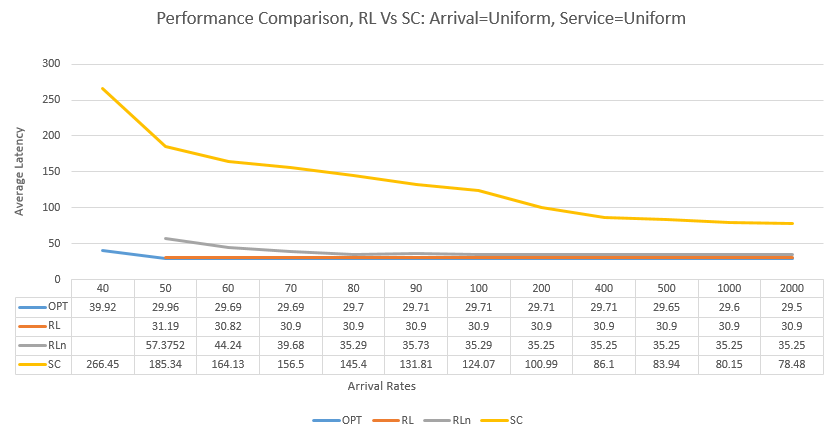
We want to see how RL and SERF paper performs for various settings to observe their behavior. Let us compare the performance of different algorithms for various cases.

This document is organized roughly as follows trying to answer the following questions:

1. For different algorithms when the window size is same, when the arrival rate changes(x-axis), how does latency (y-axis) of RL, SC, Optimal change and compare to each other?
2. For different algorithms when the arrival rate is same, for different window sizes (x-axis), how does latency (x-axis) of RL, SC and Optimal change and compare to each other?
3. Dynamic Workload: When the arrival rate changes randomly changes over time (x-axis), how does latency (y-axis) of RL, SC, Optimal Change and compare to each other?

**General Setting:** We generate requests with some distribution distributed about some mean, as given in the setting below, and process requests for both RL and SERF. We measure average latency for the optimal case, RL algorithm (one including learning curve, and another excluding the learning period) and SC algorithm. We plot the figures for various arrival rates as follows.

We now try to answer Qs 1.

