线段树

上海交通大学 方泓杰

线段树基础1

有一个n个数的序列$\{a_n\}$，要求实现三种操作：

1. 将a_x改为y；

2. 将a_x加上y；

3. 查询$\sum_{i=l}^{r} a_i$的值。

$1 \leq n, q \leq 10^5$

线段树基础2

有一个n个数的序列$\{a_n\}$，要求实现两种操作：

1. 将$a_x(l \le x \le r)$加上y；

2. 查询$\sum_{i=l}^{r} a_i$的值。

$1 \le n, q \le 10^5$

打标记 / 标记永久化

线段树基础3

有一个n个数的序列$\{a_n\}$，要求实现三种操作：

1. 将$a_x(l \le x \le r)$改为y；
2. 将$a_x(l \le x \le r)$加上y；
3. 查询$\sum_{i=l}^{r} a_i$的值。

$1 \le n, q \le 10^5$

线段树基础4

有一个n个数的序列$\{a_n\}$，要求实现四种操作：

1. 将$a_x(l \le x \le r)$改为y；
2. 将$a_x(l \le x \le r)$加上y；
3. 将$a_x(l \le x \le r)$乘上y；
4. 查询$\sum_{i=l}^{r} a_i$的值。

$1 \le n, q \le 10^5$

标记的顺序与下传

线段树基础5

有一个n个数的序列$\{a_n\}$，要求实现两种操作：

1. 令$a_i = c$（$l \le i \le r$）

2. 查询$\sum_{i=l}^{r} a_i$的值。

$1 \le n, q \le 10^5$

标记的顺序与下传

Copying Data

对于数组a和数组b，长度均为n，有m个操作，操作有两种：

1. 将a中从x开始的连续k个元素覆盖掉b中从y开始的连续k个元素。

2. 求b_x的值；

$1 \leq n, m \leq 10^5$

CodeForces 292E Copying Data

对b建立线段树，线段树每个节点存$[l,r]$。

表示这个节点所代表的b的区间被a中的$[l,r]$覆盖了。

相当于打标记操作。

时间复杂度$O(m\log n)$

Ball

有n个人，每个人有三个属性(a_i, b_i, c_i)。

如果一个人的三个属性均小于另一个人的，那么他将告辞。

问最后有多少人告辞。

$1 \leq n \leq 5 \times 10^5$

CodeForces 12D Ball

三维偏序？

树套树？

分治？

其实，只要线段树就够了！

首先将第一维离散化，第二维从大到小排序，以第一维为坐标在线段树中插入第三维的值。

由于逐个插入，第二维一定递减；

对于每个新插入的数只要保证他插入位置后面的区间最大值比这个数小就不会告辞。

时间复杂度$O(n \log n)$

XOR on segment

给出n个数$a_1, a_2, \ldots, a_n$，m个操作，操作有两种：

1. 将$a_l, a_{l+1}, \ldots, a_r$同时异或$c$；

2. 求$a_l + a_{l+1} + \cdots + a_r$。

$1 \le n \le 10^5, 1 \le m \le 5 \times 10^4, 1 \le a_i \le 10^6$

CodeForces 242E XOR on segment

区间异或不太好直接维护，于是我们考虑将所有数二进制分解。

我们把每一位单独提出来做，那么对于每一位，异或要么不变，要么取反！

对于二进制下的每一位我们建一棵线段树，那么只要维护：

1. 区间取反（0变1，1变0）
2. 区间求和

即可。这个利用线段树+标记能够很轻松的实现。

所以总复杂度为$O(nk\log n)$，其中k为二进制的位数，这里$k \leq 20$。

Queries

给一个长度为n的序列，有m次操作，要求支持两种操作：

1. 单点修改为x；

2. 求区间$[l,r]$的所有子区间$[l',r'] \subset [l,r]$的异或和，即求

$$\sum a_{l'} \text{ xor } a_{l'+1} \text{ xor } \ldots \text{xor } a_{r'}$$

$1 \leq n,m \leq 10^5, 0 \leq a_i, x \leq 1000$

CodeForces Gym 100739A Queries

将a_i拆成10个二进制位，对于每一位分别考虑。

于是每一位开一棵线段树，单点修改直接把x分解后修改即可。

对于每一棵线段树的每一个节点，我们维护：

d表示这段区间的异或和；

l_0, l_1表示子区间从左端点开始，异或和为0/1的子区间有多少个；

r_0, r_1表示子区间从右端点开始，异或和为0/1的子区间有多少个；

s_0, s_1表示异或和为0/1的子区间一共有多少个。

如何合并子树？

CodeForces Gym 100739A Queries

假设x的左儿子ls，右儿子rs，那么：

$s_{x,0} = s_{ls,0} + s_{rs,0} + r_{ls,0}l_{rs,0} + r_{ls,1}l_{rs,1}$

$s_{x,1} = s_{ls,1} + s_{rs,1} + r_{ls,1}l_{rs,0} + r_{ls,0}l_{rs,1}$

$l_{x,0} = l_{ls,0} + l_{rs,d_{ls}}, l_{x,1} = l_{ls,1} + l_{rs,d_{ls}} \text{ xor } 1$

