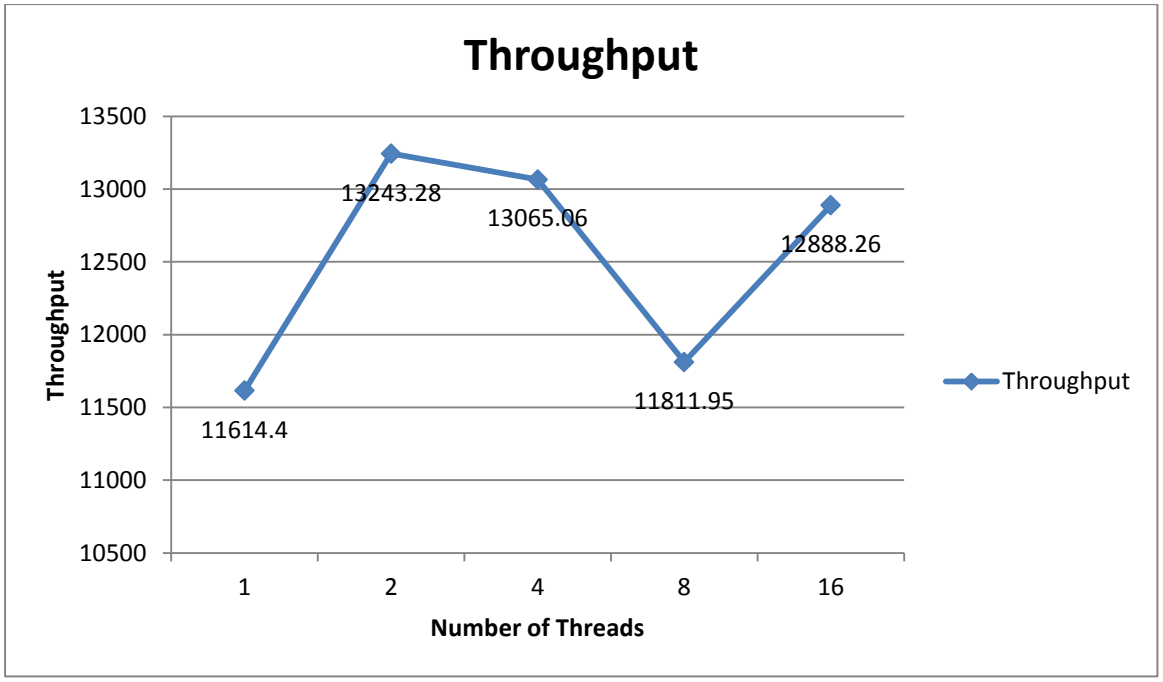
Performance Evaluation:

Sleep 0

Number of Threads	Time (sec)	Throughput
1	8.61	11614.4
2	7.551	13243.28
4	7.654	13065.06
8	8.466	11811.95
16	7.759	12888.26



The above diagram shows the throughput for sleep 0 condition.

Ideal time = Sleep time* nooftasks/no of workers

Efficiency= idealtime/actualtime

Throughput= No of tasks/No of Workers

for 10ms

Number of Threads	Number of Tasks	Time (sec)	Ideal Time	Efficiency	Efficiency(%)	Throughput
1	1000	10.39	10	0.9624639	96.24639076	96.24639076
2	2000	10.478	10	0.9543806	95.4380607	190.8761214
4	4000	10.583	10	0.9449117	94.49116508	377.9646603
8	8000	10.825	10	0.9237875	92.37875289	739.0300231
16	16000	11.228	10	0.8906306	89.06305664	1425.008906