**Setting 1: Service Time = Uniform distributed, and, Inter-arrival rate=Uniformly distributed**

For all the figure in this document,

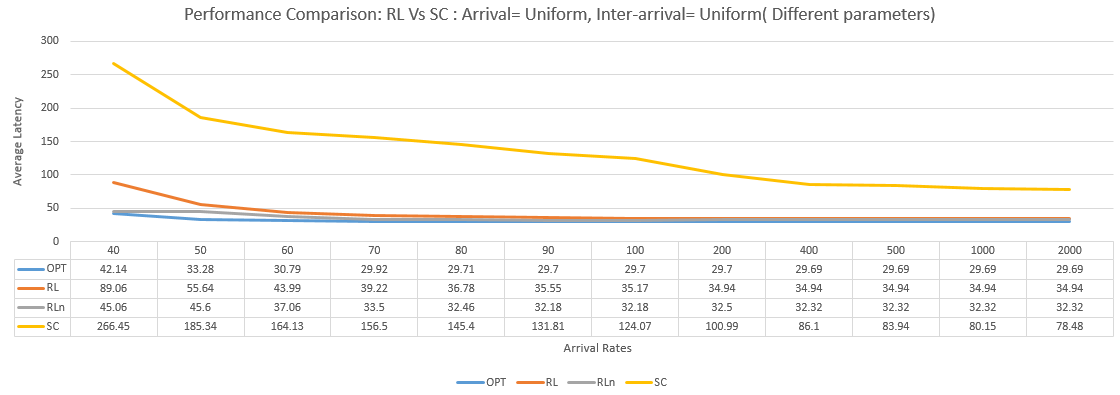
OPT=Optimal Performance

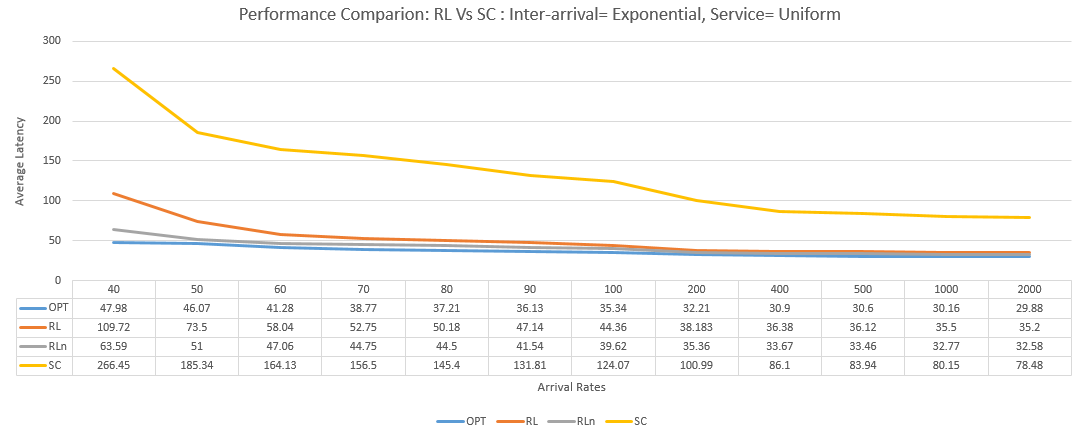
RL= Reinforcement Learning (excluding learning phase)

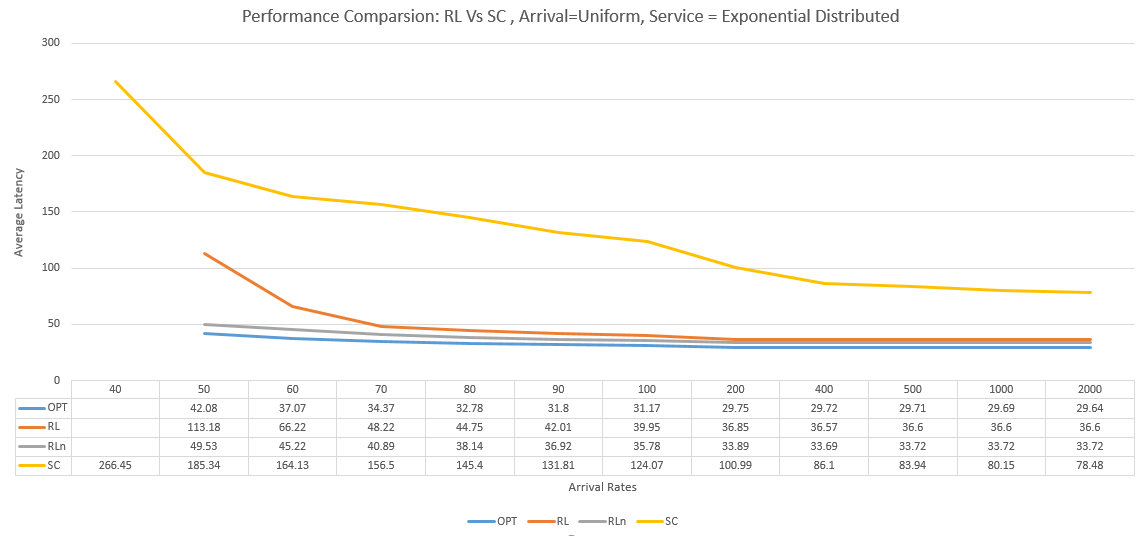
RLn= Reinforcement learning (including learning phase)

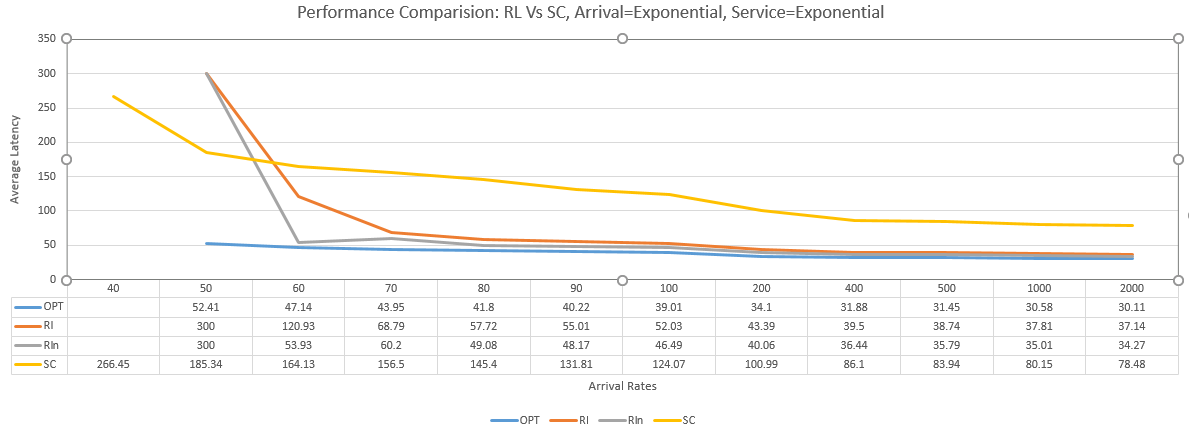
SC=SERF Algorithm

Here:

