Козак Вус та дерево

Автори та розробники: Андрій Романов, Антон Ципко, Костянтин Денисов

Блок 1. Перевіримо, чи сума ваг ребер графа не перевищує k_1. Якщо ні, то відповідь 0, інакше відповідь 1.

Блок 2. Знайдемо для кожної вершину вигоду — суму ваг ребер, які з неї виходять. Тоді нам вигідно або зовсім не фарбувати вершини в чорний колір, або пофарбувати в чорний колір вершину з найбільшою вигодою. Перевіримо, чи підійде нам один з цих двох варіантів. Якщо жоден не підійшов, то відповідь 2.

Блок 3. Напишемо динаміку `dp[x][y]` — найменша можлива сума ваг яскравих ребер, якщо ми розглядаємо вершини від 1 до x і пофарбували y з них в чорний колір. Перехід `dp[x][y] = min(dp[x-1][y-1]+edge[x-1][x], dp[x-2][y-1]+edge[x-2][x-1]+edge[x-1][x], dp[x-1][y])`, де `edge[x-1][x]` — вага ребра між вершинами $x-1$ та x. Тоді `dp[n][1]`, `dp[n][2]`, $\ldots$, `dp[n][n]` — значення мінімальних можливих сум ваг яскравих ребер, якщо пофарбовано i вершин. Застосувавши бінарний пошук на цьому масиві зможемо відповідати на запит за $O(\log n)$. Загальна складність $O(n^2 + m \log n)$.

Блок 4. Через те, що заданий всього один запит і n маленьке, можемо перебрати всі можливі комбінації вибору набору вершин, який ми пофарбуємо в чорний і знайти найкращу з них (тобто ту в якій залучена найменша кількість вершин). Складність $O(n! \cdot n)$.

Блок 5. Напишемо квадратну динаміку `dp[v][cnt][flag]` — мінімальна можлива вага яскравих ребер, якщо розглядати лише вершини піддерева v, cnt з них пофарбували в чорний колір, `flag=1`, якщо v чорна і `flag=0`, якщо v біла. Будемо поступово приєднувати до вершини v піддерева її синів. Коли ми приєднуємо піддерево вершини to, то переберемо всі можливі комбінації кількості вершин, які ми пофарбуємо. В піддереві вершини v ми можемо пофарбувати в чорний колір від I = 0 до I = n вершин, в піддереві вершини to від $J = 0$ до $J = n$ вершин. Переходами будуть:

1. `dp[v][I+J][0]=min(dp[v][I+J][0], dp[v][I][0]+dp[to][J][1])`;
2. `dp[v][I+J][0]=min(dp[v][I+J][0], dp[v][I][0]+dp[to][J][0]+edge[to][v])`;
3. `dp[v][I+J][1]=min(dp[v][I+J][1], dp[v][I][1]+dp[to][j][0])`;
4. `dp[v][I+J][1]=min(dp[v][I+J][1], dp[v][I][1]+dp[to][j][1])`;

Тепер побудуємо масив `ans[cnt]=min(dp[root][cnt][0], dp[root][cnt][1])`, де $root$ — корінь дерева. `ans[cnt]` — мінімальна можлива сума ваг яскравих ребер, якщо в чорний колір пофарбовано cnt вершин. Побудувавши даний масив можемо бінарним пошуком знаходити відповідь на кожен запит — знаходимо мінімальне таке cnt, що $ans[cnt] \leqslant k_i$. Таким чином складність цього розв'язку $O(n^3 + m \log n)$.

Блок 6. Помітимо, що I та J можемо перебирати не до n, а до `cnt[v]` і `cnt[to]` відповідно. Тут `cnt[v]` — поточний розмір піддерева вершини v, `cnt[to]` — розмір піддерева вершини to. На перший погляд, може здатися, що складність розв'язку не змінилася, але насправді тепер ми підраховуємо dp за $n \cdot n$ — перевірте це самостійно. Асимптотика рішення $O(n^2 + m \cdot \log n)$.