题目选讲

南京外国语学校 张超

QQ 28417877

例题1-核心城市

- 给定一棵n个节点的树，现在需要标记k个点作为核心点。满足这k个点可以不经过未被标记的点相互到达。
- 所有未被标记的点x，达到这k个点的最近距离为dis[x]。求如何标记，使得max(dis[x])最小，输出这个最小值。
- $1 \leqslant k < n \leqslant 10^5$

- 树上最远点距离最小，想到树的中心。
- 考虑树的中心一定会选，否则不优。
- 从树的中心扩展出去进行标记，使得所有叶子节点到达标记点的距离最小。
- 考虑使用二分，把每个叶子x的dep[x]-mid级祖先到根都标记掉。数一下标记的点是否不超过k。进行调整。
- 标记的过程，可以使用dp，判断某个点是否需要被标记，就是判断它的最远叶子的深度与它的深度差，是否超过mid。
- 二分的过程，可以直接用类似计数排序优化掉。
- 最终的复杂度O(n)

例题2-Bajtocja

- 给定 d张无向图，每张图都有 n个点。一开始，在任何一张图中都没有任何边。接下来有 m 次操作，每次操作会给出 a, b, k，意为在第 k张图中的点 a和点 b之间添加一条无向边。你需要在每次操作之后输出有序数对 (a, b)的个数，使得$1 \leq a,b \leq n$，且 a 点和 b点在 d张图中都连通。

- $1 \leq d \leq 200$，$1 \leq n \leq 5*10^5$,$1 \leq m \leq 10^6$，$1 \leq a,b \leq n$,$1 \leq k \leq d$

- 把相互连通的点看做一个集合。
- 对于点x，跟它在每张图上都连通的点y，满足在每张图上都与x在相同集合。
- 可以将一个点**在d张图上的集合**映射成一个数hash[i]，只需要找有多少对hash[i]相同即可。
- 考虑每次答案的增加只会因为新加入的这条边。如果新加入的两个点，不在同一个集合中。扫描其中一个点所在的集合，更新答案，减去原有的答案，更新加入新集合后的答案。
- 每次扫描较小的集合，合并到大集合中，进行启发式合并。
- 复杂度是O(m+ n*d* log n)。

例题3-IOI2009 Regions

- N个节点的树，有R种属性，每个点属于一种属性。有Q次询问，每次询问r1,r2，回答有多少对(e1,e2)满足e1属性是r1，e2属性是r2，e1是e2的祖先。
- N≤200000,R≤25000，Q≤200000

- 离线解决
- dfs整棵树，遍历到点x，回答x所有在颜色的所有询问
- 对于每个询问，可以采用两种方式去解决。

- (1)枚举其中一个点e2，求满足条件的所有祖先
- 递归访问到x时，栈中只有x的祖先节点。维护这些祖先节点的颜色，第一次访问的时候加，访问结束的时候减掉。

- (2)枚举其中一个点e1，求在它子树中满足条件的节点。
- 进入子树时记录答案，子树访问结束时记录答案，两个差值就是该子树的答案。

- 按照询问颜色的集合大小进行分类

- 如果r2所在集合的大小不超过S，采用(1)解决，枚举r2颜色集合中的所有e2。

- 如果r2所在集合的大小超过S，采用(2)解决
- 此时的r1不超过n/S个
- 枚举r1颜色集合中的e1

- S取$\sqrt{n}$，总的复杂度$O(n*\sqrt{n})$

例题4-路径覆盖

- 给定一颗n个节点的树，问最多可以选出多少条不相交的长度为k($1 \leqslant k \leqslant n$)的点路径（一个点也算路径），输出k行，每行一个数，表示长度为k时最多能选出多少条路径。
- $n \leqslant 10^5$

- 可以O(n)得到对于一个k的答案。树形DP，对于每个点，找到子树内传上来的最大链数和一个通向上方的最长链。
- 如果可以从传上来的若干链中合并出一个大于等于k的链，那么就合并。否则找到最长链继续上传。

中国计算机学会 China Computer Federation

- 随着k变大，答案在变小。

- 答案的种类不超过$2\sqrt{n}$，对于小于等于k的，假设每个答案都不同，也就是$\sqrt{n}$。对于大于k的，由于答案最大为$\lfloor \frac{n}{k} \rfloor$那么也只有种$\sqrt{n}$。

