高级?数据结构

dy0607

雅礼中学

September 30, 2018

Preface

NOIp中其实并不会用到很多高级数据结构，省选中常用的数据结构也只有线段树/bit/pq/set。

Preface

NOIp中其实并不会用到很多高级数据结构，省选中常用的数据结构也只有线段树/bit/pq/set。

因此这里涉及的东西不多，一些近几年已不太常用的数据结构（例如不能用STL或者pbds替代的平衡树，划分树，块状链表，也许还有LCT）就没有找题了。

Preface

NOIp中其实并不会用到很多高级数据结构，省选中常用的数据结构也只有线段树/bit/pq/set。

因此这里涉及的东西不多，一些近几年已不太常用的数据结构（例如不能用STL或者pbds替代的平衡树，划分树，块状链表，也许还有LCT）就没有找题了。

同样的，题目难度与顺序无关。

CF 443 Div1 C

Berland要举行n次锦标赛，第一次只有一个人，之后每一次会新加入一个人。锦标赛中有k种运动项目，每个人在这k种项目上都有一个能力值，每次会选择任意两个还未被淘汰的人进行某个项目的比赛，能力值高的人胜出，输的人被淘汰，直至只剩下一个人成为冠军。

给出每个人每个项目的能力值，保证它们两两不同，求每次锦标赛有多少人可能成为冠军。

$n \leq 5 \times 10^4, k \leq 10$

只要选手a在某个项目上比选手b强，a就可以淘汰b，我们可以连一条a到b的边。

只要选手a在某个项目上比选手b强，a就可以淘汰b，我们可以连一条a到b的边。

对整个图求强连通分量，缩点后一定会形成一个竞赛图，拓扑序最靠前的分量中的所有点都可能成为冠军。

只要选手a在某个项目上比选手b强，a就可以淘汰b，我们可以连一条a到b的边。

对整个图求强连通分量，缩点后一定会形成一个竞赛图，拓扑序最靠前的分量中的所有点都可能成为冠军。

每加入一个点时，我们可能需要合并拓扑序在一段区间内强连通分量。用set按拓扑序维护每个强连通分量，对每个分量记录它的大小，以及在每个项目上的最大和最小能力值，就可以直接在set上二分找到需要合并的区间。

最多只会合并$n-1$次，$O(nk\log n)$

AGC-018 C

有$X+Y+Z$个人，第i个人有A_i个金币，B_i个银币，C_i个铜币。

选出X个人获得其金币，选出Y个人获得其银币，选出Z个人获得其铜币，在不重复选某个人的前提下，最大化获得的币的总数。

$X+Y+Z \leq 10^5$

不妨先令$A_i = A_i - C_i, B_i = B_i - C_i$，问题变为选出$X$个人获得其金币，选出$Y$个人获得其银币，再将答案加上$\sum C_i$.

不妨先令$A_i = A_i - C_i, B_i = B_i - C_i$，问题变为选出$X$个人获得其金币，选出$Y$个人获得其银币，再将答案加上$\sum C_i$.

按A_i从大到小排序，枚举选出的X个人中A_i最小的人，显然这个人之前的人要么在选出的X个人中，要么在选出的Y个人中。

不妨先令$A_i = A_i - C_i, B_i = B_i - C_i$，问题变为选出$X$个人获得其金币，选出$Y$个人获得其银币，再将答案加上$\sum C_i$.

按A_i从大到小排序，枚举选出的X个人中A_i最小的人，显然这个人之前的人要么在选出的X个人中，要么在选出的Y个人中。

那么我们只要对每个位置$i, X \le i \le X + Y$计算两个信息：i之前$A_i - B_i$最大的X个人的$A_i - B_i$的和，i之后B_i最小的Z个人的B_i之和。

不妨先令$A_i = A_i - C_i, B_i = B_i - C_i$，问题变为选出$X$个人获得其金币，选出$Y$个人获得其银币，再将答案加上$\sum C_i$.

按A_i从大到小排序，枚举选出的X个人中A_i最小的人，显然这个人之前的人要么在选出的X个人中，要么在选出的Y个人中。

那么我们只要对每个位置$i, X \le i \le X + Y$计算两个信息：i之前$A_i - B_i$最大的X个人的$A_i - B_i$的和，i之后B_i最小的Z个人的B_i之和。

于是我们只需要从前往后扫一遍，用小根堆维护当前$A_i - B_i$最大的X个人，每加入一个人与堆顶比较；再从后往前算第二个信息即可。$O(n \log n)$

CF 493 Div1 E

有一个1到n的排列p，我们称一个子区间是好的，当且仅当这个子区间内的值构成了连续的一段(例如对于排列$\{1,3,2\}$，$[1,1]$, $[2,2]$, $[3,3]$, $[2,3]$, $[1,3]$是好的区间)。

q次询问，每次询问L,R, 求有多少$L \le l \le r \le R$，满足$[l,r]$是好的区间。

$n,q \le 1.2 \times 10^5, TL = 7s$

一段区间是好的，当且仅当$(Max - Min) - (r - l) = 0$.