r的合并与l类似；d的合并显然。

时间复杂度$O(m \log n)$。

Max Sum Plus Plus

给出一列数$\{a_n\}$，从中选出m个互不相交的子段，使得和最大，求这个最大值。

$1 \leq m \leq n \leq 10^6$

HDU 1024 Max Sum Plus Plus

如果正数个数小于m，则选取前m大即可。

考虑这样一种做法：

用线段树维护这个序列，每一次选取整个序列最大子段，将这个和累加到答案中。

然后把这段整体乘(-1)，下次如果选中这段，就相当于把原来选的去掉。

然后继续操作m次，若遇到负数则提前终止，最后即得答案。

如果正数个数大于等于m，那么最终选取的段的数的个数一定大于m，那么就一定可以强行拆成m段。

线段树需要维护：**区间最大值、最小值，区间前缀、后缀最大值、最小值。**

维护最小值是因为要取相反数。时间复杂度$O(m \log n)$。

k-maximum subsequences sum

给一列数$\{a_n\}$，有m个操作，要求支持两种操作：

1. 修改某个数的值；

2. 读入l, r, k，在$[l, r]$中选取至多k个不相交的子段，使这k个子段和最大，求最大值。

$1 \leq n \leq 10^5, 1 \leq m \leq 10^5, |a_i| \leq 500, 1 \leq k \leq 20$

BZOJ 3638 k-maximum subsequences sum

这个$1 \leq k \leq 20$，~~提醒我们可以暴力做~~。

和前一题一样。由于是至多k个子段，选不完也没事，就不用特判了。

时间复杂度$O(mk \log n)$。

Danil and a Part-time Job

给一棵n个节点的树，每个节点有一个灯，有亮的灯也有灭的灯。

有m个操作，操作有两种：

1 将x及其子树的灯变成相反状态（亮变暗，暗变亮）；

2 统计x及其子树有多少个灯是亮的。

$1 \le n \le 2 \times 10^5, 1 \le q \le 2 \times 10^5$

CodeForces 877E Danil and a Part-time Job

利用DFS序建立一棵线段树，那么子树在DFS序上是连续的区间

从而子树修改和子树查询转化成了序列的区间修改和区间查询！

线段树打标记即可。

时间复杂度$O(m \log n)$

攻略

给出一棵n个节点的树，根节点为1号节点，每个点有一个价值w_i。

要求选出k条从1号节点到叶子的路径，使得这些路径并的价值和最大。

$1 \leq n, k \leq 200000, 1 \leq w_i \leq 10^9$

BZOJ 3252 攻略

首先每次贪心取这个策略显然是正确的。那么我们可以处理出dfs序。

按照dfs序建立线段树，每个节点存储该节点到根的剩下价值（没被取过的）。

然后选取k次，每一次把最大值取出来后就把路径上没取过的点给取了（子树减）。

如果x被取过了，那么x的所有祖先一定被取了，这样保证每个点只被取1次。

时间复杂度$O\big((n+k)\log n\big)$。

预备知识：矩形面积并

一个平面上有n个矩形，给出每个矩形所在位置，求所有矩形面积并。

$1 \leq n \leq 10^5$

预备知识：矩形面积并

对于所有矩形的端点按照x坐标排序，然后依次扫描矩形，每次加入一条线段覆盖。

线段树查询所有区间中有线段覆盖的区间的长度即可。

时间复杂度$O(n \log n)$。

无题

给出一棵树，求有多少点对(u, v)满足其路径上不存在两个点a, b使得$a = kb$或$b = ka$。

其中k为正整数。

$1 \leq n \leq 10^5$

无题

首先，把所有形如(a, ka)的路径提出来，以下称为限制。

考虑答案路径要不合法，必定覆盖了至少一条完整的限制。

考虑覆盖限制(a, b)的路径(u, v)，设dfn_x表示x的DFS序，end_x表示x子树中最大的DFS序。

不妨设$dfn_u < dfn_v, dfn_a < dfn_b$（不满足交换即可）

1. 当u是v祖先的时候，考虑g是离u最近的路径(u, v)上的点
   - 要么$dfn_a < dfn_g, dfn_v \le dfn_b \le end_v$（一个在上方，一个在子树内）
   - 要么$dfn_v \le dfn_a \le end_v, dfn_b > end_g$（一个在子树内，一个在其他儿子）
2. 当u不是v祖先的时候，显然$dfn_u \le dfn_a \le end_u, dfn_v \le dfn_b \le end_v$

无题

1. 当u是v祖先的时候，考虑g是离u最近的路径(u,v)上的点
   - 要么$dfn_a < dfn_g, dfn_v \le dfn_b \le end_v$（一个在上方，一个在子树内）
   - 要么$dfn_v \le dfn_a \le end_v, dfn_b > end_g$（一个在子树内，一个在其他儿子）

