МФТИ, ФПМИ

Алгоритмы и структуры данных, осень 2021

Семинар №13. Splay-дерево, B-дерево

1. Поймите, как обрабатывается последовательность запросов `insert` 1, `insert` 2, ..., `insert` n в splay-дереве.

2. Покажите, что splay-дерево эффективно (линейно) работает, если использовать его как стек, то есть в качестве ключа при добавлении использовать только текущий размер дерева n, а удалять только ключ $n - 1$.

3. Приведите последовательность из n операций `insert`, после обработки которых splay-дерево вырождается в бамбук. Предположим, что процедура `erase` не вызывает `splay`. Приведите пример последовательности операций `erase`, которая в таком случае обрабатывается за $\Omega(n^2)$. Поймите, что происходит при корректной обработке этих запросов.

4. Как сделать `merge` и `split` в splay-дереве? Обратите внимание, что время обработки таких операций не должно превосходить время работы соответствующих операций splay.

5. (Неявное дерево поиска) Задан массив $a_1, \ldots, a_m$. К нему поступают запросы вида: а) вставить x в позицию pos (то есть между двумя уже существующими элементами, нумерация смещается); б) удалить число на позиции pos (нумерация смещается); в) сообщить сумму на подотрезке массива. Обработайте каждый запрос за $O(\log n)$, где n – текущий размер массива.

6. Задан массив $a_1, \ldots, a_m$. К нему поступают запросы вида: а) вставить x в позицию pos (то есть между двумя уже существующими элементами); б) удалить число на позиции pos (нумерация смещается); в) сообщить сумму на подотрезке массива; г) прибавить ко всем числам подотрезка одно и то же число; д) развернуть подотрезок. Обработайте каждый запрос за $O(\log n)$, где n — текущий размер массива.

7. (Своппер) Дан массив $a_1, \ldots, a_n$. К нему поступает q запросов одного из двух видов: а) найти сумму на отрезке; б) по данным l и r поменять местами числа a_l, a_{l+1}, поменять местами a_{l+2}, a_{l+3}, и так далее, вплоть до пары a_{r-1}, a_r (считайте, что $r - l + 1$ чётно). Ответьте на все запросы за $O((n + q) \log n)$.

1. После добавления числа i будет вызвана операция `splay`, сводящаяся к одному вызову `zig`. В итоге получится бамбук (дерево без ветвлений) влево: в корне находится i, в его левом сыне — $(i-1)$, в его левом сыне — $(i-2)$, и так далее. Каждая операция при этом выполняется за $O(1)$.

2. Удаляться в таком случае будет только корень, на его место подвешивается левый сын.

3. В первой задаче можно вызывать `erase` в порядке возрастания от всех чисел $1, 2, \ldots, n$. Понять, что происходит при корректных запусках `splay`, довольно сложно, поэтому просто нарисуйте историю дерева для $n = 10$.

4. Для реализации `merge` достаточно поднять в корень максимальный элемент левого дерева. Для реализации `split` поднимем в корень максимальное число, не превосходящее ключа-разделителя x, тогда в качестве ответа нужно вернуть корень с левым поддеревом и правое поддерево.

5. Используйте неявное дерево с возможностью делать `split` и `merge`.

6. Примените идею отложенных операций: в вершине дерева поиска можно сохранять информацию, которую когда-нибудь в будущем нужно передать детям. Эта информация проталкивается при проходе через вершину.

7. Разбейте индексы на чётные и нечётные, на каждом постройте своё неявно дерево поиска. Нужно будет вырезать какие-то куски из двух деревьев и менять их местами.