- 相同的答案是连续的，那就比较好做了。
分个块，小于S的每个k都做一遍DP，大于等于S的每次二分找到相同答案的右端点，然后直接跳到后面。

例题5-随机树

- 给定一棵n个节点的树，每条边的权值为[0,L]之间的随机整数，求这棵树两点之间最长距离不超过S的概率。
- $n \leqslant 1000, L \leqslant 10, S \leqslant 2000$

- 套用树形dp的思维方式，需要把什么信息累计到一棵子树的根节点。考虑直径的形成，是由某个节点的子树中两个最远距离较长的儿子合并过来。每棵子树的根节点需要记录最远距离，以及对应的概率。且一棵子树内不能产生直接超过S的情况。
- dp[i][j]表示i为为根的子树中，最远距离不超j的的概率。合并并子树时，只需要记录前面子树的最大值即可，同时保证两棵子树的最大值和不超S。
- 把x这棵子树的儿子y合并上来时，使用两个额外的数组
- tmp0[k]表示x到y中最远距离不超过k的概率
- tmp1[k]表示x两个最远儿子拼起来的最远距离的不超过k的概率

- 先算出tmp0[j+k] += p*dp[y][k]; (j为一条x连向y的边)
- tmp1[max(j,k)] += dp[x][j]*tmp0[k];
- 算出tmp1之后，赋值给dp[x]
- 算tmp1的复杂度是O(S^2)，可以优化。

- 枚举最大值在前面的儿子中，还是当前的儿子,再减去重复计算的即可,可以优化到O(S)。

- 最终的复杂度是O(n*S*(S+L))

例题6-Paths升级

- 有一棵n个节点的树，树上有m条路径，现在要从这些路径中选一些，选出的路径不能有公共点。每条路径有一个权值，求选出哪些路径，使得选出的总权值最大。
- $n,m \leqslant 10^5$

- 先考虑一种简单情况，如果路径的权值都一样。是否可以贪心?

- 考虑树形DP，从下往上dp,每个点记录两个值
- d[i]表示以节点i为根的子树的最优值
- sum[i]=∑(d[k] | k是节点i的儿子)
- 处理出每条链的两端点的LCA为i
- 则d[i]的转移有两种情况：
- (1)：d[i]=sum[i]
- (2): 选取一条链p(p两端点的的LCA为i)
- d[i]=max(d[i],value[p]+∑(sum[k] | k是链上的节点)-
- ∑(d[k] | k是链上的节点))

- 复杂度是链上求和的效率O(n log n)

例题7-树上旅行

- 一棵n个节点的树，每个点有一个花费时间t_i和收益w_i,经过树边需要花费时间c_i。经过一个点多次，t_i和w_i都只算一次。可以选择任意点出发，在任意点结束。求T时间内可以获得最大收益。
- $1 \leqslant n \leqslant 500$，$0 \leqslant t_i,c_i \leqslant T \leqslant 10^3$，$w \leqslant 10^6$

- 如果起点和终点已知，如何做?
- 考虑设计dp状态.
- f[i][j] 表示从 i 往子树内走并回到 i
- g[i][j] 表示从 i 往子树内走但不用回到 i .
- h[i][j] 表示从 i 内子树出发,经过 i 点,并回到 i 子树内. 就是从 i 子树中的一个节点 u , 先花费 t1 的时间走到 i , 再花费 t_2 的时间走到 i 的一个儿子 v , 最后在 v 子树内部走 $j-t_1-t_2$ 的最大价值.
- 第二维j，表示在这棵子树中花的时间。

- 考虑当前的节点为 i，处理到了儿子 v，i 到 v 的距离为 e，设在 v 的子树内走了 k 的时间.
- 如果可以走到v之后再回到i，则剩余 j-k-e*2 的时间在 v 的子树外走，那么就有:
- f[i][j]=max(f[i][j-1],f[v][k]+f[i][j-k-e*2])
- g[i][j]=max(g[i][j-1],f[v][k]+g[i][j-k-e*2])
- h[i][j]=max(h[i][j-1],max(f[v][k]+h[i][j-k-e*2],h[v][k]+f[i][j-k-e*2]))

- 如果到 v 的子树内后不回 i ,剩余 j-k-e 的时间，转移方程为:
- g[i][j]=max(g[i][j-1],f[i][j-k-e]+g[v][k])
- h[i][j]=max(h[i][j-1],g[v][k]+g[i][j-k-e])