For 1000 ms

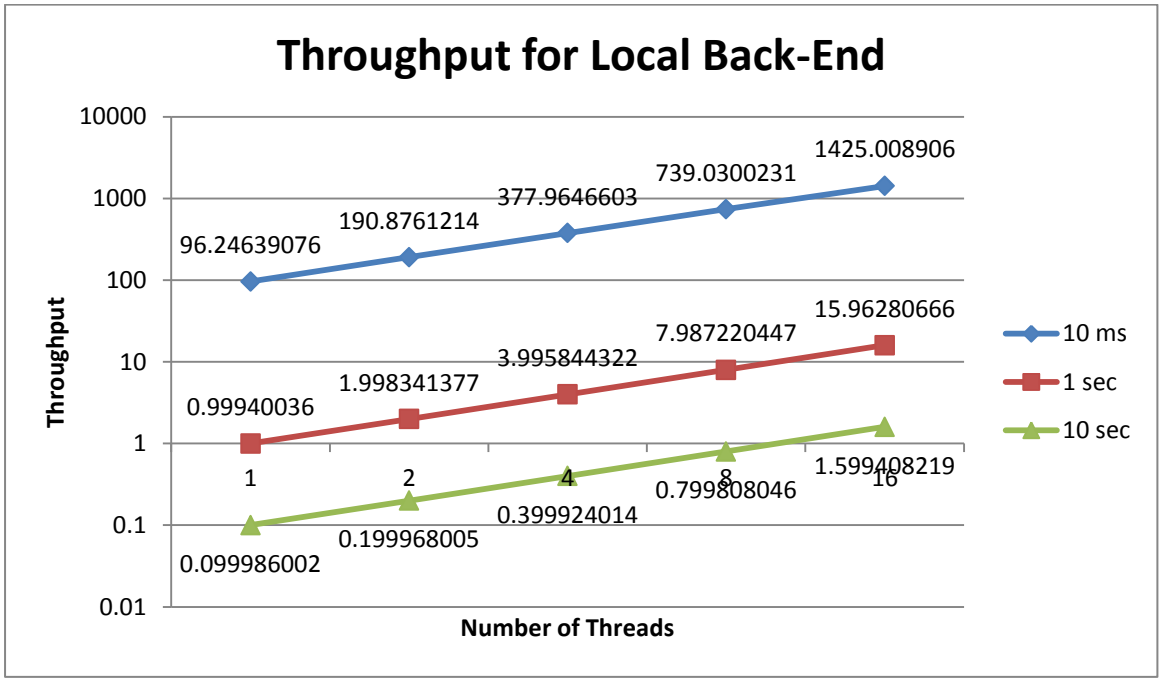
Number of Threads	Number of Tasks	Time (sec)	Ideal Time	Efficiency	Efficiency(%)	Throughput
1	100	100.06	100	0.9994004	99.94003598	0.99940036
2	200	100.083	100	0.9991707	99.91706883	1.998341377
4	400	100.104	100	0.9989611	99.89610805	3.995844322
8	800	100.16	100	0.9984026	99.84025559	7.987220447
16	1600	100.233	100	0.9976754	99.76754163	15.96280666

Sleep 10000ms

Number of Threads	Number of Tasks	Time (sec)	Ideal Time	Efficiency	Efficiency(%)	Throughput
1	10	100.014	100	0.99986	99.98600196	0.099986002
2	20	100.016	100	0.99984	99.98400256	0.199968005
4	40	100.019	100	0.99981	99.98100361	0.399924014
8	80	100.024	100	0.9997601	99.97600576	0.799808046
16	160	100.037	100	0.9996301	99.96301368	1.599408219

