

16.12.21.

Романчикова
Татьяна,
Б05-031.

① Найти в графе количество
треугольников за $O(E\sqrt{E})$.

Разобьем вершины на 2 множества:

(1) $\{x \mid \deg(x) > \sqrt{E}\}$ и $\{x \mid \deg(x) \leq \sqrt{E}\}$ (2)

Сначала перебираем вершины x (1) множества
(таких в графе $N = \frac{2E}{\deg(x)} \leq \frac{2E}{\sqrt{E}} = 2\sqrt{E}$),

внутри перебираем ребра (y, z) и проверяем,
есть ли ребра $(x, y), (y, z), (x, z)$.

Найдем какое-то кон-во треугольников,
содержащих хотя бы 1 вершину x ($\deg(x) > \sqrt{E}$).
за $O(E\sqrt{E})$.

Затем в цикле перебираем ребра $(y, z) \in E$.
Если $\deg(y) \leq \sqrt{E}$ и $\deg(z) \leq \sqrt{E}$, то внутри
перебираем x -соседей y/z (таких $\leq \sqrt{E}$) и
смотрим, есть ли ребро $(x, z)/(x, y)$ за
 $O(E\sqrt{E})$.

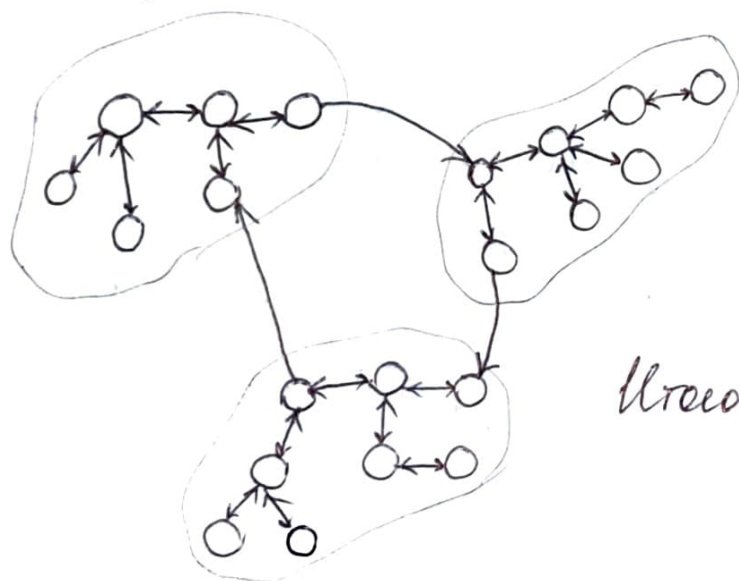
Итоговая асимптотика: $O(E\sqrt{E}) + O(E\sqrt{E}) = O(E\sqrt{E})$.

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

Сначала, если граф не связан в смысле неор. графа, работаем с каждой компонентой отдельно. Запустили на графе dfs и на ребрах, принадлежащих дереву отхода dfs поставили ориентацию в обе стороны. $O(V+E)$.

Теперь запустили алгоритм Косарайто поиска компонент сильной связности. $O(V+E)$.

Соединили однонаправленными ребрами все компоненты по циклу. $O(V) \leq O(V+E)$.



Теперь граф
сильно связан.

Итоговая асимптотика: $O(V+E)$

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

Нам нужно дополнить граф до Эйлера минимальным числом ребер*. После этого путь останется только 1, если мы удалим добавленные ранее ребра, то этот путь распарится в k реберно-непересекающихся путей. И наоборот, если мы обратно добавим эти ребра, то получим единственную путь. Поэтому задача сводится к минимизации кол-ва ребер, добавляемых в граф.

* Если наш граф связен, то паросчитаем нечётные вершины \min числом ребер. (1)
Если несвязен, то добавляем связности за $O(V+E)$ (алгоритм Косарайно), затем переходим к (1).

Итоговая асимптотика: $O(V+E)$

④ Дана некоторая пара вершин в графе найти $W[a, b]$ - такой минимальный вес, что у а в есть путь по ребрам веса $\leq W[a, b]$. $O(V^2)$

Наша задача - минимизировать макс. вес ребра в некотором пути от v до u , $v, u \in V$.

Вспомогательные алгоритмы Прима для построения минимального остовного дерева нашего графа. Все пути в дереве определяются однозначно. Так как на каждой итерации алгоритма Прима мы имеем мн-во вершин T (те, которые мы уже добавили в дерево) и мн-во V (те, которые мы ещё в дерево не добавили), то путь для каждой $t \in T$ до каждой $v \in V$ будет проходить через одно из ребер, соединяющих вершину из T с вершинами из V . По ходу алгоритма мы выбираем из них ребро с мин. весом, поэтому минимизируем вес ребра в некотором пути. $O(V^2)$.

