High Performance Computing — Homework 4

by Manyuan Tao (tm2735)

Location of my git repository:

https://github.com/tmyangel/HPC17-Homework4.git

1. MPI-parallel 2D Jacobi smoother

1.1 Weak scaling

fix number of iterations $max\_iters = 100$

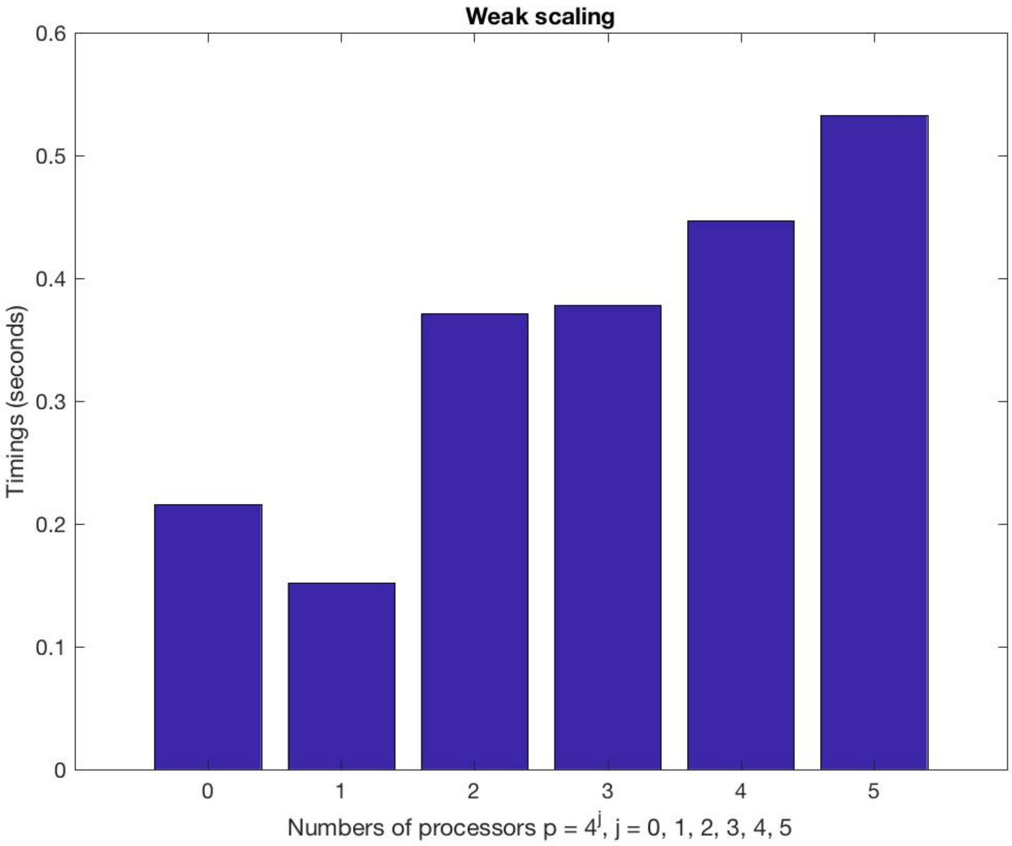
fix number of points per MPI task $N_l = 500$

run with number of MPI tasks $p = 1, 4, 16, 64, 256, 1024$

- Result

Number of processors p	1	4	16	64	256	1024
Execution time (seconds)	0.2155	0.1519	0.3711	0.3782	0.4468	0.5325

