Теоретическое решение задачи В (3-ое ДЗ)

В данной задаче нам были даны N (*1 ≤ N ≤ 105*) сундуков и M (*1 ≤ M ≤ 105*) зависимостей между ними. Зависимости подаются в виде i, j, d, что означает то, что в сундуке i на d больше монет чем в сундуке j. Нам необходимо вывести комбинацию монет в сундуках так, чтобы сумма была минимальной, либо сообщить что записи противоречивы (и сказать, где именно находится первое противоречие). Также мы должны принять за факт то, что первый сундук всегда пустой.

Первое что приходит в голову при виде этой задачи – представить все отношения в виде взвешенного графа и обойти его. Так мы сможем узнать, не противоречат ли заданные отношения друг другу. У данного способа есть существенный недостаток – ответ мы узнаем в оффлайне, т е мы сначала построим граф, а потом уже будем с ним оперировать. Есть у этого способа один значительный минус – узнать, где именно произошла ошибка не так уж и просто (хоть и не невозможно). Поэтому, мы рассмотрим другой способ

Фактически, когда мы узнаем отношение между двумя сундуками, мы можем это воспринимать как объединение вершин в одно множество. А точнее – объединения множеств, в которых лежали данные вершины. Почему так? Когда мы узнаем отношение между двумя вершинами, мы фактически создаем ребро между ними, и тем самым, объединяем их в одну компоненту связности. В середине этой компоненты между ВСЕМИ вершинами есть связь (т е от любой вершины мы можем пройти к любой другой). И эти связи не должны противоречить друг другу (как только мы добавляем ребро между вершинами в середине одной компоненты связности, которое противоречит остальным - мы не сможем однозначно восстановить вес в каждой вершине).

Мы пришли к выводу, что нам нужна структура данных, которая можем поддерживать информацию о множествах и оперативно их объединять. Воспользуемся структурой Disjoint Set Union. Будем поддерживать такую информацию – родитель вершины, разница между весом в данной вершине и весом в родителе, значения в родителях (такие чтобы при любой разнице между вершиной и родителем в середине одной компоненты связности не получалось вершин с отрицательным весом; фактически это обратное значение к минимальному весу среди вершин в компоненте связности).

Объединение вершин будет происходить по правилу присоединения к меньшому предку (с меньшим индексом). Из-за этого мы не сможем использовать ранговую эвристику, однако и без этого мы сможем влезть в ограничение по времени. Почему же именно по такому правилу? По условию задачи один из контейнеров ТОЧНО пуст (0вой). И объединяя по такому правилу мы можем гарантировать что 0ая вершина всегда будет родителем множества.

Восстановления ответа при таком подходе довольно легкое – если мы в процессе нашли ошибку, то узнаем об этом сразу и остановим выполнение программы. Если же такого не произошло, то нам для каждой вершины нужно знать вес ее предка и разницу между предком и этой вершиной. Все это мы поддерживаем в disjoint set.

Асимптотика по времени фактически сводится к асимптотике disjoint set (O(logN) с небольшой константой на запрос). Итого O(N logN). По памяти – нам нужно поддерживать несколько массивов длинны N (в disjoint set), что в итоге дает нам асимптотику О(N)

Рябцев Денис