Далее от каждой вершины построенного MST запускаем dfs (т.к. пути в дереве определены однозначно), в процессе которого обновим макс. вес ребра на пути. Пусть $MST = (V', E')$, где $V' = V$, $E' = V - 1$.

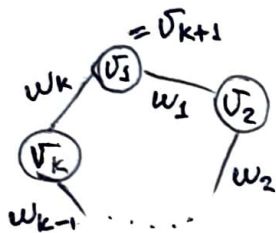
$T(V', E') = O(V' + E') = O(V)$ для каждого dfs.

Итого: $V \cdot O(V) = O(V^2)$

Итого асимптотика: $O(V^2)$.

⑥ Найти в графе циклы минимального среднего веса. $E \leq 2000$, $|w_i| \leq 10^9$.

Бинарным поиском будем искать такое min значение V , что при вычитании V из весов всех ребер существует отрицательный цикл. Этот цикл и будет циклом минимального среднего веса.



$$\frac{\sum_{i=1}^k w_i}{k} < V \Leftrightarrow \sum_{i=1}^k (w_i - V) < 0.$$

Искать циклы отрицательного веса будем алгоритмом Форда-Беллмана ($O(VE)$).

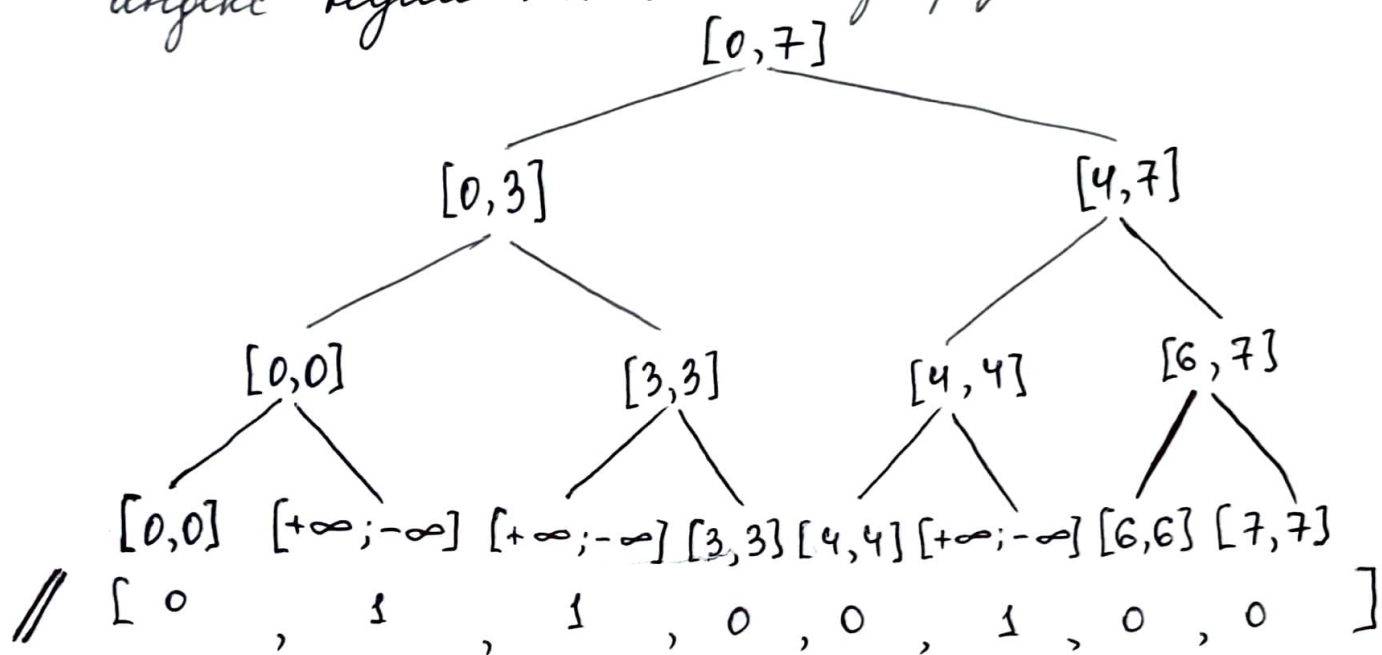
Итоговая асимптотика: $O(VE \cdot \log 10^9)$.

