Задачи от Саши

Автор и разработчик задачи: Герман Перов

Переформулировка условия

Необходимо найти количество способов разбить вершины на $k+1$ множество, в котором наименьший общий предок i-го (для $1 \leqslant i \leqslant k$) множества равняется x_i (множество под номером $k+1$ содержит вершины, соответствующие жильцам, которых Саша не позовет решать задачи).

Решение для $k = 1$

Заметим, что Саша может звать жильцов только из поддерева вершина x_1.

- если Саша позовет жильца из вершины x_1, то любого жильца из поддерева можно как звать, так и не звать. Таких вариантов $2^{sz(x_1)-1}$
- если Саша не позовет жильца из вершины x_1, то необходимо позвать хотя бы по одному жильцу хотя бы из двух поддеревьев. Чуть удобнее вычислить данное значение как $cnt_{all} - cnt_1 - cnt_0$, где cnt_{all} — общее количество способов позвать жильцов из поддеревьев (то есть $2^{sz(x_1)-1}$), cnt_1 — количество способов позвать хотя бы одного жильца только из одного поддерева (то есть $\sum\limits_{(x_1,u)\in E} (2^{sz(u)} - 1)$) и cnt_0 — количество способов не звать жильцов вообще (то есть 1).

Решение для $p_i = i - 1$

- каждая x_i обязана войти в i-е множество;
- вершина v, в которой не будут решать задачу, может войти в множество j ($j \leqslant k$) тогда и только тогда, когда x_j является предком v. Чтобы найти количество способов выбрать множество для вершины, достаточно посчитать, сколько вершин над ней находятся в массиве x. Также есть опция взять данную вершину в множество $k+1$.

Достаточно посчитать произведение числа способов выбрать множество для v по всем вершинам.

Решение за $O(nk)$

Переформулируем задачу так, что будем не определять вершины в множества, а красить их. Необходимо, чтобы наименьший общий предок вершин цвета i равнялся x_i (также будет фиктивный цвет $k+1$, для которого такое условие не требуется).

Будем считать $dp[v][c]$ — количество способов корректно раскрасить поддерево v, при этом имея c дополнительных цветов (которые в будущем потребуются для покрытия вершин из x не из поддерева v)

- если вершина v не лежит в x, то просто $dp[v][c] = c \cdot \prod\limits_{(v,u)\in E} dp[u][c]$. Это следует из того, что для получения c дополнительных цветов достаточно взять соответствующие раскраски для поддеревьев и выбрать сам цвет для v
- если вершина v лежит в x, то необходимо покрыть данную вершину. Будем обрабатывать детей по одному и поддерживать три значения: $cnt[c][0]$, $cnt[c][1]$ и $cnt[c][2]$, где $cnt[c][i]$ — количество способов получить в результате c свободных цветов, при этом покрасив в i поддеревьях детей хотя бы одну вершину в цвет вершины v (состояния при $i \geqslant 2$ для нас неотличимы, поэтому считаем, что $cnt[c][2]$ хранит суммарное число способов для всех $i \geqslant 2$). Тогда добавление очередного ребенка u выражается в
  - если в u будет использован свободный цвет (покрашен в цвет v) и должно остаться c цветов, то $newcnt[c][min(i+1, 2)]$ надо увеличить на $cnt[c][i] \cdot (dp[u][c+1] - dp[u][c])$

– если в u не будет использован свободный цвет, то $newcnt[c][i]$ надо увеличить на $cnt[c][i] \cdot dp[u][c]$

Тогда $dp[v][c] = (cnt[c][0] + cnt[c][1] + cnt[c][2]) + cnt[c][2]$. Первые три слагаемых соответствуют варианту "v покрасили в свой цвет, поддеревья можно красить произвольно". Второе слагаемое соответствует варианту "v покрасили в свободный цвет, надо раскрасить вершины хотя бы в двух поддеревьях".

Ответ на задачу будет лежать в $dp[root][1]$ (единица соответствует тому, что у нас остался свободный цвет для $k+1$).

Получили динамику, вычисляемую за $O(nk)$.

Решение за $O(n + k^2 + k^{1.5}\sqrt{n})$

Назовем вершины из x плохими, а вершины не из x хорошими.

Предположим, что у хорошей вершины v есть хороший ребенок w.