2. 当u不是v祖先的时候，显然$dfn_u \le dfn_a \le end_u, dfn_v \le dfn_b \le end_v$

情况1构成了2个矩形，情况2构成了1个矩形。

由于(a, ka)的点对不会很多，根据数学知识可以知道只有$O(n \log n)$对。

于是对$O(n \log n)$个矩形做面积并即可。

扫描线+线段树。时间复杂度$O(n \log^2 n)$。

T2

有一棵n个点的树，q次操作，每个点有点权，1号点为根。

操作如下：

1. 修改点权；
2. 修改子树内所有点权；
3. 询问点到根路径点权和。

$1 \leq n, q \leq 10^5$

如果点权改成边权？

BZOJ 4034 HAOI 2015 T2

利用树的in/out DFS序可以很方便的解决这题。

维护一个进栈/出栈的DFS序，进栈时系数为1，出栈时系数为-1。

那么三个操作分别看成：

1. 单点修改（乘以系数）；
2. 区间修改（乘以系数）；
3. 区间查询。

用线段树维护即可，时间复杂度$O(n \log n)$。

BZOJ 4034 HAOI 2015 T2

利用树的in/out DFS序可以很方便的解决这题。

维护一个进栈/出栈的DFS序，进栈时系数为1，出栈时系数为-1。

那么三个操作分别看成：

1. 单点修改（乘以系数）；
2. 区间修改（乘以系数）；
3. 区间查询。

用线段树维护即可，时间复杂度$O(n \log n)$。

点权改为边权只需要再添加$n-1$个点即可转化成点权！

T2 改编版

有一棵n个点的树，q次操作，每个点有点权，1号点为根。

操作如下：

1. 修改点权；
2. 修改子树内所有点权；
3. 询问点到根路径点权和；
4. 询问子树内点权和。

$1 \leq n, q \leq 10^5$

T2 改编版

同样的，利用DFS序，4操作我们看成区间查询（其中只询问系数为正的数的和）。

利用线段树解决，时间复杂度$O(n \log n)$。

T2 改编版

同样的，利用DFS序，4操作我们看成区间查询（其中只询问系数为正的数的和）。

利用线段树解决，时间复杂度$O(n \log n)$。

利用DFS序的转化可以将许多树上的问题转化为序列的问题，从而可以使用序列的数据结构进行转化。

另一种将树上问题转化为序列问题的方法是树链剖分，但是无论实现难度还是时间复杂度都没有DFS序+线段树优秀。

DFS+线段树也不是万能的，只能解决一些特定的书上路径问题，遇到题目需要仔细分析。

花神游历各国

有一个n个数的序列$\{a_n\}$，要求实现两种操作：

1. 将$a_x(l \le x \le r)$改为$\left[\sqrt{a_x}\right]$；

2. 查询$\sum_{i=l}^{r} a_i$的值。

$1 \le n, q \le 10^5, 1 \le a_i \le 10^9$

BZOJ 3211 花神游历各国

考虑到一个数开根号后取整若干次就会变成1。

那么每次暴力开根号，同时在线段树上记录一个s_x表示x代表的区间是否都是1。

如果都是1就不用继续开根号了。

设最多开根号p次，那么每个点最多被访问p次，那么显然复杂度是可以保证的。

还有另外一种做法。

BZOJ 3211 花神游历各国

考虑记录区间最大值和最小值。

如果一个区间最大值开根和最小值开根相同，那么打区间覆盖标记。

否则，暴力往下更新。

这样复杂度也是正确的。

Rikka with Sequence

有一个n个数的序列$\{a_n\}$，要求实现三种操作：

1. 将$a_x(l \le x \le r)$改为$\left[\sqrt{a_x}\right]$；
2. 将$a_x(l \le x \le r)$加上y；
3. 查询$\sum_{i=l}^{r} a_i$的值。

$1 \le n, q, a_i \le 10^5$

HDU 5828 Rikka with Sequence

考虑记录区间最大值和最小值。

如果一个区间最大值开根和最小值开根相同，那么打区间覆盖标记。

否则，暴力往下更新。

这样复杂度正确吗？

考虑：$n = 10^5, m = 10^5, a = \{1,2,1,2, \dots, 1,2\}$

每次先将$[1, n]$集体+2，然后集体开根，重复5×10^4次。

复杂度明显为$O(n^2)$。

HDU 5828 Rikka with Sequence

我们考虑对于区间$max - min \leq 1$的情况进行维护。其他仍然暴力做。

当区间$max = min$时，区间内的数相同，直接做即可。

当区间$max = min + 1$时候，如果$\sqrt{max} = \sqrt{min}$，那么变成区间覆盖，也是可以做的。

如果$\sqrt{max} = \sqrt{min} + 1$，那么$max - \sqrt{max} = min - \sqrt{min} = d$，转化为区间减$d$。

这样复杂度就对了吗？似乎只是排除了刚刚那种特殊情况？

HDU 5828 Rikka with Sequence

每次区间开根号，区间里一定存在两个相邻的数，他们的差减小。

而每次区间加只会改变两个差。

开根号减小次数和log差不多，所以复杂度是$O(n \log^2 n)$。

Gorgeous Sequence · 改

给定一个长度为n的序列，编号从1到n。m次操作，要求支持下面几种操作：

1. 把一个区间$[L,R]$里小于x的数变成x；

2. 求区间$[L,R]$的和；

3. 求区间$[L,R]$的最小值。

$1 \le n,m \le 10^6, |a_i| < 2^{31}$

Gorgeous Sequence · 改

Segment Tree Beats!（WC2016营员交流）

维护最小值、最小值出现次数和次小值。

每次区间对p取max：

如果当前节点满足$min < p < sec$，就打一个区间max标记，而且可以很方便算出改变量。

否则暴力往下继续做。

标记下传考虑每个儿子，要么满足$p < min$，要么满足$min < p < sec$，直接维护即可。

虽然时间复杂度是$O(n \log^2 n)$的，但是跑的和$O(n \log n)$差不多快！

原题（HDU 5306）是区间对p取min和求最大值。

和谐序列

给定一个n个元素的序列$\{a_i\}$。

定义“和谐序列”为序列的任何两个相邻元素相差不超过k。

求a_i的子序列中“和谐序列”的个数。

$1 \leq n \leq 10^5$

和谐序列

f_i表示以第i个数为结尾的和谐序列的个数。

$f_i = 1 + \sum_{j=1}^{i-1} f_j[|a_j - a_i| \le K]$

拆开绝对值，即$a_i - K \le a_j \le a_i + K$

那么每次更新完f_i，将f_i插入到BIT（树状数组）上a_i对应位置。

查询的时候只需要查找区间和即可。

如果a_i很大，需要离散，并在查询时候二分。

新型计算机

输入序列中有一些数字（都是自然数，包括 0）。

计算机先读取第一个数字s_1，然后顺序向后读入s_1个数字； 接着再读一个数字s_2，顺序向后读入s_2个数字……依此类推。

只有计算机正好将输入序列中的数字读完，它才能正确处理数据，否则计算机就会进行自毁性操作！

Tim 现在有一串输入序列，但可能不是合法的，也就是可能会对计算机造成破坏。 于是他想对序列中的每一个数字做一些更改，加上一个数或者减去一个数。当然，也可以仍然保持其为原来的数。 使得更改后的序列为一个新型计算机可以接受的合法序列。