一段区间是好的，当且仅当$(Max - Min) - (r - l) = 0$.

考虑从小到大枚举r，用线段树对每个l维护上式的值。维护两个栈，分别保存l在每段区间中时Max和Min的值，维护栈的同时在线段树上修改对应区间的值。

一段区间是好的，当且仅当$(Max - Min) - (r - l) = 0$.

考虑从小到大枚举r，用线段树对每个l维护上式的值。维护两个栈，分别保存l在每段区间中时Max和Min的值，维护栈的同时在线段树上修改对应区间的值。

线段树上需要维护最小值以及最小值数量，若最小值为0(实际上全局最小值一定是0)，则所有最小值的位置对于当前的r都是合法的左端点。于是线段树上存一个"所有最小值的位置的答案"的加标记，并维护区间答案之和。

一段区间是好的，当且仅当$(Max - Min) - (r - l) = 0$.

考虑从小到大枚举r，用线段树对每个l维护上式的值。维护两个栈，分别保存l在每段区间中时Max和Min的值，维护栈的同时在线段树上修改对应区间的值。

线段树上需要维护最小值以及最小值数量，若最小值为0(实际上全局最小值一定是0)，则所有最小值的位置对于当前的r都是合法的左端点。于是线段树上存一个"所有最小值的位置的答案"的加标记，并维护区间答案之和。

将询问离线按R排序，我们就可以在R处查询区间答案之和来回答询问。$O(n \log n)$.

Avito Code Challenge 2018 G

有n个初始为空的'魔法'可重集，向一个'可重集'加入元素时，若该元素未出现过，则将其加入；否则该可重集中所有元素的个数都会翻倍。例如将2加入$\{1,3\}$会得到$\{1,2,3\}$，将2加入$\{1,2,3,3\}$会得到$\{1,1,2,2,3,3,3,3\}$.

q次操作，每次操作要么向一个区间内的所有可重集加入某个元素，要么询问一个区间内可重集的大小之和。

$n,q \leq 2 \times 10^5$

翻倍的性质并不会影响元素是否出现，那么我们开n个set或者线段树维护某个元素在哪些区间是出现过的。

翻倍的性质并不会影响元素是否出现，那么我们开n个set或者线段树维护某个元素在哪些区间是出现过的。

加入元素时，对于出现过这个元素的区间，实际上就是进行区间乘法；没出现过则是区间加法。线段树维护加标记和乘标记以及区间和即可。

翻倍的性质并不会影响元素是否出现，那么我们开n个set或者线段树维护某个元素在哪些区间是出现过的。

加入元素时，对于出现过这个元素的区间，实际上就是进行区间乘法；没出现过则是区间加法。线段树维护加标记和乘标记以及区间和即可。

$$O(q \log n)$$

Codefest 2017 F

有10^5个初始为空的集合。对于一个集合，若其中没有元素x满足$x > 0$，或没有元素x满足$x < 0$，则定义其优美度为0;否则其优美度等于最小的正的元素减去其最大的负的元素。

q次操作，每次操作要么向一个区间内的集合加入某个元素，要么询问一个区间内集合的优美度之和。

$q \leq 5 \times 10^4, TL = 4s$

对于正的和负的元素分开考虑。如果每个集合都有正的和负的元素，那么这道题就是要支持区间取min以及区间求和。

对于正的和负的元素分开考虑。如果每个集合都有正的和负的元素，那么这道题就是要支持区间取min以及区间求和。

这可以用吉司机线段树来解决，也就是每个位置维护最大值Max和次大值sec，以及最大值数量和一个取min的标记。如果现在要对x取min，分情况讨论：

对于正的和负的元素分开考虑。如果每个集合都有正的和负的元素，那么这道题就是要支持区间取min以及区间求和。

这可以用吉司机线段树来解决，也就是每个位置维护最大值Max和次大值sec，以及最大值数量和一个取min的标记。如果现在要对x取min，分情况讨论：

- 若$x \geq Max$，直接忽略该操作。
- 若$sec < x < Max$，更新最大值，根据记录的最大值数量快速算出新的区间和。
- 若$sec \leq x$，直接暴力下传。可用势能分析证明暴力下传次数为$O(q \log n)$级别。

