

线段树部分总结

Pyh

2017 年 7 月 6 日

目录

1 按位置建线段树的基础应用	2
1.1 一类难点在区间合并与维护信息的线段树问题	2
1.1.1 从简单情况开始分析	2
1.1.2 分析题目性质	3
1.2 一类难点在处理标记的线段树问题	4
1.2.1 从全局出发, 一种“操作若干次就不再有效”的问题	4
1.2.2 标记永久化	5
1.2.3 区间最值操作与历史最值询问	5
2 权值线段树的基础应用	6
2.1 充当普通平衡树	6
3 可持久化线段树的基础应用	6
3.1 一类维护序列中某个区间信息的问题	6
4 动态开点线段树的基础应用	7
5 线段树合并的基础应用	7
5.1 一类考虑子树对父亲的贡献的问题	8
6 线段树套线段树的基础应用	9
7 线段树作为辅助数据结构的一些问题	9
7.1 用线段树优化 dp	9
7.2 用线段树判断完美匹配	10
7.3 用线段树模拟费用流	11
8 线段树问题的一些小技巧	11
8.1 下标有关	11
8.2 一类关于子串的问题	12

1 按位置建线段树的基础应用

有一些典型的数据结构题，即给出几种操作，动态询问区间的某些信息。下面我总结了几类按位置建线段树能够解决的难题。

1.1 一类难点在区间合并与维护信息的线段树问题

有一类问题，初看到问题会无从下手，往往修改操作十分简单，但是询问的内容比较复杂，甚至某些问题完全没有修改操作，只有区间的动态询问。

下面我总结了一些解决这类问题的方法。

1.1.1 从简单情况开始分析

这是一种通用的方法，我们可以首先从简单情况分析起，最后类比推理到复杂的情况。

例题 1. *k-Maximum Subsequence Sum*¹

给出一个长度为 n 的序列 A , m 次操作，操作有如下两种：

1. 给出 i, val , 把 A_i 变成 val 。
2. 给出 l, r, k , 询问把区间 $[l, r]$ 划分成不超过 k 个不相交的区间，这些区间中数的和的最大值。

数据范围： $n, m \leq 10^5, k \leq 20$

这里介绍一种 $O(k^2(n+m)\log n)$ 的算法（下面将会介绍更优的算法）。

因为是单点修改，我们考虑怎么将两个子区间的信息合并到当前区间。

$k=1$ 的情况就是区间最大子段和，是一个经典问题，不再赘述。

考虑 $k=2$ 的情况。

我们发现选的两个子区间要么都不跨左右子区间的边界，要么一个跨了另一个没跨。

这就相当于在一个子区间中是最大后缀（前缀）和，在另一个子区间中是一端贴着边界，总共包含两个子区间的最大和。

这个相当好维护。那么能不能推广呢？

首先我们在每个子区间维护一个数组 $maxx[1..k]$, $maxx[i]$ 记录该区间划分成 i 个子区间的最大和。

显然首先当前区间划分成的 k 段区间可以完全在左子区间和右子区间，直接统计答案即可。

然后假设选的所有区间不跨两个子区间的边界，我们可以枚举左子区间放了多少个区间，直接统计答案。

跨边界的情况怎么办呢？我们注意到跨边界的那个区间可以拆分成两个部分，一个在左子区间，一个在右子区间。

于是我们对于一个区间，维护一个信息，表示当前区间划分了 k 个子区间，其中最右边的子区间是贴着右边界的最大和。

同理，维护一个最左边的子区间是贴着左边界的最大和（这些都是数组，因为一个子区间可能是放 $1..k$ 中的任意多个区间）。

为了维护上面的这两个“贴着边界最大和”的信息，我们需要再次考虑跨过中点的情况：这次轻车熟路了，我们只需要维护两端都贴着边界的最大和即可。

而维护“两端都贴着边界最大和”的信息，我们可以直接通过左右子区间维护的上述信息维护出来。

¹Codeforces 280D

因为计算维护的信息数组的每一项需要枚举左右区间各自放了多少个区间，所以区间合并的复杂度是 $O(k^2)$ 的，整体的复杂度为 $O(k^2(n+m)\log n)$ 。

