线段树合并

lsty

广州市第六中学

2023 年 3 月 28 日

目录

开始之前

线段树合并这个知识点虽然板子题都是紫色的，但是由于 NOI 大纲上没有这个东西所以甚至可能出现在 NOIP 里面
NOIP2021 就考了这个东西

开始之前

线段树合并这个知识点虽然板子题都是紫色的，但是由于 NOI 大纲上没有这个东西所以甚至可能出现在 NOIP 里面

NOIP2021 就考了这个东西

不过这东西内容知识点很简单，相信大家都是能学的会的，前置知识点只有权值线段树和动态开点就行了

而且功能很强大，可以用来做很多东西，基本上不少可重集合并都可以用线段树合并维护

另外值得一提的是这东西在这几年的 NOI 中三年考了两次，而且都在送分题的位置，题目都放到习题了

开始之前

线段树合并这个知识点虽然板子题都是紫色的，但是由于 NOI 大纲上没有这个东西所以甚至可能出现在 NOIP 里面

NOIP2021 就考了这个东西

不过这东西内容知识点很简单，相信大家都是能学的会的，前置知识点只有权值线段树和动态开点就行了

而且功能很强大，可以用来做很多东西，基本上不少可重集合并都可以用线段树合并维护

另外值得一提的是这东西在这几年的 NOI 中三年考了两次，而且都在送分题的位置，题目都放到习题了

其实本来还打算讲一下 DSU On Tree(树上启发式合并) 的但是写稿子实在是写不完了，所以鸽了

权值线段树

相信大家都会权值线段树
权值线段树维护的是序列的值域
考虑计数排序，对桶建一个线段树就是权值线段树
线段树的一个节点维护的就是一个权值区间，统计的信息通常是里面数的数量
在线段树上二分就可以 $O(\log n)$ 获取全局第 k 大
或者 $O(\log n)$ 查询一个数的排名之类的
而且这种做法很有前途，可持久化一下就可以查询区间第 k 大，这个等讲可持久化权值线段树的时候再说

动态开点

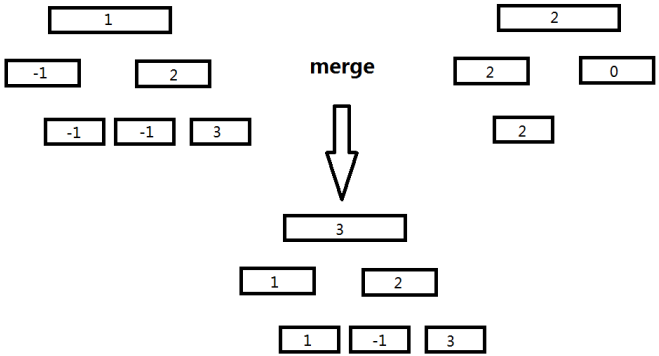
相信大家也很熟悉动态开点线段树
上面的权值线段树在值域高达 10^9 甚至更高的时候空间就很显然开不下了
然后发现询问有限的时候实际上用到的节点不多，每次询问最多会用到 $\log n$ 个节点
对于每个用到的节点查询到的时候再开辟就行了
最多有 $q \log n$ 个节点
实际上权值线段树 $+$ 动态开点可以平替平衡树，比如[NOIP 2017 D2T3] 列队

