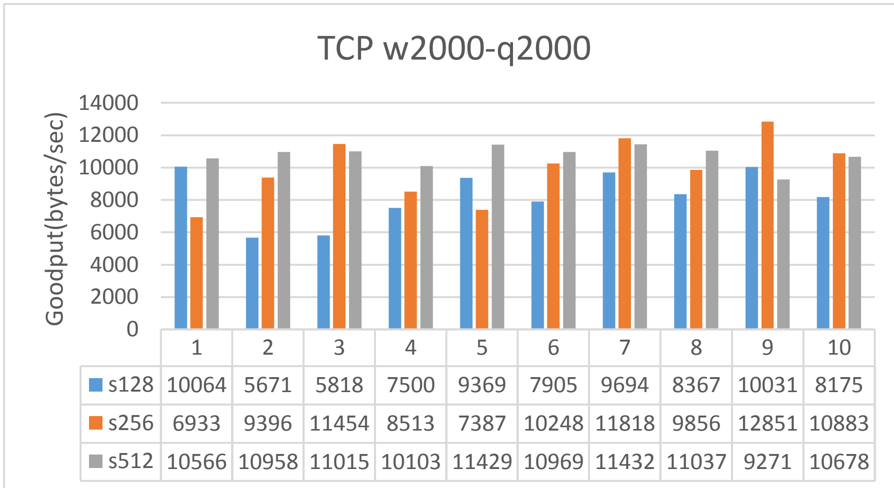
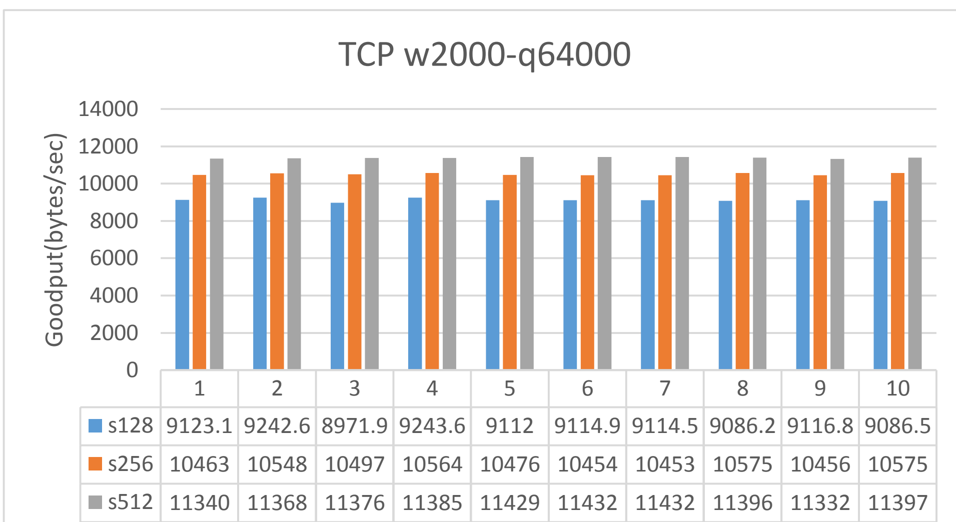


	1	2	3	4	5	6	7	8	9	10
s128	10064	5671	5818	7500	9369	7905	9694	8367	10031	8175
s256	6933	9396	11454	8513	7387	10248	11818	9856	12851	10883
s512	10566	10958	11015	10103	11429	10969	11432	11037	9271	10678

In the first case, when the max window size is limited to 2000 we can see that the goodput obtained is also constrained by the window size. Some flows receive fair service while some of them suffer due to dropped packets. Specifically, the flows with the smaller start time have a greater goodput, this is possibly because of the fact that packets from the other flows might be dropped frequently.