- 注: 对于以 x 为根的子树，g[x] 显然不劣于 f[x] .
- 最终对所有的 h[i][j] 取 max 即可

- 复杂度O(nT^2)

例题8-HDU 6133

- 给定一棵n个节点的二叉树。对于一个大小为m的子树,其中所有的点权排序后为v[i],则这棵子树的价值为$\sum v[i]*(m-i)$，输出每个子树的价值。
- $n \leqslant 10^5$

- 需要计算出每个子树的答案
- 考虑使用dsu on tree

```
void solve(int x) {
    找到重儿子big
    for each y is a son of x
        if(y ≠ big)solve(y),删除y这棵子树
    end for
    solve(big);//不删除big对应的子树
    for each y is a son of x
        if(y ≠ big)加入y这棵子树
    end for
    加入x节点
    回答在x节点上的询问
}
```

- 考虑一个全局的容器，支持插入和删除一个权值，维护答案。

- 一个权值插入时，会让比它小的权值都移动一位，乘以的下标都增大1。此时对答案的影响就是这些比它小的权值和。
- 可以使用两个树状数组维护和与个数。

- 复杂度是$O(n \log^2 n)$

- 另一种更简单的方法：线段树合并

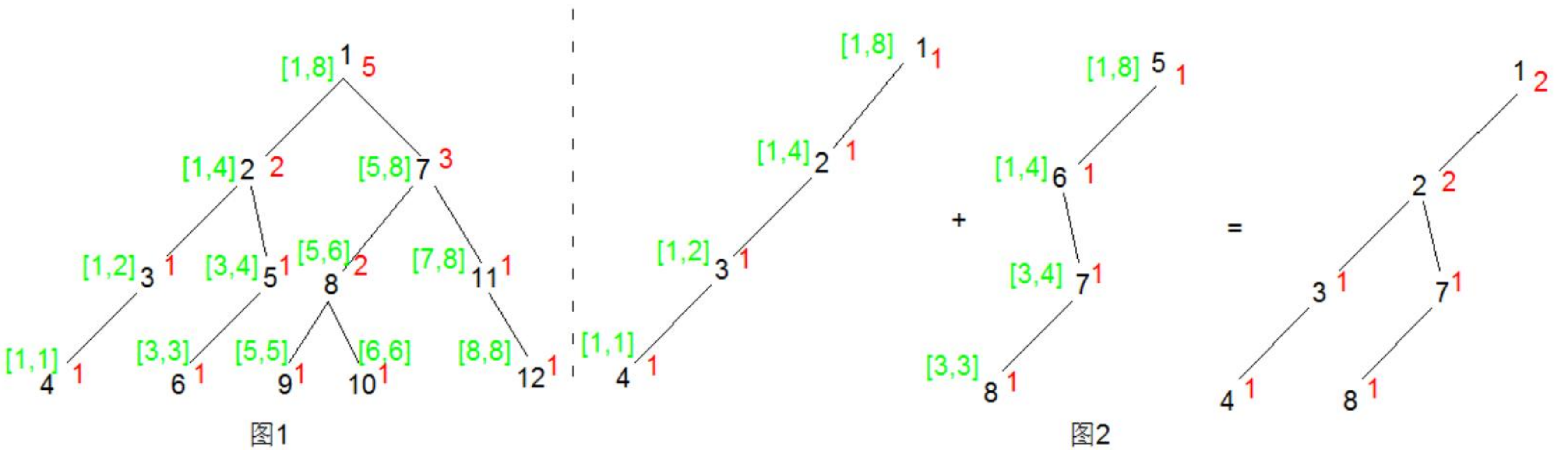


图1为1,3,5,6,8形成的权值线段树，图2为1和3对应线段树合并过程

- 每棵子树用一个线段树记录和与个数。
- 合并两个节点时，直接更新答案，用左儿子的总和乘以右儿子的个数。
- 一次合并的复杂度，为两棵线段树的公共节点个数。 合并过程中，就相当于把公共节点的权值+1。
- 所以：“这个过程的开销不会比向一棵空树顺序 插入n个整数来的大”。
- 也可以这么理解: 每调用一次Merge，节点个数会减少1， 所以合并的总复杂度与初始节点个数同阶。
- 整个n-1次合并的过程，总复杂度是O(n log n)。