线段树合并

实际上就是上面说的并查集 + 线段树合并维护联通块
线段树合并的过程很简单，看下面这张图

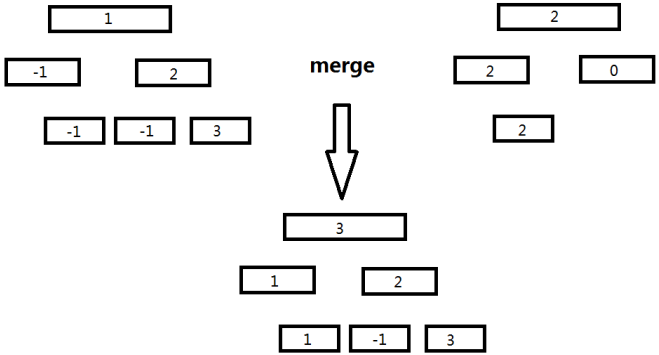
线段树合并

实际上就是上面说的并查集 + 线段树合并维护联通块
线段树合并的过程很简单，看下面这张图



线段树合并

实际上就是上面说的并查集 + 线段树合并维护联通块
线段树合并的过程很简单，看下面这张图
两个线段树结构相同，如果两个节点他们都有，那就合并上面的信息（这里是加法）
如果只有一棵树有，那就取有的那颗树的信息



一个例子

```
int merge(int a, int b, int l, int r) {
    if (!a) return b;
    if (!b) return a;

    //叶子结点
    if (l == r) {
        // do something
        return a;
    }
    int mid = l + r >> 1;
    ls(a) = merge(ls(a), ls(b), l, mid);
    rs(a) = merge(rs(a), rs(b), mid + 1, r);
    pushup(a);
    return a;
}
```

用处

权值线段树维护序列的值域的过程实际上是维护了整个序列的数集上的信息
因为维护值域相当于维护每个数在序列中出现的次数

用处

权值线段树维护序列的值域的过程实际上是维护了整个序列的数集上的信息
因为维护值域相当于维护每个数在序列中出现的次数
通过线段树合并把两棵权值线段树合并到一起相当于合并了两个数集
线段树合并就可以很方便的维护两个集合信息并在一起了
其实根据这个一个很经典的应用线段树合并 + 并查集维护联通块，这个等下就会见到

[HNOI2012] 永无乡

给你一张 n 个点 m 条边的图，点的编号为 $1 \sim n$，点有点权，你需要完成 q 次询问：
1. 从 x 到 y 连一条边
2. 查询点 x 所属联通块中点权的第 k 小点的编号

$1 \leq n, m \leq 10^5, q \leq 3 \times 10^5$

[HNOI2012] 永无乡

给你一张 n 个点 m 条边的图，点的编号为 $1 \sim n$，点有点权，你需要完成 q 次询问：
1. 从 x 到 y 连一条边
2. 查询点 x 所属联通块中点权的第 k 小点的编号

$1 \leq n, m \leq 10^5, q \leq 3 \times 10^5$

这题主要是讲线段树合并的实现

[HNOI2012] 永无乡

给你一张 n 个点 m 条边的图，点的编号为 $1 \sim n$，点有点权，你需要完成 q 次询问：
1. 从 x 到 y 连一条边
2. 查询点 x 所属联通块中点权的第 k 小点的编号

$1 \leq n, m \leq 10^5, q \leq 3 \times 10^5$

你对于每个联通块需要维护一个数集，支持查询集合中的第 k 大，这可以用权值线段树维护
询问 1 实际上就是你要合并两个数集，自然会想到线段树合并

[HNOI2012] 永无乡

给你一张 n 个点 m 条边的图，点的编号为 $1 \sim n$，点有点权，你需要完成 q 次询问：
1. 从 x 到 y 连一条边
2. 查询点 x 所属联通块中点权的第 k 小点的编号

$1 \leq n, m \leq 10^5, q \leq 3 \times 10^5$

具体实现可以看代码

CF600E

Lomsat gelral

给你一个 n 个节点，节点 1 为根的有根树，第 i 个节点有颜色 c_i
如果一个颜色在 x 为根的子树中出现次数最多，其在以 x 为根的子树中占主导地位
显然可能一个子树有多种占主导位置的颜色
每次询问你一个 i，问你以 i 为根的子树中所有占主导地位的颜色的编号和
$n \leq 10^5, c_i \leq n$

CF600E

Lomsat gelral

给你一个 n 个节点，节点 1 为根的有根树，第 i 个节点有颜色 c_i
如果一个颜色在 x 为根的子树中出现次数最多，其在以 x 为根的子树中占主导地位
显然可能一个子树有多种占主导位置的颜色
每次询问你一个 i，问你以 i 为根的子树中所有占主导地位的颜色的编号和
$n \leq 10^5, c_i \leq n$

