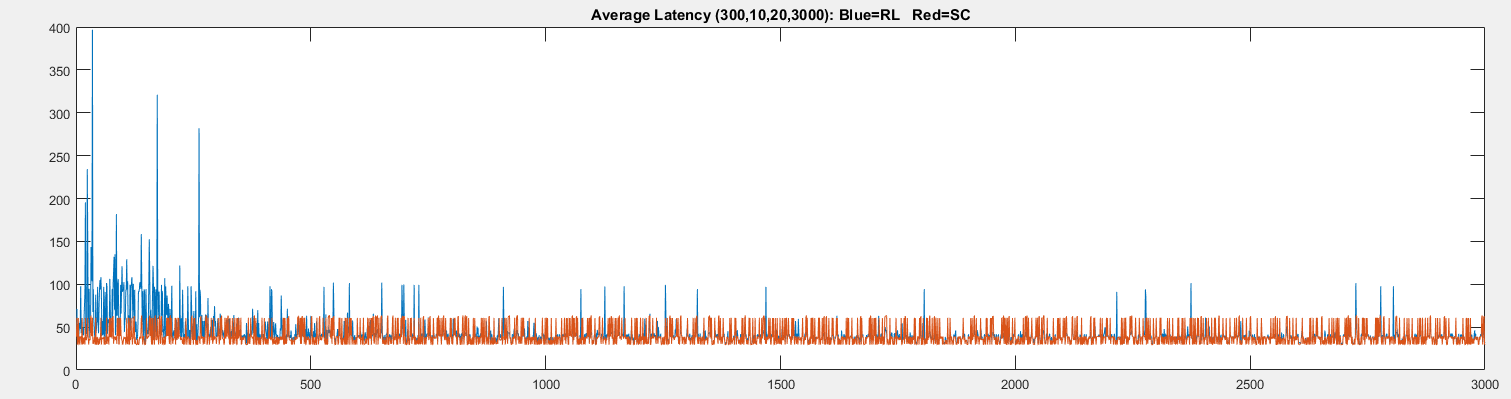
We can see that the performance of RL algorithm improves with time.

The figure below plot the cumulative response time for RL and SC algorithm. The performance gap increases during the initial learning phase, while it seems to be parallel at the end.



Now, I plot the performance for different setting.

Setting 2 : Each arrival rate is repeated 300 times randomly. There are 10 requests in each time frame, and there are 30000 total requests. There are a total of 10 different arrival rates { 50,60,70,80,100,150,200, 400,1000,2000 }.



We can see the randomly selected actions in the beginning of the plot by RL algorithm. As it learns, they become least frequent. There are random spikes later as well, which is probabilistic in nature. As we zoom in, we can see later that there are some cases where RL performs better than SC, I don’t see why that can be true. I am still looking into it.

Now the cumulative time latency is shown in the following figure. We can see the offset begins due to the performance gap between RL and SC paper during the learning phase. Later, we can see that the lines are almost parallel, suggesting that the performance of the RL algorithm is converging to SC.