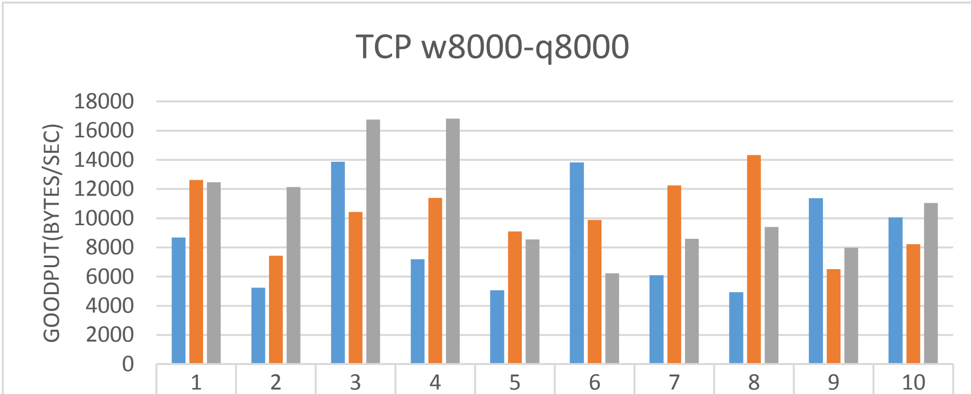
EFFECT OF SEGMENT SIZE ON THE GOODPUT:



	1	2	3	4	5	6	7	8	9	10
s128	9123.1	9242.6	8971.9	9243.6	9112	9114.9	9114.5	9086.2	9116.8	9086.5
s256	10463	10548	10497	10564	10476	10454	10453	10575	10456	10575
s512	11340	11368	11376	11385	11429	11432	11432	11396	11332	11397

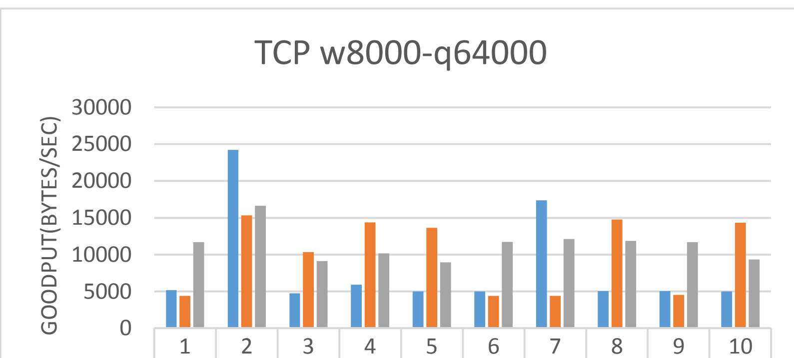
In this case we can see that the queue size is large and the window size is small, this means that there is a small stream of packets from all the flows which are queued and serviced fairly. The average goodput is uniform over all flows. The segment size plays a major role in this case because TCP Tahoe tends to enter the 'Slow Start' phase a lot of times during retransmissions and hence the window size drops to 1xMSS (maximum segment size). Thus we see that the flows that have slightly better segment sizes have slightly better goodputs on average.

EFFECT OF QUEUE SIZE ON THE GOODPUT:

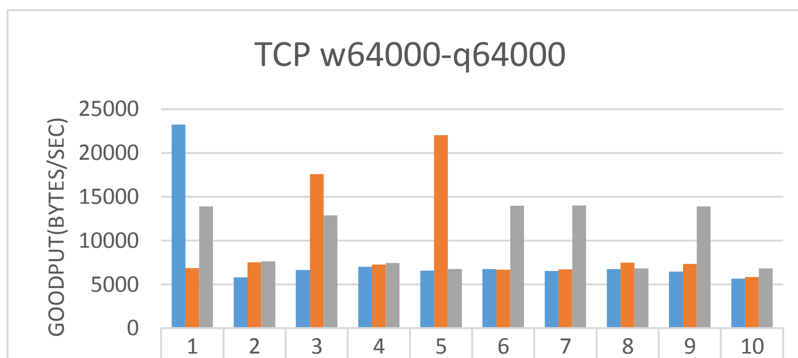


	1	2	3	4	5	6	7	8	9	10
s128	8672.1	5235.8	13872	7179.5	5057.9	13827	6089.2	4928.1	11370	10062
s256	12628	7424.8	10420	11385	9086.3	9887.4	12256	14322	6515.7	8213.8
s512	12473	12136	16754	16821	8545.7	6231.1	8599.6	9394.2	7983.7	11037