同样也是一道板子题，对每个子树开一个线段树维护颜色种类的序列，然后就是一个取 $\max$ 的线段树了，在 DFS 的同时统计就行了，具体看代码

CF1009F

Dominant Indices

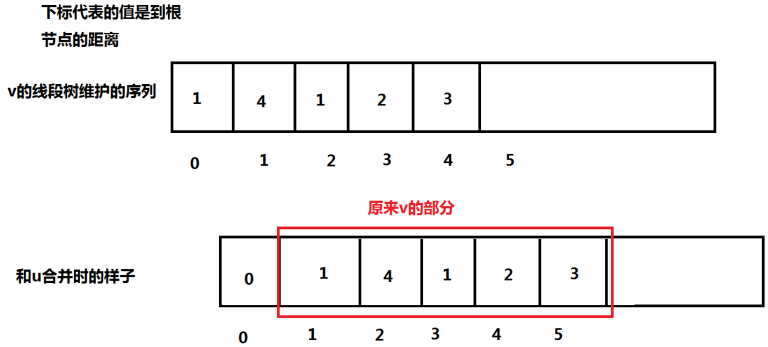
给你一个以 1 为根，n 个节点的无边权树，记 $d(u,x)$ 为 u 的子树中到 u 距离为 x 的节点数
对于树上的每个点 u, 你需要选择一个最小的 k 使得 $d(u,k)$ 最大
$n \leq 10^6$

CF1009F

Dominant Indices

给你一个以 1 为根，n 个节点的无边权树，记 $d(u,x)$ 为 u 的子树中到 u 距离为 x 的节点数
对于树上的每个点 u, 你需要选择一个最小的 k 使得 $d(u,k)$ 最大
$n \leq 10^6$

首先可能会直接想到用对每一个根节点开一个线段树维护到根节点的每个距离有多少数，取 $\max$
然后会发现这个东西不好合并，把父节点和子节点的线段树合并的时候要平移序列


下标代表的值是到根
节点的距离
v的线段树维护的序列
1
4
1
2
3
0
1
2
3
4
5
原来v的部分
和u合并时的样子
0
1
4
1
2
3
0
1
2
3
4
5

给一张图

Dominant Indices

给你一个以 1 为根，n 个节点的无边权树，记 $d(u,x)$ 为 u 的子树中到 u 距离为 x 的节点数
对于树上的每个点 u, 你需要选择一个最小的 k 使得 $d(u,k)$ 最大
$n \leq 10^6$

既然上面的方法不好做，那可以考虑换一个维护的信息

CF1009F

Dominant Indices

给你一个以 1 为根，n 个节点的无边权树，记 $d(u,x)$ 为 u 的子树中到 u 距离为 x 的节点数
对于树上的每个点 u, 你需要选择一个最小的 k 使得 $d(u,k)$ 最大
$n \leq 10^6$

既然上面的方法不好做，那可以考虑换一个维护的信息
设 $dep[k]$ 是 k 在整棵树内的深度
对以 u 为根的子树内的所有节点 v 维护的信息是 $dep[u] - dep[v]$
对于同一颗子树实际上 $dep[u]$ 是不会变的, 所以对每个节点实际上直接维护 $dep[v]$ 就可以了
直接统计出现次数最多的 $dep[v]$

CF1009F

Dominant Indices

给你一个以 1 为根，n 个节点的无边权树，记 $d(u,x)$ 为 u 的子树中到 u 距离为 x 的节点数
对于树上的每个点 u, 你需要选择一个最小的 k 使得 $d(u,k)$ 最大
$n \leq 10^6$

既然上面的方法不好做，那可以考虑换一个维护的信息
设 $dep[k]$ 是 k 在整棵树内的深度
对以 u 为根的子树内的所有节点 v 维护的信息是 $dep[u] - dep[v]$
对于同一颗子树实际上 $dep[u]$ 是不会变的，所以对每个节点实际上直接维护 $dep[v]$ 就可以了
直接统计出现次数最多的 $dep[v]$
参考代码