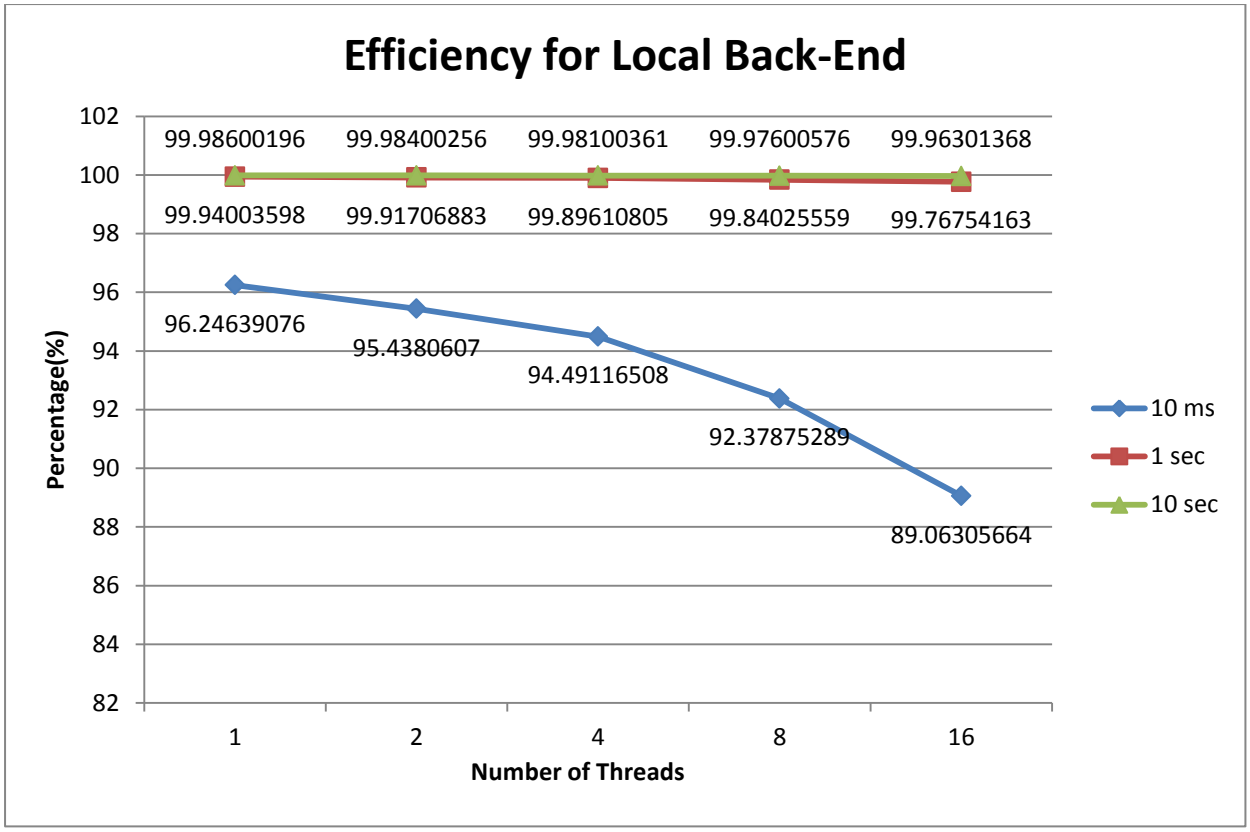
Graph:

Throughput for 10ms, 1 sec and 10 sec



The above graph shows the throughput for Local Back-End. The graph is taken in log-scale. Here we observe the increase in the throughput as the number of thread increases for tasks that are performing foe 10 ms, 1 sec and 10 sec . Here for 10ms the number of tasks given is 1000 and as the number of threads increases correspondingly the number of tasks also increases. i.e for 2 threads, number of tasks is 2000 and so on. For 1 sec the number of tasks given is 100 for 1 thread and so on and for 10sec the number of tasks is 10 for 1 thread and so on.

Efficiency for 10ms, 1 sec and 10 sec



The above graph shows the efficiency for Local Back-End . Here we observe a slight the decrease in the efficiency as the number of thread increases for tasks that are performing foe 10 ms, 1 sec and 10 sec . Here for 10ms the number of tasks given is 1000 and as the number of threads increases correspondingly the number of tasks also increases. i.e for 2 threads, number of tasks is 2000 and so on. For 1 sec the number of tasks given is 100 for 1 thread and so on and for 10sec the number of tasks is 10 for 1 thread and so on. If worker increase task scheduling is difficult, some workers remain idle hence efficiency decreases

Remote Back-End Performance:

For 10 ms:

Number of Threads	Number of Tasks	Time (sec)	Ideal Time	Efficiency	Efficiency(%)	Throughput
1	1000	153.39	10	0.0651933	6.519329813	6.519329813
2	2000	151.947	10	0.0658124	6.581242144	13.16248429
4	4000	496.154	10	0.020155	2.015503251	8.062013004
8	8000	1194.587	10	0.0083711	0.837109394	6.696875154
16	16000	2886	10	0.003465	0.346500347	5.544005544

For 1 sec:

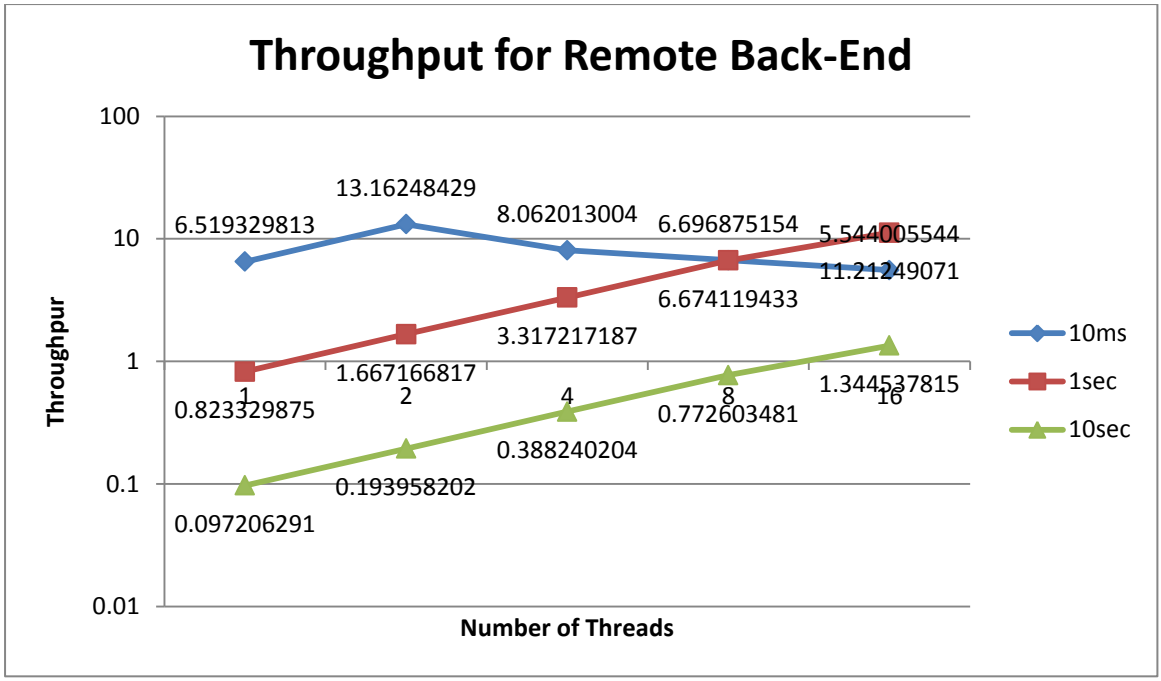
Number of Threads	Number of Tasks	Time (sec)	Ideal Time	Efficiency	Efficiency(%)	Throughput
1	100	121.458	100	0.8233299	82.33298753	0.823329875
2	200	119.964	100	0.8335834	83.35834084	1.667166817
4	400	120.583	100	0.8293043	82.93042966	3.317217187
8	800	119.866	100	0.8342649	83.42649292	6.674119433
16	1600	142.698	100	0.7007807	70.07806697	11.21249071

For 10sec:

Number of Threads	Number of Tasks	Time (sec)	Ideal Time	Efficiency	Efficiency(%)	Throughput
1	10	102.874	100	0.9720629	97.20629119	0.097206291
2	20	103.115	100	0.969791	96.979101	0.193958202
4	40	103.029	100	0.9706005	97.06005105	0.388240204
8	80	103.546	100	0.9657544	96.57543507	0.772603481
16	160	119	100	0.8403361	84.03361345	1.344537815

