# 树

罗进

长沙市一中

2018 年 4 月 1 日

# Outline

# 点分治

- 一棵树的重心为删掉该点之后剩下的最大子树节点最少的点.
- 重心一定存在, 最多只有两个.

# 点分治

- 一棵树的重心为删掉该点之后剩下的最大子树节点最少的点.
- 重心一定存在, 最多只有两个.
- 每次找到重心, 处理经过当前重心的答案, 把重心删掉再递归处理剩下的子树.
- 分治的层数为 $O(\log N)$.

# Codechef CUTTREE

- 定义森林的强度为连通块大小的平方和.
- 第 $0$ 天有一棵 $N$ 个节点的树, 每一天大厨会随机删掉森林中一条边, 共进行 $N-1$ 天, 对于 $i=1\cdots N$ 求出第 $i$ 天结束时这棵树的强度的期望值, 求答案对 998244353 取模的结果.
- $N \leq 10^5$.

## Solution

- 如果有序点对 $(u, v)$ 满足 $u$ 和 $v$ 连通，那么它就对树的强度有 $1$ 的贡献.
- 求出第 $i$ 期望有多少个点对满足条件.

# Solution

- 如果有序点对 $(u, v)$ 满足 $u$ 和 $v$ 连通，那么它就对树的强度有 $1$ 的贡献.
- 求出第 $i$ 期望有多少个点对满足条件.
- $u, v$ 连通的条件是 $u$ 到 $v$ 的路径上的点都不能被删.
- $(u, v)$ 对第 $i$ 天的贡献是 $\frac{(N-1-d(u,v))^{\underline{i}}}{(N-1)^{\underline{i}}}$.

# Solution

- 如果有序点对 $(u, v)$ 满足 $u$ 和 $v$ 连通，那么它就对树的强度有 $1$ 的贡献.
- 求出第 $i$ 期望有多少个点对满足条件.
- $u, v$ 连通的条件是 $u$ 到 $v$ 的路径上的点都不能被删.
- $(u, v)$ 对第 $i$ 天的贡献是 $\frac{(N-1-d(u,v))^{\underline{i}}}{(N-1)^{\underline{i}}}$.
- 如果能够求出距离为 $d$ 的点对数量 $cnt_d$
- $ans_i = \sum_{d=0}^{N-1} \frac{(N-1-d)!(N-1-i)!}{(N-1-d-i)!(N-1)!} \times cnt_d$
- $N-1-d-i+d = N-1-i$ 是一个卷集的形式，可以用 NTT 优化.

# Solution

- 考虑使用点分治求 $cnt_d$.
- 求过分治中心的路径也相当于是在做卷集, 可以用 NTT 优化.

# 边分治

- 参考点分治，每次找到一条边使得删掉这条边之后两个子树的最大节点树尽可能少.
- 优点很明显，每次只需要合并两棵子树的信息，较点分治而言更好实现.
- 但是如果类似星形树复杂度就没有保证.

# 边分治

- 参考点分治，每次找到一条边使得删掉这条边之后两个子树的最大节点树尽可能少.
- 优点很明显，每次只需要合并两棵子树的信息，较点分治而言更好实现.
- 但是如果类似星形树复杂度就没有保证.
- 可以采用加点来使得平衡，但是会损失一些性质，没有点分治那么通用.

