树上问题

wx

T1

问题描述：

有一棵点数为 n 的树。

有 q 次询问，每次询问有多少个点到 a,b 距离相等。

$1 \le n,q \le 500000$

T1

问题分析：

- 设询问 a,b 两点直接的路长度为 d。
- 如果 d 为奇数，那么无解，d 为偶数有解。
- 考虑以下几种情况：
  - a,b 到 LCA(a,b) 的距离相等，那么 LCA(a,b) 中除了包含 a,b 的两棵子树上的点，其他点都满足。
  - a,b 到 LCA(a,b) 的距离不相等，假设 a 到 LCA(a,b) 距离更远，则中点到 a,b 的距离为 d/2。从 a 为起点往上走 d/2 的距离，设该点为 c，那么 c 为根的子树，除了 a 所在的子树，其他点都可以作为中点。
- 时间复杂度为 O(qlogn)。
- 来源：[CF519E]A and B and Lecture Rooms

T2

问题描述：

给你一个以 1 为根的有根树。

每回询问 k 个节点 $v_1, v_2 \dots v_k$。

求出是否有一条以根节点为一端的链使得询问的每个节点到此链的距离均 <= 1 。

只需输出可行性, 无需输出方案。

$n,k,\sum k \le 200000$

T2

问题分析：

- 设深度最大的点为 x，那么链的端点为 1,x 肯定满足情况。
- 考虑 x 和其他点 y 之间的关系。
- 可以发现 lca(x,y) 肯定在链上，符合条件的点 y 到 lca(x,y) 之间的距离不超过 1。
- 时间复杂度为 O(klog n)。
- 来源：[CF1328E]Tree Queries

T3

问题描述：

给定一棵 n 个节点的树，每个节点都有一枚硬币。

每次可以选择树上任意一个点 x，将 x 上的硬币移除，其他点的硬币往相邻且离 x 最近的点移动 。

问先手必胜还是后手必胜。

$n \le 200000$

T3

问题分析：

- 考虑一条链的情况，移除非端点，节点减少 2，移除端点，节点减少 1，直到节点为 0 结束。
- 对于树上的情况，考虑树上的一条最长链，最终需要将最长链变为 0 结束，即直径。
- 删除叶子节点，链上节点减少 1，删除非叶子节点，链上的节点减少 2。
- 最终只剩两个点的时候，只能减少 1，此时先手必败。

T3

问题分析：

- 设树的直径节点数为 d，如果 d mod 3 为 2 ，后手必胜，否则先手必胜。
- 当长度模 3 等于 0 时，先手先减 1 ，后手可以减 1 (那先手减 2)，后手减 2 (先手减 1) ，最后一定会剩下两个，先手获胜。
- 模 3 等于 1 同理，先手直接减 2。
- 模 3 等于 2 ，先手减 1 ，后手减 2，先手减 2 ，后手减 1 ，后手必胜。
- 所以最后我们只需要求出直径然后判断其长度就行了。
- 来源：[AGC033C]Removing Coins

T4

问题描述：

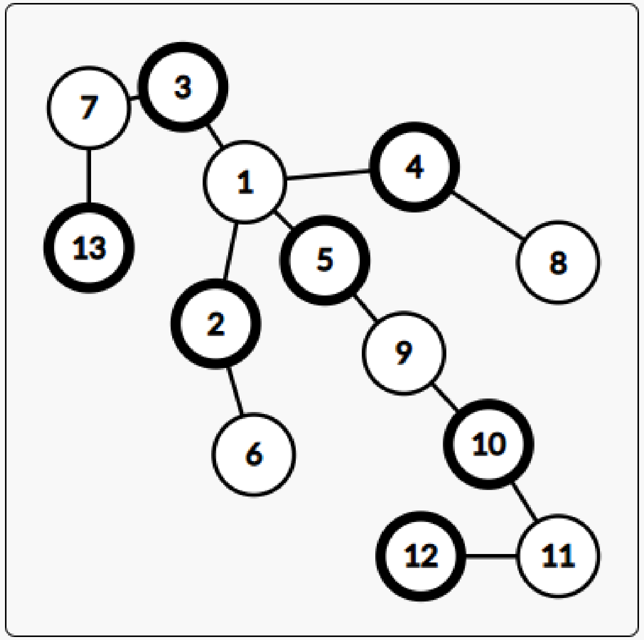
给一棵 n 个节点的树，每个点为黑色或白色，一次操作可以使一个相同颜色的连通块变成另一种颜色，求使整棵树变成一种颜色的最少操作数。