Tim 还希望更改的总代价最小，所谓总代价，就是对序列中每一个数改变的绝对值之和。

$1 \leq n \leq 10^6$

BZOJ 2259 新型计算机

从前往后不好做，考虑从后往前。

f_i表示$i \to n$合法的最小代价，那么$f_i = \min\{f_j + |a_i - (j - i - 1)|\}$

当$a_i > j - i - 1$，也就是$a_i + i + 1 > j$，那么有

$$f_i = \min\{f_j - j\} + a_i + i + 1$$

当$a_i < j - i - 1$，也就是$a_i + i + 1 < j$，那么有

$$f_i = \min\{f_j + j\} - (a_i + i + 1)$$

那么以j为下标，用BIT/线段树维护$f_j - j$和$f_j + j$最小值即可。

每次分情况查找。时间复杂度$O(n \log n)$

该题还有其他做法，在这里不一一叙述。

折线统计

二维平面上有 n 个点 (x_i, y_i)。

现在这些点中取若干点构成一个集合 S，对它们按照 x 坐标排序，顺次连接。

将会构成一些连续上升、下降的折线，设其数量为$f(S)$。

现给定 k，求满足 $f(S) = k$ 的 S 集合个数。

$1 \leq n \leq 50000, k \leq 10, 1 \leq xi, yi \leq 10^5$

BZOJ 3688 折线统计

令$f_{i,j,0/1}$表示到第i个数，选了j段，当前折线是上升/下降。

$$f_{i,j,0} = \sum_{k=1}^{i-1} f_{k,j,0} + f_{k,j-1,1}(if\ Y_k > Y_i)$$

$$f_{i,j,1} = \sum_{k=1}^{i-1} f_{k,j,1} + f_{k,j-1,0}(if\ Y_k < Y_i)$$

开$2k$棵（对后两维）BIT每次求出f的时候在对应位置插入即可。

然后每次查询求和。复杂度$O(nk\log n)$

Pillars

有n个元素：$h_1, h_2, \ldots, h_n$。

求一个最长子序列，要求相邻两个数满足：$\left|h_{i_k} - h_{i_{k-1}}\right| \geq d$。

并要求打印出任意一个最长的序列。

$1 \leq n \leq 10^5, 1 \leq d, h_i \leq 10^9$

CodeForces 474E Pillars

设$f(i)$表示取到第i个，且第i个一定取的最长长度。

那么：

$$f(i) = \max_{1 \le j \le i}\left(f(j)\right) + 1 \quad \left(\left|h_i - h_j\right| \ge d\right)$$

分析以下，其实就是$h_j \le h_i - d$或$h_j \ge h_i + d$。相当于有两个区间要查询最大值。

问题是h是10^9级别的，无法直接用数据结构。

实际上我们可以直接对h离散，然后对于$h_i - d$和$h_i + d$去离散列表中二分即可得到区间。

然后用线段树即可实现查询。时间复杂度$O(n \log n)$。

Sail

有一艘船，船上n个旗杆，每个旗杆上有h_i个小节。每根旗杆上会挂k_i张帆。

每个小节上最多挂一个帆。 在风中，帆的不同排布方式会产生不同的推动力。

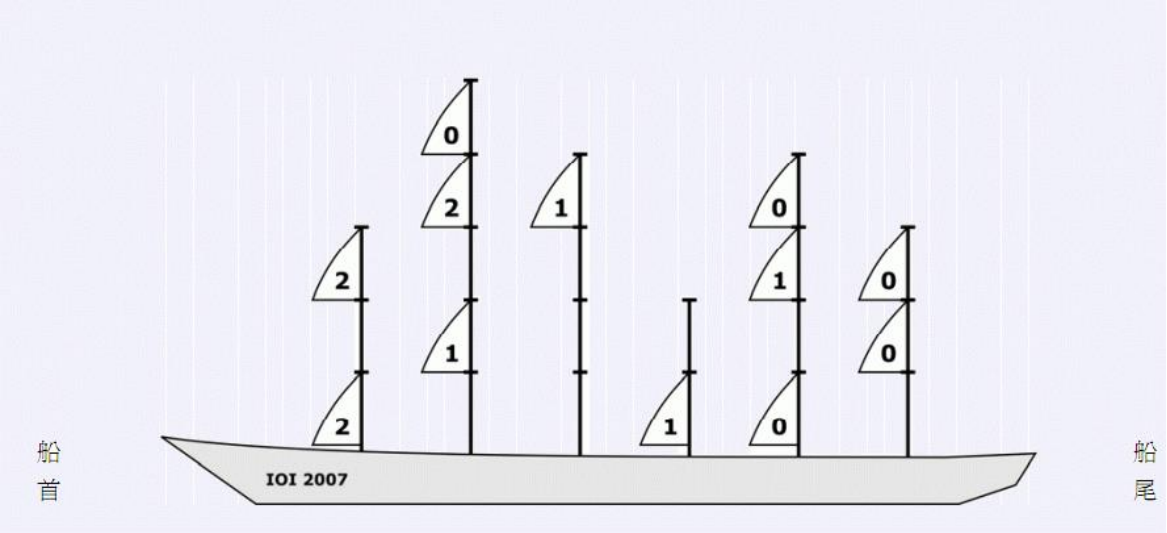
对于任意一张帆，它的**推动力折扣**等于在它后面并且和它在同一高度的帆的数目。