# 动态点分治

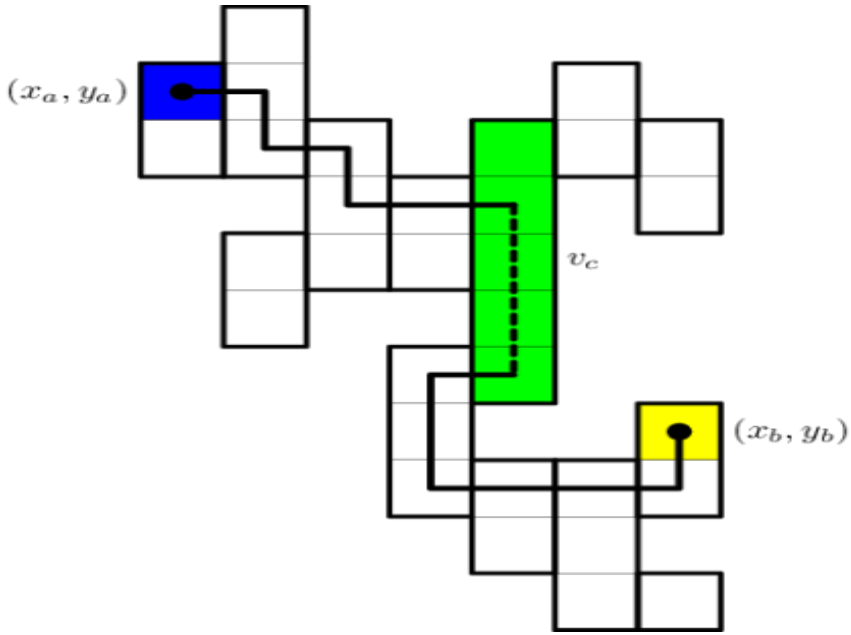
- 如果每个重心都与它所在子树上一层的重心连边，就会构成一个树结构.
- 对于每个重心，维护它所代表的子树内的点的信息.
- 由于点分治的层数是 $O(\log N)$ 的，所以修改一个点最多只会对 $O(\log N)$ 个分治树上的节点有影响.

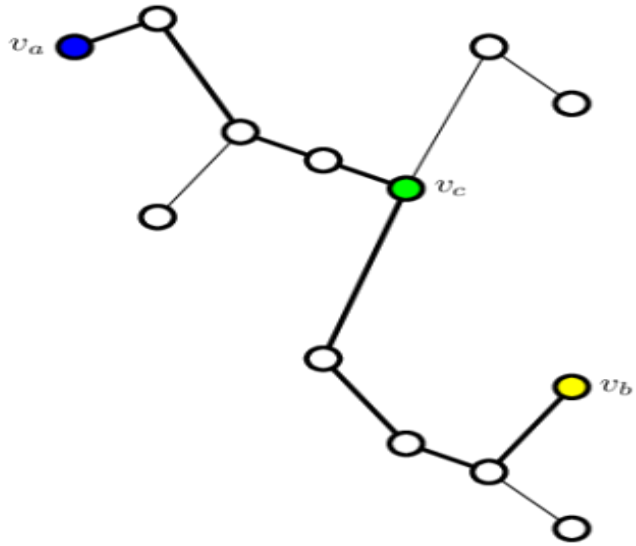
# Codeforecs 936 E

- 一个网格图上有 $N$ 个好的格子，这些好的格子是四连通的，不是好的格子也是四连通的. 现在有 $q$ 个操作，为一下两种.
- 1 x y：标记坐标为 $(x, y)$ 的，保证 $(x, y)$ 是个好的格子.
- 2 x y：在只能走好的格点的情况下，从 $(x, y)$ 走到最近的一个被标记的格子要走多远，保证 $(x, y)$ 是一个好的格子.

# Solution

- 把横坐标相同且相邻的格子缩成一个点，再还原连通性，那么原图就成了一个树结构.



## Solution

- 两个点之间的某条经过重心的路径一定是分别走到重心上离自己最靠近位置.
- 设 $d_u$ 为 $u$ 走到重心上最少要走多远, $z_u$ 为重心上离 $u$ 最近的点的位置.

# Solution

- 两个点之间的某条经过重心的路径一定是分别走到重心上离自己最靠近位置.
- 设 $d_u$ 为 $u$ 走到重心上最少要走多远, $z_u$ 为重心上离 $u$ 最近的点的位置.
- 那么 $u$ 到 $v$ 的路径长度就是 $d_u + d_v + |z_u - z_v|$.
- 每个重心维护一个线段树即可解决, $d_u$ 和 $z_u$ 能在最开始分治的时候求出.