$n \le 200000$

T4

问题分析：

- 缩点，并查集维护。
- 缩点后建出一棵新的树，树上每条边的端点颜色均不同。
- 右图需要 4 次。
- 每次选择直接中间的点，往外一层一层扩散。
- 求解新树的直径 d，答案为直径的一半，即 (d+1)/2。
- 来源：[CF734E]Anton and Tree



T5

问题描述：

一棵树初始只有一个编号为 1 的根结点。

n 次操作，每次新增一个点作为 pi 的子结点，询问更新后有多少点可以作为树直径的端点。

$n \le 300000$

T5

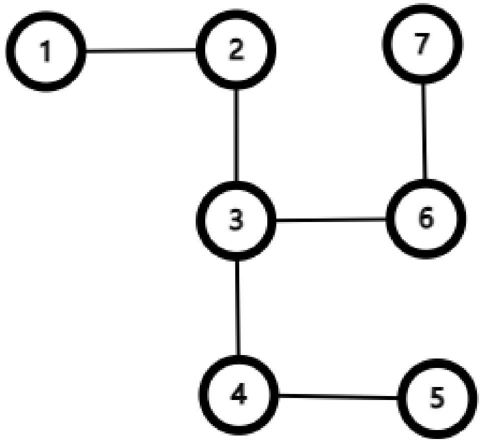
问题分析：

- 树的多条直径肯定存在公共点或者公共边。
- 维护两个点集 S_1,S_2，分别表示直径的端点，S_1 和 S_2 中任取一个点都构成一条直径。
- 设当前直接为 d，现加入一个点 x。
- 求解出 x 到 S_1 的距离 d_1，到 S_2 的距离 d_2。

T5

问题分析：

- 如果 $\max(d_1,d_2)> d$，更新直径。假设 $d_1> d_2$，那么 x 就是新的点集 S_2，注意，原 S_2 的点可能和 x 构成直径，需要遍历一遍。
- 例如：$S_1=\{1\}$，$S_2=\{5,7\}$，添加一个新点 x， 它父亲为 7， x 称为新的 S_2，但和 5 仍然可以构成直径。
- 如果 $\max(d_1,d_2)=d$，那么将 x 加入到相应点集中。
- 时间复杂度为 $O(n\log n)$。
- 来源：[CF842E]Nikita and game



T6

问题描述：

有一棵点数为 n 的树，以点 1 为根。

有 m 个操作，分为两种：

1 x a：将 x 为根的子树涂上 a 号颜色(颜色不会覆盖，全部共存)

2 x：询问 x 为根的子树颜色丰富度，表示为所有子树节点颜色种类数相加(每个点单独计算)

$n,m \le 100000$

T6

问题分析：

- 求解出 DFS 序列，每个点的颜色丰富度可以看作点的权值如果染色没有重复，每一次对 x 为根的子树染色，实际上就是 DFS 序的区间增加 1，询问就是区间和。
- 需要解决的问题是，如何处理颜色相同的染色？
- 一个点 x 染色 a，如果 x 的祖先节点染色成 a，那么这个操作是无效的。
- 一个点 x 染色 a，如果 x 的子树中已经有被染色成 a 的，就会重复。

T6

问题分析：

- 对每一个颜色都创建一个 set，存放相同染色的子树根节点的 in[] 值
- 对节点 x 进行染色 a：
  - 找到染色 a 的集合中是否有 x 的祖先节点：在集合中二分找，如果找到则跳过本次操作
  - 找到染色 a 的集合中是否有 x 的子节点：在集合中二分找，找到，则删除。
  - 最后对节点 x 进行染色 a操作，区间+1
- 询问是区间和，使用线段树维护。
- 来源：[USACO19DEC]Bessie's Snow Cow P

