Домашнее задание №10

Томинин Ярослав, 778 группа

5 мая 2018 г.

1.

a)Поймем, что каждое остовное дерево состоит из n-1 ребра(где n- количество вершин), из этого следует, что все остовные деревья увеличатся на (n-1)*w. Именно поэтому все остовные деревья и только они, которые были минимальными,и только они останутся минимальными.

b)Да, это так, для этого достаточно посмотреть на первую итерацию алгоритма Крускала. Мы выбираем это легкое ребро и делаем разрез, согласованный со множеством A(в котором на данный момент 0 ребер). Это разрез из одной вершины из легкого ребра и всех остальных. Далее мы выбираем из пересекающих разрез ребер минимальное. По теореме это ребро безопасно и, если оно не входит в какое-то минимальное остовное дерево, то проведем следующую операцию. Назовем вершины легкого ребра u,v. Найдем путь из u в v в остове, которое не содержит легкое ребро. Тогда существует пересекающее ребро в нашем разрезе(если это не так, то граф распадается на 2 компоненты, а такого быть не может), удаляем это ребро и заменяем на наше. Из условия задачи следует, что мы заменим на меньшее ребро и мы придем к противоречию, потому что то остовное дерево было не минимальным. Из этого следует, что все минимальные остовные деревья проходят через легкое ребро.

в)Уберем ребро e, тогда наше дерево разобьется на 2 компоненты. В силу того, что этот разрез согласован с A, то по теореме минимальное ребро из пересекающих будет безопасным, а мы знаем, что наше ребро безопасно, следовательно, оно минимально.

г)Нет, контрпример: треугольник со сторонами 2,4,5. Самая длинная сторона- кртачайшее расстояние между этими вершинами, но при этом путь из двух других ребер образует минимальное остовное дерево длины 6. А остовное дерево веса 6 не может содержать ребро длины 5.

2.

Попробуем доказать это от противного. Допустим, что это не так, тогда рассмотрим ребро, которое входит в T, но не входит ни в какое минимальное остовное дерево подгафа H(ребро соединяет вершины u,v). Удалим это ребро и наше дерево разобьется на 2 компоненты. Это будет разрез согласованный со множество ребер дерева T, кроме 1 ребра. Рассмотрим путь, в остовном дереве подграфа H через вершины u,v. Мы точно знаем, что в этом пути будет пересекающееся ребро(в разрезе согласованном со множеством ребер дерева T, кроме 1 ребра) и именно поэтому мы можем сказать, что ребро, которое мы нашли, меньше ребра которое мы удалили(ведь иначе остовное дерево подграфа H не было бы минимальным). Теперь просто добавим ребро, которое мы нашли в граф, полученный из T удалением ребра, и получим остовное дерево которое станет меньше. Мы пришли к противоречию, потому что изначально брали уже минимальное остовное дерево. Следовательно, все ребра, входящие в H и T, должны входить и в какое-то остовное дерево H.

3.

Для начала поймем сколько потомков может быть у вершиы ранга k. На лекци мы уже разбирались, поэтому предоставим ответ без доказательства-не меньше, чем 2^k.

Поймем еще один факт: у дерева, корень которого вершина ранга к есть путь длины к-1. Это почти сразу следует из факта: у вершины рангом к всегда есть потомок ранга к-1(ведь иначе у нее ранг не увеличился бы на 1).

Сформулируем и докажем лемму. Лемма: мы можем посторить граф с корнем ранга к за $2^k - 1$ итераций функции union(при этом за меньшее количество итераций построить такое дерево нельзя). Доказательство: сначала нижняя оценка. По индукции.

База. для к=1 нужна хотя бы 1 итерация.

Переход. Если у нас есть вершина ранга к. Мы знаем, что она образовалась при сливании к-1 и к-1. На к-1 по предположению индукции тратится хотя бы $2^{k-1} - 1$. Если мы умножим на 2 и прибавим 1, то получим как раз наше утверждение.

Достижимость сделать просто: надо сливать 0 ранг с 0, потом 1 с 1 и так далее, тогда на к ранг затратится в точности $2^k - 1$ итераций.

Теперь перейдем к самой задаче. Поймем, что операция union весит O(1), поэтому мы должны сделать так, чтобы операция find была сложностью хотя бы log(n).(Иначе оценка не достижима). Мы знаем, что сделать мы это можем минимум за $2^k - 1$ итераций, где к=log(n). Поэтому количество итераций равно n-1. Если же m меньше, чем n, то наша оценка не достижимая(так как сложность find будет меньше log(n)). Если же m будет отличаться на константу от n, то сложность будет максимум O(m+C log(n)), если же m будет равно cn, где c>1, то мы всегда сможем достичь такой ассимтотики, сделав n-1 операцию union, а остальные потратить на find.

4.

Алгоритм: Выкинем из графа все ребра, которые содержат хотя бы 1 вершину из U, ну и сами вершины из U(назовем граф K'). После этого найдем минимальное остовное дерево с помощью алгоритма Крускала, после этого добавим постоим разрезы для каждой вершины из U: в одной половине будет эта вершина, в другой вершины, которое содержит минимальное остовное дерево. Найдем для кадого такого разреза минимальное пересекающее ребро и добавим эти ребра в наш граф(если же в каком-то разрезе не будет пересекающих ребер, то скажем, что такое дерево постоить нельзя).

Корректность: рассмотрим наш граф и минимальное остовное дерево T, которое соответствует условию задачи. Теперь проделаем следующую операцию: выкинем из графа все ребра, которые содержат хотя бы 1 вершину из U, ну и сами вершины из U(назовем граф K'). Поймем, что наше остовное дерево T потеряет листья(не все) и ребра, идущие к ним. Докажем от противного, что полученное дерево T' является минимальным остовным для графа K'.
Если это не так, то найдем с помощью алгоритма Прима минимальное остовное дерево для K'. После этого рассмотрим разрезы с вершинами из K' и с вершиной из U. И добавим наименьшее пересекающее ребро в каждом разрезе. Мы знаем, что наше дерево будет остовным, но при этом оно будет меньше нашего изначального. Противоречие.
Докажем, почему нельзя сделать меньше. В силу предыдушего пункта, дерево, полученное удалением вершин U и ребер, содержащих их, всегда будет минимальным остовным для K'. Далее поймем, что к нашим вершинам из U всегда идет ребро, которое является ппересекающим для разреза, состоящего из вершин K' и одной вершины из U(вершины из U должны быть листом по условию). Поэтому, если наш алгоритм не строит минимальный граф, то это означает, что какое-то из пересекающих ребер выбрано не минимальным, но наш алгоритм выбирает минимальные. Противоречие.
Сложность по времени: Наш алгоритм удаляет ребра за O(E) и проводит алгоритм Прима за O(E log E). И находит минимальное пересекающееся ребро в U разрезах за за O(E). Следовательно, общая сложность O(E log E).