Algorithms for Massive Data Sets

Student name and id: Alexander R. Johansen (s145706)

Collaborator name(s) and id(s): Jonas (s142957)

Hand-in for week: 5

1 Path Sum problem

The Path Sum Problem can be solved in linear space using a a nearest common ancestor(NCA) data structure and a level ancestor(LA) data structure. The algorithm can be solved using a constant time on such data structure.

1.1 Analysis of problem

Any node in a tree is connected by one unique path(Cormen et al., 2009).
The $NCA(v_1, v_2)$ algorithm returns the ancestor node lying on the unique path between the two nodes.(prooved by contradiction)
As the unique path between two nodes must have one ancestor node in common(the placement in the tree in which the depth traversal along the unique path switches from inclining to declining) the ancestor node must have the largest depth of all ancestors to the two nodes given.
If an ancestor chosen did not lie on the path, the NCA algorithm did not pick the correct NCA(appendix A).
Having the ancestor node inbetween two nodes, computing the sum of weights from the root, minus the parent edge, will provide the weight inbetween the ancestor and the nodes(explained in detail on the specific algorithm)
Thus I will use the NCA and weight from the top of the tree to find the weight on the unique path between a pair of nodes.

1.2 Specific data structure

Given an empty tree T
1. Compute the depths of all nodes in T, and store it in the nodes.
Space: $O(n)$
2. Compute weighted edge traversal from the root to every node, and store their individual wieghts in the nodes, such that every node has a weight reference.
Space: $O(n)$
3. Compute level ancestor on the tree T.
Space: $O(n)$
4. Compute the NCA data structure on tree T.
Space: $O(n)$

1.2.1 Evaluation data structure

Step 1: $O(n)$
Step 2: $O(n)$
Step 3: $O(n)$
Step 4: $O(n)$
Total: $O(n)$

1.3 Specific algorithm

Given a two leafs l_1, l_2 and integers k_1, k_2
1. Use the Level Ancestor structure to find the k'th ancestor of l, such that:
$v_1 = LA(l_1, k_1)$
$v_2 = LA(l_2, k_2)$
time: $O(1)$
2. Using step 1: Compute the nearest common ancestor of v_1 and v_2, such that:
$v_n = NCA(v_1, v_2)$
time: $O(1)$
3. Compute the difference between v_1, v_n and v_2, v_n, such that:
$res = (v_1.weight - v_n.weight) + (v_2.weight - v_n.weight)$
time: $O(1)$

1.3.1 Evaluating algorithm

Step 1: $O(1)$
Step 2: $O(1)$
Step 3: $O(1)$
Total: $O(1)$

1.4 Appendix A

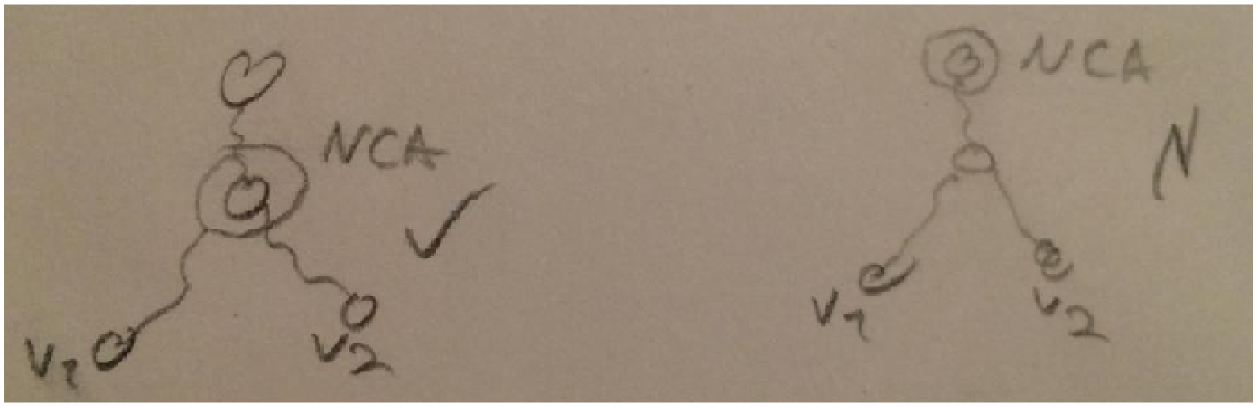

Figure 1: Contradiction of NCA on unique path.