

Другие

Автор задачи и разработчик: Егор Юлин

В задаче по сути требуется отвечать для вершин на запросы количества их потомков на определенной глубине от них. Будем решать задачу в оффлайне — прочитаем все запросы сразу же, и для каждой вершины дерева сохраним, на каких глубинах мы хотим получить ответ.

Тогда решение будет иметь следующий вид: спускаемся по дереву и в вершине храним **map**, где ключом является высота, а значением — количество детей на такой высоте. Достаточно хранить только информацию про глубины, которые интересны в этой вершине или в каком-то из ее предков.

Пересчитывать эту информацию для родителей через детей может потребовать много ($\Omega(n^2)$) времени, но можно при выходе из детей объединять **map** детей в одну при помощи *small-to-large* оптимизации: всегда будем сохранять наибольшего по количеству информации детей и «докладывать» в него информацию о количестве потомков других детей.

Тогда каждая такая операция обновления информации для каждого элемента **map** случится не более $\log n$ раз. Действительно, если считать «размером» **map**'ы количество детей в поддереве текущей вершины, то после переноса элементов из меньшего множества в большее, они оказываются в множестве, как минимум в два раза большем исходного. Понятно, что такое не может случиться больше $\log_2 n$ раз.

После обхода всех детей мы можем пройти по всем запросам для текущей вершины и записать ответ для них. Это решение будет работать за $O(n \log^2 n + q \log n)$.