所有帆的任意一种位置组合的推动力折扣和等于在该位置下所有帆的推动力折扣的和。

求所有位置组合中最小的推动力折扣和。

（图上是一个例子）

$2 \le n \le 10^5, 1 \le h \le 10^5, 1 \le k \le h$



BZOJ 1805 IOI 2007 Sail

贪心。设每一个位置上的旗子数量为S_i，那么按照旗杆长度从小到大排序。

每次贪心的找到可行的位置中，旗子数量最少的k个放旗子即可（即$S_i \mathrel{+}= 1$）。

最后$\sum_i \frac{S_i(S_i-1)}{2}$就是答案了。时间复杂度$O(\sum_{i=1}^{n} h_i)$。

如何优化？Hint：线段树。

BZOJ 1805 IOI 2007 Sail

按照S_i维护一个排序后的数列，那么每次就相当于区间加1。

区间加1后可能会使得序列不再有序，怎么办？

（一个例子）

原来的S_i：1 1 1 1 3

[1,3]加1后的S_i：1 2 2 2 3

实际上也是区间加法！我们只要找到之后第一个比当前这个数大的即可。

利用线段树即可实现。时间复杂度$O(n \log n)$

Yet Another Maxflow Problem

给出一张图，这个图的点击被分为两部分，记作A和B，每部分都有n个点。

分别记作：$A_1, A_2, \dots, A_n$和$B_1, B_2, \dots, B_n$。

连边：$(A_i \to A_{i+1}), (B_i \to B_{i+1})$，容量分别为$x_i$和$y_i$。

A点集和B点集间有m条边：$(A_{u_i} \to B_{v_i})$，容量w_i。

接下来有q次操作，每次操作改变$(A_{t_i} \to A_{t_i+1})$的容量为p_i。

求每次操作后$A_1 \to B_n$的最大流。

$2 \le n, m \le 2 \times 10^5, 0 \le q \le 2 \times 10^5, 1 \le x_i, y_i, w_i, p_i \le 10^9$

CodeForces 903G Yet Another Maxflow Problem

最大流转最小割。得出以下简单结论：

在A中，若割掉了$(A_x \to A_{x+1})$，那么割掉$(A_y \to A_{y+1})(y > x)$没有意义；

在B中，若割掉了$(B_x \to B_{x+1})$，那么割掉$(B_y \to B_{y+1})(y < x)$没有意义。

因此，A中至多1条边属于割集，B中也至多1条边属于割集。

设这两条边为$(A_x \to A_{x+1}), (B_y \to B_{y+1})$，那么$(A_u \to B_v)(u \le x, y < v)$都属于割集。