[NOIP 2016] 天天爱跑步

给你一颗 n 个节点的树，有 m 条路径 $s_i \to t_i$
对于每个点 j，问你有多少条路径的第 $w_j + 1$ 个点是 j
$n \leq 3 \times 10^5$

[NOIP 2016] 天天爱跑步

给你一颗 n 个节点的树，有 m 条路径 $s_i \to t_i$
对于每个点 j，问你有多少条路径的第 $w_j + 1$ 个点是 j
$n \leq 3 \times 10^5$

这题有不用线段树合并的方法，但是比较困难

[NOIP 2016] 天天爱跑步

给你一颗 n 个节点的树，有 m 条路径 $s_i \to t_i$
对于每个点 j，问你有多少条路径的第 $w_j + 1$ 个点是 j
$n \leq 3 \times 10^5$

显然要用 LCA 把路径拆成两个部分 $s_i \to LCA$ 和 $LCA \to t_i$
指定 1 为树根，对每个子树开一个线段树，维护这个子树内每个深度有多少个路径的端点
两种路径都可能给答案产生贡献，所以给 $dep[s_i], dep[t_i]$ 的位置 $+1$，LCA 被加了两次要 -1
这条路径不影响高于 LCA 的位置，$fa[LCA]$ 处还要 -1

[NOIP 2016] 天天爱跑步

给你一颗 n 个节点的树，有 m 条路径 $s_i \to t_i$
对于每个点 j，问你有多少条路径的第 $w_j + 1$ 个点是 j
$n \leq 3 \times 10^5$

考虑所有 $s_i \to LCA$ 的路径对答案的贡献，对每个点 u 统计有多少个路径端点深度为 $dep[u] + w_u$
考虑所有 $LCA \to t_i$ 的路径对答案的贡献，对每个点 u 统计有多少个路径端点深度为 $dep[u] - w_u$(从上向下走 w_u 步)
注意第二个式子可能是负数，但是动开权值线段树自带离散化，值域开 $[-n, n]$ 即可
其实就是用线段树合并维护树上差分

[NOIP 2016] 天天爱跑步

给你一颗 n 个节点的树，有 m 条路径 $s_i \to t_i$
对于每个点 j，问你有多少条路径的第 $w_j + 1$ 个点是 j
$n \leq 3 \times 10^5$

考虑所有 $s_i \to LCA$ 的路径对答案的贡献，对每个点 u 统计有多少个路径端点深度为 $dep[u] + w_u$
考虑所有 $LCA \to t_i$ 的路径对答案的贡献，对每个点 u 统计有多少个路径端点深度为 $dep[u] - w_u$(从上向下走 w_u 步)
注意第二个式子可能是负数，但是动开权值线段树自带离散化，值域开 $[-n, n]$ 即可
其实就是用线段树合并维护树上差分
代码见这里

[NOIP 2021] 棋局

给定 $n \times m$ 的棋盘，棋盘上有 4 种类型的边，分别代表不可通行、只能走一步、只能一直沿一个方向往前走和可以任意走

棋子有两种颜色 col_i 和等级 lv_i，棋子间可以吃子，规定只能吃颜色不同且等级不高于自己的棋子，且吃完子后不能继续向前走

同时规定每次走子时经过的边类型必须相同。初始棋盘是空的，有 q 次操作，每次往棋盘上放一个棋子，问这个棋子能走到多少个格子

多组测试数据

$2 \leq n, m \leq 10^5, 4 \leq n \times m \leq 2 \times 10^5, q \leq \min(10^5, n \times m), lv_i \leq q, col_i \in \{0, 1\}$

