毕绿传说 ~Extra~ 题解V2

我们分几个阶段来解决这一问题。这也是我刚想到这个问题时的思考过程。以下令N代表点的个数（=M），V代表值域范围（0到100，故V=101），Q代表操作次数。

1. 修改只有单点的场合

暴力修改每个点，然后线段树处理即可。和普通的线段树题目并没有什么区别。因为我们知道一个点的具体的值是多少，所以可以得出其变化后的值是多少，并据此维护线段树一段的和。

这一思路不能被推广到有段改的场合，根本原因在于无法得知某一个线段内部的点的值。实际上当我们进行一次加k的时候，我们相当于把所有大小在的数都变成，然后再（用某种方法）全体加k；减法也一样。如果不知道每个线段内值在任意的数的个数，这一操作将不可能实现，因为即使打上了lazy标记，也无法维护这一线段内所有数的和。

2. 修改和查询都只有固定一段的场合

然而如果要改的段永远是固定的一段，就不存在什么问题。可以这样处理：预处理出段内所有数字的值，用一颗线段树保存值为0,1,...,100的数各自有多少个，并维护它们的和；定义一个add的标记表示整个段被整体加减的值，初始为0；加k的时候，先令add += k并卡到100，再取得线段树区间内的数字个数并清零整个区间（用lazy标记），然后加到的点上去；减法也类同。求和的时候，用线段树的和，加上区间长度len乘以add的积。

我们希望原题目可以被分成若干个固定长度的段来推广这一算法。注意到树状数组和线段树本身的每一个线段是固定线段；与此相似的，伸展树和非旋转树堆在任何一个特定的时间点上每个节点代表一个固定线段。此外，块状数组或者静态块状链表的每一块是一个固定线段。

因此一个朴素的想法是，树套树。主体依然是一颗线段树，在线段树的每个节点上用数组或者线段树一类的暴力维护区间内值为0,1,...,100的数各自有多少个。这样每次上传维护的时候复杂度都是，总复杂度，时间似乎可以接受，但是遗憾的是这一算法的空间复杂度具有相同的数量级（），对只有64MB的网教来说这太勉强了。

剩下的就只有块状数组，把N个位置分成个长度为的块，每一块内部都是一个固定段。段查段改的时候，中间被完整包含的段用值域线段树整体查改的策略，这样的段有个；两边没有完整包含的段用暴力解决，涉及到的元素个数有个。所以总的时间复杂度是，空间复杂度是，听起来似乎不错，然而这里还有一个问题。

3. 段查段改的场合

如果采用了块状数组的策略，那么对某个特定的块，如果先整体操作再独立操作，块本身的lazy标记（就是add），有强制求值的必要。然而单凭add是无法做到这一点的。见下例：

|  |  |
| --- | --- |
| 初始值 | 2 1 3 add=0 和是6 |
| + 98 | 1个1 2个2 add=98 和是299 |
| - 100 | 3个2 add=-2 和是0 |
| + 10 | 3个2 add=12 和是36 |
| force | 14 13 15 add=0 和是42 |
| 真实的情况 | 12 12 12 add=0 和是36 |

不可能在线段树里直接记录哪些数被修改了，但是注意到每次整体操作导致的修改一定是以下两种：1.把大于100-k的数全部变成100-k 2.把小于-k的数全部变成-k。因此可以额外定义一个上界HiBound和下界LoBound，整体操作的时候维护这两个值，强制求值的时候先求值这两个标记再处理add。考虑到会有的场合（比如说先+100再连续-100），这种情况下相当于把整个线段变成一个数，变成的数是多少依赖于最后一次修改标记的操作，因此增设标记Cover处理这种情况。

最后，当一个块被强制求值并进行内部处理之后，还需要重建线段树以便于下一次的整体操作。

于是这道题就做完了。总体思路是：先分块，每个块有内部各个点上数的值，标记add,LoBound,HiBound,Cover和一个值域建立的线段树，维护整体和；块内线段树支持插入若干个指定值的数、查询值在区间内数的个数、清除所有值为区间内的数共3种修改操作，并维护所有数的和。总复杂度，。

*好吧刚出完这题多校就被打脸。这道题是有更好的做法的。思路可以参照2015年多校第二场的1007（HDU5306）。*

*但是照抄是不可能的，特别是，像原题那样的O(NlogN)算法是不存在的（请自行思考为什么）。这里依然要用值域限制在[0,100]的条件，把暴力复杂度卡到O(100N+NlogN)*

*懒得写原理了，自己参看标程2代码。我是SB不要理我……*