Name: Zishi Deng (Steve)
netID: zd475

Parallel Computing Lab1 Report

Table I

# of unknown	10	100	1000	10000	100000
1	0.958	1.034	2.403	166.5	N/A
2	1.040	1.098	2.676	146.0	N/A
10	2.260	2.043	3.965	178.7	N/A
20	N/A	3.123	4.913	200.6	N/A
40	N/A	N/A	7.690	220.1	N/A

*All data was produced after averaging over 5 repeated results
*100000 unknown data was not able to be generated due to CIMS quota limit

Table II

Speedup	10	100	1000	10000	100000
1	1.000	1.000	1.000	1.000	N/A
2	0.921	0.942	0.897	1.140	N/A
10	0.424	0.506	0.606	0.932	N/A
20	N/A	0.331	0.489	0.830	N/A
40	N/A	N/A	0.312	0.756	N/A

3. a) There is no speedup at all running on small to medium sized data, namely from 10 unknowns to 1000 unknowns. At the same time, creating more processes will slow down the program to a larger extent and inversely affecting the performance.

   b) It is because, there are very large overheads in creating extra processes and in synchronizing the communication among these processes. After calculating the value of variables in one iterations, the program has to update the value of them in all processes to be used for the calculations in the next iteration. So there will be a lot of communication involved in gathering these data to all processes. Hence, when running paralleled code on small to medium sized data, there is no speedup at all in creating more processes and run them in parallel.

c) There is no significant speedup except running on relatively large dataset (10000) using two processes, we can get a speedup of 1.14 . In all other cases, either running on smaller dataset or using more number of processes will slow down the program. In general, we can discover the trend that when running on larger problem size, the speedup generally improves across same number of processes

d) The reason for the speedup is that the efficiency in solving the linear equation of 10000 unknowns using 2 processes overcomes the overheads involved in creating two processes and the cost of communication and synchronization among these two processes. As when the problem size increases, a single process takes longer time to get the computation task done. At the same time, when there is less number of processes, the communication of data groups into larger chunk which reduces the number of communication as well as the overhead in creating and synchronizing these processes. Hence, it works for very large problem size with relatively little number of processes.