- Plot the timings



1.2 Strong scaling

fix number of iterations $max\_iters = 100$

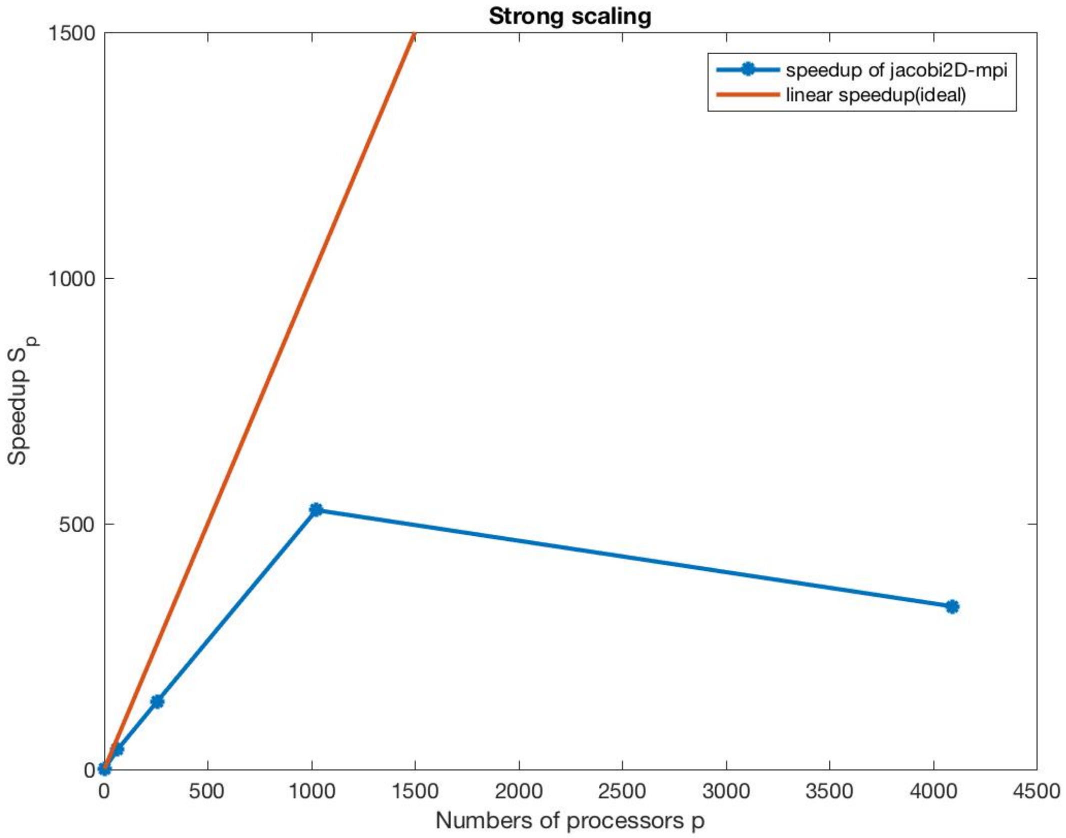
fix total number of points $N = 8000$

run with number of MPI tasks $p = 1, 64, 256, 1024, 4096$

- Result

Number of processors p	Execution time T_p (seconds)	Speedup $S_p = T_1/T_p$
1	61.107787	1
64	1.526234	40.038282
256	0.442216	138.185382
1024	0.115862	527.418714
4096	0.184381	331.421280

- Plot the speedup compared to the ideal speedup



1.3 Non-blocking

fix number of iterations $max\_iters = 100$

run blocking/non-blocking versions on Stampede with 64 processors

run with total number of points $N = 100,$

- Comparison in the run time

Total number of points N	Blocking	Non-blocking
200	0.037775 s	0.047191 s
400	0.027834 s	0.040573 s
800	0.036927 s	0.037595 s
1600	0.067784 s	0.062654 s
3200	0.179468 s	0.180133 s
6400	0.907950 s	0.897840 s

- Observation

As seen from the above table, I didn't notice much time difference between blocking and non-blocking algorithms in my experiment.

Probably we can run on more processors for comparison.

2. Parallel sample sort

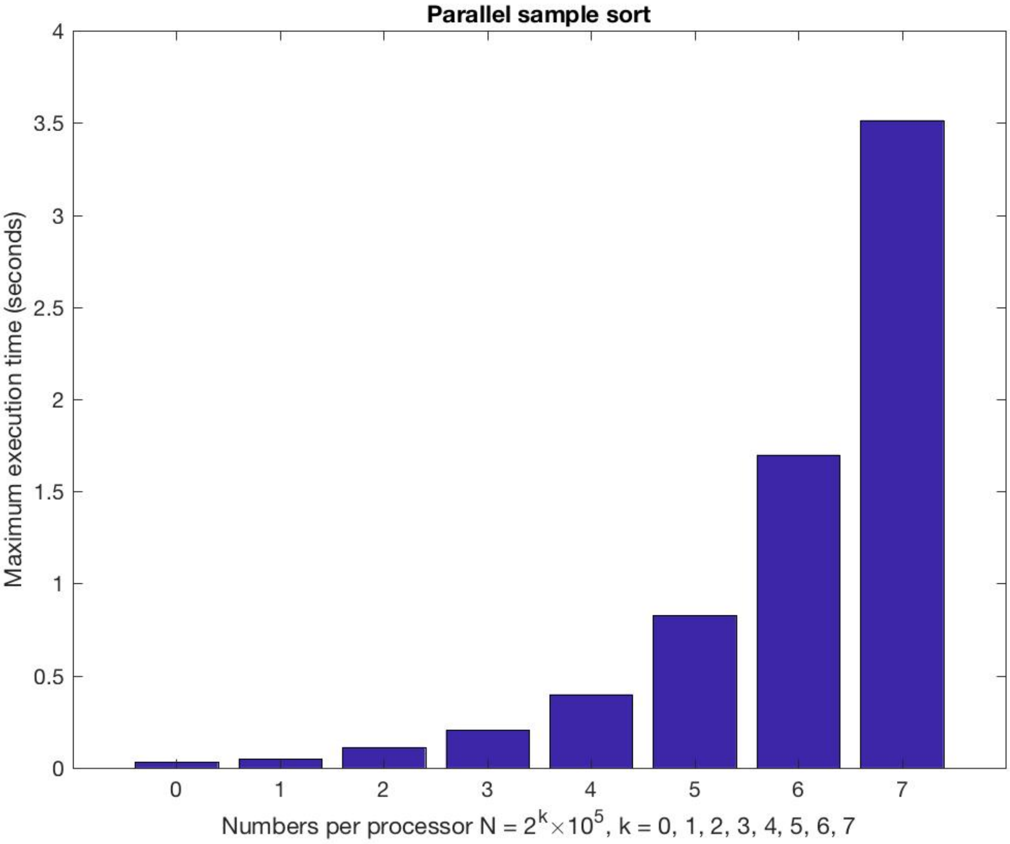
run on 64 processors of Stampede

run with number of elements per processor $N = 1, 2, 4, 8, 16, 32, 64, 128 \times 10^5$

- Present timings

Value of $N(\times 10^5)$	Maximum execution time (seconds)
1	0.032763
2	0.049816
4	0.112097
8	0.204249
16	0.398289
32	0.828677
64	1.698663
128	3.514385

- Plot


Parallel sample sort
Maximum execution time (seconds)
Numbers per processor N = 2^k×10^5, k = 0, 1, 2, 3, 4, 5, 6, 7