High Performance Computing: Homework 6

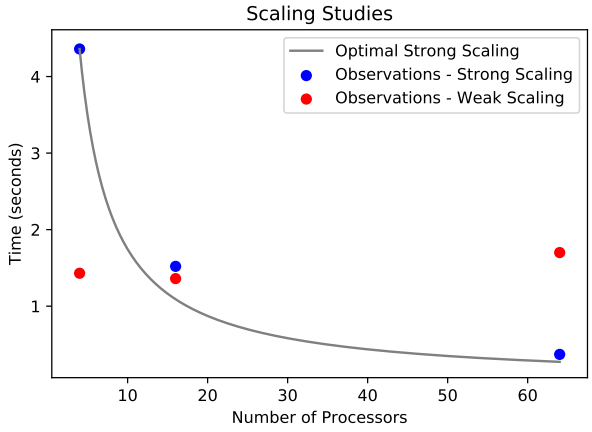
Dmitriy (Tim) Kunisky [dk3105]

Problem 0: Project Status

I have finished an implementation of the branch-and-bound method for solving MaxCut using an iteratively refined semidefinite programming relaxation. I have also added instrumentation to do basic modulation of how actively new "useful" constraints are propagated to be included in future relaxations, allowing a tradeoff between relaxation accuracy and communication overhead. This is done in a naive "all-or-nothing" way to keep the implementation simple: if adding some constraints achieves a certain threshold of objective function improvement, then those constraints are broadcast to all other nodes. My remaining plan is to run timing and performance studies to evaluate the tradeoff associated with this strategy, and to try more refined strategies if time permits.

Problem 1: MPI 2D Jacobi Smoothing

In both strong and weak scaling tests I used 2000 Jacobi iterations and tested with 4, 16, and 64 processors. In the weak scaling test I fixed $N_\ell = 500$, and in the strong scaling test I fixed $N = 2000$. In the weak scaling test, the timings remain roughly constant as the number of processors increases, as we expect since the work per processor remains roughly constant (with the exception of growing communication costs measured across the entire computation, which may begin to affect timing when cost of traffic over the network plays a role). In the strong scaling test, the timings decrease by approximately the optimal factor of 4 when the number of processors increases by a factor of 4.



Problem 2: MPI Sample Sort

I tested running on 64 cores (one per node) on Prince, for $N = 10^4, 10^5, 10^6$. The respective times recorded are 1.97, 2.23, and 3.30 seconds. As expected, the parallel implementation scales much better than the direct serial sorting algorithm with $O(N \log N)$ time guarantees, with which we would expect to see each successive time be at least ten times greater than the previous.