T7

问题描述：

地灵殿门口有一个小挂件，小挂件可视作一棵树，初始只有 1 个编号为 1 的节点。

接下来恋恋会做 $n-1$ 次操作，第 i 次操作添加一个编号为 $i+1$ 的节点，并在节点 $i+1$ 与 p_i 之间连边。

现在她想知道，在每一次操作之后，至少需要做多少次操作(每次操作可以添加一个点和一条边)，才能使这棵树具有两个重心。

由于恋恋忙着给挂件做装饰，所以你要帮她解决这个问题。

本题有 t 组数据。

$n, \sum n \le 500000$

问题分析：

- 树的重心有以下性质：
  - 树的重心如果不唯一，则至多有两个，且这两个重心相邻。
  - 在一棵树上添加或删除一个叶子，那么它的重心最多只移动一条边的距离，如果移动，会向重儿子所在子树移动。

T7

问题分析：

- 设当前有一个重心为 u ，u 的重儿子为 v 。
- 如果需要两个重心，那么贪心的在重儿子所在子树增加节点，使得最终两个重心为 u 和 v。因为在非重儿子的子树增加节点，此时重心是没有任何变化的，只有有了新的重儿子才会发生变化。
- 当有两个重心 u,v ，u 为根的子树(不包含 v 为根的子树)的大小 = v 所在子树(不包含 u 为根的子树)的大小。

问题分析：

- 如果只有一个重心 u 。
- 设当前节点数为 n ， 重儿子所在子树大小为 x，那么此时 u 为根的子树(不包含 v 为根的子树)的大小为 n-x ， 因此，还需要在 v 的子树添加 n-2x 个节点。当 n-2x=0 时，表示已经存在了两个重心。
- 因此，需要维护当前树的重心，以及重儿子的子树大小。
- 每进行一次操作，求解重心，时间复杂度太高，离线处理。
- 需要支持单点修改，新增一个点看作点权 +1 ，初始时点权为 0 ；查询子树大小。

T7

问题分析：

- 使用 dfs 序和树状数组维护
- 算法流程：
  - 1. 直接建立最终的树，所有节点点权初始值为 0 。
  - 2. 求解 dfs 序。
  - 3. 依次添加第 k=1~n 个点：
  - 记录当前重心为 s ， 重儿子为 son，分两种情况讨论：

T7

西南大学附属中学 High School Affiliated to Southwest University

问题分析：

- k 在 s 的子树：
- 1. 如果 k 所在子树的大小超过重儿子 son 子树大小，更新重儿子 son=k 所在子树根节点
- 2. 如果重儿子 son 子树大小超过总节点数的一半，更新重心，此时重儿子变成新树的重心 s=son，新的重儿子变为原来的重心 son=s。

T7

问题分析：

- k 不在 s 的子树：
- 1. k 添加到 s 父节点 fa[s] 所在子树，如果 fa[s] 所在子树的大小超过重儿子 son 子树大小，更新重儿子 son=fa[s]
- 2. 如果重儿子 son 子树大小超过总节点数的一半，更新重心，交换重心与重儿子。
- 输出答案。
- 维护子树大小用树状数组
- 时间复杂度为 O(nlogn)
- 来源：[CF1827D]Two Centroids

