Домашнее задание №3
Алгоритмы. 5 курс. Весенний семестр.

Горбунов Егор Алексеевич

2 марта 2016 г.

1 Мои решения

Задание №1 Доказать, что путь из любой вершины в корень пересекает не более чем $\log n$ путей из покрытия «Heavy-Light-Decomposition», где n — количество вершин.

Решение: Во первых, такие пути покрытия не пересекаются, т.е. не может у двух путей быть общее ребро, т.к. это значит, в силу того, что пути идут только по тяжёлым рёбрам, что есть вершина с 2-мя тяжёлыми рёбрами, но это невозможно (легко показывается от противного). Заметим, что любой путь покрытия от вершины v вверх к корню может закончиться либо в этом корне, либо закончиться где-то раньше, пройдя по лёгкому ребру. Т.е. любой путь в дереве от вершины v лежит на нескольких путях покрытия, переходы между которыми происходят по лёгким рёбрам. Это значит, что для оценки числа путей покрытия, которые пересекает путь из v в корень, нужно оценить число лёгких рёбер на этом пути. Рассмотрим путь:

$$root, v_2, v_3, v_4, \ldots, v_{k-1}, v$$

Это путь от v к корню $root$. Посмотрим на соответствующие вершинам пути размеры поддеревьев в этих вершинах $size(v)$:

$$n > size(v_2) > size(v_3) > \ldots > size(v_{k-1}) > size(v)$$

Это убывающая последовательность целых чисел начинающаяся с n. Если какое-то ребро на v_i, v_{i+1} пути было лёгким, то $size(v_{i+1}) < \frac{size(v_i)}{2}$. Сколько раз в такой убывающей последовательности может произойти уменьшение в двое? Ясно, что максимум $\log n$ рах можно поделить n на 2, чтобы остаться в целых числах. Таким образом число лёгких рёбер не более чем $\log n$, а значит и число путей из покрытия, пересекающих путь из v в корень. ■

Задание №2 Дано дерево, у каждой вершины есть вес. Запросы: изменить вес вершины; найти максимум на пути из u в v за $\langle \mathcal{O}(n \log n), \mathcal{O}(\log^2 n) \rangle$.

Решение: Построим Heavy-light decomposition на данном дереве. Получим какие-то непересекающиеся пути p_i. Каждый такой p_i будем рассматривать как массив, индексация в котором идёт от самой верхней вершины на p_i к самой нижней, а элементы этого массива — веса соответствующих вершин. На каждом таком пути-массиве p_i построим дерево отрезков с операцией максимум на подотрезке. Так же поддержим, например, двоичные подъёмы для нахождения LCA за $\mathcal{O}(\log n)$. Так же позаботимся о том, чтобы можно было за $\mathcal{O}(1)$ находить путь декомпозиции, которому принадлежит вершина v и её индекс в массиве, соответствующем этому пути.

Построение Heavy-light декомпозиции занимает $\mathcal{O}(n)$ операций, т.к. мы просто выбираем те вершины, их которых не идёт вних тяжёлых рёбер и поднимаемся от них вверх. Далее, т.к. пути не пересекаются, то мы суммарно строим деревья отрезков над n элементами, что занимает $\mathcal{O}(n \log n)$, аналогично с двоичными подъёмами. Т.е. предподсчёт занимает $\mathcal{O}(n \log n)$.

Чтобы отвечать на запрос обновления веса в вершине мы находим за $\mathcal{O}(1)$ путь-массив, в котором она лежит и делаем запрос на изменение к дереву отрезков на этом массиве — это $\mathcal{O}(\log n)$. Запрос о максимуме на пути из v в u обрабатывается так: находим $x = LCA(v, u)$. Далее разбиваем, в силу декомпозиции, пути от v к x и от u к x на $\mathcal{O}(\log n)$ путей покрытия и на каждом таком пути запрашиваем максимум на нужном подотрезке (т.к. путь мог захватывать только части путей из покрытия), в итоге имеем $\mathcal{O}\left(\log^2 n\right)$. ■

Задание №3 Для разных вариантов Эйлерова обхода придумать, как за $\mathcal{O}(\log n)$ обновить его при переподвешивании дерева за другую вершину.

Решение:

- Эйлеров обход содержащий каждое посещение вершины. Пусть есть некоторый Эйлеров обход подвешенного дерева с корнем в r. Будем переподвешивать его за вершину x. Для этого найдём в Эйлеровом обходе первое и последнее вхождение вершины x:

  $$rLxMxRr$$

  Тут просто заметить, что с новым корнем x часть Эйлерова пути xMx полностью сохраниться. А так же, часть rL олицетворяет «спуск» к вершине x, а Rr «подъём» от этой вершины (ясно, что там могут быть всякие заходы в поддеревья), но тогда в дереве, где корнем будет x, те вершины, которые были затронуты в rL и вRr образуют новое поддерево для которого обходом будет RrL (там, где мы раньше поднимались, теперь будут спуски, поэтому переставились местами части пути). Описывать это не так просто, но если порисовать картинки, всё становится ясно (честно!). Таким образом итоговым Эйлеровым обходом будет:

  $$xMxRrLx$$

  В конце добавили x т.к. это новый корень и на нём всё кончается.

Таким образом, всё, что мы сделали — это поменяли некоторые подотрезки местами. Такую штуку можно реализовать на декартовом дереве по неявному ключу с использованием сплитов и слияний за время $\mathcal{O}(\log n)$. ■

- Эйлеров обход содержащий рёбра. Смотри предыдущий пункт в силу того, что обход содержащий каждое посещение вершины получается их обхода содержащего рёбра схлопыванием соседних рёбер по правому-левому концам. Тут будет даже проще, т.к. не придётся добавлять/удалять никаких вершин, а достаточно просто поменять местами подотрезки. ■

Задание №4 Дан лес неподвешенных деревьев. Запросы online:

- Соединить ребром вершины v и u разных деревьев
- Удалить ребро между вершинами v и u
- Проверить, в одной ли компоненте лежат вершины u и v

Решение: Посчитаем для каждого дерева Эйлеров обход (который позволяет переподвешивания за $\mathcal{O}(\log n)$, начиная с произвольной вершины, которую будем считать начальным корнем. Теперь запрос «соединить ребром вершины v и u» сведём к запросу «подвесить дерево с корнем в v к вершине u другого дерева» путём переподвешивания первого дерева за v за время $\mathcal{O}(\log n)$. Запрос «удалить ребро между вершинами v и u» в силу того, что деревья у нас теперь подвешенные, эквивалентен запросу «отрезать поддерево с корнем в вершине v от её дерева». Ну а запрос о принадлежности к одной компоненте эквивалентен запросу «Проверить, в одном ли дереве лежат вершины u и v». Таким образом мы свели эту задачу к задаче 8 из практики по этой же теме, а значит она так же решается за $\mathcal{O}(\log n)$ на запрос при помощи декартовых деревьев по неявному ключу, построенным по Эйлеровым обходам. Для поддержки переподвешивания нам нужно уметь быстро вырезать подотрезок из массива, что мы умеем. Итого обработка за $\mathcal{O}(\log n)$ на любой запрос. ■