Parallel Progrmg: Sci & Engrg

Machine Problem 3

yhyuan2

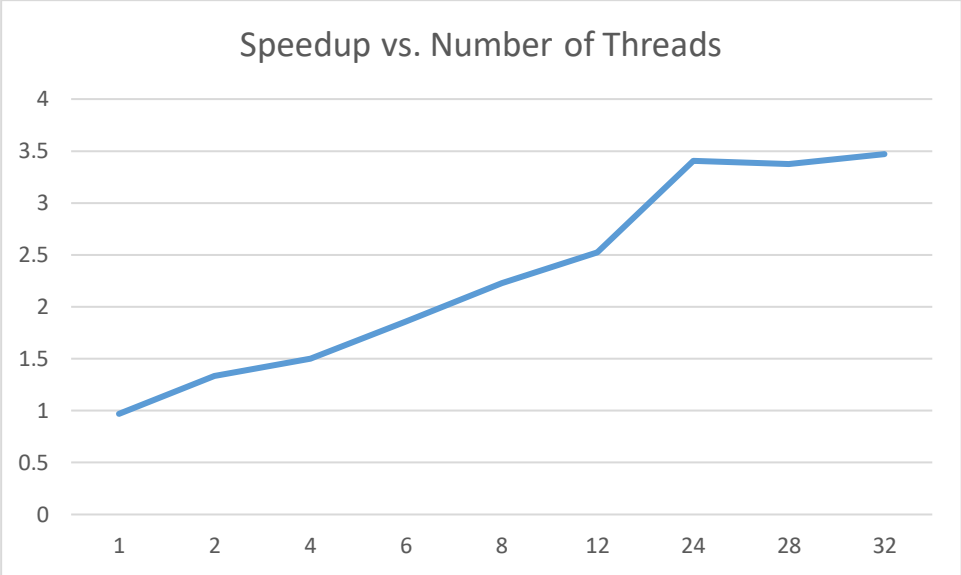
A. Q: Try running the benchmark program with the following data sizes: 256, 512, 1024, 2048, 4096. Briefly comment about how changing the data size affects the performance of the parallel version with four threads, and include the output of the benchmarking program for data size 2048 in your report.
A: When data size is 512, speedup is the lowest. But while increasing data size, speedup will be about 1.6.

	256	512	1024	2048	4096
speedup	1.632	1.199	1.286	1.498	1.581

Generating a 2048x2048 mandelbrot with thread counts: [1, 2, 4, 6, 8, 12, 24, 28, 32]
The serial version ran for 1.8015558279876132 s.
The parallel version, with 1 thread(s), ran for 1.858623828011332 s, a speedup of 0.969x.
The parallel version, with 2 thread(s), ran for 1.3504135670082178 s, a speedup of 1.334x.
The parallel version, with 4 thread(s), ran for 1.2028670339786913 s, a speedup of 1.498x.
The parallel version, with 6 thread(s), ran for 0.9705147970234975 s, a speedup of 1.856x.
The parallel version, with 8 thread(s), ran for 0.8097687760018744 s, a speedup of 2.225x.
The parallel version, with 12 thread(s), ran for 0.7136495879967697 s, a speedup of 2.524x.
The parallel version, with 24 thread(s), ran for 0.5291206499969121 s, a speedup of 3.405x.
The parallel version, with 28 thread(s), ran for 0.5339066930173431 s, a speedup of 3.374x.
The parallel version, with 32 thread(s), ran for 0.5192355140170548 s, a speedup of 3.470x.

B. Q: For data size 2048... Plot Speedup vs. Number of Threads using the results of the benchmark program.

A:



C. Q: For data size 2048... How did the performance of the parallel version with a single thread compare to the serial version? Briefly explain your results.
A: Single thread is slower than serial version because it has an overhead of preparing arguments and creating threads.

D. Q: For data size 2048... How did the performance of the parallel version with multiple threads compare to serial version? Briefly explain your results.
A: Multithread version is faster because it can compute many data simultaneously.

E. Q: Using the campus cluster documentation to find the number of cores for the compute nodes, what happened when the number of threads exceeded the number of available cores? What would you expect to happen?
A: Using benchmark.py, I found there is 24 cores. Some threads will be halted because there isn't enough cores for running all threads. This will stop speedup from increasing when number of threads exceeds number of cores.