那么在A中从小到大枚举A_x，问题在于在B中找到最优决策点y使得剩下代价最小。

CodeForces 903G Yet Another Maxflow Problem

每加入一条$(A_u \to B_v)$边，对答案造成的影响为连续一段，相当于区间加；

那么可以用线段树维护出对于所有x的剩下代价的最小值。

注意到修改A边容量，这个剩下代价最小值不会改变。

因此用一个堆/set支持修改即可。时间复杂度$O(n \log n + q \log n)$。

Legacy

有n个星球，q条路径，每条路径有以下三种形式：

1. $u \to v$，费用w；

2. $u \to [l, r]$，费用w；

3. $[l, r] \to v$，费用w。

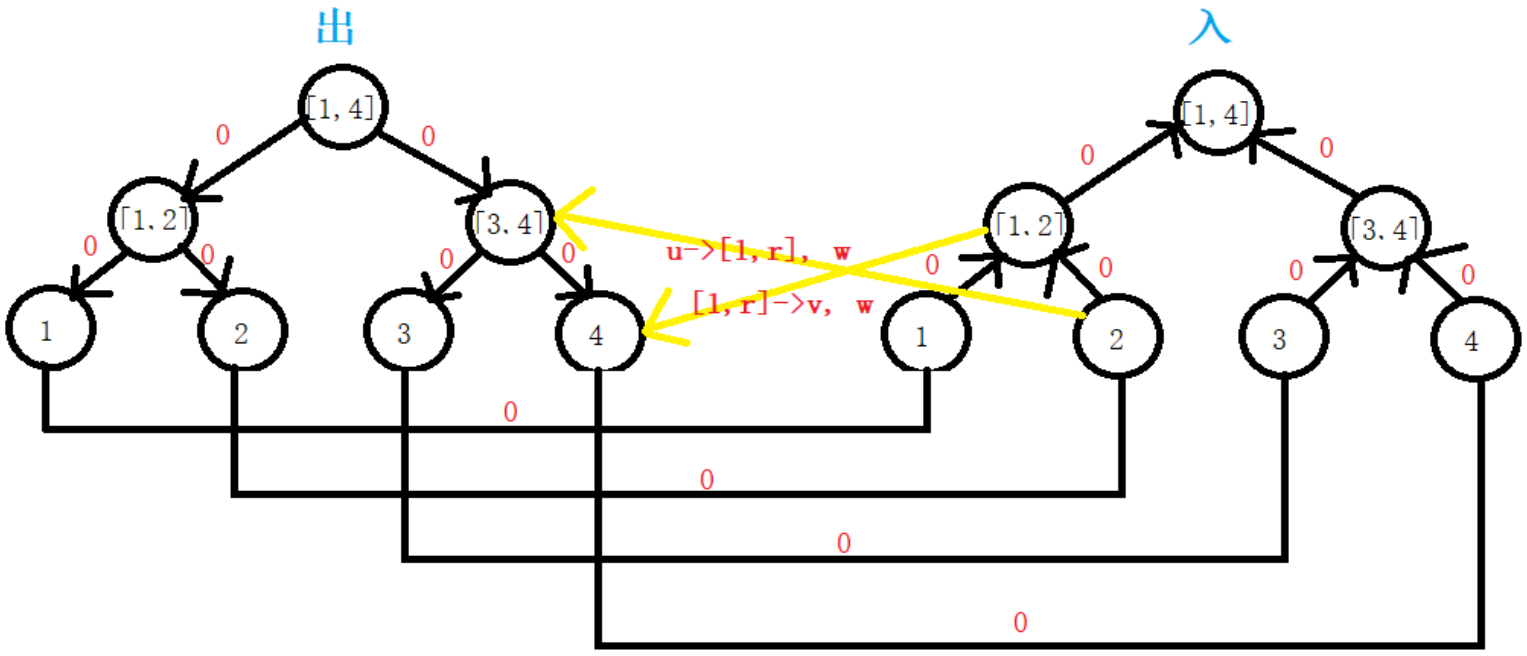
求s星球到其他星球最短路。

$1 \le n, q \le 10^5$

CodeForces 787D Legacy

线段树优化最短路建模。

建立两棵线段树，如下连边：



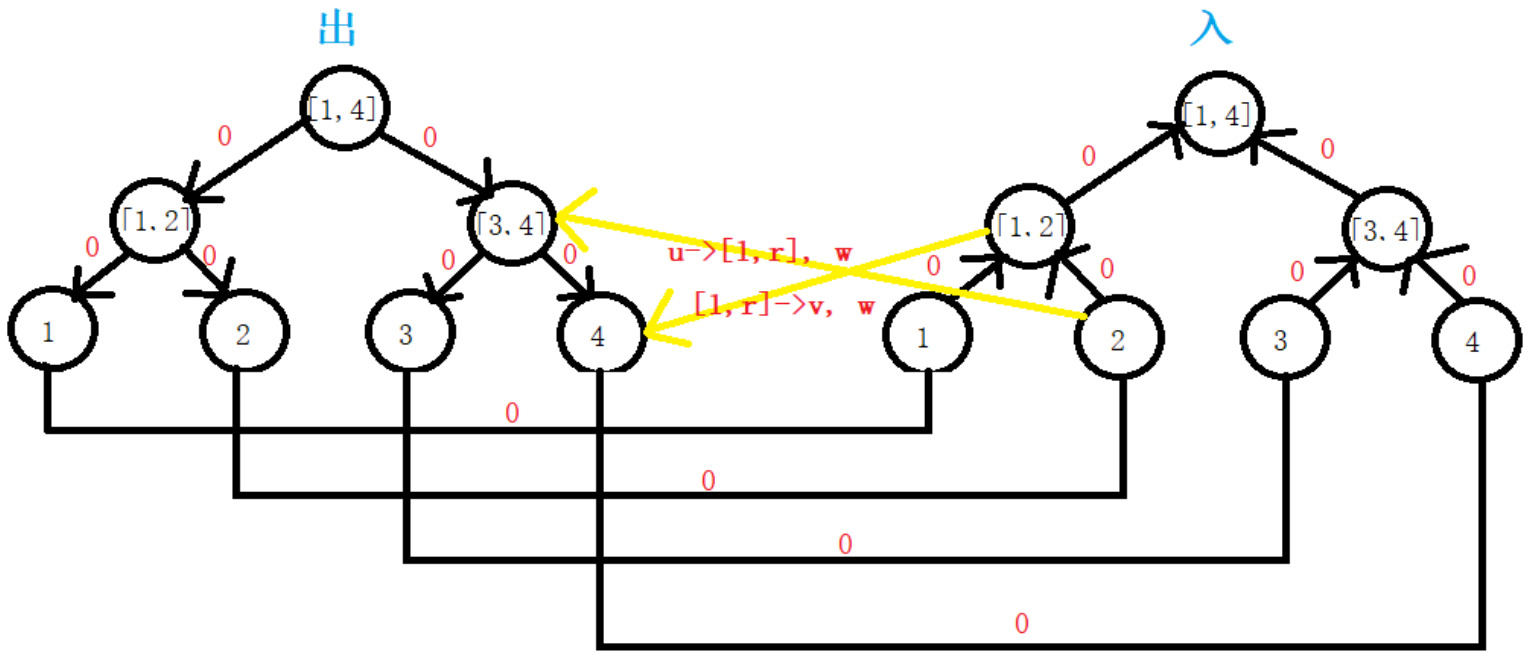
CodeForces 787D Legacy

共有$O(n \log n)$个点；

$O(n \log n + q \log n)$条边；

做一遍dijkstra算法即可。

时间复杂度$O(n \log^2 n)$



Journeys

有n个星球，m条路径，每条路径形式为：

$[l_1, r_1] \to [l_2, r_2]$，表示$[l_1, r_1]$的所有星球都可以花费1时间通往$[l_2, r_2]$的所有星球。

求从p星球开始的单源最短路。

$1 \le n \le 5 \times 10^5, 1 \le m \le 10^5$

BZOJ 3073 PA 2011 Journeys

和上一题的思路完全一样，同样建立两棵树，那么如何处理区间到区间的边呢？

对于每条边，建立两个虚拟节点p_1, p_2，然后：

入树向p_1连边，费用0；p_2向出树连边，费用0；p_1向p_2连边，费用1。

Bajtman i Okrągły Robin

有n个强盗，第i个强盗会在$[a_i, b_i]$时间段中选出一个长度为1时间段抢劫，并计划抢走c_i元。

在每一段长度为1的时间内最多只能制止一个强盗，那么最多可以挽回多少损失呢？

$1 \leq n \leq 5000, 1 \leq a_i \leq b_i \leq 5000, 1 \leq c_i \leq 10000$

BZOJ 4276 ONTAK 2015 Bajtman i Okrągły Robin

转化：有n个用时为1的任务，每个任务需要在$[a_i, b_i]$时间内完成，得c_i收益，求最大收益。

注意：以下连边$(flow, cost)$表示流量和花费。

建两组点，一组为任务，一组为时间。

从源点向任务点连边$(1, 0)$的边，从时间点向汇点连$(1, 0)$的边。

从每个任务向所有可完成的时间连$(1, c_i)$的边。

然后做最大费用最大流即可。这样的话边数为$O(n^2)$的。

利用线段树优化以下边数就可以降到$O(n \log n)$级别的了（和前面优化方法相同）

然后就可以通过本题。

Pustynia

给出一个长度为n的正整数序列$a_1, a_2, \ldots, a_n$，其中$a_i \in [1, 10^9]$。

告诉你其中s个数，同时给出m条信息，每条信息包括l, r, k以及接下来k个正整数，表示：$a_l, a_{l+1}, \ldots, a_r$里的这$k$个数中的任意一个都比任意一个剩下的$r - l + 1 - k$个数大。

构造出任何一组满足条件的方案，或判断无解。

$1 \le s \le n \le 10^5, 1 \le m \le 2 \times 10^5, \sum k \le 3 \times 10^5$

BZOJ 4383 POI 2015 Pustynia

朴素建图：新建点p，向所有的条件给定的x_i连边：$(p \to x_i, 0)$

同时向所有$l \sim r$中任意一个不是x_i的点t连边：$(t \to p, 1)$。

那么一个点i的最小值$f_i = \max(a_i, f_j + w_{j \to i})$，拓扑排序即可。

很明显会发现这样边数是$O(n^2)$级别的。

