МФТИ, ФПМИ

Алгоритмы и структуры данных, осень 2021

(Дополнительный) Семинар №10. Корневая оптимизация

1. Разбейте массив длины n на отрезки длины k (если n не делится на k, то последний отрезок будет короче). Храня сумму на каждом таком отрезке, научитесь отвечать на два типа запросов: а) изменение в точке; б) сумма на отрезке. Подберите оптимальное k, чтобы суммарное время ответа на все запросы было минимальным.

2. Дан массив $a_1, \ldots, a_n$. За $O((n+q)\sqrt{n+q})$ обработайте на q запросов трёх типов: а) вставить число (по индексу i и числу x вставить число x между a_{i-1} и a_i, нумерация сдвигается); б) удалить число (по индексу i удалить a_i, нумерация сдвигается); в) найти сумму на отрезке (по индексам l и r найти $a_l + \ldots + a_r$).

3. Дан массив $a_1, \ldots, a_n$. За $O((n+q)\sqrt{n+q}\log(n+q))$ обработайте на q запросов трёх типов: а) вставить число (по индексу i и числу x вставить число x между a_{i-1} и a_i, нумерация сдвигается); б) удалить число (по индексу i удалить a_i, нумерация сдвигается); в) найти на отрезке количество чисел, меньших данного (по индексам l и r, а также числу y найти количество чисел среди $a_l, \ldots, a_r$, которые меньше y).

4. Дан пустой граф на n вершинах. Обработайте q запросов трёх типов: а) добавить ребро между вершинами u и v; б) удалить ребро между u и v; в) проверить, существует ли в графе путь из u в v. Считайте, что все запросы известны заранее.

5. Дан статический массив $a_1, \ldots, a_n$. Пусть $f(l, r)$ — такая функция от $a_l, \ldots, a_r$, что $f(l, r\pm1)$ и $f(l\pm1, r)$ легко пересчитываются через $f(l, r)$. Ответьте на q запросов вычисления f на отрезке (считайте, что все запросы известны заранее), если

а) f — количество различных чисел на отрезке;

б) f — сумма различных чисел на отрезке;

в) $f(l, r)$ — количество пар (i, j), таких что $l \leqslant i \leqslant j \leqslant r$, и при этом $a_i \oplus \ldots \oplus a_j = k$, где k — общая константа;

г) $f(l, r)$ — сумма чисел, лежащих в отрезке $[l, r]$, которые при этом не превосходят k, где k — общая константа. Что делать, если k меняется от запроса к запросу?

1. Изменение в точке можно обработать за $O(1)$, а сумму можно найти за $O(k + n/k)$. Последнее минимально при $k = \sqrt{n}$.
2. Разбейте весь массив на блоки, длина каждого из которых лежит в отрезке от k до $2k$, где $k \approx \sqrt{n+q}$. Если блок переполняется, его надо разбить на два. Если блок становится слишком маленьким, его надо объединить с одним из соседей (и при необходимости затем разбить на два примерно равных).
3. Вновь разбейте массив на блоки, только теперь каждый блок представляйте в виде отсортированного массива. Тогда достаточно будет делать бинарный поиск в каждом блоке.
4. Разобьём **запросы** на блоки длины k. Считаем блок целиком. Запомним рёбра, которые будут присутствовать в течение всего этого блока (то есть те, которые уже есть в графе и которые при этом не удалятся в этом блок). Сожмём компоненты связности на этих рёбрах. Затем на каждый вопрос можно отвечать простым `dfs` по сжатым компонентам.
5. Воспользуйтесь алгоритмом Мо. Пусть известен ответ (хранится какая-то структура) для отрезка $[l, r]$. При движении одного из концов отрезка появляется или исчезает одно число, соответственно нужно изменить ответ (структуру). Отсортируйте все отрезки по $(\lfloor l/k \rfloor, r)$, где $k \approx \sqrt{n}$. Тогда внутри блока запросов с равным значением $\lfloor l/k \rfloor$ значение правой границы будет только возрастать, а потому пройдёт $O(n)$ положений.