这是一个很巧妙的算法，但是因为 k^2 比较大而无法通过本题。下文我将会讲述这个题目的正确解法，也是使用了线段树，不过是作为辅助数据结构来使用的。

1.1.2 分析题目性质

解决这类问题的另一种通用方法是分析题目中询问的信息的性质，来达到便捷地维护区间信息的目的。

例题 2. 楼房重建²

在二维平面内，给出 m 个操作。

每个操作给出 x, y ，把横坐标为 x 处的线段删除，并新建一条从 $(x, 0)$ 到 (x, y) 的线段，每个操作完成后询问在 $(0, 0)$ 能够看到多少条线段的顶部（线段可以互相遮挡）。

数据范围： $m \leq 10^5$ ，横坐标 $\leq 10^5$

显然问题可以转化成满足下面这个条件的线段的个数：前面的所有线段的 y/x 均严格小于它的 y/x 。

所以我们不妨把所有线段的高度变成 y/x 。

因为是单点修改，我们同样思考怎样合并两个子区间的信息。

假设当前考虑的区间的前面所有线段的高度最大值为 d ，

若左儿子的高度最大值 $< d$ ，则左儿子的所有线段完全被挡住，左儿子的贡献为 0，但是右儿子的贡献可能不为 0，递归右儿子；

若左儿子的高度最大值 $\geq d$ ，则左儿子的这个最大的线段完全挡住了左边的区间，在左边区间的修改完全不会影响到右儿子的线段，直接用右儿子在修改前对当前区间的贡献，而左儿子的线段可能会受影响，递归左儿子。

于是，在每次区间合并的过程中，我们要么递归左儿子要么递归右儿子，一次区间合并的复杂度为 $O(\log n)$ ，总的复杂度为 $O(n \log^2 n)$ ，可以通过本题。

这个问题的解决我们巧妙地应用了线段遮挡问题的性质，即修改一条线段对后面线段的影响是很少的，所以才能设计出数据结构解决本题。

例题 3. 捉迷藏³

给出一棵 n 个点的树，每个点有黑或白一种颜色。

有 m 个操作，每个操作改变一个点的颜色，询问最远的黑点对的距离。

数据范围： $n \leq 10^5, m \leq 5 \times 10^5$

这个题目又是单点修改，我们可以考虑用线段树来解决它。

首先这是一个树上的问题，我们首先用 dfs 序把它转化成区间问题。

感觉还是不好做，因为树上的距离不好直接在线段上表示出来。

这个时候，我们可以考虑树的一个关于距离的性质：在树的括号序列中，一个点到另一个点的距离恰好等于两点之间删去匹配括号之后剩下的括号的数量。

²bzoj2957

³ZJOI2007

于是对于每个区间，我们维护：

- 删去匹配括号之后的左括号、右括号的数量；
- 所有黑点到左端点、右端点左、右括号的数量的最大值；
- 所有黑点到左端点、右端点左括号数量-右括号数量、右括号数量-左括号数量的最大值。
- 该区间中最远黑远点对的距离。

怎么从子区间合并上述信息并不难，在此不再赘述。

这个算法的时间复杂度为 $O(n \log n)$ ，可以完美地解决本题。

1.2 一类难点在处理标记的线段树问题

除了上面这种维护的信息比较复杂的问题外，还有一类修改操作比较特别，需要使用懒标记来解决的问题。

1.2.1 从全局出发，一种“操作若干次就不再有效”的问题

这是一类通用的问题，其特点在于操作一旦超过若干次之后作用就可以直接处理。

其一般解决方法就是尝试进行全局操作。

例题 4. Rikka with Phi⁴

给出一个长度为 n 的序列 A ，有 m 个操作。

1. 给出 l, r ，将所有的 $A[i], l \leq i \leq r$ 全部变为 $\varphi(A[i])$ ；
2. 给出 l, r, x ，将所有的 $A[i], l \leq i \leq r$ 全部变为 x ；
3. 给出 l, r ，询问 $\sum_{i=l}^r A[i]$