[NOIP 2021] 棋局

给定 $n \times m$ 的棋盘，棋盘上有 4 种类型的边，分别代表不可通行、只能走一步、只能一直沿一个方向往前走和可以任意走
棋子有两种颜色 col_i 和等级 lv_i，棋子间可以吃子，规定只能吃颜色不同且等级不高于自己的棋子，且吃完子后不能继续向前走
同时规定每次走子时经过的边类型必须相同。初始棋盘是空的，有 q 次操作，每次往棋盘上放一个棋子，问这个棋子能走到多少个格子
多组测试数据
$2 \le n, m \le 10^5, 4 \le n \times m \le 2 \times 10^5, q \le \min(10^5, n \times m), lv_i \le q, col_i \in \{0, 1\}$

我就是为了这碟醋才包的饺子.jpg
这题相当不建议各位写，代码实在是太长太困难了，看完分析就算了

[NOIP 2021] 棋局

给定 $n \times m$ 的棋盘，棋盘上有 4 种类型的边，分别代表不可通行、只能走一步、只能一直沿一个方向往前走和可以任意走
棋子有两种颜色 col_i 和等级 lv_i，棋子间可以吃子，规定只能吃颜色不同且等级不高于自己的棋子，且吃完子后不能继续向前走
同时规定每次走子时经过的边类型必须相同。初始棋盘是空的，有 q 次操作，每次往棋盘上放一个棋子，问这个棋子能走到多少个格子
多组测试数据
$2 \le n, m \le 10^5, 4 \le n \times m \le 2 \times 10^5, q \le \min(10^5, n \times m), lv_i \le q, col_i \in \{0, 1\}$

根据吃完子后不能继续向前走这一条规定我们知道，落子相当于分割棋盘
众所周知维护动态图连通性的做法都相当麻烦，万幸的是这题只用维护删边，那把询问操作离线下来倒着做变成每次加边
这可以用并查集简单维护

[NOIP 2021] 棋局

给定 $n \times m$ 的棋盘，棋盘上有 4 种类型的边，分别代表不可通行、只能走一步、只能一直沿一个方向往前走和可以任意走
棋子有两种颜色 col_i 和等级 lv_i，棋子间可以吃子，规定只能吃颜色不同且等级不高于自己的棋子，且吃完子后不能继续向前走
同时规定每次走子时经过的边类型必须相同。初始棋盘是空的，有 q 次操作，每次往棋盘上放一个棋子，问这个棋子能走到多少个格子
多组测试数据
$2 \leq n, m \leq 10^5, 4 \leq n \times m \leq 2 \times 10^5, q \leq \min(10^5, n \times m), lv_i \leq q, col_i \in \{0, 1\}$

接着分别考虑四类边
不可通行的直接当不存在，后面三种类型的边放到同一个数据结构里面维护不好做
走边的方法差别太大，考虑分别开一些数据结构维护

[NOIP 2021] 棋局

给定 $n \times m$ 的棋盘，棋盘上有 4 种类型的边，分别代表不可通行、只能走一步、只能一直沿一个方向往前走和可以任意走
棋子有两种颜色 col_i 和等级 lv_i，棋子间可以吃子，规定只能吃颜色不同且等级不高于自己的棋子，且吃完子后不能继续向前走
同时规定每次走子时经过的边类型必须相同。初始棋盘是空的，有 q 次操作，每次往棋盘上放一个棋子，问这个棋子能走到多少个格子
多组测试数据
$2 \le n, m \le 10^5, 4 \le n \times m \le 2 \times 10^5, q \le \min(10^5, n \times m), lv_i \le q, col_i \in \{0, 1\}$

对于一类边的情况比较好处理，只要维护四个方向上能走到的点有没有已经走到的点
对于二类边，维护这个点四个方向上连续一段能通过二类边走到的点，这可以横纵各开一个并查集维护，然后维护一个编号的最小值和最大值
对于三类边，维护其组成的联通块，同时维护联通块的大小