T8

问题描述：

给定 n 个点的树，起初每个节点都是黑色。

有 q 次操作，操作有三种：

1. 把下标为 [l,r] 的点染成白色；

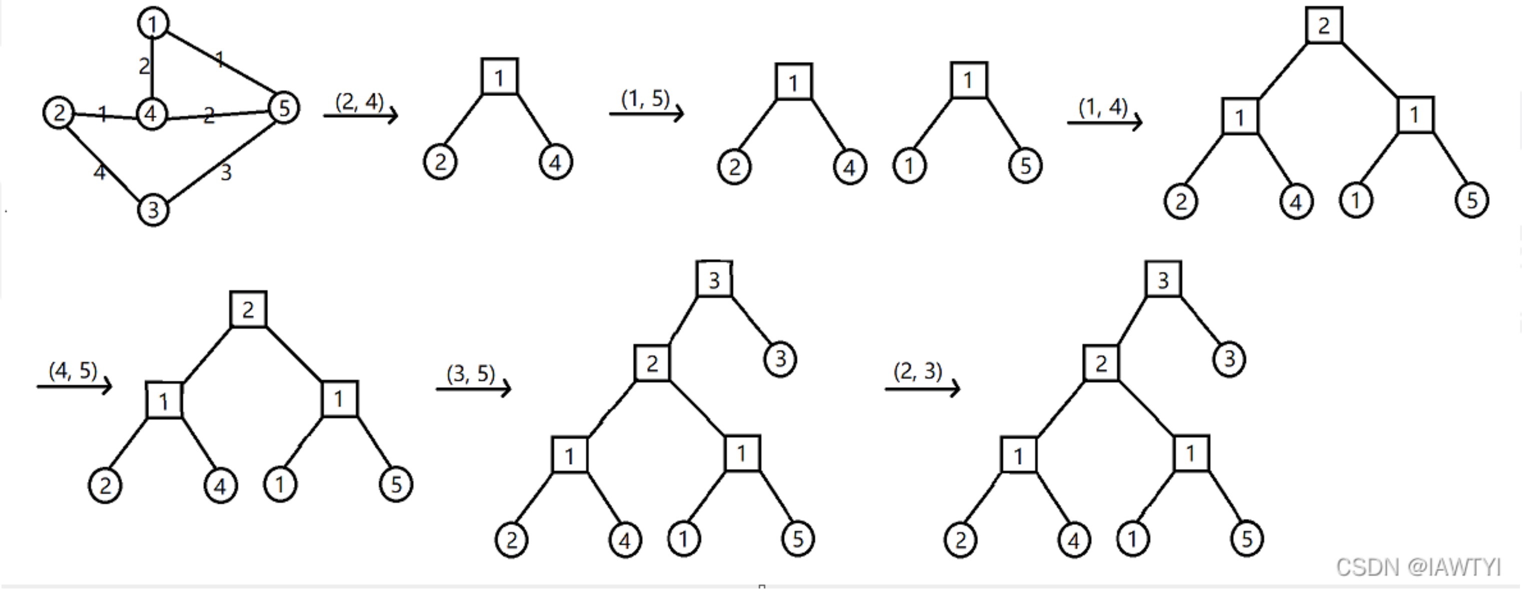
2. 把下标为 [l,r] 的点染成黑色；

3. 询问从节点 x 出发到达任意一个白色节点的简单路径上经过的边，最大可能的权值。不存在则输出-1。

$n,q \le 300000$

kruskal重构树

- 1. 将边权从小到大排个序
- 2. 按边权从小到大依次遍历每一条边，如果边连接两个节点 u , v，u,v 不在同一个并查集内，我们就新建一个节点 p，令 u , v 分别为 p 的左右儿子。这个点的点权就是连接 u , v 这条边的边权。这样构造出来的树我们称为kruskal 重构树。



kruskal重构树

- 性质：
  - 1. 二叉树
  - 2. 两点 x , y 的最近公共祖先 lca(x, y) 的点权就是原图中 x , y 满足最大边最小的路径上的边的最大值。
  - 3. 任一点的权值大于左右儿子的权值（也就是满足大根堆的性质）

T8

问题分析：

- 对于这种简单路径边权最大值，使用Kruskal 重构树。
- 对于Kruskal 重构树，原图中两点间所有简单路径最大边权的最小值=Kruskal 重构树上两点 LCA 的权值。
- 我们构建 Kruskal 重构树通过查询多点最近公共祖先维护路径边权最大值。

T8

问题分析：

- 如果知道若干白点的 LCA呢？
- 求解树的 dfs 序，dfs 序最小的点和最大的点的 LCA，就是所有白点的 LCA 。
- 现在需要维护区间白点 dfs 序最小值和最大值。
- 使用线段树维护。
- 时间复杂度为 O(qlog n)。
- 来源：[CF1628E]Groceries in Meteor Town

T9

问题描述：

给定一棵 n 个点的树，给第 i 个点染上颜色 c_i，其中，c_i 为 $[1,n]$ 的一个整数。

现在，对于每一种颜色 k，你要求出有多少条简单路径满足路径上至少有一个点的颜色为 k。

$n \le 200000$

T9

问题分析：

- 对于颜色 x，统计经过 x 的路径困难，考虑反过来，统计不经过 x 的路径的方案数，用总方案减去就是答案。
- 如何统计不经过 x 的路径方案数。
- 将 x 删除，剩下若干个连通块，连通块内的路径就是不经过 x 的路径。
- 设连通块的大小 k，则路径数为 $k(k+1)/2$（一个点的情况也包含）。

