Cameron Calv
CS260 Data Structures
Section 002
8/12/2019

Programming Assignment 2 Timing Analysis

Results and Conclusions:
For both operations of pre-order and post-order, there were a couple of data points taken. The reason for this is that for a full tree of a particular height, there is an drastic increase in number of nodes to traverse as the height grows larger. The results shown below showcase the timing in seconds of pre-order and post-order traversals of a full 3-level tree of height three and four with a total of 40 and 121 nodes respectively.

Table 1: Pre-Order Traversal Timing Results

Height	LOC Tree	LCRS Tree
3	0.0001751249999999982	0.00013373099999999805
4	0.0012672650000000035	0.0009278419999999982

Table 2: Post-Order Traversal Timing Results

Height	LOC Tree	LCRS Tree
3	0.00026364200000000837	0.00021078700000000394
4	0.0010220910000000027	0.0005374739999999961

From what can be observed, there is a slight increase in the amount of time that it takes for shorter trees from pre-order to post-order traversals. However, as the trees get larger, the post-order traversal becomes faster than the pre-order traversal. The increase in time per increase in height of the tree is also greater for the pre-order traversal versus the post-order traversal. For an increase of one in height pre-order, there is a corresponding increase of about 0.0009 seconds versus a increase of about 0.0005 seconds for the post-order traversal. The time it takes for either traversal is greater for the list of children (LOC) implementation than it is for the leftmost-child right-sibling (LCRS) implementation. This may have to do with the way that each implementation receives information about parent nodes and subsequent child nodes.