数据范围： $n, m \leq 3 \times 10^5$

初看觉得十分不可做，但是我们尝试把所有操作看成全局操作：

覆盖操作会导致所有数字变成一样；

取欧拉函数操作最多 $\log(x_i)$ 次就会使所有数变成 1。

于是我们维护当前区间的最大值和最小值，如果最大值等于最小值那么直接打上覆盖标记；否则递归处理下面的区间。

我们来分析复杂度。单看欧拉函数操作，每个数最多 $\log(x_i)$ 次就会变成 1，复杂度显然是 $O(n \log n)$ 的。

如果加上覆盖操作，根据我们的算法，我们可以把相同的一段区间看作一个数字。

那么每一次覆盖操作最多增加一个数字，均摊下来不会降低取欧拉函数操作的复杂度。

所以总的时间复杂度是 $O(n \log n)$ 。

例题 5. 基础数据结构练习题⁵

给出一个长度为 n 的序列 A ，有 m 个操作。

1. 给出 l, r, x ，将所有的 $A[i], l \leq i \leq r$ 变成 $A[i] + x$ 。
2. 给出 l, r ，将所有的 $A[i], l \leq i \leq r$ 变成 $\lfloor \sqrt{A[i]} \rfloor$
3. 给出 l, r ，询问 $\sum_{i=l}^r A[i]$

⁴hdu5634

⁵uoj228

我们再次把操作看成全局操作。

我们发现两个数如果开根号的话差会变成 $\sqrt{a} - \sqrt{b} = (a - b)/(\sqrt{a} + \sqrt{b})$ 。

可以发现最多 $O(\lg \lg n)$ 次就可以变成 0

这样不停地开根号两个数的差就会很快减小到 0。

但是我们发现如果两个数的差为 1，则开根号之后可能差还为 1，所以需要特判差为 1 的情况（发现正好就是区间减法）。

而区间加法只不过是把区间两端的差给变化了。

综上所述，维护区间的最大值和最小值，如果最大值与最小值的差小于等于 1，则直接打上加法标记，否则递归处理子区间。

这种类似的操作还有很多，例如区间取模⁶；区间除法⁷；区间每个数 a_i 都变成 c^{a_i} ， c 为常数⁸等等，都可以利用这个思想来解决这些问题，有些还需要特殊的技巧。

