Editorial for CCC '22 S5 - Good Influencers

Remember to use this editorial **only** when stuck, and **not to copy-paste code from it**. Please be respectful to the problem author and editorialist.
Submitting an official solution before solving the problem yourself is a bannable offence.

In the first subtask, the students and their friendships form a line.

One approach to this subtask involves dynamic programming. Let $DP[i]$ be the minimum cost required for the first i students to end up intending to write the CCC (ignoring any later students). Note that $DP[0] = 0$, and our answer will be $DP[N]$.

For each value of i from 0 to $N-1$, we'll consider transitions onwards from that state, with the subarray of students from $i+1$ to j (for each possible j such that $i < j \leq N$) ending up intending to write the CCC. If at least one of those students has a P value of `Y`, then it's possible to perform this transition by choosing a `Y` student and "expanding their influence" to all others, updating $DP[j]$ accordingly.

For each i, we can consider all possible values of j while computing their transitions' minimum costs in a total of $\mathcal{O}(N)$ time, resulting in an overall time complexity of $\mathcal{O}(N^2)$.

To solve the remaining subtasks, we'll use a different dynamic programming formulation, this time on the tree structure formed by the students and their friendships. We'll consider the tree to be rooted at an arbitrary node (student), such as node 1.

Let $DP[i][j]$ (defined for $1 \leq i \leq N$ and $0 \leq j \leq 2$) be the minimum cost required for all students in i's subtree to end up intending to write the CCC, such that:

- if $j = 0$, i will be influenced by its parent
- if $j = 1$, i will have no particular interaction with its parent
- if $j = 2$, i will influence its parent

Our answer will then be $DP[1][1]$.

As is typical for DP on trees, we'll recurse through the tree, and for each node i, we'll compute $DP[i][0 \ldots 2]$ based on the DP values of i's children. Each value $DP[i][j]$ must be computed carefully, with different logic depending on the values of $P[i]$ and j.

An implementation of this algorithm may take $\mathcal{O}(N^2)$ time (with some DP transitions taking $\mathcal{O}(N)$ time each to compute), but optimizations can be applied to reduce it to an overall time complexity of $\mathcal{O}(N)$ and have it earn full marks.

Comments

There are no comments at the moment.