Cuda Implementation

Our cuda implementation of the shortestPath function utilizes the GPU to find all ancestors of a given pair of nodes and then construct the shortest path between them. Our implementation assumes pc43 is being used, with 1024 threads per block and 1 GB of global memory. For simplicity’s sake, we assume the given data can fit in our GPU’s global memory. Our solution utilizes the fact that the shortest path between two nodes in a tree must converge at the lowest common ancestor of both nodes. In cases, where one node is an ancestor of another, then the shortest path is then found by traversing the parents of the child node. We parallelized our code by finding both sets of ancestors of the given nodes at the same time. Since, neither node needs to know about the other to find its own ancestors, this problem can be done independently of each other. Both sets of ancestors are then traversed to find the shortest path. We were unable to parallelize this part of the solution since each list of ancestors must be checked to find overlapping elements.

**TALK ABOUT TIME DIFFERENCES //gonna take a guess here**

Compared to the serial version, the cuda implementation performed slower in most test cases. While the cuda version can compute both given node’s ancestors at the same time, it must also load the entire tree into the GPU’s memory.