CS133 Lab2 Report
Xiaohui, Zhou
104-014-248

a) Please briefly explain how you have partitioned the data and the computation among the processors. Also, briefly explain how the communication among processors is being done.

In my algorithm, the data are partitioned by rows. Each processor will be assign with N/16 rows of matrix for calculation. In the beginning, the root processor will share corresponding N/16 rows of A with each processor and then broadcast the whole B matrix with them. Then, after all processor finished their own work, the root processor will collect all the calculated results and assemble them into matrix C.

b) Please provide the runtime using blocking (MPI_Send, MPI_Recv), buffered blocking (MPI_Bsend, MPI_IRecv), and non-blocking (MPI_Isend, MPI_Irecv) communication. Which type of communication gives best result? Please explain why. Note that the final submission file should use the type of communication that gives the best result.

Blocking: ~ 7 – 8 GFlops
Buffered Blocking: ~ 7 – 8 GFlops
Non-blocking: ~ 8 – 9 GFlops

Buffered Blocking has a better performance than Blocking because of the existence of buffer. The root processor doesn't need to wait for other processors to get ready for sending messages. However, if the processors can start working at the same time the root processor want to send message, then direct blocking would be better as they no longer need to do copying with buffer.
However, non-blocking is even better because it does not need to wait nor copying data with buffer. In this case, as the tasks are all independent, so non-blocking should be have the best performance, as we expected.

c) Please report the scalability of your algorithm using mpirun –np 4, 8, 16, 32. Do you get linear speedup? If not, why? Note that the final submission file should be optimized for np=16.

4: ~7 – 8 GFlops
8: ~7 – 8 GFlops
16: ~8 - 9 GFlops
32: ~8 - 9 GFlops

I did not get a linear speed up when as the number of processors increases, mainly due to the overhead during initial and final communications. It will take a noticeable amount of time to copying data for the root process and so the overall performance would be affected.

d) Please list all optimization you have used, and quantify the impact of each optimization. You are welcome to use any optimization that increases the performance.

- I tried to flip the matrix B so that newB[i][j] = oldB[j][i] to reduce the cache miss rate, but the result performance will not change much when n= 2046

e) Challenges you encountered and how you solved them

None.