[NOIP 2021] 棋局

给定 $n \times m$ 的棋盘，棋盘上有 4 种类型的边，分别代表不可通行、只能走一步、只能一直沿一个方向往前走和可以任意走
棋子有两种颜色 col_i 和等级 lv_i，棋子间可以吃子，规定只能吃颜色不同且等级不高于自己的棋子，且吃完子后不能继续向前走
同时规定每次走子时经过的边类型必须相同。初始棋盘是空的，有 q 次操作，每次往棋盘上放一个棋子，问这个棋子能走到多少个格子
多组测试数据
$2 \le n, m \le 10^5, 4 \le n \times m \le 2 \times 10^5, q \le \min(10^5, n \times m), lv_i \le q, col_i \in \{0, 1\}$

上面的维护方法会导致有些点会被算两次以上，考虑去重
对于第一类边能走到的点去重是比较好做的，如果在第二类或者第三类里面统计了不算进去就行
第二类边和第三类边之间的去重需要注意到一个性质：第二类边组成的联通块里横坐标或者纵坐标总是连续的
把第二类边之间的所有点分别按横坐标和纵坐标排序，里面编号连续的就是一组二类边联通块了

[NOIP 2021] 棋局

给定 $n \times m$ 的棋盘，棋盘上有 4 种类型的边，分别代表不可通行、只能走一步、只能一直沿一个方向往前走和可以任意走

棋子有两种颜色 col_i 和等级 lv_i，棋子间可以吃子，规定只能吃颜色不同且等级不高于自己的棋子，且吃完子后不能继续向前走

同时规定每次走子时经过的边类型必须相同。初始棋盘是空的，有 q 次操作，每次往棋盘上放一个棋子，问这个棋子能走到多少个格子

多组测试数据

$2 \leq n, m \leq 10^5, 4 \leq n \times m \leq 2 \times 10^5, q \leq \min(10^5, n \times m), lv_i \leq q, col_i \in \{0, 1\}$

去重操作实际上在三类边组成的联通块中查询上面两种排序方式中编号连续的点的数量

在维护三类边组成联通块的同时维护两棵线段树，分别存两种排序方式里的编号，集合合并的时候线段树合并

[NOIP 2021] 棋局

给定 $n \times m$ 的棋盘，棋盘上有 4 种类型的边，分别代表不可通行、只能走一步、只能一直沿一个方向往前走和可以任意走

棋子有两种颜色 col_i 和等级 lv_i，棋子间可以吃子，规定只能吃颜色不同且等级不高于自己的棋子，且吃完子后不能继续向前走

同时规定每次走子时经过的边类型必须相同。初始棋盘是空的，有 q 次操作，每次往棋盘上放一个棋子，问这个棋子能走到多少个格子

多组测试数据

$2 \le n, m \le 10^5, 4 \le n \times m \le 2 \times 10^5, q \le \min(10^5, n \times m), lv_i \le q, col_i \in \{0, 1\}$

接下来考虑吃子的情况，发现通过一二类边能吃的子每个方向至多一个，但是通过三类边能吃的子可以有很多个

考虑三类边，对每个三类边联通块维护可以直达的棋子的集合，对黑白分别维护

这里需要去重第二次，注意合并集合的时候一个棋子可能同时属于两个集合，这里可以先把等级 lv 离散化方便去重

查询就是查询和当前棋子颜色相反的集合之中等级小于等于其的有多少个

一二类边能到达的点在线段树里面查是否存在即可

[NOIP 2021] 棋局

给定 $n \times m$ 的棋盘，棋盘上有 4 种类型的边，分别代表不可通行、只能走一步、只能一直沿一个方向往前走和可以任意走

棋子有两种颜色 col_i 和等级 lv_i，棋子间可以吃子，规定只能吃颜色不同且等级不高于自己的棋子，且吃完子后不能继续向前走

同时规定每次走子时经过的边类型必须相同。初始棋盘是空的，有 q 次操作，每次往棋盘上放一个棋子，问这个棋子能走到多少个格子

多组测试数据

$2 \le n, m \le 10^5, 4 \le n \times m \le 2 \times 10^5, q \le \min(10^5, n \times m), lv_i \le q, col_i \in \{0, 1\}$