考虑到是向一些区间连边，那么把这些区间放在线段树上，然后向线段树节点连边即可。

这样边数就是$O(n \log n)$级别了。

然后用相同的方法，拓扑排序后dp即可。

时间复杂度$O(n \log n)$

炸弹

在一条直线上有n个炸弹，每个炸弹的坐标是x_i，爆炸半径是r_i。

当一个炸弹爆炸时，如果另一个炸弹所在位置x_j满足： $x_i - r_i \leq x_j \leq x_i + r_i$

那么，该炸弹也会被引爆。

现在，请你帮忙计算一下，先把第i个炸弹引爆，将引爆多少个炸弹呢？

$1 \leq n \leq 5 \times 10^5, |x_i| \leq 10^{18}, 0 \leq r_i \leq 2 \times 10^{18}$

BZOJ 5017 SNOI 2017 炸弹

如果一个炸弹能引爆另一个，那么就从这个炸弹向引爆的炸弹连一条有向边。

Tarjan缩点后拓扑排序，在拓扑序上倒着统计答案就可以知道能引爆的炸弹的区间了。

发现一个炸弹引爆另一个炸弹一定是一个区间，那么用线段树建边即可。

时间复杂度$O(n \log n)$。

Ultimate Weirdness of an Array

给一个n个数的序列a_i。

定义一个序列的奇怪值为最大的$\gcd(a_i, a_j)$，其中$i, j \in [1, n]$且$i \neq j$。

定义一个序列的终极奇怪值为$\sum_{i=1}^{n} \sum_{j=i}^{n} F(i, j)$。

其中$F(i, j)$表示a序列除去$a_k (i \leq k \leq j)$的序列的奇怪值。

求这个序列的终极奇怪值。

$1 \leq n \leq 2 \times 10^5, 1 \leq a_i \leq 2 \times 10^5$

CodeForces 671C Ultimate Weirdness of an Array

$h(i)$表示$f(l,r) \le i$的(l,r)对数，那么显然$h(i)$单调不降。

$next(i,l)$表示满足$f(l,r) \le i$的最小的r，那么$h(i) = \sum_{l=1}^{n} n - next(i,l) + 1$

显然$next(i,l)$随着i减小而单调不降，且$next(i,l)$随着l增加单调不降！

当$i = \max(a_j)(1 \le j \le n)$时，显然$next(i,l) = l$。考虑如何从$i$得到$i-1$。

我们找出约数有i的数，设他们下标$x_1 < x_2 < \cdots < x_m$，则从i变为$i-1$的时候：

任何一个$[l,r]$中至少包含$m-1$个x_j！（**这样才能使得剩下一个没法构成数对**）

所以$[x_2+1,n]$这段区间的$next(i-1,l)$应该全部改为$n+1$（**因为x_1, x_2没删，都不能选！**）

$[x_1+1,x_2]$的$next(i-1,l)$应该改为$\max(next(i,l), x_m)$

$[1,x_1]$的$next(i-1,l)$应该改为$\max(next(i,l), x_{m-1})$（**可以剩下一个！**）

由于$next(i,l)$随l增加单调不降，每次区间取max修改的是连续的一段区间！

我们利用线段树，就可以把复杂度优化到$O(n\log n)$。

gss2

给定n个元素的序列。

给出m个询问：求$[l,r]$的最大子段和（可选空子段）。

这个最大子段和有点特殊：一个数字在一段中出现了两次只算一次。

$1 \leq n,m \leq 10^5$

BZOJ 2482 SPOJ 1557 gss2

离线，按右端点排序，考虑枚举区间右端点，线段树上每个节点存区间左端点的值。

加入一个右端点，只有$[pre_i+1,i]$会增加贡献a_i，其中pre_i表示a_i上一次出现的位置。

当询问右端点的时候，相当于询问区间内的历史最大值。

这题的特殊之处在于，历史最大值是从开始到当前的最大值。

维护：当前值，当前tag，历史最大值，历史最大tag。

历史最大tag：上一次下传标记到这一次下传这个区间的标记这么一段时间区间里，存在的tag的最大值。

那么原始数就是还没有加历史最大tag的当前值

把原始数加上历史最大tag和历史最大值取max即可。

历史最大tag也要相应更新！时间复杂度$O(n\log n)$.

