What is DP on trees?

Dynamic Programming(DP) is a technique to solve problems by breaking them down into overlapping sub-problems which follow the optimal substructure. We all know of various problems using DP like subset sum, knapsack, coin change etc. We can also use DP with trees to solve some specific problems.

We define functions for nodes of the trees, which we calculate recursively based on the children of a node. One of the states in our DP usually is node i, denoting that we are solving for the subtree of node i.

General Structure of Tree DP

We define a function *f* that calculates some required value in the subtree of node i. We are usually interested in f(root) where the *root* is the root of the tree. We try to divide the function f of some node into subproblems of subtrees of each of the children. So we have

$$f\,(node) = merge(f(v_1),f(v_2).....f(v_k)\ )$$

where v_1,v_2.....v_k are the immediate children of the node and merge is some function that is used to merge the subproblems of each of the children.

Again the actual merge function may differ and can depend on the implementation of the problem, but we generally try to formulate such a recurrence relation when we try to solve problems using tree DP.

Let us now look at some standard applications of DP on trees to understand the core idea behind it.