Бабушкин А.

# 1 Обязательные задачи

1. Сожмём всё по целым дорогам, на оставшемся графе найдём MST.

2. За (E - n)M(n). Для каждого ребра не из остова надо проверить, нельзя ли им улучшить остов.

3. Запустим ДФС. Когда мы будем возвращаться из вершины во время обхода, мы вернём кучу, где каждый элемент – какое-то множество в СНМе, для которого текущий ответ одинаков (текущий ответ для вершины u – это если я стою в вершине v, то какой максимум на пути до u от v. u находится ниже v.) Когда мы получили все кучи от детей v, то для всех пар <v, u> мы можем найти ответ – просто ответ для множества, в котором сейчас лежит u. Как обновить ответ? Посмотрим на ребро в v. Пусть у него вес w. Тогда для всех вершин, у которых ответ был меньше w, он стал w, а для остальных не изменился. Смёржим все кучи от детей в одну, затем вытащим какое-то количество множеств оттуда (пока ответ верхнего элемента меньше w) и все эти множества сольём с множеством, в котором лежит v. Для такого множества ответом будет w. Запихнём это множество в кучу и вернём её. Если в качестве кучи выбрать skew heap, то всё отработает как раз за $(m + n) \log n$.
   Или так: отсортим рёбра по возрастанию и будем мёржить по ним вершины. Когда ребро соединило две компоненты, то оно – ответ для всех пар вершин, где одна вершина из первой компоненты, а другая – из второй. Как найти такие быстро? Выберем ту компоненту, где меньшее число неотвеченных запросов, и переберём все запросы в ней, если они нам подходят – ответим на них. Тогда это $(m+n) \log n$, потому что после каждого мёржа число запросов в компоненте хотя бы удвоится. Это работает даже для произвольных графов. А если $\log^2 n$, то с помощью персистентного СНМа и бинпоиска можно для произвольного графа в онлайне отвечат.
   Или так: для массива легко сделать разделяйкой, обобщим на дерево с помощью центроиды. Бонусом научились без поиска lca решать даже для не вертикальных путей, а если позволить $n \log n$ памяти, то даже в онлайне.
   Или так: построим ХЛД, но раз ДО нельзя, то напишем Фарах-Колтона-Бендера на куждом пути. И снова онлайн!
   Чудесная, чудесная задача!

4. Построим MST, а затем скажем, что для вершин a и b ответ это как раз путь в MST (иначе можно обновить каким-то ребром). Как найти путь? Подвесим остов и обойдём его, записав времена входа и выхода. Теперь за O(1) мы умеем проверять, является ли вершина предком другой. Тогда от вершины a будем подниматься вверх до первой вершины, которая предок b, а затем от b до неё же. Получили ответ.

5. Не дольше, чем квадрат, очевидно. Но на некоторых тестах как раз за квадрат –

например, пусть n – степень двойки, и мы сначала смёржим вершины по парам, потом эти пары в четвёрки, и т.д. Матожидание высоты дерева, полученного на k-ом шаге – $E_k = 1/2(E_{k-1}+1)+1/2E_{k-1} = O(k)$, потому что с вероятностью $1/2$ мы выберем увеличим ранг, и с вероятностью $1/2$ – нет. Значит, $\Theta(n^2)$

# 2 Дополнительные задачи

1.

2. Будем держать два СНМа без сжатия путей. В первом компоненты связности, во втором – двусвязности. Текущий ответ – разность количества компонент в первом СНМе и втором. Когда мы добавляем ребро, то:
   a) Если вершины не соединены в первом СНМе, соединеняем.
   b) Если соединены в обоих, то оно бесполезное.
   c) Если в первом да, а во втором нет, то мы должны смёржить во втором их и все на пути между ними. Как? Находим LCA a и b. Если смёржить все веришины на путях от a до lca и от b до lca, то это долго ($\log^2 n$). Но можно подняться от a до первой вершины, которая во втором СНМе лежит в том же множестве, что и lca, и от b тоже. И смёржить всё только на этих путях. Так суммарно мы во втором СНМе сделаем $O(n)$ мёржей. LCA можно найти тупым параллельным подъёмом от двух вершин, высота дерева не больше лога $\rightarrow$ мы посетим не больше $4 \log n$ вершин, прежде чем найдём lca.