Получим формулу

$$dp[v][c] = c \cdot \prod_{(v,u)\in E} dp[u][c] = c \cdot \left(\prod_{w \neq u \ (v,u)\in E} dp[u][c] \right) \cdot dp[w][c]$$

$$dp[v][c] = c \cdot \left(\prod_{w \neq u \ (v,u)\in E} dp[u][c] \right) \cdot c \left(\prod_{(w,t)\in E} dp[t][c] \right)$$

Заметим, что можно отдельно сгруппировать c в некоторой степени и произведение состояний динамик при фиксированном c. Это соответствует тому, что если есть ребро (v, w) между двумя хорошими вершинами, то можно удалить нижнюю из вершин и поддеревья w подвесить к v (при этом надо не забыть учесть, что можно красить и саму вершину w; для этого предлагается в вершинах поддерживать вес — скольким вершинам соответствует данная).

После такой обработки дерева не будет двух хороших вершин, связанных ребром. Значит, в итоговом дереве будет $2k$ вершин плюс сколько-то (возможно, n) хороших листьев. При этом заметим, что суммарный вес хороших листьев будет не может быть больше n.

Таким образом, получили решение за $O(n + k^2 + leaves \cdot k)$, где $leaves$ — количество хороших листьев после вышеописанной обработки дерева. Осталось научиться эффективно обрабатывать вершину, к которой подвешено множество хороших листьев какого-то веса.

Посчитаем массив $cnt[c][i]$ на хороших листьях. Для удобства будем считать, что к вершине подвешено l листьев весов $w_1, w_2, \ldots, w_l$. Пока что мы умеем вычислять $cnt[c][i]$ для всех c за $O(k \cdot l)$ и хотим научиться быстрее.

- $cnt[c][0] = c^{\sum w_j}$. Соответствует тому, что просто в каждом поддереве красим вершины в c свободных цветов.
- $cnt[c][2] = (c+1)^{\sum w_j} - cnt[c][0] - cnt[c][1]$. Следует из того, что $cnt[c][0]$, $cnt[c][1]$ и $cnt[c][2]$ покрывают всевозможные варианты покрасить поддерева детей, оставив c свободных цветов. Таких вариантов, в свою очередь, $(c+1)^{\sum w_j}$
- осталось разобраться с подсчетом $cnt[c][1]$. Заметим, что при добавлении очередного w, получается $newcnt[c][1] = cnt[c][0] \cdot ((c+1)^w - c^w) + cnt[c][1] \cdot c^w = newcnt[c][0] \cdot \left(\left(\frac{c+1}{c} \right)^w - 1 \right) + cnt[c][1] \cdot c^w$. Отсюда следует равенство $\frac{newcnt[c][1]}{newcnt[c][0]} = \frac{cnt[c][1]}{cnt[c][0]} + \left(\frac{c+1}{c} \right)^w - 1$

  Значит, можно поддерживать значение $\frac{cnt[c][1]}{cnt[c][0]}$ и обновлять его при очередном весе. Благодаря такому трюку можно добавлять не один вес, а сразу несколько одинаковых весов.

Имея массив $w_1, \ldots, w_l$ посчитаем, сколько раз какой вес встречался и будем скармливать в алгоритм не только веса, но и их количества. Получим подсчет $cnt[c][i]$ для всех c за $O(k \cdot distinct(w_1, \ldots, w_l)) \leqslant O\left(k \cdot \sqrt{\sum w_j}\right)$ для фиксированной вершины.

Мы получили оценку $O\left(k \cdot \sqrt{\sum w_j}\right)$ для обработки хороших листьев одной плохой вершины. Теперь оценим общее время обработки всех листьев.

Пусть W_i — суммарный вес хороших листьев, подвешенных к вершине x_i. Так как всего вершин не больше n, имеем ограничение $\sum_{i=1}^{k} W_i \leqslant n$, а время работы равняется $\sum_{i=1}^{k} k\sqrt{W_i}$. Худшим случаем будет $W_i = \frac{n}{k}$ для всех i, и общее время работы обработки листьев составит $O\left(\sum_{i=1}^{k} k\sqrt{\frac{n}{k}}\right) = O\left(k^{1.5}\sqrt{n}\right)$

Таким образом, общее время работы всего алгоритма составляет $O\left(n + k^2 + k^{1.5}\sqrt{n}\right)$