CS41 Final Project Online Algorithms
Initial Report and Proposed Action Plan
Tahmid Rahman, Dylan Jeffers

Results

Reasoning
Implementation Tests Results

Reasoning Despite the speed benefits Splay trees provide over frequenly accessed data, its runtime is signifcantly compromised when its average height gets large compared to its size. We felt this drawback was enough to consitute the construction of a new tree, one that both takes advantage of locality of reference like a Splay tree, yet can also rebalance itself like an AVL tree when its height becomes large . More specifically, our tree would use the ratio of tree height over max alternate to decide between a splaying or balancing. Additionally, the lack of literature on hybridized trees interested us in pursuing this model.

From an implementation perspective, Splay trees and AVL trees operate on a set of similar node rotations, making them good candidates for hybridization. To make things even simpler, we settled on a rotation model that requires only two unique rotations to fully satisfy the splay and balance functions. In addition to added simplicity, we chose this rotation to fulfil a herusitic goal slightly different from that proposed by Sleator Tarjan. Their implementation takes advantage of spatial locality by prioritizes all nodes in the path between the root and accessed node, while our implementation known as the "rotate to root" method takes advantage of temporal locality by prioritizing previously accessed nodes. (may want to add bit about tree height here) An example of this is presented on the next page.

Implementation For our tests, we implemented first implemented a Splay Tree, and then used this model to construct our hybrid tree which we will refer to as SPLAVL (pronounced as spla-vel).

Splay Our Splay tree uses a recursive stack frame model, meaning that each step in the path from the root to the accessed node is remembered by each stack frame. This decision was influenced by our work in CS35, where we implemented an AVL Tree with this model. Since the rotation functions are

(Note about Splay tree): To ensure a consistent model for testing purposes, we wrote our Splay Tree with

SPLAVL