# 可持久化点分治

- 由于动态点分治的分治树结构层数为 $O(\log N)$，所以可以考虑持久化.

# 可持久化点分治

- 由于动态点分治的分治树结构层数为 $O(\log N)$，所以可以考虑持久化.
- 点的度数可能很大，复制信息的时候复杂度无法保证.

# 可持久化点分治

- 由于动态点分治的分治树结构层数为 $O(\log N)$，所以可以考虑持久化.
- 点的度数可能很大，复制信息的时候复杂度无法保证.
- 可以通过加辅助节点把每个点的度数限制在 3 以内.

# Codeforces 757 G

- 给定一棵 $n$ 个节点的树，和一个长为 $n$ 的排列 $a_i$，有 $q$ 个询问，为一下两种类型.
- 1 l r v: 求 $\sum_{i=l}^{r} dist(a_i, v)$.
- 2 x: 交换 $a_x, a_{x+1}$.
- 强制在线.
- $n, q \leq 2 \times 10^5$

# Solution

- 如果没有交换就直接上可持久化点分治.
- 交换操作只会对两棵点分树有影响, 所以可以直接修改.

# 树上莫队

- 树上莫队可以处理一类询问树上路径的问题.
- 如果要从询问 $(u, v)$ 转移到询问 $(u', v')$, 只需要将路径 $(u, u')$ 和路径 $(v, v')$ 上点的状态取反.

# 树上莫队

- 树上莫队可以处理一类询问树上路径的问题.
- 如果要从询问 $(u, v)$ 转移到询问 $(u', v')$, 只需要将路径 $(u, u')$ 和路径 $(v, v')$ 上点的状态取反.
- 如果我们能够将树上的点分成很多大小为 $O(S)$ 的块, 使得块内的点之间路径长度不超过 $O(S)$.
- 将询问按照 $(blockid(u), dfn(v))$ 排序, 复杂度即为 $O(QS + \frac{N}{S}N)$, 当 $Q$ 和 $N$ 同阶时, $S$ 取 $\sqrt{N}$ 复杂度为 $O(N\sqrt{N})$.

# 树上莫队

- 树上莫队可以处理一类询问树上路径的问题.
- 如果要从询问 $(u, v)$ 转移到询问 $(u', v')$, 只需要将路径 $(u, u')$ 和路径 $(v, v')$ 上点的状态取反.
- 如果我们能够将树上的点分成很多大小为 $O(S)$ 的块, 使得块内的点之间路径长度不超过 $O(S)$.
- 将询问按照 $(blockid(u), dfn(v))$ 排序, 复杂度即为 $O(QS + \frac{N}{S}N)$, 当 $Q$ 和 $N$ 同阶时, $S$ 取 $\sqrt{N}$ 复杂度为 $O(N\sqrt{N})$.
- 分块方法可以使用 BZOJ1086 的方法, 可以保证块的大小在 $[S, 3S]$ 之间.
- 用处的话就是能够把序列上的问题放到树上了.

- 树上路径上不同颜色的个数.
- 树上路径的 mex.

- 一个 $n$ 个点的树，每个点有一个颜色，$q$ 个询问，每次询问 $(u, v)$ 路径上出现次数最多的颜色的出现次数.
- $O(N\sqrt{N})$.

- 树上莫队.
- 维护颜色的出现次数以及出现次数的出现次数就行了.

# 虚树

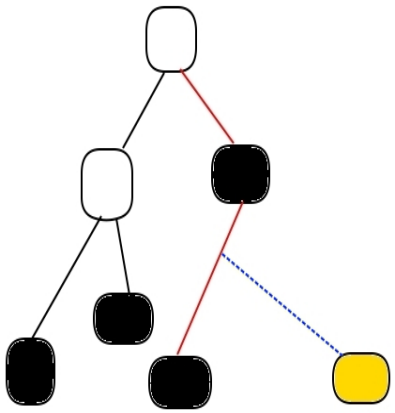
- 给出一颗树，每次询问一个点集，求这些点集在树上的信息，只保证所有点集的大小之和为 $O(N)$ 级别.