1.2.2 标记永久化

标记永久化是一种特殊的思想，其主要原理是把标记打到节点上，然后询问的时候自上而下的经过每个节点的同时更新答案。

例题 6. Segment⁹

要求在平面直角坐标系下维护两个操作：

1. 在平面上加入一条线段。记第 i 条被插入的线段的标号为 i 。

2. 给定一个数 k ，询问与直线 $x = k$ 相交的线段中，交点最靠上的线段的编号。

数据范围：操作数 $\leq 10^5$ ，直线的左右端点的横坐标 ≤ 39989 ，纵坐标 $\leq 10^9$ 。

强制在线。

因为是单点查询，所以我们只需要考虑如何插入一条线段并维护信息。

首先找到这个区间所代表的 $O(\log n)$ 个节点。

对于一个节点，如果上面没有线段，直接把线段信息记录在节点上；

如果上面有线段，则比较两根线段。

如果在这个区间中，当前插入的线段完全优于原线段，则覆盖掉节点信息；

如果当前插入的线段完全劣于原线段，则直接退出；

否则比较两根线段的交点，把影响区域大的线段信息记录在节点上，另一根线段的信息递归下放。

这样每经过一个结点需要 $O(\log n)$ 次下放，总的复杂度是 $O(n \log^2 n)$ 。

每次询问的时候，从上而下进行访问，经过的所有节点的信息就是覆盖在当前节点上的线段的信息。

所以我们可以 $O(n \log^2 n)$ 的时间内解决这道题。

1.2.3 区间最值操作与历史最值询问

这是一种玄妙的姿势，就是通过记录一些奇怪的信息来支持区间取 min、取 max、询问历史最值的操作。

具体的实现方法在吉如一的 2016 年国家集训队论文里面有详细论述。

⁶某一次省选模拟考试

⁷某一次省选模拟考试

⁸Shoi2017 相逢是问候

⁹Heoi2013 day1

2 权值线段树的基础应用

2.1 充当普通平衡树

权值线段树即按权值建树，适用于一些不要求询问区间的题目（充当普通平衡树的作用），或者通过树套树或线段树合并乃至可持久化来支持询问一些特殊的东西。

充当普通平衡树，其实就是把元素的权值当作下标插入到线段树中，然后利用线段树来维护这些元素的一些性质。

因为这个很水所以我就不举例子了。

3 可持久化线段树的基础应用

如果两棵线段树之间有许多相同的节点可以共用，那么我们可以考虑用可持久化线段树让它们共用节点。

可持久化线段树最基本的应用就是通过区间的差分来维护给定区间的一些信息，当然也可以用树套树……。

3.1 一类维护序列中某个区间信息的问题

例题 7. 网络管理¹⁰

给出一棵 n 个点的树，每个点有一个值 w_i 。有 m 个操作。

1. 给出 a, b ，将 w_a 变成 b 。
2. 给出 a, b, k ，询问从 a 到 b 的路径中第 k 大的值。

数据范围： $n, m \leq 80000, w_i \leq 10^8$ 。

这是一个经典问题。

我们对于每个节点开一棵权值线段树，维护从根节点到自己的所有 w 。

因为每个节点相对于父亲节点只是插入了一个 w ，所以可以用可持久化线段树，每次插入复杂度为 $O(\log n)$ 。

那么询问的话就是两个端点的权值线段树减去 LCA 的权值线段树（注意细节即可）。

修改一个点的值只会影响它的子树。

于是我们按照 dfs 序建树状数组，树状数组的每个节点维护一棵权值线段树。

修改点值就相当于在它的子树这一段连续的 dfs 序区间中每个点修改值，用树状数组的区间修改、单点查询的方法就能够在 $O(n \log^2 n)$ 的时间内解决本题。

这就是伟大的树上带修改第 k 大。

这类问题满足的性质就是维护的信息满足可加减性，然后就可以用可持久化线段树来动态维护信息了。

例题 8. k seq¹¹

给出一个长度为 n 的序列 A 和 k 。

定义一个子串的 w 为子串中，相同的数字只计算一次，所有的数的和。

求第 k 大的 w 。

数据范围： $n, k \leq 10^5$

¹⁰CTSC2008

¹¹hihocoder 1046

因为一个子串总能被表示成一个后缀的前缀，所以我们对于每个位置 i 维护一个普通线段树 $S[i]$ ，里面下标为 j 的元素是 $w[i..j]$ 的值。同时维护一个最大值。

首先我们先暴力预处理出普通线段树 $S[1]$ 。

然后往后枚举每一个位置，对于每一个位置 i ，记上一个位置的那个元素下一次出现的位置为 pos 。

则对于 $j \in [i, pos - 1]$ ， $S[i]$ 中下标为 j 的元素的值就等于 $S[i - 1]$ 中下标为 j 的元素的值减去 $val[i - 1]$ ，其他下标的元素的值不变。

