1 Notes on Labeling the nodes of a tree

Consider the following problem: we have a rooted tree T with a label drawn from a set A assigned to each leaf. We want to assign labels to the non-leaf nodes so as to minimize the cost of the lableling. We incur a cost of 1 for each place pair (x, y) in T such that x is $y's$ parent, and the label of x doesn't match the label of y.

To compute the optimal labelling we use the following intermediate results:

Let $C(i, X)$ be the minimum total cost of the subtree rooted at i if we label vertex i with symbol X.

Let $C(i)$ be the minimum possible cost of the subtree rooted at i (thus it is the minimum of the $C(i, X)$ values).

We can compute the C values and find a min-cost labelling as follows:

Do a postorder traversal of the tree (so a node is processed after all its children are done). For a leaf C$(i, X) = 0$ for its label X, otherwise infinity.

For a non-leaf we compute $C(i, X)$ as follows: For each child j of i let cost$(j, X) =$ MIN $\{C(j, X), 1 + C(j)\}$. This represents the best way to label the subtree rooted at j when its parent has label X: label node j with X at no cost and do the best you can below j, or pay 1 for a mismatch with the parent and label j optimally.

Now $C(i, X) = cost(j, X)$ summed over all children j of i. Thus if we have $C(j, X)$ and $C(j)$ already computed, we can compute $C(i, X)$ in time proportional to its number of children. Each node has at most 1 parent, so we only compute cost(j, X) once for each j, X pair, and cost(j, X) is computed in O(1) time. Thus the total time is O(nA) where A is the size of the alphabet and n is the number of nodes in the tree.

This gives the best value for the root eventually. We can easily get the best labeling by retracing the decisions we made to get the values (thus for C(1) we have an associated symbol X for the root. We then look at which of the two terms for each child of the root gave the min for cost(j, X) and so on.