# 虚树

- 给出一颗树，每次询问一个点集，求这些点集在树上的信息，只保证所有点集的大小之和为 $O(N)$ 级别.
- 不能够每次遍历整棵树，可以建出一颗类似原树的结构，只保留关键点以及它们两两的 LCA.
- 可以证明，如果点集的大小为 $K$，那么两两之间的 LCA 个数为至多为 $K-1$，所以虚树的大小是 $O($点集大小$)$ 的.

# 虚树

- 给出一颗树，每次询问一个点集，求这些点集在树上的信息，只保证所有点集的大小之和为 $O(N)$ 级别.
- 不能够每次遍历整棵树，可以建出一颗类似原树的结构，只保留关键点以及它们两两的 LCA.
- 可以证明，如果点集的大小为 $K$，那么两两之间的 LCA 个数为至多为 $K-1$，所以虚树的大小是 $O(\text{点集大小})$ 的.
- 将点集按照 $dfs$ 序排成升序，每次维护最右链来构建，单次构建复杂度 $O(K \log K)$.



# HNOI2014 世界树

- 给出一棵 $N$ 个点的树，有 $M$ 个询问，每次给出点集 $S$，树上的每个点会被距离它最近的点集中的点管辖 (距离相同的选标号较小的)，对于点集中的每个点求出它管辖了多少个点.
- $N \leq 3 \times 10^5, sum(|S|) \leq 3 \times 10^5$.

## Solution

- 建出虚树，然后树形 dp，dp 出虚树上每个点最近的关键点是谁，以及距离是多少.
- 虚树与原树不同的地方在于边上也是有点的，这里还必须把根节点点也加入虚树中.

## Solution

- 建出虚树，然后树形 dp，dp 出虚树上每个点最近的关键点是谁，以及距离是多少.
- 虚树与原树不同的地方在于边上也是有点的，这里还必须把根节点点也加入虚树中.
- 对于虚树上的节点，我们知道它以及它那些没有出现在虚树中的子树被谁管辖.
- 对于边上的点，可以二分出边界，然后计算点数.
- 复杂度 $O(N \log N)$

# Codeforces Bear and Chemistry

- 有一个 $n$ 个点 $m$ 条边的图，有 $q$ 个询问，每次询问给出一个点集 $V$ 和一个边集 $E$，求在原图上加入 $E$ 中的边之后，$V$ 中有多少点对在同一个边双连通分量内.
- $n, m \leq 3 \times 10^5, sum(|V|), sum(|E|) \leq 3 \times 10^5$.

## Solution

- 先对原图求出双连通分量，如果把双连通分量缩成一个点，那么就成了一颗树.
- 对每个询问给出的关键点建虚树，然后在跑一遍 tarjan 求双连通分量.
- 统计每个双连通分量中的点的个数即可.
- 复杂度 $O(N \log N)$

# POI Tourism

- 给出一个 $N$ 个点 $M$ 条边的无向图，保证不存在一条长度大于 10 的简单路径. 每个点有个点权 $c_i$. 你需要选择一些点，对于每个点都要满足要么它被选，要么与它相邻的某个点被选，最小化被选点的点权之和，输出和最小是多少.
- $n \leq 20000, m \leq 25000$.

# Solution

- 由于不存在一条长度大于 10 的简单路径，所以存在一棵 dfs 树的深度不超过 5.
- 无向图的 dfs 树不存在横向跨过子树的边，所以一个点只会对他的祖先和子树有影响.

# Solution

- 由于不存在一条长度大于 10 的简单路径，所以存在一棵 dfs 树的深度不超过 5.
- 无向图的 dfs 树不存在横向跨过子树的边，所以一个点只会对他的祖先和子树有影响.
- 对于每个祖先，记录 0/1/2，分别代表它没有满足条件/自己被选/某个相邻的被选，由于深度不大，所以复杂度是可以承受的.

