CS 4230
Qixiang Chao
Programming Assignment Report

The running time of cannon's algorithm in ms of different process numbers and different matrix size.

	4 procs	9 procs	16 procs	25 procs	36 procs	49 procs
4x4	0.405	0.410	0.604	0.788	81.399	169.569
16x16	1.708	1.533	1.564	1.506	88.376	106.647
128x128	57.219	57.022	72.534	65.477	148.783	247.823
256x256	895.906	949.224	353.038	253.146	364.050	474.016
512x512	12575.997	8430.494	1060.528	6436.113	5785.429	4015.343
1024x1024	26910.633	29075.112	25239.190	31593.332	21725.683	32911.654

The running time of the sequential code with different matrix size.

4x4	0.134
16x16	2.78
128x128	184.389
256x256	816.283
512x512	8374.974
1024x1024	26314.247

After comparing the running time by using different processes or matrix size, I found that when the matrix size is small, and the process number is small, the running time of cannon's algorithm is similar or slower than sequential code. And also if matrix size is small, the running time of mpi code with large process number is significant slower than small process number. With the increasing of the matrix size, the running time of mpi code began faster than sequential code, especially with 16, 25 and 36 processes, the running speed is fast. Consequently, my result shows that if the matrix size is small, the sequential code and mpi code have very similar running speed. If the matrix is large, the mpi code has higher efficiency than sequential code. And more processes for mpi code does not mean higher efficiency. Lots of processes might decrease the running speed.