使用可持久化线段树，使用懒标记，于是每个位置只需要花费 $O(\log n)$ 的时间就可以构造出线段树。

我们把每个位置的线段树的最大值放到一个堆里面。

每次找出最大的那个位置，然后将最大的那个后缀的最大的那个前缀的位置上的值改成无穷小，求出最大值之后再次插入堆中。

这个操作重复 k 次即找出第 k 大值。

复杂度为 $O((n + k) \log n)$ 。

4 动态开点线段树的基础应用

一般动态开点线段树的应用情况就是要开的节点太多，但是实际用到的节点特少的情况下，我们可以用指针树的方法动态开线段树的节点。

动态开点线段树的应用很广泛，可持久化线段树若有若无地用到了这个思想，线段树合并则是这个东西的具体应用。

下面我给出一道体现“开点开不下”的例题。

例题 9. 旅行¹²

给出一棵 n 个点的树， q 个操作。每个点有两个值。

1. 给出 x, c ，将第 x 个点的第一个值改成 c 。
2. 给出 x, c ，将第 x 个点的第二个值改成 c 。
3. 给出 x, y ，询问第 x 个点到第 y 个点经过的路径中，第一个值与 x 的第一个值相同的点，第二个值的和。
4. 给出 x, y ，询问第 x 个点到第 y 个点经过的路径中，第一个值与 x 的第一个值相同的点，第二个值的最大值。

数据范围： $n, q \leq 10^5, c \leq 10^5$

好吧这就是一个水题，我们对于每个“第二个值”开一棵线段树，即 10^5 棵线段树维护每个位置，然后用树链剖分即可。

因为每棵线段树的极限空间是 $O(n)$ 的，所以空间复杂度是 $O(nc)$ 的。

但是因为总的节点数量不超过 $O(n)$ ，所以动态开点即可，空间复杂度为 $O(n)$ 。

下面应该还有一些要用到动态开点的题目。

5 线段树合并的基础应用

线段树合并就是动态开点的线段树合并在一起，使用条件是总点数为 $O(n)$ ，则复杂度为 $O(n \log n)$ 。

具体实现方法就是哪里有点就往哪里合并。

¹²Sdoi 2014

5.1 一类考虑子树对父亲的贡献的问题

这种题目看起来就像树形 dp，就是从每个点维护一棵线段树，然后合并到父亲那里去。

例题 10. 天天爱跑步¹³

给出一棵 n 个节点的树， m 条有向路径，每个节点有一个值 v_i 。

对于每个节点，询问有多少条经过它的有向路径，且它到这条路径的起点的距离为 v_i 。

这个题我学习了一下线段树合并的做法。

对于一条有向路径，将它的信息记录在两个端点上，因为路径上的其它点一定是两个端点其中一个的父亲，于是可以用树形 dp 的思想。

对于这条路径的信息，我们经过左右端点的时候记录在线段树中，在经过它们的 LCA 的父亲的时候删除。

经过每个点，我们在线段树中查找，是否有一条路径走到它的时候距离为 v_i 。

这个查询需要用到下文中“下标有关”的思想，其实也很容易，将距离与自己的深度拆开即可。

于是经过一个节点，首先合并子树，然后删除信息，最后进行查询。

复杂度为 $O(n \log n)$ 。

下面是一个更加容易看出这种思想的题目。

例题 11. Tree Rotations¹⁴

现在有一棵二叉树，所有非叶子节点都有两个孩子。在每个叶子节点上有一个权值 (有 n 个叶子节点，满足这些权值为 $1..n$ 的一个排列)。

可以任意交换每个非叶子节点的左右孩子。

要求进行一系列交换，使得最终所有叶子节点的权值按照遍历序写出来，逆序对个数最少。

首先对于每个节点维护一个权值线段树。

因为对于一个节点交换其左右子树对于左右子树中自己的逆序对没有影响，所以我们可以用类似树形 dp 的方法。

一个节点的逆序对数量等于左右子树中最少的逆序对数量，加上是否交换左右子树产生的跨树的逆序对的数量。

这个跨树的逆序对我们只需要在合并左右两颗权值线段树的时候顺带统计一下就好了。

于是可以在 $O(n \log n)$ 的时间内解决此题。

下面是一个另一种风格的题目。

例题 12. 安全路径¹⁵

给出一张点数为 n ，边数为 m 的无向图，对于每个点，询问从第一个点到这个点的，不经过最短路的最后一条边的最短路的长度。

保证对于每个点只有一条最短路。

数据范围： $n \leq 10^5, m \leq 2 \times 10^5$

首先构出最短路径树，因为不经过最后一条边，所以删掉这条边之后，第一个点到这个点在最短路径树上不连通。

¹³noip2016

¹⁴Poi2011

¹⁵Usaco Jan09

那么只能从这个点的子树中的某个点往上走到这个点。

对于它（编号为 j ）的子树中的一个点 i ，和这个点的一条进来的非树边的起点 e ，贡献是 $dis[1][e] + dis[e][i] + depth[i] - depth[j]$

