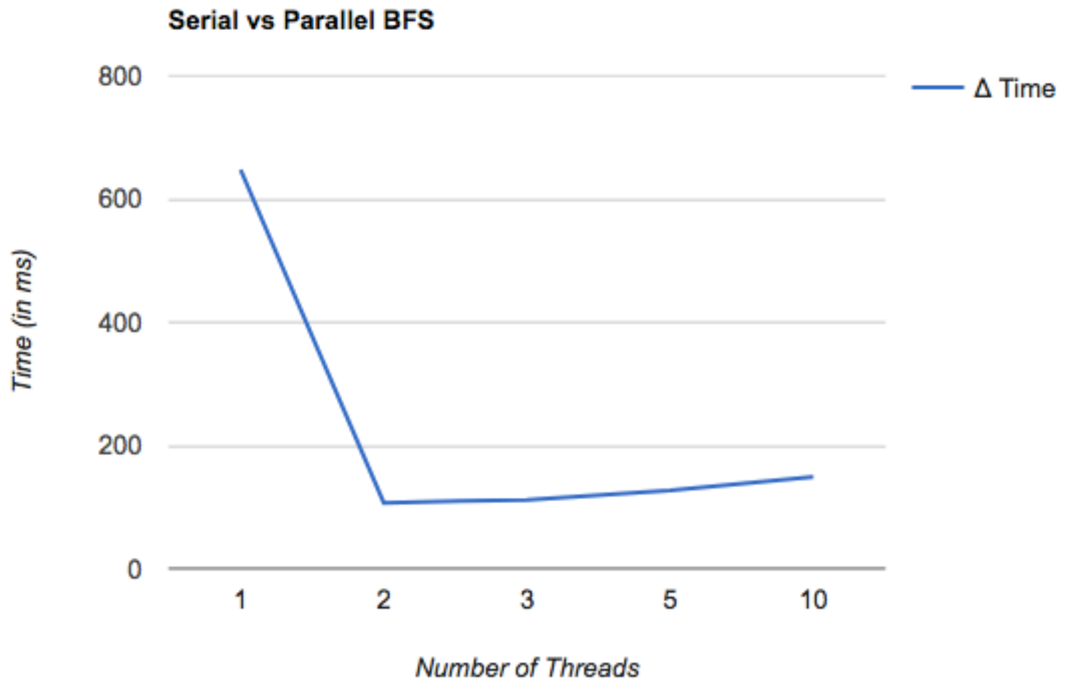
Daniel Acevedo (daacevedo) and Sashi Thapaliya (sbthapaliya)

CS 599-03

Parallel Breadth-First Search

For the following graph, we used the file g.txt, which is a file containing 99,999 vertices and 99,998 vertices, and chose a common source node, 0. The "1" label on the horizontal axis represents the serial run time, while the other labels represent a different amount of threads used for the parallelized version. The parallel BFS implementation's run time was significantly better than the serial version in all tests. The implementation was optimized using two threads on the campus cluster.



We attempted various implementations of parallel BFS, and while most implementations were faster than the serial version, they were not as significant as we would've liked. We found that our final solution, which included using an array of locks and a Boolean array, finally gave us the significant speed up that we set out to accomplish.

We made an observation that a parallel BFS implementation was not necessarily suitable for long-chain structured connected graphs, such as A->B->C->D->E, where each node is connected only by one node, with no two nodes having direct edges to the same node. In this case, because there was only one node at each level, our level synchronous parallelization was not beneficial.