# 例题

- 有一棵树，每个节点上有一颗糖，拿走这颗糖可以获得 $a_i$ 的收益．如果一个节点上的糖被拿走，那么它的祖先中就不能有糖被拿走．最多拿 $k$ 颗糖，求最大收益．
- $n \times k \leq 10000000$

# Solution

- 拿了一颗糖之后它的子树中就不能有糖被拿.
- 弄出 dfs 序之后, 相当于有 $n$ 个区间, 每个区间有一个权值, 区间不能相交.
- $O(NK)$ 解决.

# 例题

- 给出一个 $N$ 个节点的树，每次分治时可以随意选择分治重心，分治最深层数最小是多少.
- $N \leq 100000$.

# Solution

- 相当于给每个节点标个数字，使得如果 $u, v$ 数字相同，那么 $u$ 到 $v$ 的路径上至少有一个数字大于它们的数字.
- 设 $f[i][S]$ 为以 $i$ 为根的子树中，没有被覆盖的数字集合为 $S$ 是否有可能 (被覆盖指这个数字到根节点的路径上有一个比它大的数).
- 如果 $i$ 的儿子的状态分别为 $S_1, S_2 \cdots, S_k$，首先 $S$ 应该等于 $S_1 | S_2 | \cdots | S_k$，如果有个数字在多于一个集合中出现，那么 $i$ 上的数字应该大于它们.
- $i$ 同样不能选 $S$ 中出现了的数字，如果确定了 $i$ 上的数字，那么就可以把 $S$ 小于它的数字删掉.

## Solution

- 如果 $f[i][S] = 1, f[i][T] = 1, T \subset S$, 那么 $S$ 是不优的.
- 如果将 $S$ 中的元素写成二进制的形式, 那么可以证明只需要保留最小的 $S$ 即可.

# Solution

- 如果 $f[i][S]=1, f[i][T]=1, T \subset S$, 那么 $S$ 是不优的.
- 如果将 $S$ 中的元素写成二进制的形式, 那么可以证明只需要保留最小的 $S$ 即可.
- 如果 $f[i][T]=1, f[i][S]=1, T<S, T \not\subset S$, 那么考虑 $S$ 和 $T$ 最高的不相同的位 $k$, $S$ 在该位为 1, $T$ 在该位为 0.
- 因为 $T$ 不是 $S$ 的子集, 所以 $T$ 在这一位后面还有 1, 那么就说明 $i$ 上的数字小于 $k$, 把 $i$ 上的数字改为 $k$ 即可得到 $S$ 的一个子集 (可能会得到 $S$, 但是这个时候 $T$ 是要比 $S$ 优的).
- 由于点分治层数的上界是 $O(\log N)$ 的, 所以这题复杂度为 $O(N \log N)$.

# SDOI 2017 苹果树

- 有一个 $N$ 个点的有根树，第 $i$ 个点上有 $a_i$ 个苹果，每个苹果有 $v_i$ 的幸福度，你需要在树上拿一些苹果，必须满足如果一个点上至少被拿了一个苹果，那么它的父亲也要至少被拿一个苹果.
- 还有一个参数 $k$，如果一共取走了 $t$ 个苹果，且至少被取了一个苹果的最深节点的深度为 $d$，那么要求 $t - d \leq k$. 求最多能收获多少幸福度.
- $N \leq 20000, K \leq 500000, NK \leq 250000000$.

## Solution

- 先把所有点拆成两个点，一个点上有 $1$ 个苹果，另一个点上有 $v_i - 1$ 个苹果，前一个点为后一个点的父亲.
- $t - d \leq k$ 的条件相当于一条链上的苹果可以不占用空间，其他节点的至多选 $k$ 个苹果.