T9

问题分析：

- 对于当前颜色 x，遍历所有孩子节点，将孩子节点子树大小减去子树中颜色 x 的子树大小，这就是一个连通块，可以求解出不经过 x 的路径方案数。
- 通过 dfs 预处理出每个节点离它最近且不同子树的颜色相同的节点。
- 时间复杂度为 O(n)。
- 来源：[ABC163F]path pass i

T10

问题描述：

给定 n 个节点的树，1为根节点，每个节点是一个‘(’或者‘)’。

1到每一个节点x的路径是一个括号序列，记括号序列中子串是合法括号序列的数量为k_x。

计算出$(1\times k_1)$xor $(2\times k_2)$ xor $(3\times k_3)$ xor $\cdots$ xor $(n\times k_n)$。

$n \le 500000$

T10

问题分析：

- 考虑是一条链的情况。
- 如果快速求出一个括号序列的子串中的合法括号序列数量？
- ()()()，每一位的贡献是：0,1,0,2,0,3，总贡献是：0,1,1,3,3,6
- ())()，每一位的贡献是：0,1,0,0,1，总贡献是：0,1,1,1,2
- ()(())，每一位的贡献是：0,1,0,0,1,2，总贡献是：0,1,1,1,2,4
- 一个右括号如果能匹配一个左括号，假设这个左括号的前1位同样有一个已经匹配了的右括号，那么我们势必可以把当前的匹配和之前的匹配序列合并，当前的这个右括号的贡献值，其实就等于前面那个右括号的贡献值+1

T10

问题分析：

- lst[i]表示第i位括号的贡献，sum[i]表示c[1]~c[i]的总贡献。
- 从左往右遍历每个括号，使用栈存储左括号
- 如果c[i]为右括号，栈中有左括号s[top]，那么lst[i]=lst[s[top]-1]+1，左括号出栈。
- 如果c[i]为左括号就入栈
- sum[i]=sum[i-1]+lst[i]
- sum[i]就是c[1]~c[i]子串合法括号的数量。
- 时间复杂度为O(N)

T10

问题分析：

- 将链转换为树
- lst[i]=lst[s[top]-1]+1 转化为 lst[i]=lst[fa[s[top]]+1
- sum[i]=sum[i-1]+lst[i] 转化为sum[i]=sum[fa[i]]+lst[i]
- 进行dfs求解
- 但是会出现回溯后，栈里的信息不完整了，记录一下栈之前的信息，回溯时将其复原
- 时间复杂度为O(N)
- 来源：[CSP-S2019]括号树

T11

问题描述：

给定一棵 n 个节点的树，编号 $1\sim n$，第 i 个点的权值为 w_i。

现在有一张完全图，两点 x,y 之间的边权为 $w_x+w_y+dis\{x,y\}$，其中 $dis\{x,y\}$ 表示树上两点的距离。

求完全图的最小生成树的边权和。

$n \le 200000$

T11

问题分析：

- 有一个结论：
- 对于一个完全图 G=(V,E)。
- 将边集 E 分成若干个边集：$E_1,E_2,\ldots,E_k$，这些边集覆盖了 E。
- 对每个边集求最小生成树，得到新的边集 $E_1',E_2',\ldots,E_k'$；再继续对新的边集求最小生成树

  .....
- 最终得到的最小生成树一定也是 E 的最小生成树。

T11

问题分析：

- 证明：
- 反证法证明。假设存在一种比边集 $E_1',E_2',...,E_k'$ 建出的最小生成树更小。那么存在一条边 x,y 不包含在任何一个边集中。
- 在边 x,y 所在集合中，一定存在一条链连接 x,y，比边 x,y 更优。因此由这条链替代边 x,y 更优。
- 与假设矛盾。

T11

问题分析：

- 使用点分治。分治的过程其实就是将边集拆分的过程。
- 设当前分治中心为 s。每个点的点权为 $p_i=dis\{i,s\}+w_i$。
- 考虑两棵不同子树点与点之间连边得到的最小生成树的边集。

T11

问题分析：

- 不同子树的两个点的边权为 p_i+p_j。
- 设最小的点权为 p_x。那么直接将另外一棵子树上的所有点全部连到 x，此时最小。
- 最后将得到的所有边集求一个最小生成树。
- 每一层最多产生 n 条边，点分治有 log n 层，最多有 nlog n 条边。
- 总时间复杂度为 $O(n\log^2 n)$。
- 来源：[Atcoder Code Festival 2017 Final J]Tree MST