Задача 2 (на 30.11.2015)

Воспользуемся декартовым деревом по неявному ключу. В вершине дерева будем хранить ключ, размер поддерева $size$, сумму k-ых степеней в поддереве $0 \leqslant k \leqslant 4$, переменную $toadd$, содержащую число, которое надо прибавить ко всем вершинам в поддереве, флаг $isAssigned$ и переменную $toAssign$, (если флаг истинен то это означает, что всем вершинам в поддереве надо присвоить значение $assign$).

Чтобы реализовать операции $split$ и $merge$ нам надо понять, как проталкиваются значения переменных $toadd$ и $assign$, и как пересчитываются значения остальных переменных. В начале каждой операции с вершиной будем делать следующее:

1. Если флаг $isAssigend$ истинен, то обнуляем его, обнуляем переменную $toadd$ в двух потомках (если они есть, конечно), присваиваем 1 переменной $isAsigned$ в потомках и присваиваем нашему ключу и переменным $assign$ в потомках значение нашей переменной $assign$. Значения k-ых степеней рассчитываем зная размер поддерева и значение переменной $assign$.

2. Если $toadd$ не равно нулю, то прибавляем к переменным $toadd$ в потомках значение нашей переменной $toadd$, к сумме первых степеней поддерева прибавляем

$size \cdot toadd$. Пусть sum и sum_2 - старые суммы первых и вторых степеней поддерева. Тогда $sum_2 = x^2 + y^2 + \ldots$. Если ко всем ключам надо прибавить $toadd$, то новая сумма

$$sum_2' = (x + toadd)^2 + (y + toadd)^2 + \ldots =$$

$$= (x^2 + y^2 + \ldots) + 2 \cdot toadd \cdot (x + y + \ldots) + size \cdot toadd^2 =$$

$$= sum_2 + 2 \cdot toadd \cdot sum + size \cdot toadd^2$$

Таким образом значения суммы k-ых степеней в поддереве вычисляем зная размер поддерева и старую сумму в поддереве всех степеней $\leqslant k$.

Мы научились делать $split$ и $merge$. Выразим через них другие операции.

* Сумма k-ых степеней на отрезке: два split'а, получаем значение нужной суммы из среднего получившегося дерева, два merge'а.

* Вставка элемента: split, создаём новое дерево из одной вершины, два merge'а.

* Присвоение элемента на отрезке: два split'а, присваиваем нужное значение переменной $assign$ в среднем получившемся дереве, два merge'а.

* Прибавление на отрезке: два split'а, присваиваем нужное значение переменной $toadd$ в среднем получившемся дереве, два merge'а.

* Циклический сдвиг на отрезке: два split'а, получили дерево X — отрезок, который нам надо циклически сдвинуть. Рассмотрим случай циклического сдвига влево на k. Делаем такой merge, чтобы получить два дерева L размера k и R, далее делаем merge деревьев R и L (то есть в обратном порядке), получили новое

дерево X — циклически сдвинутый отрезок, далее два merge'а.

Это были пункты а) и б)

Теперь рассмотрим пункт в)

Разобьём наш массив $array[n]$ на пять массивов $parts[5][k]$: $\forall i \in [0 \dots n)$ $array[i] = parts[i\ mod\ 5][i\ div\ 5]$. По каждому из этих пяти массивов построим декартово дерево как в предыдущем пункте. Рассмотрим некоторые операции:

* Циклический сдвиг на отрезках длины 5: в каждом из пяти массивов делаем два split'а, получили массив из пяти деревьев: $Trees[5]$, i-ое дерево содержит i-ые элементы каждого отрезка из пяти элементов, соответственно чтобы сделать циклический сдвиг каждого отрезка длины пять надо циклически сдвинуть массив $Trees[5]$. Далее для каждого дерева из массива $Trees[5]$ (уже циклически сдвинутого) делаем два split'а с соответствующими деревьями (которые получились в результате двух merge'ей).

* Вставка элемента x: рассмотрим участок массива $array$, в который надо вставить элемент:

$$\dots a_1 b_1 c_1 d_1 e_1\ a_2 b_2 c_2 d_2 e_2\ a_3 b_3 c_3 d_3 e_3 \dots$$

a_1, a_2, a_3 — элементы массива $parts[0][k]$
b_1, b_2, b_3 — элементы массива $parts[1][k]$
...
Пусть нам надо вставить элемент между b_2 и c_2. В каждом из пяти массивов $parts[5]$ делаем split, так чтобы все элементы, которые после вставки x в массив $array[n]$ окажутся справа от x оказались в правом дереве split'а. Вот что получится в нашем случае ($LTrees[i]$

и $RTrees[i]$ — результат split'а дерева построенного на массиве $parts[i]$)

$$LTrees[0] = \dots a_1, a_2 \quad RTrees[0] = a_3 \dots$$

$$LTrees[1] = \dots b_1, b_2 \quad RTrees[1] = b_3 \dots$$

$$LTrees[2] = \dots c_1 \quad RTrees[2] = c_2, c_3 \dots$$

$$LTrees[3] = \dots d_1 \quad RTrees[3] = d_2, d_3 \dots$$

$$LTrees[4] = \dots e_1 \quad RTrees[4] = e_2, e_3 \dots$$

После вставки массив $array$ будет выглядеть следующим образом:

$$\dots a_1 b_1 c_1 d_1 e_1 \ a_2 b_2 x c_2 d_2 \ e_2 a_3 b_3 c_3 d_3 \ e_3 \dots$$

Поэтому сделаем split дерева из одного элемента x с соответствующим деревом из $RTrees[5]$, в нашем случае это будет $RTrees[1]$:

$$RTrees[1]' = x, b_3 \dots$$

Пусть после предыдущего шага элемент x находится в дереве $RTrees[i]$. Тогда теперь нам надо сделать merge следующих деревьев:

$$LTrees[i+1] \ c \ RTrees[i]$$

$$LTrees[i+2] \ c \ RTrees[i+1]$$

$$\dots$$

$$LTrees[0] \ c \ RTrees[4]$$

Вот что получится в нашем случае ($Trees[i]$ — результат merge'а дерева $LTrees[i]$ и $RTrees[(5+i-1) \, mod \, 5]$):

$$Trees[0] = \dots a_1, a_2, e_2, e_3 \dots$$

$$Trees[1] = \ldots b_1, b_2, a_3 \ldots$$
$$Trees[2] = \ldots c_1, x, b_3 \ldots$$
$$Trees[3] = \ldots d_1, c_2, c_3 \ldots$$
$$Trees[4] = \ldots e_1, d_2, d_3 \ldots$$

Заметим, что получившиеся пять деревьев $Trees[5]$ — именно такие, как если бы мы их строили по новому массиву $array$ (с элементом x).