CS 218 : Design and Analysis of Algorithms

Lecture 14: Problems on Trees

Lecturer: *Sundar Vishwanathan*
Computer Science & Engineering Indian Institute of Technology, Bombay

1 Problem Statement

Input: A tree with each vertex having a weight w_i.
Output: Maximum weight subset of vertices such that no two adjacent vertices are selected.

We call our initial procedure: MaxWtTree (T)

1.1 Recursion

We start with the by now familiar idea: the required subset may contain a vertex or it may not.

If it contains the first vertex, then discard its neighbours and the edges incident on its neighbours. This results in some number of smaller subtrees. We recurse on each and add up the solutions and the weight of the first element to get our solution to this part. Similarly if it does not contain the first vertex, we discard only this vertex (and remove edges incident on it), get a set of subtrees and recurse on each. Output the maximum of the values output by the two recursive calls.

What about the number of recursive calls? This could be as high as the number of subtrees of a tree. Exercise: Construct a tree with an exponential number of subtrees.

So, we need to order the input. For trees a natural order is to pick a vertex as the root and now the order is away from the root (or towards it). It is easy to see that now, we have one distinct call for each tree rooted at a vertex. That is n of them. Consider the subtree rooted at v. Suppose its children are $v_1, \ldots, v_p$ and grandchildren $u_1, \ldots, u_q$. Then the recurrence is given below:

Maxsubset(v)
Choose the maximum of the following two quantities:

$$W_v + \Sigma_j M\text{axsubset}(u_j)$$

$$\Sigma_k M\text{axsubset}(v_j)$$

The first is when the root is chosen and the second when it is not.

Exercise. Write the complete procedure including the base case. Define the table and analyse. In this case, the iterative version is also immediate. Solve the problem for the children before solving it for a vertex in the tree.