例题9-树上统计

- 给定一棵n个节点的树。定义Tree[L,R]表示为了使得编号为L~R之间的点两两连通，最少需要选择的边的数量,求$\sum_{L=1}^{n}\sum_{R=L}^{n} Tree[L,R]$

- 首先，选一个点为根DFS。对于一条边，我们记这条边离根近的那个点为A，离根远的为B。那么若这条边对一对[L,R]有贡献，那么[L,R]中一定有至少一个点在B或B的子树内，且有至少一个点不在B的子树内。

- 此时我们把所有点分类，若它在B或在B的子树内，就标为1，否则标为0。然后我们就可以得到一个01序列。然后问题就转化为求有多少个[L,R]满足区间内又有0又有1。

- 但是这样对于我们过于困难。正难则反，我们可以求所有的[L,R]区间数量减去区间内只有一种数的区间。这样就可以用线段树来维护了。记录每个区间左边和延伸出来的0串和1串的长度合并即可，利用分治的思想来理解。

- 知道如何求一条边的贡献之后，就开始考虑如何快速求出所有边的贡献。我们关注到这个序列中的1的位置都是在一个点的子树中的。
- 可以用DSU来维护这个序列，全局的容器用线段树。
- 也可以使用线段树合并来实现。
- 最优的复杂度为$O(n \log n)$

中国计算机学会
China Computer Federation

例题10-SwitchGrass

- 给定一张n,m带权(权为正)无向图，每个点有一个颜色k(k≤n)，有q次操作，每次改变一个点的颜色，要求你在操作后输出这个图中最近异色点对之间的距离。最近异色点对定义为：一对点颜色不同，且距离最小.
- n,m,q≤$2*10^5$

- 首先题目询问的其实就是最短的一条边，满足两端颜色不同。
- 但是每次修改的边很多，不方便维护。
- 不难发现答案中出现的边一定是在最小生成树上的边。因为如果有一个答案不在最小生成树上，而它两端颜色不同，那么最小生成树上一定存在一条边，两端颜色不同且权值比这条边小。
- 那么现在只需要维护一个树。树的好处就是：改变一个点的权值，只会影响父亲和儿子。儿子有很多，但是我们可以在父亲节点上维护儿子的信息。

- 这题主要是和颜色有关，那么我们记录颜色。
- 改变一个点x之后，要找到儿子中颜色和x不同且边权最小的一条边。
- 于是要记录它儿子中某个颜色的所有边，同一个颜色的儿子的边权存在一个multiset中。然后还要查询除了x的颜色以外最小的边权。那么还是需要一个数据结构，这可以用一个动态开点的线段树实现（下标为颜色，查询的时候也就是询问[1,y-1]∪ [y+1,K]的最小值）

- 注意multiset只在叶子节点用到，不需要开很多。
- 然后改变一个点x，他还会影响他的父亲。
- 对于这个我们可以在父亲的线段树中先删去原来的颜色，再加入新的颜色，然后重新询问一遍。
- 那么此时知道了每个点到儿子的最小合法的边。接着用一个全局的multiset维护，维护每个点到它儿子的边中最小颜色不同的边。每次修改都要在这个multiset里修改。
- 只需要先把原来的答案删掉，改完之后再询问一遍放进去就可以了
- 复杂度为O((n+q)log K)

例题11-数星星

- 给定一棵n个节点的树，每个节点有点权。有m路径，编号为1到m。有q次询问，每次询问编号在[L,R]的路径，覆盖树上节点的点权总和。
- $n,m,q \leqslant 10^5$

中国计算机学会
China Computer Federation

- 考虑一个经典问题：一个序列，多组询问，每次询问一个区间内所有出现过 的元素之和，重复的只算一次。
- 我们的解法是按照询问右端点离线，然后按照右端点从左到右的顺序， 更新每个元素最后一次出现的位置，然后用树状数组求和。
- (1)如果树随机，对于序列中所有的路径长度是 log n的，所以我们可以直接 暴力向上跳。更新每一个节点的最后出现的位置，即如果它有，那么要在树状数组上减掉。 查询的时候直接用树状数组询问左端点即可，复杂度 $O(n \log^2 n)$。