注意给棋盘压维，$(10^5)^2$ 显然是开不下的

代码不放在这里了，有需要的人可以找我要

线段树合并优化 DP

线段树能通过维护区间/集合的信息来加速 DP 的递推过程
如果其中涉及到集合信息的情况就需要再加上线段树合并来优化 DP 了

CF1455G

Forbidden Value

已知初始值 $x = 0$，给定下面 2 种命令：
1. set y v，令 $x = y$，或花费 v 元钱删除该命令；
2. if y ...end，如果 $x == y$，执行 if...end 中的命令，否则跳过该 if...end。
你需要使用最少的花费，使得无论运行到哪里，都有 $x \neq s$。

CF1455G

Forbidden Value

已知初始值 $x = 0$，给定下面 2 种命令：
1. set y v，令 $x = y$，或花费 v 元钱删除该命令；
2. if y ...end，如果 $x == y$，执行 if...end 中的命令，否则跳过该 if...end。
你需要使用最少的花费，使得无论运行到哪里，都有 $x \neq s$。

众所周知，对于这种序列的区间嵌套的形式实际上是一个树形的结构
把这个序列挪到树上进行树形 DP
考虑一个 $if\ x_0$ 内部的解，也就是说一棵子树内的解
容易想到设 $dp_{u,i}$ 为从在 u 为根的子树中取到 $x = i$ 的最小代价

CF1455G

Forbidden Value

已知初始值 $x = 0$，给定下面 2 种命令：
1. set y v，令 $x = y$，或花费 v 元钱删除该命令；
2. if y ...end，如果 $x == y$，执行 if...end 中的命令，否则跳过该 if...end。
你需要使用最少的花费，使得无论运行到哪里，都有 $x \neq s$。

对于 set
$f[u][y] = \min\{f[u][i]\}$
$f[u][i] = f[u][i] + v$

CF1455G

Forbidden Value

已知初始值 $x = 0$，**给定下面** 2 **种命令**：
1. set y v，**令** $x = y$，**或花费** v **元钱删除该命令**；
2. if y ...end，**如果** $x == y$，**执行** if...end **中的命令，否则跳过该** if...end。
你需要使用最少的花费，使得无论运行到哪里，都有 $x \neq s$。

对于 set
$f[u][y] = \min\{f[u][i]\}$
$f[u][i] = f[u][i] + v$
对于 if...end
$f[u][y] = f[u][y] + f[v][y]$
$f[u][i] = \min(f[u][i], f[v][y] + f[u][i]), i \neq y$

代码见这里

CF1455G

Forbidden Value

已知初始值 $x = 0$，给定下面 2 种命令：
1. set y v，令 $x = y$，或花费 v 元钱删除该命令；
2. if y ...end，如果 $x == y$，执行 if...end 中的命令，否则跳过该 if...end。
你需要使用最少的花费，使得无论运行到哪里，都有 $x \neq s$。

对于 set
$f[u][y] = \min\{f[u][i]\}$
$f[u][i] = f[u][i] + v$
对于 if...end
$f[u][y] = f[u][y] + f[v][y]$
$f[u][i] = \min(f[u][i], f[v][y] + f[u][i]), i \neq y$
暴力转移 $O(n^2)$，但是注意到只有子树内 set 过的值才会有用
那实际上转移的时候只用合并有用的数集，然后在里面查询就可以了
所以对每个节点 u 开一个线段树维护，转移的时候就是单点修改，全局加，单点查询
复杂度就是 $O(n \log n)$

