Задача 1. Найти в графе количество треугольников за $O(E^{1.5})$

Решение.

Назовем **тяжелой** вершину, имеющую более $\sqrt{E}$ соседей. Все оставшиеся вершины назовем **легкими**.

Утверждение: В графе тяжелых вершин не более $2\sqrt{E}$.

Докозательство: Пусть в графе более $2\sqrt{E}$ тяжелых вершин. Тогда число ребер в графе больше, чем $\frac{2\cdot\sqrt{E}\cdot\sqrt{E}}{2} = E$, чего не может быть.

Мысленно разобьем все вершины графа на легкие и тяжелые. Заметим, что треугольников, образованных только тяжелыми вершинами, всего $O(E^{1.5})$, как $C^3_{\sqrt{E}}$. Теперь рассмотрим треугольники, которые содержат в себе легкие вершины. В таком треугольнике точно будут два ребра, которые выходят из легкой вершины. Всего таких ребер $O(E)$, при этом для каждого ребра парными могут быть только $O(E^{1.5})$ ребер, в силу степени легкой вершины. Таким образом, число треугольников в графе равно $O(E^{1.5})$. Теперь найдём их.

Утверждение: Из каждой вершины выходит не более $O(\sqrt{E})$ ребер.

Докозательство: Степень легких вершин $O(\sqrt{E})$, а из тяжелых вершин ребра идут только в тяжелые, которых всего $O(\sqrt{E})$.

Теперь для каждой вершины пометим ее соседей, после чего запустим поиск путей длины 2 и будем фиксировать треугольник при нахождении пометки. Для каждого первого ребра пути мы посмотрим на $O(\sqrt{E})$ ребер, поэтому итоговая сложность алгоритма $O(E^{1.5})$.

Задача 2. Ориентировать неор граф так, чтобы он стал сильносвязным за $O(V+E)$ или сказать, что это невозможно

Решение.

Запустим DFS(он работает за $O(V+E)$) и ориентируем рёбра так, как на них впервые посмотрит DFS. Получится, что рёбра дерева DFS будут ориентированы вниз, а обратные вверх. Из корня достижимо всё. Откуда угодно можно попасть в корень, потому что откуда угодно можно подняться выше: если нельзя, то между поддеревом вершины и тем, что над ней, есть только одно ребро, тогда ориентация была исходно невозможна.

Задача 3. Разбить все ребра неорграфа на минимальное число путей за $O(V+E)$

Решение.

Для начала рассмотрим решение задачи о дополнении неорграфа до Эйлерова минимальным числом ребер за $O(V+E)$.

Если граф связен –- любым образом паросочетаем нечётные вершины. Если несвязен, выписываем компоненты связности по циклу и сперва добиваем связности, добавляя рёбра между соседними компонентами. Если какая-то компонента содержит только чётные вершины, в какую-нибудь вершину проведём два ребра. Мы везде используем DFS, поэтому асимптотика $O(V+E)$.

Вернёмся к нашей задаче.

Все компоненты связности обработаем независимо. Дополним связный граф до Эйлерова, найдём Эйлеров цикл, удалим добавленные ранее рёбра, цикл распался на пути. Путей будет столько же, сколькими ребрами дополнили до Эйлерова. Наоборот, имея пути, можно их по циклу соединить и получить Эйлеров. Так что минимизация этих задач эквивалентна. Мы везде используем DFS, поэтому асимптотика $O(V+E)$.

Задача 8. Есть массив из нулей и единиц. В *online* за $O(\log n)$ отвечать на запросы: поменять элемент; найти ближайший слева/справа ноль к позиции i.

Решение.

Будем использовать дерево отрезков, его операции удовлетворяют условию задачи $O(\log n)$. В узлах ДО будем хранить позицию соответственно левого и правого нуля на отрезке (если нулей на отрезке нет, то -1).

При изменении элемента просто обнавляем значение в листе и поднимаемся по его родителям, в которых также обновляем значения.

Теперь запрос на ближайший ноль. При запросе поднимаемся в рдителя и смотрим его второго сына: если мы пришли из правого (левого) сына, то посмторим на значение крайнего правого (левого) нуля на данном отрезке. Если такой нашёлся, то мы нашли искомое положение. Повторяем процесс, пока не дойдём до корня. Если мы не нашли крайний правый (левый) элемент, то он не существует.

/
*
*
*Лопатин Артемий
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
* Лопатин Артемий
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
*
/