对于正的和负的元素分开考虑。如果每个集合都有正的和负的元素，那么这道题就是要支持区间取min以及区间求和。

这可以用吉司机线段树来解决，也就是每个位置维护最大值Max和次大值sec，以及最大值数量和一个取min的标记。如果现在要对x取min，分情况讨论：

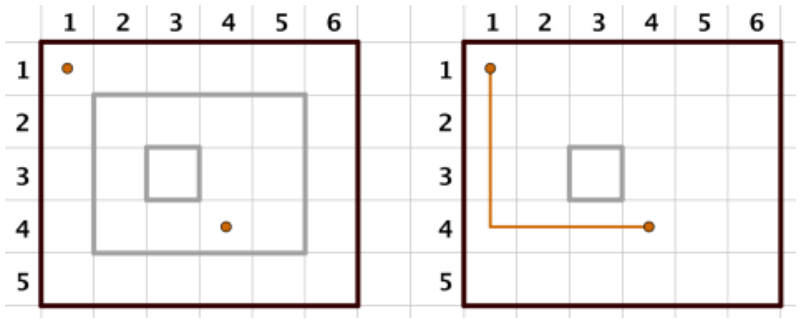
- 若$x \geq Max$，直接忽略该操作。
- 若$sec < x < Max$，更新最大值，根据记录的最大值数量快速算出新的区间和。
- 若$sec \leq x$，直接暴力下传。可用势能分析证明暴力下传次数为$O(q \log n)$级别。

回到本题，唯一要注意的是如果一个集合还没有正的或负的元素就不要计入它的信息，当一个集合第一次加入正的或负的元素时，可以暴力处理。$O((n+q)\log n)$

CF 439 Div2 E

有一个$n \times m$的网格，三种操作：

- 添加一个矩形围栏，保证所有矩形的边界没有公共点。
- 删去一个存在的矩形围栏。
- 询问从网格中的某个位置是否可以走到另一个位置（不能跨过围栏）。



$n, m \leq 2500, q \leq 10^5$

本题有多种做法，这里介绍一种比较简单的。

考虑暴力，我们只需要check是否存在一个矩形，起点和终点分别在这个矩形的内部和外部。

本题有多种做法，这里介绍一种比较简单的。

考虑暴力，我们只需要check是否存在一个矩形，起点和终点分别在这个矩形的内部和外部。

于是两个位置互相可达当且仅当它们覆盖它们的矩形是相同的。

本题有多种做法，这里介绍一种比较简单的。

考虑暴力，我们只需要check是否存在一个矩形，起点和终点分别在这个矩形的内部和外部。

于是两个位置互相可达当且仅当它们覆盖它们的矩形是相同的。

考虑哈希，对每个矩形随机一个2^{64}内的权值，维护覆盖每个位置矩形的异或和。异或和相同便视为覆盖它们矩形相同。

本题有多种做法，这里介绍一种比较简单的。

考虑暴力，我们只需要check是否存在一个矩形，起点和终点分别在这个矩形的内部和外部。

于是两个位置互相可达当且仅当它们覆盖它们的矩形是相同的。

考虑哈希，对每个矩形随机一个2^{64}内的权值，维护覆盖每个位置矩形的异或和。异或和相同便视为覆盖它们矩形相同。

用二维树状数组维护即可，$O(q \log n \log m)$.

CF 371 Div1 D

给出一个$n \times m$的0/1矩阵，q次询问，每次询问一个矩形内最大的全为0的正方形边长。

CF 371 Div1 D

给出一个$n \times m$的0/1矩阵，q次询问，每次询问一个矩形内最大的全为0的正方形边长。

$n, m \leq 10^3, q \leq 10^6$

首先求出以每个点为右下角的最大全0正方形的边长：

$$f[i][j] = min\{f[i-1][j], f[i][j-1], f[i-1][j-1]\} + 1$$

首先求出以每个点为右下角的最大全0正方形的边长：

$$f[i][j] = min\{f[i-1][j], f[i][j-1], f[i-1][j-1]\} + 1$$

二分答案k，那么我们每次只需要判断一个矩形内是否存在$f[i][j] \geq k$。

首先求出以每个点为右下角的最大全0正方形的边长：

$$f[i][j] = min\{f[i-1][j], f[i][j-1], f[i-1][j-1]\} + 1$$

二分答案k，那么我们每次只需要判断一个矩形内是否存在$f[i][j] \geq k$。

用二维RMQ快速处理最大值，$O(nm \log n \log m + q \log n)$

SDOI 2017 树点涂色

Bob有一棵n个点的有根树，其中1号点是根节点。Bob在每个点上涂了颜色，并且每个点上的颜色不同。

定义一条路径的权值是：这条路径上的点（包括起点和终点）共有多少种不同的颜色。

Bob可能会进行q次操作，共有三种：

- 把点x到根节点的路径上所有的点染上一种没有用过的新颜色。
- 求x到y的路径的权值。
- 在以x为根的子树中选择一个点，使得这个点到根节点的路径权值最大，求最大权值。