代码实现

```
ll w[SN], tag[SN];
ll hw[SN], htag[SN];  // history
# define ls (x<<1)
# define rs (x<<1|1)
inline void up(int x) {
    w[x] = max(w[ls], w[rs]);
    hw[x] = max(hw[ls], hw[rs]);
}
inline void pushtag(int x, ll tg, ll htg) {
    htag[x] = max(htag[x], tag[x] + htg);
    hw[x] = max(hw[x], w[x] + htg);
    w[x] += tg; tag[x] += tg;
}
inline void down(int x) {
    if(!tag[x] && !htag[x]) return ;
    pushtag(ls, tag[x], htag[x]);
    pushtag(rs, tag[x], htag[x]);
    tag[x] = htag[x] = 0;
}
```

```
inline void edt(int x, int l, int r, int L, int R, int d) {
    if(L <= l && r <= R) {
        pushtag(x, d, d);
        return ;
    }
    down(x);
    int mid = l+r>>1;
    if(L <= mid) edt(ls, l, mid, L, R, d);
    if(R > mid) edt(rs, mid+1, r, L, R, d);
    up(x);
}
inline ll query(int x, int l, int r, int L, int R) {
    if(L <= l && r <= R) return hw[x];
    down(x);
    int mid = l+r>>1;
    if(R <= mid) return query(ls, l, mid, L, R);
    else if(L > mid) return query(rs, mid+1, r, L, R);
    else return max(query(ls, l, mid, L, mid), query(rs, mid+1, r, mid+1, R));
}
```

Math Puzzle

给定n个数，q次操作，操作有两种：

$1\ l\ r\ x$：询问是否能改变一个数的值，使得$[l,r]$区间的gcd等于x；

$2\ i\ y$：将i位置的数改成y。

$1 \le n \le 5 \times 10^5, 1 \le q \le 4 \times 10^5, 1 \le a_i \le 10^9$

CodeForces 914D Math Puzzle

用线段树维护每一段的gcd，单点修改直接做，考虑如何处理查询。

结论1：如果当前某个区间的gcd可以被x整除，那么一定可以修改一个数使gcd变为x。

证明显然，随便找一个数修改成x即可。

结论2：如果一个区间的gcd不能被x整除，那么其中至少存在一个数不能被x整除。

结论3：如果一个区间的gcd可以被x整除，那么这个区间内所有数都可以被x整除。

那么按照这个思路，我们查询时只要考虑是否有多于一个数不能被x整除即可。

由于最多只会找到叶子2次，所以复杂度可以保证。

时间复杂度：$O(n\log n + Q\log n)$

Addition on Segments

有一个长度为n的数列，初始都为0。一共可以进行q次操作。

每次操作将$[l_i, r_i]$中的数加上x_i。

现在只做q次操作的子集，不能全做。

求问完成后数组中最大数可能是多少，列出所有可能。

$1 \leq n, q \leq 10^4, 1 \leq x_i \leq n$

CodeForces 981E Addition on Segments

性质：求最大值的所有可能值，相当于求所有元素的所有可能值。

证明：对于某个元素的所有可能值，将其他元素不进行任何操作即可。

线段树记录下每个区间可以进行的操作。

用bitset从上往下搜索，将所有可能都存在bitset中即可。

时间复杂度$O\left(\frac{n^2 \log n}{64}\right)$

统计

有n个数$\{a_n\}$，给出一个数p，有两种操作：

1. 将下标在$[a,b]$的数均改成y；

2. 询问所有数字中，在区间$[a,b]$出现率大于等于$p\%$的所有数。

一共有m次操作。

为了让这道题变得简单，当出现询问操作时，你可以输出k个数，只要这k个数**完全包含答案**就判定为正确。注意：需要满足$k \times p \leq 100$。

$1 \leq n,m \leq 1.5 \times 10^5, 20 \leq p \leq 100$

统计

一个区间内，如果有一个数出现次数大于$\frac{n}{2}$，我们怎么用$O(1)$的空间复杂度，$O(n)$的时间复杂度求出来？

统计

一个区间内，如果有一个数出现次数大于50%，我们怎么用$O(1)$的空间复杂度，$O(n)$的时间复杂度求出来？

每次的数和当前的答案两两抵消，最后剩下来的就是答案！

如果有一个数出现次数等于50%，怎么找出这个数？

统计

一个区间内，如果有一个数出现次数大于50%，我们怎么用$O(1)$的空间复杂度，$O(n)$的时间复杂度求出来？

每次的数和当前的答案**两两抵消**，最后剩下来的就是答案！

如果有一个数出现次数等于50%，怎么找出这个数？

同样，我们存储两个数，每次**三个不同的数相互抵消**，最后剩下的两个数一定包含答案！

统计

那么我们只要储存$\frac{100}{p}$个数，每次把$\left(\frac{100}{p}+1\right)$个不同的数抵消即可。

利用线段树，每个区间维护这$\frac{100}{p}$个数，剩下的就是区间合并了。

如果两个区间加起来还没有$\frac{100}{p}$个数，那大区间就把这$\frac{100}{p}$个数简单取并集即可；

如果两个区间的数的并集大于$\frac{100}{\mathrm{p}}$，那么按出现次数排序后模拟消除情况即可。

时间复杂度$O(m\log n)$