**Setting 2: Service Time = Uniform distributed, and, Inter-arrival rate=uniformly distributed** (different parameters compared to setting 1)

**Setting 3: Service Time = Uniform distributed, and, Inter-arrival rate=Exponentially distributed**

**Setting 3: Service Time =Exponential distributed, and, Inter-arrival rate=Uniformly distributed**

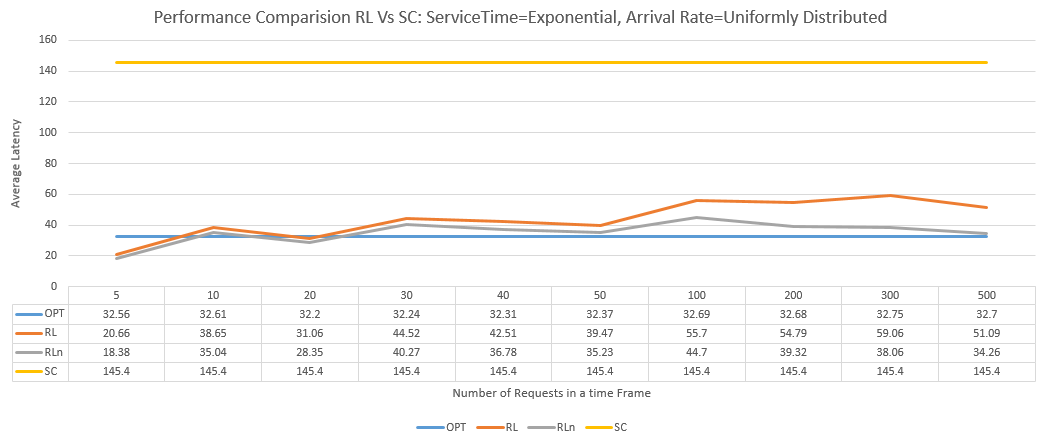
**Setting 4: Service Time =Exponential distributed, and, Inter-arrival rate=Exponentially distributed**

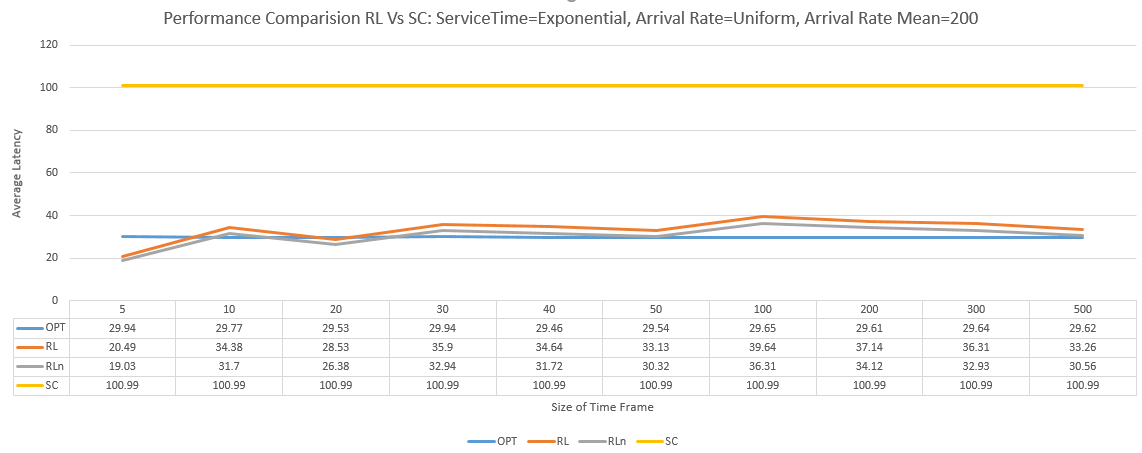
Here, Rl didn’t converge for arrival rate of 50. For the figure, I have changed the values to 300.