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

Допустим есть массив: $[0, 1, 1, 0, 0, 1, 0, 0]$

Построим по нему следующее ДО:

в каждой вершине будем хранить \min и \max индекс нуле на этом подотрезке, то есть:



- (1) Поменять значение в позиции $i =$

$$= \begin{cases} [+inf; -inf] \rightarrow [i, i], & \text{если 1 заменим на 0.} \\ [i, i] \rightarrow [+inf; -inf], & \text{если 0 заменим на 1.} \end{cases}$$

+ Поднимаемся из элемента к вершине и обновляем все значения $[\min, \max]$ за $O(\log N)$.

- (2) Найти ближайший 0 справа от позиции $i =$
 $=$ найти \min из значений a ($\text{node} = [a, b]$)
 на отрезке $[i+1, N-1]$ за $O(\log N)$.
 \uparrow \min \uparrow \max

- (3) Найти ближайший 0 слева от позиции $i =$
 $=$ найти \max из значений b ($\text{node} = [a, b]$)
 на отрезке $[0, i-1]$ за $O(\log N)$.

- ⑦ Дан массив чисел. За $O(\log N)$ обрабатывать запросы:
- 1) сумма кубов на отрезке
 - 2) $+x$ на отрезке.
 - 3) значение i -го элемента.

Построим по массиву сегментированное ДА:
 в вершинах каждого уровня будут стоять
 моды, в которых храним $[a_i, a_i^2, a_i^3]$.
 Далее за $O(N)$ обновим значения в
 родителях, поднимаясь к корню, суммируя
 все числа, квадраты и кубы.

- (1)* Запрос суммы кубов работает за $O(\log N)$
 (2)* Обновление $+x$ на отрезке будем хранить
 в виде обещаний и пропускать их в коде
 выполнения запросов.

Пусть мы прибавим a на отрезке $[L, R]$.
 В рекурсивном спуске мы на каждой вершине,
 отвечающую за отрезок $[l, r] \subseteq [L, R]$. Тогда,
 перед тем, как передавать обещание детям,
 нужно обновить значения в самой вершине.

Пусть вершина = $[\sum_{i=l}^r a_i, \sum_{i=l}^r a_i^2, \sum_{i=l}^r a_i^3]$, тогда

мы хотим получить новые значения $[\sum_{i=l}^r (a_i+a), \sum_{i=l}^r (a_i+a)^2, \sum_{i=l}^r (a_i+a)^3]$

Пусть $\sum_{i=l}^r a_i = h_1; \sum_{i=l}^r a_i^2 = h_2; \sum_{i=l}^r a_i^3 = h_3;$

$\sum_{i=l}^r (a_i+a) = \text{new } h_1; \sum_{i=l}^r (a_i+a)^2 = \text{new } h_2; \sum_{i=l}^r (a_i+a)^3;$

$$(1) \text{new } h_3 = h_3 + 3a^2 \cdot h_1 + 3ah_2 + (r-l+1) \cdot a^3;$$

$$(2) \text{new } h_2 = 2a \cdot h_1 + (r-l+1) \cdot a^2 + h_2;$$

$$(3) \text{new } h_1 = h_1 + (r-l+1)a;$$

$$\begin{aligned}
 (1) \sum_{i=l}^r (a_i + a)^3 &= \sum_{i=l}^r (a_i^3 + a^3 + 3a^2 a_i + 3a a_i^2) = \\
 &= \sum_{i=l}^r a_i^3 + (r-l+1)a^3 + 3a^2 \sum_{i=l}^r a_i + 3a \sum_{i=l}^r a_i^2 = \\
 &= h_3 + (r-l+1)a^3 + 3a^2 \cdot h_1 + 3a \cdot h_2.
 \end{aligned}$$

$$\begin{aligned}
 (2) \sum_{i=l}^r (a_i + a)^2 &= \sum_{i=l}^r (a_i^2 + 2a a_i + a^2) = \\
 &= \sum_{i=l}^r a_i^2 + 2a \sum_{i=l}^r a_i + (r-l+1)a^2 = \\
 &= h_2 + 2a \cdot h_1 + (r-l+1)a^2.
 \end{aligned}$$

$$(3) \sum_{i=l}^r (a_i + a) = h_1 + (r-l+1)a;$$

Все эти обновления в одной вершине за $O(1)$.
 Суммарно Update (+a) будет работать, как и
 запрос суммы за $O(\log N)$.

(3)* Получить значение i -го элемента мы
 сможем просто по индексу, т.к. в Д.О храним
 на каждом уровне массив размера соответст-
 вующей степени двойки.