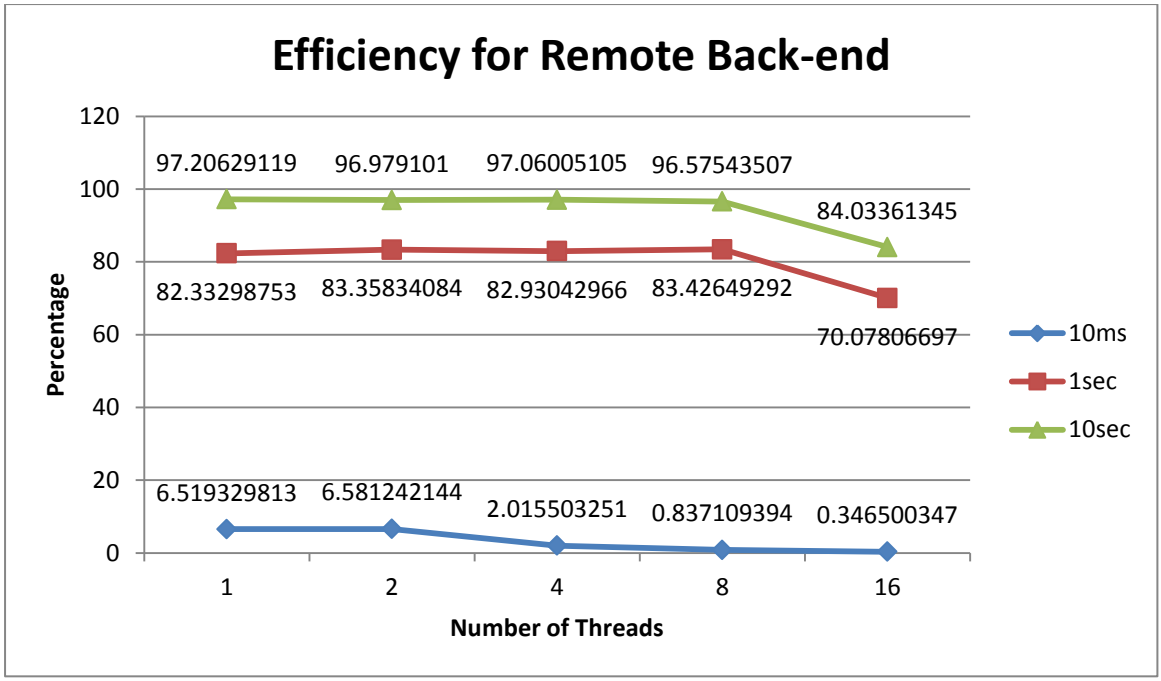
Graph:

Throughput for 10ms, 1 sec and 10 sec



The above graph shows the throughput for Remote Back-End. The graph is taken in log-scale. Here we observe the increase in the throughput as the number of thread increases for tasks that are performing for 1 sec and 10 sec .Whereas there is a decrease for 10 ms . Here for 10ms the number of tasks given is 1000 and as the number of threads increases correspondingly the number of tasks also increases. i.e for 2 threads, number of tasks is 2000 and so on. For 1 sec the number of tasks given is 100 for 1 thread and so on and for 10sec the number of tasks is 10 for 1 thread and so on.

Efficiency for 10ms, 1 sec and 10 sec



The above graph shows the efficiency for Remote Back-End . Here we observe a slight the decrease in the efficiency as the number of thread increases for tasks that are performing foe 10 ms, 1 sec and 10 sec . Here for 10ms the number of tasks given is 1000 and as the number of threads increases correspondingly the number of tasks also increases. i.e for 2 threads, number of tasks is 2000 and so on. For 1 sec the number of tasks given is 100 for 1 thread and so on and for 10sec the number of tasks is 10 for 1 thread and so on. . If worker increase task scheduling is difficult, some workers remain idle hence efficiency decreases