[PKUWC 2018] Minimax

小 C 有一棵 n 个结点的有根树，根是 1 号结点，且每个结点最多有两个子结点。
定义结点 x 的权值为：
1. 若 x 没有子结点，那么它的权值会在输入里给出，保证这类点中每个结点的权值互不相同。
2. 若 x 有子结点，那么它的权值有 p_x 的概率是它的子结点的权值的最大值，有 $1-p_x$ 的概率是它的子结点的权值的最小值。
现在小 C 想知道，假设 1 号结点的权值有 m 种可能性，权值第 i 小的可能性的权值是 V_i，它的概率为 $D_i(D_i \geq 0)$，求：

$$\sum_{i=1}^{m} i \cdot V_i \cdot D_i^2 \tag{1}$$

你需要输出答案对 998244353 取模的值。

LG P5298

比较套路的，设 $f_{u,w}$ 表示节点 u 的权值为 w 小的概率
设 ls 是左子节点，rs 是右子节点

LG P5298

比较套路的，设 $f_{u,w}$ 表示节点 u 的权值为 w 小的概率
设 ls 是左子节点，rs 是右子节点
分别考虑左右子节点的贡献
令可能的权值个数为 m，对于左子节点的每个 w，转移上去有两种可能：

LG P5298

比较套路的，设 $f_{u,w}$ 表示节点 u 的权值为 w 小的概率
设 ls 是左子节点，rs 是右子节点
分别考虑左右子节点的贡献
令可能的权值个数为 m，对于左子节点的每个 w，转移上去有两种可能：
1. 和右子节点权值取 min

$$p_u \sum_{k=1}^{w-1} f_{rs,k}$$

LG P5298

比较套路的，设 $f_{u,w}$ 表示节点 u 的权值为 w 小的概率
设 ls 是左子节点，rs 是右子节点
分别考虑左右子节点的贡献
令可能的权值个数为 m，对于左子节点的每个 w，转移上去有两种可能：
1. 和右子节点权值取 min

$$p_u \sum_{k=1}^{w-1} f_{rs,k}$$

2. 和右子节点权值取 max

$$(1-p_u) \sum_{k=w+1}^{m} f_{rs,k}$$

LG P5298

比较套路的，设 $f_{u,w}$ 表示节点 u 的权值为 w 小的概率
设 ls 是左子节点，rs 是右子节点
分别考虑左右子节点的贡献
令可能的权值个数为 m，对于左子节点的每个 w，转移上去有两种可能：
1. 和右子节点权值取 min

$$p_u \sum_{k=1}^{w-1} f_{rs,k}$$

2. 和右子节点权值取 max

$$(1-p_u) \sum_{k=w+1}^{m} f_{rs,k}$$

然后再乘上自己的的概率，贡献最终是

$$f_{ls,w} \times (p_u \sum_{k=1}^{w-1} f_{rs,k} + (1-p_u) \sum_{k=w+1}^{m} f_{rs,k})$$

对右子节点也是同样的道理

对右子节点也是同样的道理
最终就有式子:

$$f_{u,w} = f_{ls,w} \times (p_u \sum_{k=1}^{w-1} f_{rs,k} + (1-p_u) \sum_{k=w+1}^{m} f_{rs,k}) + f_{rs,w} \times (p_u \sum_{k=1}^{w-1} f_{ls,k} + (1-p_u) \sum_{k=w+1}^{m} f_{ls,k})$$

对右子节点也是同样的道理
最终就有式子:

$$f_{u,w}=f_{ls,w}\times(p_u\sum_{k=1}^{w-1}f_{rs,k}+(1-p_u)\sum_{k=w+1}^{m}f_{rs,k})+f_{rs,w}\times(p_u\sum_{k=1}^{w-1}f_{ls,k}+(1-p_u)\sum_{k=w+1}^{m}f_{ls,k})$$

这个东西既有前缀和又有后缀和，还要支持全局乘，那考虑用线段树维护这个东西
然后就线段树合并维护就做完了
需要代码的话可以找我要

习题

以下题目写这份东西的人都没有做几道，看着做就行

简单题

- P4566 雨天的尾巴
- P8496 [NOI2022] 众数

比较难写/毒瘤的题目

- CF246E Blood Cousins Return
- P5612 [Ynoi2013] Ynoi

下面的题目由 mod998244353 推荐

- CF1777F Comfortably Numb
- P6773 [NOI 2020] 命运