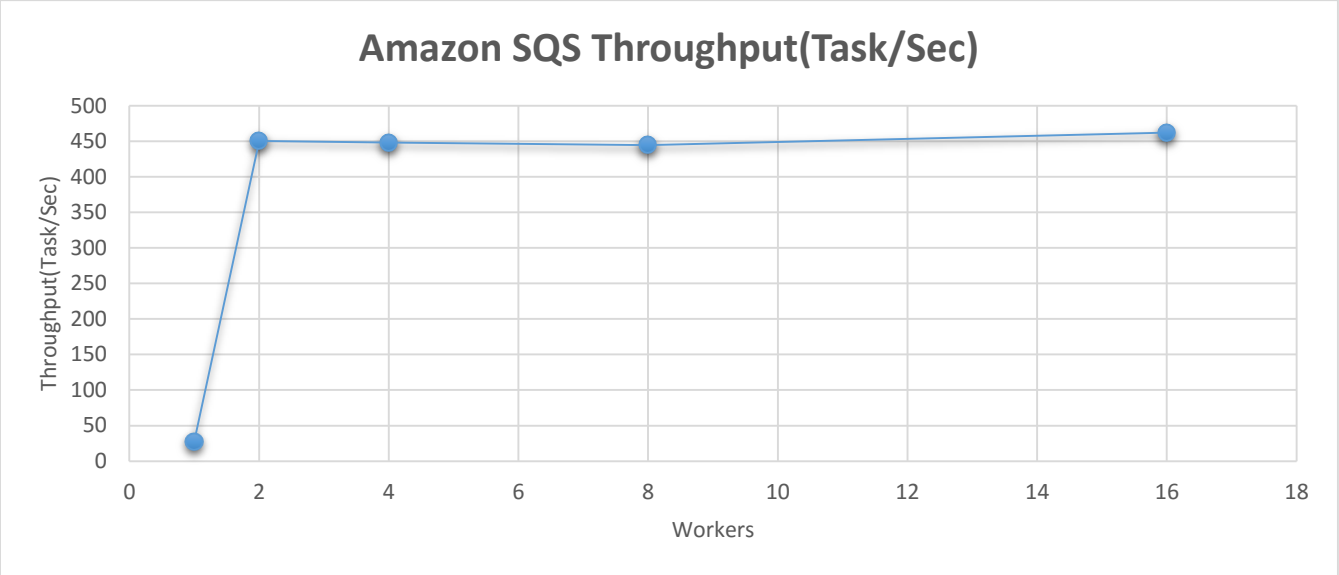
Performance Evaluation

1. Remote Worker:

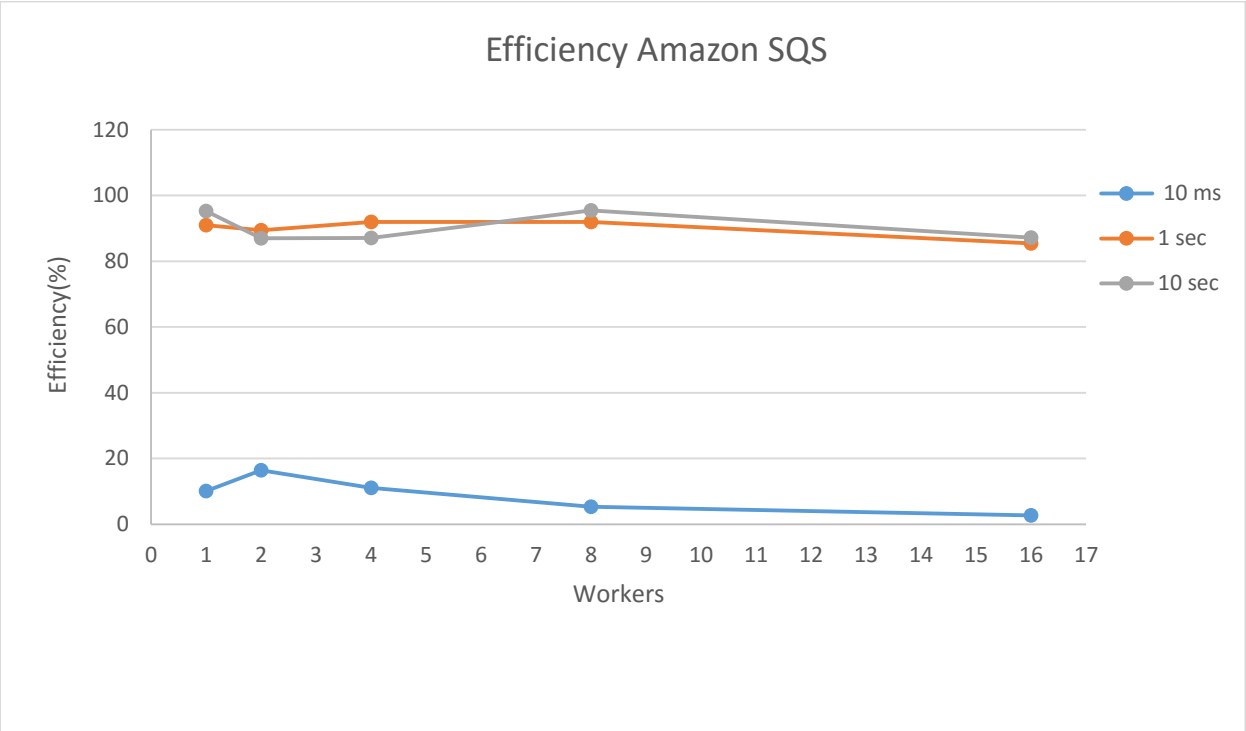
a. Throughput:



Worker	Remote	Task
1	27.91892	10000
2	450.2679	100000
4	448.3782	100000
8	444.7014	100000
16	461.9791	100000

As can be seen from the above graph, the throughput increases as the number of workers increases. This is because of dividing the work and execute it individually and that ends up getting higher throughput values.

b. Efficiency:

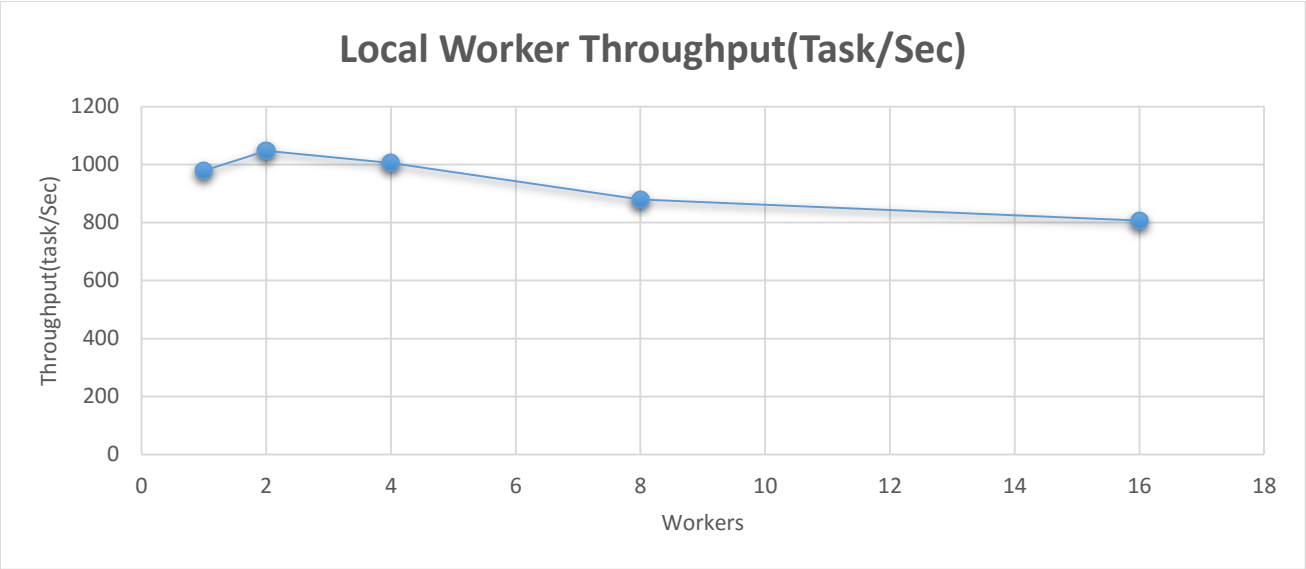


Time	1	2	4	8	16
10 ms	10.117361	16.398819	11.053388	5.3550391	2.7250927
1 sec	90.983532	89.38947	91.979397	91.99632	85.426277
10 sec	95.27439	87.032202	87.077673	95.438061	87.168759

The above graph shows that efficiency is significantly less in “sleep 10ms” task when it is run on remote and decreases exponentially as we increase the number of Threads. The “sleep 10ms” task might not incur much time but the updations in dynamoDB and putting in Result SQS and get data from queue are too high which results in deduction of efficiency. In other two tasks “sleep 1s” and “sleep 10s” the number of task per worker is less (100 and 10). This helps these two tasks to get high efficiency as the overheads of DynamoDB.

Local Worker:

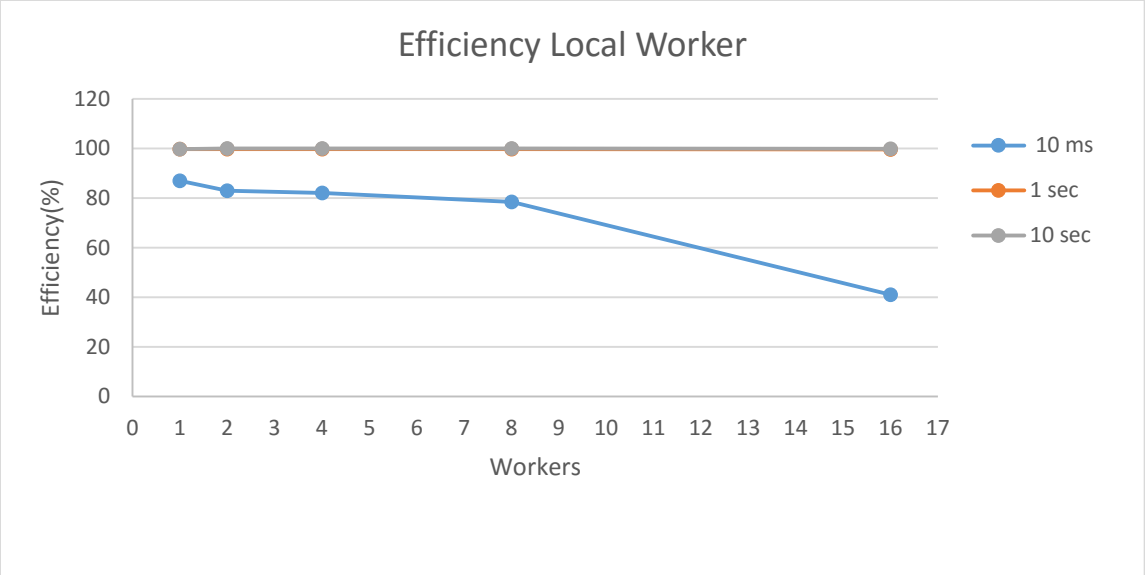
a. Throughput:



Worker	Local	Task
1	979.4319	10000
2	1047.559	100000
4	1005.126	100000
8	879.7396	100000
16	806.7769	100000

Above graph shows that throughput decreases as number of threads increases from 1 to 16. We can observe from the graph that as number of threads are doubled the throughput instead of increasing, keeps on decreasing. Here upto 2 threads I am getting good throughput and it's increasing but after that graph plummets. This might be because of T2.Micro Instance and it's processor.

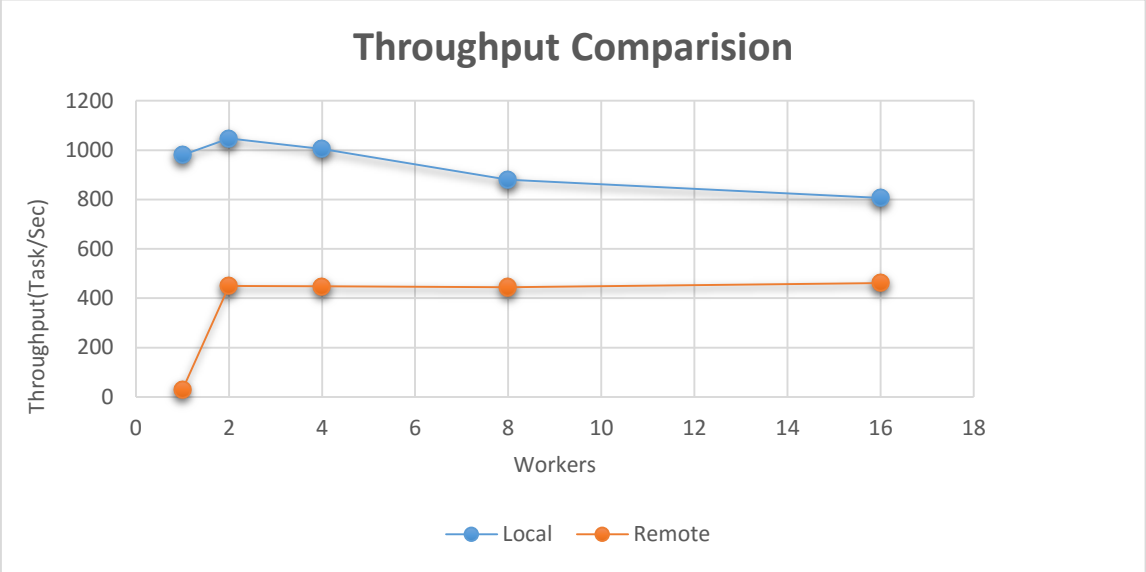
b. Efficiency:



Time	1	2	4	8	16
10 ms	86.956522	83.015109	82.101806	78.492936	41.017227
1 sec	99.820323	99.740674	99.710839	99.730727	99.581757
10 sec	99.740674	99.974007	99.978005	99.96901	99.951024

Above graph shows that efficiency remains on higher side and it gradually drops as the number of threads increases. In case of “Sleep 10s” task, my experiments produce a constant efficiency of 99%. Higher variation is seen in “Sleep 10ms” task as for this task each worker does 1000 task which results in other overheads leading to a visible drop as number of threads increases and at the end for 16 thread, it gives only 41% accuracy.

Throughput Comparison:



For comparison part, Local is performing better than Remote worker. As can be seen from the above graph, the best performance is seen for 4 threads. It is because there are 4 hyper-threads on the system. The throughput decreases as the number of threads increases from 4 onwards. The efficiency decreases as the number of threads increases. The reason is that more time is being wasted in synchronization.