$n, q \leq 10^5$

首先，由于每次都用的是一种新的颜色，每种颜色的节点一定分别构成了从链底端到某个祖先的一条链。

首先，由于每次都用的是一种新的颜色，每种颜色的节点一定分别构成了从链底端到某个祖先的一条链。

对每个节点v维护$f(v)$表示v到根的路径上的颜色数，路径权值就是$f(x)+f(y)-2f(lca)+1$。

首先，由于每次都用的是一种新的颜色，每种颜色的节点一定分别构成了从链底端到某个祖先的一条链。

对每个节点v维护$f(v)$表示v到根的路径上的颜色数，路径权值就是$f(x)+f(y)-2f(lca)+1$。

考虑怎么在染色过程中维护f，对x到根的路径上的每个颜色段分开考虑，可以发现每次只需要将两个子树的f进行加减。同时还需要用线段树维护颜色。

首先，由于每次都用的是一种新的颜色，每种颜色的节点一定分别构成了从链底端到某个祖先的一条链。

对每个节点v维护$f(v)$表示v到根的路径上的颜色数，路径权值就是$f(x)+f(y)-2f(lca)+1$。

考虑怎么在染色过程中维护f，对x到根的路径上的每个颜色段分开考虑，可以发现每次只需要将两个子树的f进行加减。同时还需要用线段树维护颜色。

这样复杂度为$O(\log n\times$染色操作中的$\sum f(x))$，可证明其复杂度为$O(q\log^2 n)$

首先，由于每次都用的是一种新的颜色，每种颜色的节点一定分别构成了从链底端到某个祖先的一条链。

对每个节点v维护$f(v)$表示v到根的路径上的颜色数，路径权值就是$f(x)+f(y)-2f(lca)+1$。

考虑怎么在染色过程中维护f，对x到根的路径上的每个颜色段分开考虑，可以发现每次只需要将两个子树的f进行加减。同时还需要用线段树维护颜色。

这样复杂度为$O(\log n\times$染色操作中的$\sum f(x))$，可证明其复杂度为$O(q\log^2 n)$???

注意到这里的染色操作实际上与LCT的access操作类似，只需要将颜色相同的链视为实链. 我们只需要证明LCT中一条虚边变为实边的次数是$O(q \log n)$的。

注意到这里的染色操作实际上与LCT的access操作类似，只需要将颜色相同的链视为实链. 我们只需要证明LCT中一条虚边变为实边的次数是$O(q \log n)$的。

我们知道任意一个点到根的路径上，轻儿子的个数是$O(\log n)$的，那么我们每次最多将$O(\log n)$个轻儿子由虚边变为实边，最多将$O(\log n)$个重儿子由实边变为虚边。

这样我们将重儿子虚边变为实边的总次数也是$O(n + q \log n)$的，所以复杂度是对的。

CF 483 Div1 E

一棵n个点的树，树上有m条双向的公交线路，每条公交线路都在两个节点之间沿最短路径往返。

q次询问从一个点要到达另一个点，在只坐公交的情况下，至少需要坐几辆公交车；或者判断无法只坐公交到达。

$n, m, q \leq 2 \times 10^5$

对于每个点，先预处理出从这个点坐一次公交车能最远到达哪个祖先。对于一条公交线路(u, v)，将lca的信息挂在u, v上，dfs一遍向上更新信息即可。

对于每个点，先预处理出从这个点坐一次公交车能最远到达哪个祖先。对于一条公交线路(u,v)，将lca的信息挂在u,v上，dfs一遍向上更新信息即可。

通过倍增算出从某个点坐2^k次最远能到达哪个祖先。这样对于一条路径(u,v)，我们就能快速算出u,v走到lca分别至少需要多少辆车，假设答案分别为a,b，那么询问的答案要么是$a+b$，要么是$a+b-1$。

对于每个点，先预处理出从这个点坐一次公交车能最远到达哪个祖先。对于一条公交线路(u,v)，将lca的信息挂在u,v上，dfs一遍向上更新信息即可。

通过倍增算出从某个点坐2^k次最远能到达哪个祖先。这样对于一条路径(u,v)，我们就能快速算出u,v走到lca分别至少需要多少辆车，假设答案分别为a,b，那么询问的答案要么是$a+b$，要么是$a+b-1$。