于是我们把子树中的所有点以及它们的非树边的 $dis[1][e] + dis[e][i] + depth[i]$ 放到权值线段树中。

则一个点的答案就是权值线段树中的最小值减去 $depth[j]$ 的值。

我们一路线段树合并上去就能在 $O(n \log n)$ 的时间内解决此题。

6 线段树套线段树的基础应用

线段树套线段树其实就是在在线段树的每个节点都开一棵线段树。

这种题目也是烂大街了，可持久化线段树的题目一般都能改成树套树，复杂度不变或增大几个 \log 。

这里举一个特殊一点的树套树的题目的例子。

例题 13. K 大数查询¹⁶

有 n 个位置， m 个操作。操作有两种，每次操作如果是 $1\ a\ b\ c$ 的形式表示在第 a 个位置到第 b 个位置，每个位置加入一个数 c

如果是 $2\ a\ b\ c$ 形式，表示询问从第 a 个位置到第 b 个位置，第 C 大的数是多少。

数据范围： $n, m \leq 50000$

这个题目如果想要每个位置分别维护元素的话显然元素个数要爆炸。

所以我们考虑怎么每个元素分别维护。

于是我们用树套树，权值线段树套位置线段树，即每一个权值我们开一棵线段树，表示哪些区间中有这个权值。

询问的时候从权值线段树自上往下走，查询个数的时候深入到位置线段树中即可。

时间复杂度为 $O(n \log^2 n)$

7 线段树作为辅助数据结构的一些问题

除了裸的数据结构题之外，还有一些题目线段树是作为辅助数据结构来使用的，主体算法往往不是线段树。

7.1 用线段树优化 dp

这个是老生长谈了。

我就举一道考试题作为例子吧。

例题 14. 永远亭的竹笋采摘¹⁷

辉夜原本是生活在月宫的月之公主。

在永远亭中，有一排新鲜的 n 个竹笋。辉夜对他们很是喜欢，于是决定采摘一些来制作 k 份料理。她找出了一个曾经在月都使用过的工具，凑合凑合可能可以用来采摘竹笋。这个工具每次可以采摘连续的一段竹笋，并用这些竹笋自动做出一份料理。但为了避免对工具造成损坏，这连续的

¹⁶Zjoi 2013

¹⁷test20170331

一段中的竹笋必须都是未经过采摘的。辉夜依次给每颗竹笋标注了卡路里 a_0, \dots, a_{n-1} ，拥有很棒很棒身材的她当然不想摄入过多的能量。用竹笋制作出的料理的能量是所有选出的竹笋中卡路里相差最小的两个卡路里不同的竹笋。每次采摘的竹笋中至少要有 2 个卡路里不相同的笋子。

辉夜想知道，她使用工具采摘竹笋能够获得的最少能量是多少。

$1 \leq n \leq 50000, k \leq 1000, k \leq n/2$ ，保证数据可以制作 k 份料理。 a_i 的值均是正整数且不超过 n 。

萌萌的 DP 题。

那么不妨设 $f_{i,j}$ 表示在 $1..j$ 的范围内使用了 i 个区间的最小答案。不难写出转移方程：

$$f_{i,j} = \min\{f_{i,j-1}, f_{i-1,k} + g(k,j)\}$$

其中 $g(k,j)$ 代表区间 $k..j$ 中最小的两个不同的数的差。

那么不难得到 $g(k,j)$ 的转移方程：

$$g(k,j) = \min(|a_k - a_j|, g(k+1,j), g(k,j-1)) | a_k \neq a_j$$

不难发现，这样 DP 是做了冗余操作的，即假设 $k..j$ 之间差最小的是 $|a_x - a_y|$ ，那么对于那些 $k \leq p < x$ 或 $y < p \leq j$ 的点，他们既可以被选入这个区间，又可以不被选入，也就是答案相同的区间我们进行了 $(x-k) \times (j-y)$ 次枚举，实际上只有 1 次是有意义的。