# Solution

- 先把所有点拆成两个点，一个点上有 $1$ 个苹果，另一个点上有 $v_i - 1$ 个苹果，前一个点为后一个点的父亲.
- $t - d \leq k$ 的条件相当于一条链上的苹果可以不占用空间，其他节点的至多选 $k$ 个苹果.
- 考虑枚举这条链，它把整棵树分成了两部分，这两部分的点在 dfs 序上是连续的.
- 对整棵树 dfs 的前缀和后缀分别 dp.

# Solution

- 我们对每个点 $u$ 计 $f[u][i]$ 为 $dfs$ 到 $u$ 的时候，选了 $i$ 个点的最多能获得多少收益.
- 对于一个点被选，它的父亲必须被选的限制条件，我们可以在第一次访问到 $u$ 的时候令 $u$ 必须选，求出 $f[u]$，对于 $u$ 的儿子 $son$，我们先把 $f[u]$ 复制给 $f[son]$，然后再去递归 $son$ 的子树. 当回溯到 $u$ 的时候，我们将 $f[u]$ 与 $f[son]$ 取最大值再赋值给 $f[u]$.
- 这样保证了求每个节点 $u$ 的 dp 数组时，它的父亲已经被选了，后面的取最大值保证了肯定是一棵子树不被选.

# IOI2017 training

- 一个无向图由一颗树和一些非树边组成，每条非树边有一个权值，你需要删掉一些非树边使得图中没有偶环，并最小化被删边的权值之和．保证每个点的度数不超过 10.
- $|V| \leq 1000, |E| \leq 5000.$

# Solution

- 选的边权值之和尽可能大，每条边至多出现在一个环上.

# Solution

- 选的边权值之和尽可能大，每条边至多出现在一个环上.
- 每条树边至多被覆盖一次，每选一条非树边就会把树分成很多部分.
- 设 $f[i][S]$ 为 $i$ 的子树中只儿子集合为 $S$ 的情况.

# Solution

- 选的边权值之和尽可能大，每条边至多出现在一个环上.
- 每条树边至多被覆盖一次，每选一条非树边就会把树分成很多部分.
- 设 $f[i][S]$ 为 $i$ 的子树中只儿子集合为 $S$ 的情况.
- 选一条跨过 $i$ 的非树边只会影响两个子树，先枚举所有非树边求出 $f[i][S](|S| \leq 2)$，然后再枚举子集 $f[i][T] + f[i][S - T]$ 对 $f[i][S]$ 取 max.

# NOI2013 快餐店

- 给出一个环套树，每条边有边权．你需要找到一个位置，使得这个位置到其它节点最短路的最大值最小．这个位置不一定要在节点上．
- $N \leq 10^5$.

## Solution

- 如果是树的话，答案就是直径除 2.
- 对于环套树，环上肯定有一条边是没有用的，考虑枚举这条边然后求树的直径.

# Solution

- 如果是树的话，答案就是直径除 2.
- 对于环套树，环上肯定有一条边是没有用的，考虑枚举这条边然后求树的直径.
- 对于环上每个点求出它到子树内的最远距离.
- 把环倍长，求前缀和，断掉一条边就相当于只考虑一段长为 $len$ 的区间.
- 求这段区间内 $s_i - s_j + d_i + d_j (i < j) = (s_i + d_i) + (-s_i + d_j)$ 的最大值.

# Solution

- 如果是树的话，答案就是直径除 2.
- 对于环套树，环上肯定有一条边是没有用的，考虑枚举这条边然后求树的直径.
- 对于环上每个点求出它到子树内的最远距离.
- 把环倍长，求前缀和，断掉一条边就相当于只考虑一段长为 $len$ 的区间.
- 求这段区间内 $s_i - s_j + d_i + d_j (i < j) = (s_i + d_i) + (-s_i + d_j)$ 的最大值.
- $i > j$ 显然不会成为答案，记录最大值和次大值可以避免计算 $i = j$ 的情况.