J. Valentina and the Gift Tree

Editorial

In this problem, you are given a tree T consisting of N vertices. Each vertex v has assigned an integer G_v. Your task is to answer Q queries. For a single query (u, v), your have to find the maximum sum of a continuous subpath of the path from u to v. More formally, for a given two vertices u, v, let $P_{u,v} = v_1, v_2, \ldots, v_k$ be the path from u to v. Your task is to find the value of $v_i + v_{i+1} + \ldots + v_j$, which is maximized over all $1 \leq i \leq j \leq k$.

First, let's solve a simple problem. Assume that a given tree is a path. Then we can represent it as an array $A[1, \ldots, N]$, and the problem which we need to solve, is for a given query (i, j), find the maximum sum of a subarray of $A[i, \ldots, j]$.

This problem can be easy solved with a segment tree. Let v be a vertex of a segment tree representing the range $A[l, \ldots, r]$. Then we will store the following values in v:

- sum of elements of the maximal subarray of $A[l, \ldots, r]$
- sum of all elements in $A[l, \ldots, r]$
- sum of elements of the maximal subarray of $A[l, \ldots, r]$ beginning at index l
- sum of elements of the maximal subarray of $A[l, \ldots, r]$ ending at index r

Notice, that we can compute each of the above values for a vertex v, if we know these values for its children. If v does not have any children, then it is a leaf, and all these values are determined. We can easy use this fact to handle both insertion to the segment tree as well as querying it for the maximum sum of a subarray of any range.

Now, let's back to the original problem defined on a tree. The idea here is to process all queries offline, and use the solution for a problem defined on a path.

In more details, any query (u, v) can be divided into queries (u, x) and (v, x), where $x = LCA(u, v)$. Thus from now, we assume that each query corresponds to a path (v, x) of the tree. We will compute answers to all the queries while traversing the tree using `DFS` and maintaining the following invariant: while entering a vertex v, we have a path from the root of the tree to v stored as a segment tree like in the problem defined for a path. Then, if we enter a vertex v, first we insert its value to the segment tree. After that, we can answer each query of the form (x, v) in $O(\log N)$ time using the segment tree - notice that x is on the path from the root to v, so we can do this.

The total complexity of this method is $O((Q + N) \cdot \log N)$, because this is the time needed to decompose all queries into path queries and answer them offline using a segment tree.