In the case of the queue size being equal to the window size, the goodput is higher, on average for all flows. All flows are getting serviced fairly and equally on average. This might be because the queue is buffering exactly the right amount and there is no loss of packets from each of the flows. There still seem to be certain odd cases where there are peaks and drops but the average performance is evidently better than the other cases. Therefore, there seems to be both an upper and lower bounds for queue size to ensure optimal performance.



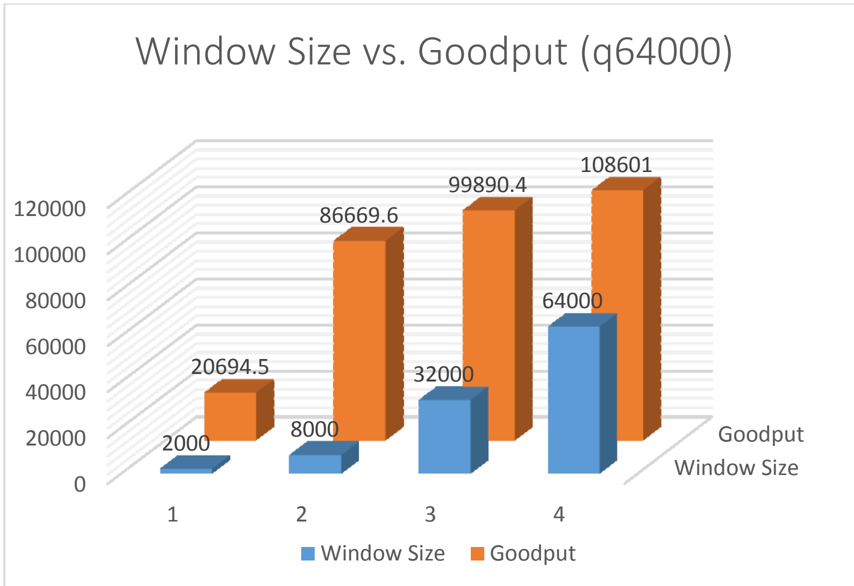
	1	2	3	4	5	6	7	8	9	10
s128	5167	24220	4732	5897	5019	5008	17354	5031	5048	4980
s256	4381	15310	10316	14385	13642	4377	4377	14759	4507	14323
s512	11700	16642	9101	10154	8958	11741	12101	11858	11692	9343



	1	2	3	4	5	6	7	8	9	10
s128	23233	5799	6658	7026	6589	6746	6540	6763	6464	5660
s256	6855	7527	17582	7256	22034	6695	6720	7469	7340	5827
s512	13917	7630	12876	7436	6744	13956	14007	6827	13907	6828

In the case of the queue size being much large, we see that it results in unfairness and poor performance amongst all the flows. This might be due to the fact that certain flows flood the queue and certain flows are not being served at all. This causes a large queueing delay to build up in the other flows and hence causes a lot of loss in goodput when viewed over a short timespan like 10 seconds, hence short term fairness is not present.

EFFECT OF WINDOW SIZE ON THE GOODPUT:



This is the case of a single flow in the case of the **queue size being fixed at 64000**. It is clear that window size is directly proportional to the goodput.

References:

[1] Amogh Dhamdhere, Hao Jiang, Constantinos Dovrolis. Networking and Telecommunications Group. Georgia Institute of Technology. *"Buffer Sizing for Congested Internet Links"*.

[2] Anshul Kantawala, Jonathan Turner. Department of Computer Science. Washington University. "*Efficient Queue Management for TCP Flows"*

[3] Chi Zhang, Lefteris Mamatas. School of Computer Science. Florida International University. Department of Electrical and Computer Engineering, Demokritos University. "*The Interaction between Window Adjustment Strategies and Queue Management Schemes*".