Name: Zishi Deng (Steve)
netID: zd475

Parallel Computing Lab2 Report

Table I

N=10,000

#Thread	1	2	5	10	20	40	80	100
Time	0.000358	0.000435	0.000544	0.000750	0.00119	0.00197	0.00453	0.00549
Speedup	1.000	0.823	0.658	0.477	0.301	0.182	0.0790	0.0652

Table II

N=100,000

#Thread	1	2	5	10	20	40	80	100
Time	0.00379	0.00323	0.00305	0.00323	0.00358	0.00465	0.00676	0.00775
Speedup	1.000	1.173	1.243	1.173	1.059	0.815	0.561	0.489

For Table I, we can see that speedup decreases as number of threads increases. It is because there is very large overhead in spawning new threads, scheduling tasks of crossing out multiples in parallel and joining threads before termination.
Although each thread is doing roughly equal among of work, there are many repeated work as it will cross out the multiple again even it was crossed out in the previous iteration. In total, the time saved running in parallel of generating numbers and crossing out multiples of numbers was offset by the time consumed due to these extra overheads.

For Table II, as the problem size grow by 10 times, we can see the speedup increases initially from 2 to 5 threads, and then decreases as number of threads increases but to a much smaller extent as compared to Table I. Overall, we can achieve a positive speedup from 2 to 20 threads. It shows that there is still large overhead in spawning new threads, scheduling tasks among the threads and synchronization of threads before termination which offset the amount of time saved in crossing out the multiples of a number in parallel when we pawn threads beyond the optimal level which is 5 in this case. However as the problem size is getting larger, we can see there is a large

improvement in the performance as compared to running the same parallel algorithm on a smaller data set.