Observation for Qs 1:

1. RL is always better than SC for the above combination of service distribution and arrival rates by some degree.
2. For certain combination of service and arrival distribution, RL doesn’t converge. It might be because the initially selected configuration might take very long to process, and before the time frame changes, the waiting time is really high. Therefore, as new requests are generated, it never converges. While for SERF, because it already selected better configuration, it doesn’t produce high waiting time and hence can process all the requests.

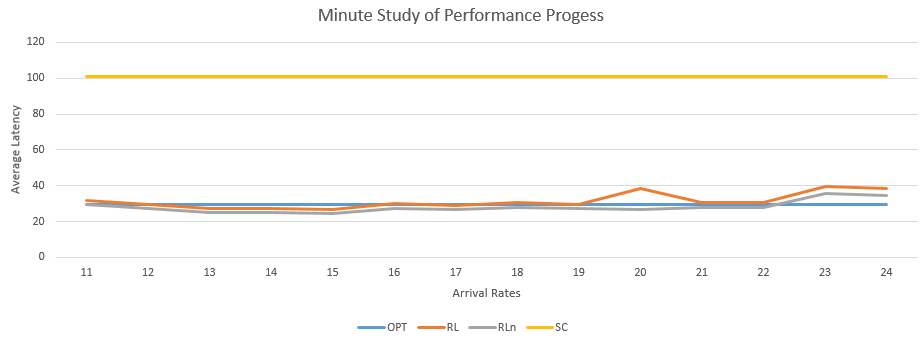
Now, we study the performance of RL and SC when we change the window size ( or size of time frame) keeping arrival rate constant, and answer question 2.

**Setting 1: Service Time =Exponential distributed, and, Inter-arrival rate=Uniform distributed, Arrival Rate= 80**

**Setting 2: Service Time =Exponential , and, Inter-arrival rate=Uniform distributed, Arrival Rate= 200**

Observations:

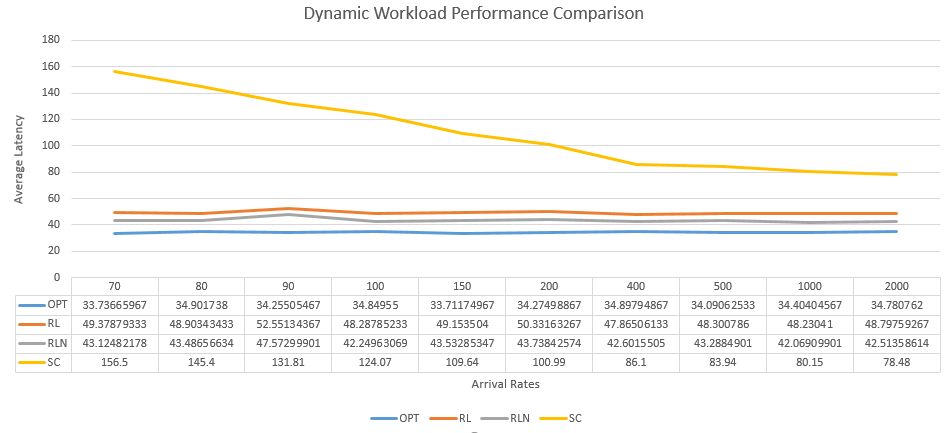
1. For both arrival rates, 80 and 200 (first figure and second figure), we see that as the size of time frames increase from 5 to 20, we see that RL is better than optimal. But after that, RL tries to approximate optimal. This is counter-intuitive. I also tried changing the seed for random number and re-run the experiments, but this resulted similar results. May be this is due to some random reason. I tried to study the change between 5-25 window size more minutely, and here is the results.