那么我们把两个 DP 写在一起：

$$f_{i,j} = \min\{f_{i,j-1}, f_{i-1,k} + |a_j - a_k|\} | a_j \neq a_k$$

终于有个靠谱一点的方程了，考虑优化。

把绝对值拆开，把 a_j 提到 \min 外面，可得：

$$f_{i,j} = \begin{cases} \min\{f_{i,j-1}, f_{i-1,k} - a_k\} + a_j & a_j > a_k \\ \min\{f_{i,j-1}, f_{i-1,k} + a_k\} - a_j & a_j < a_k \end{cases}$$

至此我们可以用线段树优化至 $O(nk \log n)$ ，即将 DP 值以 a_i 为关键字记录在线段树里面即可，但这个算法常数较大。

7.2 用线段树判断完美匹配

一些特殊的完美匹配问题可以用线段树来解决。

例题 15. lyz¹⁸

滑冰俱乐部初始有 $1..n$ 号码溜冰鞋各 k 双，已知 x 号脚的人可以穿 $x..x+d$ 号码的鞋子。现在在 m 次操作，每次两个数 r, x ，表示 r 号脚的人来了 x 个， x 为负表示离开。对于每次操作，输出溜冰鞋是否足够。

数据范围： $n, k, m \leq 5 \times 10^5, k \leq 10^9$

我们把人看作一个点集，溜冰鞋看作另一个点集，则题目转化为判断是否有一个匹配满足人的点集被鞋的点集完美匹配。

根据 Hall 定理，这个成立当且仅当对于人的每一个子集，可以连到的鞋的点集的点的数量不小于这个子集的大小。

¹⁸POI2009

于是对于任意一个区间 $[l, r]$, $num \leq (r + d - l + 1) \times k$, 其中 num 表示这个区间的脚的人的数量。

所以 $num - len \times k \leq d \times k$

于是我们把每个点的初值设为 $-k$, 然后每次找最大子段和判断是否大于 $d \times k$ 即可。

复杂度为 $O(n \log n)$ 。

7.3 用线段树模拟费用流

例题 16. k -Maximum Subsequence Sum¹⁹

给出一个长度为 n 的序列 A , m 次操作, 操作有如下两种:

1. 给出 i, val , 把 A_i 变成 val 。
 2. 给出 l, r, k , 询问把区间 $[l, r]$ 划分成不超过 k 个不相交的区间, 这些区间中数的和的最大值。
- 数据范围: $n, m \leq 10^5, k \leq 20$

这个题目之前讲了 $O(k^2 n \log n)$ 的做法, 但是会超时, 现在我介绍用费用流解决这个问题的方法。

我们把每个点拆成两个节点, 一个入点和一个出点。

对于序列中的一个点, 其出点向汇点连一条容量为 1, 费用为 0 的边, 其出点向下一个点的入点连一条容量为 1, 费用为 0 的边;

