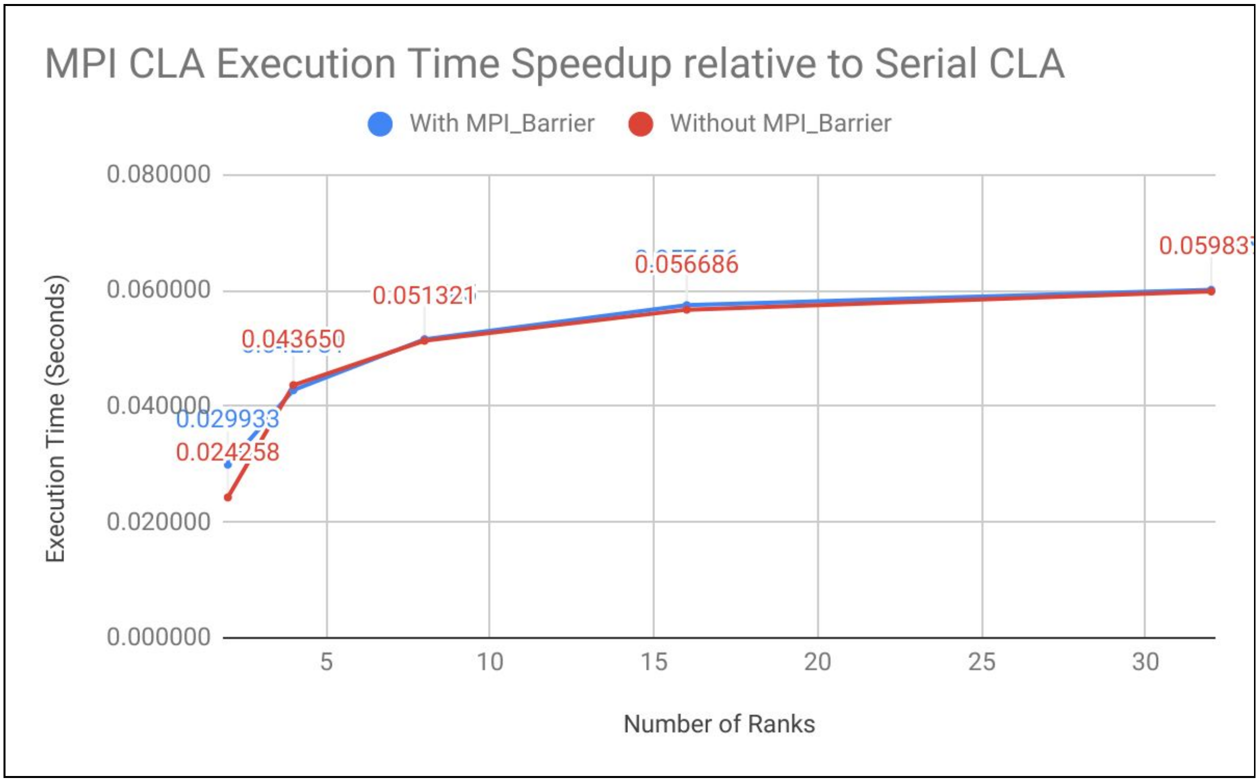
Harrison Lee
19Feb2019
Parallel Computing