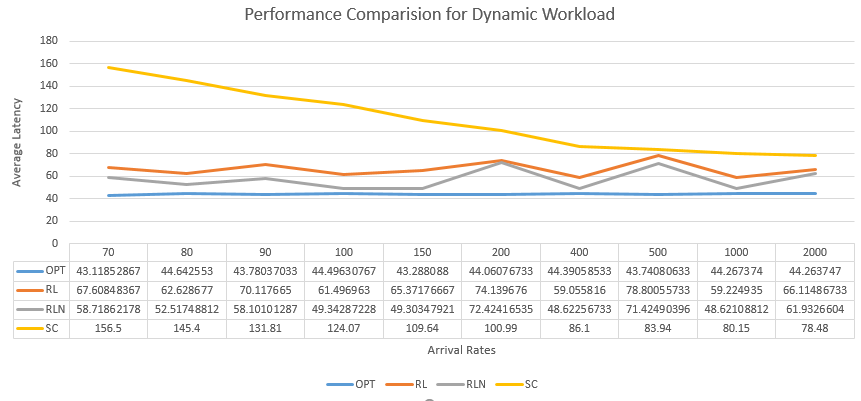


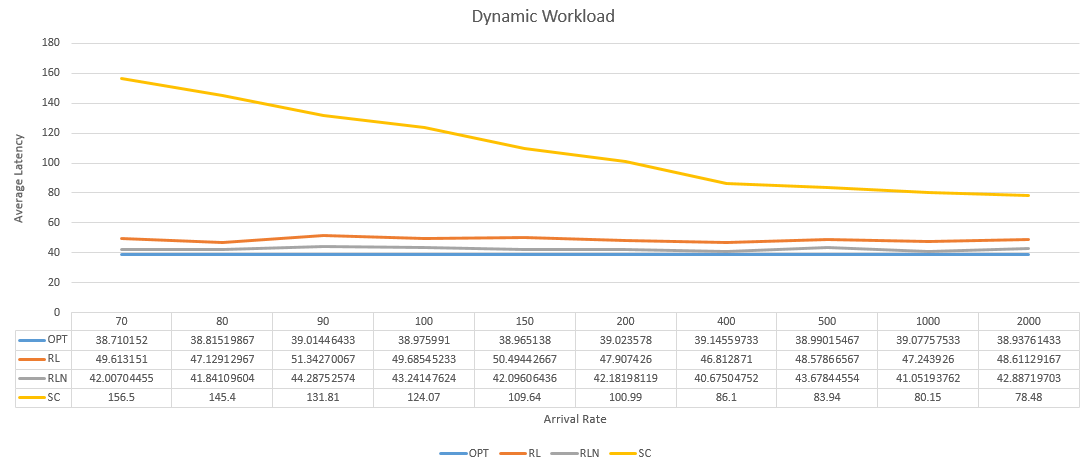
The change is gradual, and I think cannot be said random. There might be something about this window size, which I cannot postulate as of now.

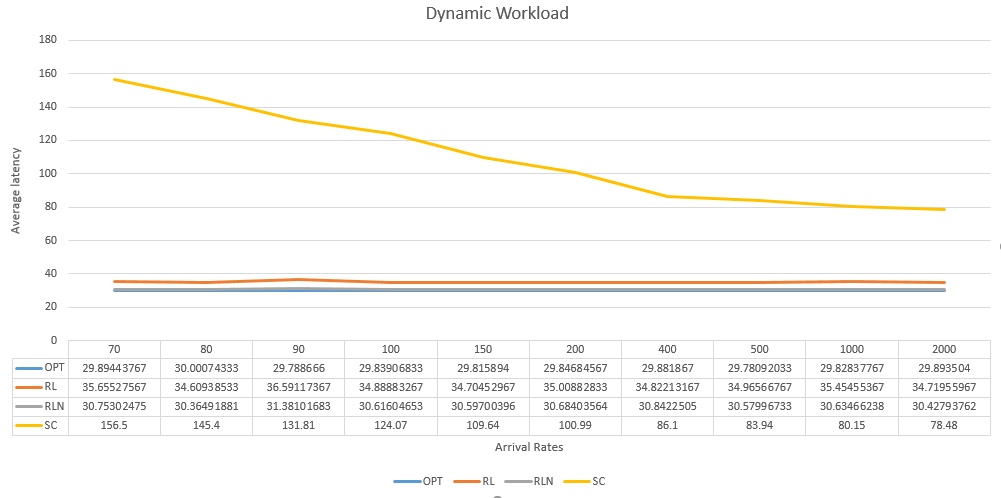
1. SC performs same for all window sizes, and is always worse than RL.

Dynamic Work load Performance Comparision: Question 3

**ServiceTime=Exponential, Inter-Arrival=Uniform**

**Setting: ServiceTime=Exponential, Inter-arrival=Exponential**

**Setting: ServiceTime=Uniform, Inter-arrival=Exponential**

**Setting: ServiceTime=Uniform, Inter-arrival=Uniform**

Observations:

1. RL performs better than SC and closely approximates optimal performance.