判断的方法也很简单，先算出u向上坐$a-1$次，v向上坐$b-1$次，最远能到达哪个节点。若存在一条线路经过这两个节点，则答案为$a+b-1$。也就是判断是否有一条线路的两端分别在这两个节点的子树中。

对于每个点，先预处理出从这个点坐一次公交车能最远到达哪个祖先。对于一条公交线路(u,v)，将lca的信息挂在u,v上，dfs一遍向上更新信息即可。

通过倍增算出从某个点坐2^k次最远能到达哪个祖先。这样对于一条路径(u,v)，我们就能快速算出u,v走到lca分别至少需要多少辆车，假设答案分别为a,b，那么询问的答案要么是$a+b$，要么是$a+b-1$。

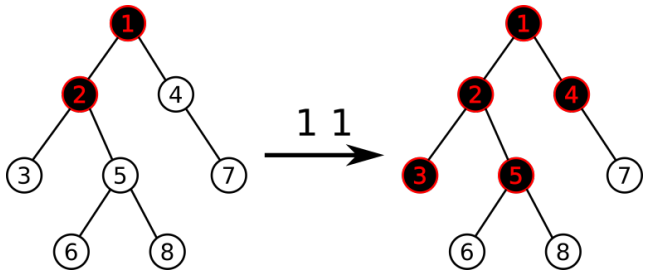
判断的方法也很简单，先算出u向上坐$a-1$次，v向上坐$b-1$次，最远能到达哪个节点。若存在一条线路经过这两个节点，则答案为$a+b-1$。也就是判断是否有一条线路的两端分别在这两个节点的子树中。

于是变成了一个二维数点问题，$O((n+m+q)\log n)$。

CF 502 Div1+Div2 G

一棵n个点的树，一开始所有点都是白色，接下来进行q次操作，有三种类型：

- 1 u 若u为白色，则将其染成黑色；否则对所有儿子做此操作；
- 2 u 将u的子树的所有点染白；
- 3 u 询问节点u的颜色。



$n, q \leq 10^5$

这题也有多种做法，这里介绍一种基于动态dp思想的。

暂且不管2操作。设$f(v)$表示在节点v上进行的1操作数量，$c(v)$表示直接在节点v上进行的（不是从父亲传下来的）1操作数量. 那么

$$f(v) = \max\{f(fa[v]) - 1, 0\} + c(v)$$

当$f(v) > 0$时，v为黑色，否则为白色。

这题也有多种做法，这里介绍一种基于动态dp思想的。

暂且不管2操作。设$f(v)$表示在节点v上进行的1操作数量，$c(v)$表示直接在节点v上进行的（不是从父亲传下来的）1操作数量. 那么

$$f(v) = \max\{f(fa[v]) - 1, 0\} + c(v)$$

当$f(v) > 0$时，v为黑色，否则为白色。

对于一条从上往下的链$v_0, v_1, v_2, ..., v_k$，通过合并信息我们一定可以将$f(v_k)$表示成$\max\{f(v_0)+a, b\}$的形式。

对于一条从上往下的链$v_0, v_1, v_2, ..., v_k$，通过合并信息我们一定可以将$f(v_k)$表示成$\max\{f(v_0)+a, b\}$的形式。

树链剖分，线段树上每个区间维护对应的a, b。对于询问操作，即可通过$O(\log n)$次重链上的线段树询问算出$f(v)$。

对于一条从上往下的链$v_0, v_1, v_2, ..., v_k$，通过合并信息我们一定可以将$f(v_k)$表示成$\max\{f(v_0)+a, b\}$的形式。

树链剖分，线段树上每个区间维护对应的a, b。对于询问操作，即可通过$O(\log n)$次重链上的线段树询问算出$f(v)$。

初始时对于所有线段树的叶子节点有$a=-1, b=0$。对于1操作，将对应点的a, b加1。

对于一条从上往下的链$v_0, v_1, v_2, ..., v_k$，通过合并信息我们一定可以将$f(v_k)$表示成$\max\{f(v_0)+a, b\}$的形式。

树链剖分，线段树上每个区间维护对应的a, b。对于询问操作，即可通过$O(\log n)$次重链上的线段树询问算出$f(v)$。

初始时对于所有线段树的叶子节点有$a=-1, b=0$。对于1操作，将对应点的a, b加1。

对于2操作，先询问$f(v)$，然后将其a, b减去$f(v)$，并将子树内的其它点的信息设为初始值，也就是强制将v变为白色，并抹掉所有之前子树内的操作。

$O(n \log^2 n)$

Thanks