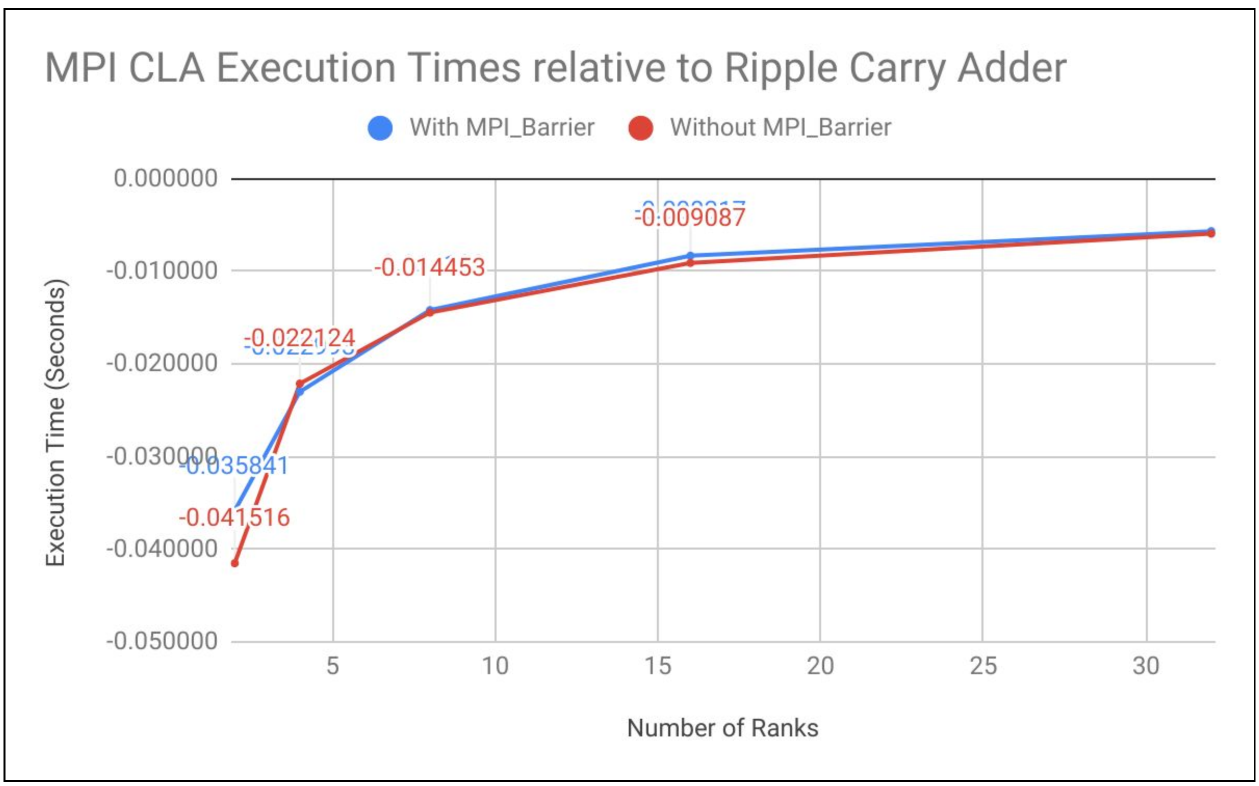
Homework 2 MPI_CLA (Carry Lookahead Adder) Performance Study:

Graphs:

Graph 1: Raw MPI CLA Execution Times



Graph 2: CLA Execution Time Speedups to Serial CLA



Graph 3: CLA Slowdowns Relative to Ripple Carry Adder

Discussion:

In graph 1, it is visible that our Carry Lookahead Adder (CLA) sped up with a greater number of ranks as a result of increased parallelization. As in Amdahl's Law, where speed up = 1/ ((1-p) + p/s), where p is the proportion of the program that is parallelizable and s is the increase in resources available for those parallelizable code segments. While s is directly related to speed up, it does not have a linear relationship, as shown by graph 1. Graphs 2 and 3 move the line by subtracting by the corresponding run times for a serial CLA adder and for a Ripple Carry Adder (RCA), 0.074292s and 0.008518 respectively. The speed up relative to a serial CLA adder is an intuitive result of the same speed up as the number of ranks increase. The slowdown relative to the RCA run times is likely a result of the bootstrapping needed to set up parallelization. This bootstrap time is a result of us simulating CLA, rather than running it on physical transistors that wouldn't be affected by this setup time. There might also be some hidden optimizations in the hardware or compiler, in addition to slow down resulting from the more complex code to implement CLA. Usage of MPI_Barriers had a negligible effect, with the strongest effect being visible with two ranks, with the testing where MPI_Barrier were used being faster. This did not have a clear explanation to me, and I expect the effect to be reduced with more test runs. In this case, there were 10 test runs, with additional potential interference from the available shared server resources fluctuating. We had to prune some outliers that were more than an order of magnitude longer than the majority of our execution times.