这个点的入点向出点连一条容量为 1, 费用为 A_i 的边。

从源点向每个点的入点连一条容量为 1, 费用为 0 的边。

于是增广 k 次后的最大费用流即为答案。

这个的复杂度特别高, 但是我们发现可以用线段树来模拟。

根据费用流的算法, 每次找到最大子段和, 然后将其取反, 重复 k 次这个操作即可得到答案。

于是复杂度为 $O(kn \log n)$, 可以解决本题。

8 线段树问题的一些小技巧

8.1 下标有关

有些问题中修改的量与下标有关系, 这时候一般的解决方法就是考虑如何将修改的量与下标拆分开来, 即表示成下标的系数与常数的和。

例题 17. 无名²⁰

给出一棵 n 个点的树, m 个操作。

1. 给定 x, k , 第 x 个点的子树中的每个点 j 的权值增加这个点到第 x 个点的距离的 k 倍。
2. 给定 x , 询问第 x 个点的权值。

i 的子树中的第 j 个点的距离是 $depth[j] - depth[i]$, 增加的权值就是 $k * depth[j] - k * depth[i]$, 于是我们对于每个点维护一个系数 $k[j]$ 和常数 $b[j]$, 每次修改 $k[j]$ 区间加上 k , $b[j]$ 区间加上 $-k * depth[i]$ 。

然后询问的时候直接输出 $k[j] \cdot depth[j] + b[j]$ 即可。

¹⁹Codeforces 280D

²⁰自己随便出的水题, 应该是原题

8.2 一类关于子串的问题

这类问题是我总结出来的一种有通用解决方法的问题。有些与子串无关的问题也可以用这个方法来解决。

例题 18. 序列²¹

给出一个长度为 n 的序列， m 个询问。

每个询问给出 l, r ，询问序列中区间 $[l, r]$ 中所有子串的最小值的和是多少。

数据范围： $n, m \leq 10^5$

对于这种关于“区间的所有子串”的题目，我们把一个区间 $[l, r]$ 看作是平面上的一个点 (l, r) ，则区间 $[l, r]$ 的所有子串就是一个左下角为 (l, l) ，右上角为 (r, r) 的矩形。

因为一个点能够影响（那些以它为最小值的区间）的区间也能构成平面上的一个矩形，所以我们把这个问题转化成了平面上的矩形加、矩形查询问题。

这个问题离线可以用线段树在时间 $O(n \log n)$ 、空间 $O(n)$ 的复杂度下解决，在线可以套一个可持久化线段树在时间 $O(n \log n)$ 、空间 $O(n \log n)$ 的复杂度下解决。

例题 19. 影魔²²

给出一个长度为 n 的序列， m 个询问。

对于一个子区间，定义它的值为：如果子区间中除了端点之外的所有元素中最大的元素同时小于左右端点，则值为 $p1$ ；如果子区间中除了端点之外的所有元素中最大的元素在左右端点的元素之间，则值为 $p2$ 。

每个询问给出 l, r ，询问区间 $[l, r]$ 的所有子串的值之和。

保证序列中没有相同的元素。

数据范围： $n, m \leq 10^5$

同上面的“序列”那题。

我们考虑一个元素（位置为 i ）的贡献。我们找到左边第一个比它大的元素的位置记作 l ，右边第一个比它大的元素的位置记作 r ，则区间 $[l, r]$ 贡献为 $p1$ ，区间 $[l, j], j \in [i+1, r]$ 、 $[j, r], j \in [l, i-1]$ 的贡献均为 $p2$ 。

我们利用上面转化成矩形操作的思想就可以在 $O(n \log n)$ 的复杂度内解决本题。

例题 20. 接水果²³

给定一棵 n 个节点的树，和若干条路径，每个路径有一个权值。

m 个询问，每个询问给定一条路径，询问路径的子路径中权值第 k 大的路径。

数据范围： $n, m \leq 40000$

如果一条路径 i 的两端的 LCA 不为两端的两个节点之一，则能够包含它的路径的两端一定在路径 i 的两端的子树中。

如果我们用 dfs 序表示，一条路径的两端节点的 dfs 序看作平面上的一个点，则包含路径 i 的路径又是平面上的一个矩形。

如果路径 i 的两端的 LCA 为两端的两个节点之一，则能够包含它的路径的一端在路径 i 的下面那个端点的子树中，另一个端点只要不在路径 i 的上面那个端点往下面那个端点的方向走一步到达的节点的子树中即可。这是两个矩形。

²¹Hnoi2016

²²Hnoi2017

²³Hnoi 2015

因为要求第 k 大，所以我们可以用扫描线 + 树状数组套主席树在 $O(n \log^2 n)$ 的复杂度内解决此题。