中国计算机学会 China Computer Federation

- (2)如果树是一条链，可视为一个区间,考虑改动上述算法。
- 我们要维护每个元素最后一次出现的位置，需要支持两种操作：
- 1. 区间覆盖
- 2. 在区间覆盖[L,R]的同时，对于每个i⊆[L,R]，在树状数组上把A_i这一位置进行单点加减
- 由于第二个操作的存在，直接用线段树处理区间覆盖就不再方便了，我们不妨换一个思路处理区间覆盖。对于一个序列，维护序列上目前有哪些颜色块，并记录这些块的颜色。覆盖时，将这个区间内的所有颜色块全部都修改为目标颜色，并且把这个块内的元素c在树状数组上的c位置进行单点加减。

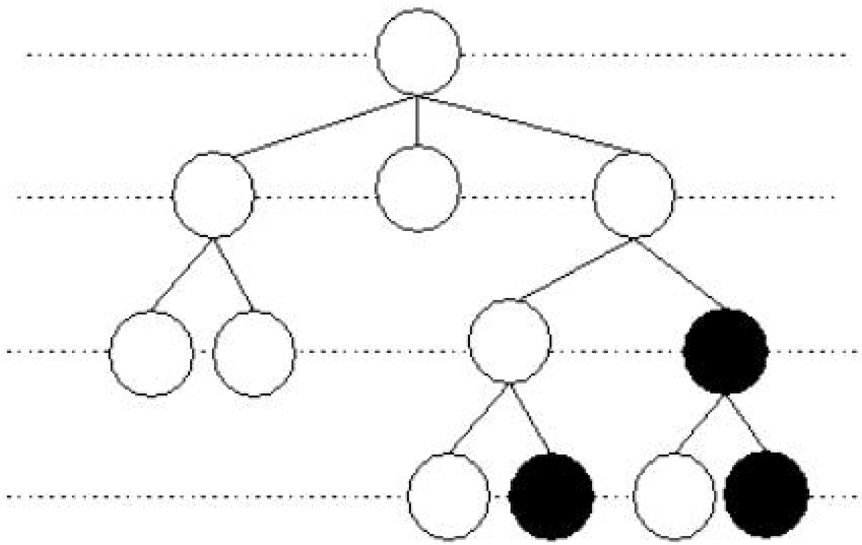
- 当然，如果两个端点在块的内部，需要额外断开块。
- 简单讲一下这个东西的复杂度：加入和删除一个块的复杂度都是O(log n)，而每次调用区间覆盖，最多只会多出两个块
- 所以加入块的总数是O(n)的，因此加入复杂度为 O(n log n) 删除块的复杂度为均摊O(n log n) 。维护颜色块等价于维护分割点的前驱后继，可以用set实现。
- 复杂度O(n log n)。

- (3)一般情况
- 可以用树链剖分把每条路径划分成log n区间
- 实现方法与链的情况就类似了。
- 总的复杂度$O(n \log^2 n)$

例题12-七彩树

- 给定一棵n个节点的树，每个节点有一个颜色c[i]，m个询问，每次询问子树 X_i 内，与 X_i 距离不超过 D_i 的所有节点的不同颜
- 色数。强制在线。
- $1 \leqslant n, m \leqslant 10^5, 1 \leqslant c[i] \leqslant n$

- 本题中，先考虑离线。
- 每个询问实际上对应一个dfs序区间和深度区间。
- bfs遍历整棵树，按深度一层一层加入每个点，把当前点对应的dfs序位置+1，删除与它同色并且形成lca最深的节点，可通过set查找dfs序离当前点最近的点，在lca对应的dfs序位置上-1。用树状数组维护dfs序，用于统计区间和。



- 加完本层以后，回答所有以该层作为深度右端的询问。

•解法一

- 按深度建一棵前缀线段树，线段树上存储的是dfs序，把前面所有层的信息都继承到当前层，加入当前点时，把同色中与它形成lca最深的lca删掉，这些都在当前层主席树中完成。
- 询问时，在深度区间右端点对应的主席树上进行查询。

- 解法二
- 使用线段树合并，每个节点用两棵线段树存储子树的信息。
- 一棵以颜色为键值维护每种颜色的最小深度
- 一棵以深度为键值维护不同颜色的种数。
- 把子树信息收集上来，先把深度对应的线段树合并。再合并颜色，遇到相同颜色时，在深度线段树上把深度大的减掉，保证从当前节点向下查询时候，每种颜色只需要一个深度最小的。
- 两棵树都要可持久化，存下所有的信息，每次Merge时，新建一个节点，时空复杂度均为O(n log n)。

中国计算机学会
China Computer Federation

谢谢观看！