Solution

ExfJoe

福建省长乐第一中学

March 13, 2017

Outline

传销组织

传销组织

- 实际上就是用第一类边判断是否满足第二类边

传销组织

- 实际上就是用第一类边判断是否满足第二类边
- 首先强连通缩点建 DAG

传销组织

- 实际上就是用第一类边判断是否满足第二类边
- 首先强连通缩点建 DAG
- 拓扑排序，暴力统计每个点能到达哪些点，bitset 压位优化

传销组织

- 实际上就是用第一类边判断是否满足第二类边
- 首先强连通缩点建 DAG
- 拓扑排序，暴力统计每个点能到达哪些点，bitset 压位优化
- 由于内存不够，所以将所有点分块，统计每个点能到块中的哪些点即可

Outline

守卫

守卫

- $k_i = 1$：即询问有多少个点距离 u 在 r 之内

守卫

- $k_i = 1$：即询问有多少个点距离 u 在 r 之内
- 考虑点分治，对于每个重心 i，我们查询有多少个 v 满足 $dis(u,i) + div(i,v) \le r$

守卫

- $k_i = 1$：即询问有多少个点距离 u 在 r 之内
- 考虑点分治，对于每个重心 i，我们查询有多少个 v 满足 $dis(u, i) + div(i, v) \le r$
- $C(i, dep)$ 表示 i 为重心时满足 $dis(i, v) \le dep$ 的 v 的数量，这个数组可以在 $n \log n$ 的时间内维护出来，进而可以统计出答案

守卫

- $k_i = 1$：即询问有多少个点距离 u 在 r 之内
- 考虑点分治，对于每个重心 i，我们查询有多少个 v 满足 $dis(u, i) + div(i, v) \leq r$
- $C(i, dep)$ 表示 i 为重心时满足 $dis(i, v) \leq dep$ 的 v 的数量，这个数组可以在 $n \log n$ 的时间内维护出来，进而可以统计出答案
- 当 $k_i > 1$ 时，我们考虑对当前的守卫建立一棵虚树

守卫

- $k_i = 1$：即询问有多少个点距离 u 在 r 之内
- 考虑点分治，对于每个重心 i，我们查询有多少个 v 满足 $dis(u,i) + div(i,v) \le r$
- $C(i, dep)$ 表示 i 为重心时满足 $dis(i,v) \le dep$ 的 v 的数量，这个数组可以在 $n \log n$ 的时间内维护出来，进而可以统计出答案
- 当 $k_i > 1$ 时，我们考虑对当前的守卫建立一棵虚树
- 对于虚树中的一条边，我们找到它的分界点，在分界点处加上一个贡献为负的守卫

守卫

- $k_i = 1$：即询问有多少个点距离 u 在 r 之内
- 考虑点分治，对于每个重心 i，我们查询有多少个 v 满足 $dis(u,i) + div(i,v) \leq r$
- $C(i, dep)$ 表示 i 为重心时满足 $dis(i,v) \leq dep$ 的 v 的数量，这个数组可以在 $n \log n$ 的时间内维护出来，进而可以统计出答案
- 当 $k_i > 1$ 时，我们考虑对当前的守卫建立一棵虚树
- 对于虚树中的一条边，我们找到它的分界点，在分界点处加上一个贡献为负的守卫
- 依然点分治，对于所有点上的询问我们依次处理即可

守卫

- $k_i = 1$：即询问有多少个点距离 u 在 r 之内
- 考虑点分治，对于每个重心 i，我们查询有多少个 v 满足 $dis(u, i) + div(i, v) \leq r$
- $C(i, dep)$ 表示 i 为重心时满足 $dis(i, v) \leq dep$ 的 v 的数量，这个数组可以在 $n \log n$ 的时间内维护出来，进而可以统计出答案
- 当 $k_i > 1$ 时，我们考虑对当前的守卫建立一棵虚树
- 对于虚树中的一条边，我们找到它的分界点，在分界点处加上一个贡献为负的守卫
- 依然点分治，对于所有点上的询问我们依次处理即可
- 处理重复的部分，可以分奇偶情况讨论，也可以使用主席树暴力减去

Outline

树

树

- 将 $(\sum a_i)^k$ 暴力展开，相当于在连通块的点权集合中可重的选出 k 个数的积之和

树

- 将 $(\sum a_i)^k$ 暴力展开，相当于在连通块的点权集合中可重的选出 k 个数的积之和
- 设 $f(i,j)$ 表示 i 为根的子树中，连通块包含点 i，已重复选出 j 个数的贡献

树

- 将 $(\sum a_i)^k$ 暴力展开，相当于在连通块的点权集合中可重的选出 k 个数的积之和
- 设 $f(i,j)$ 表示 i 为根的子树中，连通块包含点 i，已重复选出 j 个数的贡献
- 将子树合并时，其实就是将两个有序序列合并成一个：

树

- 将 $(\sum a_i)^k$ 暴力展开，相当于在连通块的点权集合中可重的选出 k 个数的积之和
- 设 $f(i,j)$ 表示 i 为根的子树中，连通块包含点 i，已重复选出 j 个数的贡献
- 将子树合并时，其实就是将两个有序序列合并成一个：
- $$f(i,j) = (1-p)\cdot f(i,j) + p\cdot\sum_{x=0}^{j}(f(son,x)\cdot f(i,j-x)\cdot\binom{j}{x})$$

树

- 将 $(\sum a_i)^k$ 暴力展开，相当于在连通块的点权集合中可重的选出 k 个数的积之和
- 设 $f(i,j)$ 表示 i 为根的子树中，连通块包含点 i，已重复选出 j 个数的贡献
- 将子树合并时，其实就是将两个有序序列合并成一个：
- $$f(i,j) = (1-p)\cdot f(i,j) + p\cdot\sum_{x=0}^{j}(f(son,x)\cdot f(i,j-x)\cdot\binom{j}{x})$$
- 每个点的贡献为 $f(i,k)\cdot(1-p_{fa})$

树

- 将 $(\sum a_i)^k$ 暴力展开，相当于在连通块的点权集合中可重的选出 k 个数的积之和
- 设 $f(i,j)$ 表示 i 为根的子树中，连通块包含点 i，已重复选出 j 个数的贡献
- 将子树合并时，其实就是将两个有序序列合并成一个：
- $$f(i,j) = (1-p) \cdot f(i,j) + p \cdot \sum_{x=0}^{j} (f(son,x) \cdot f(i,j-x) \cdot \binom{j}{x})$$
- 每个点的贡献为 $f(i,k) \cdot (1-p_{fa})$
- 将上述方程变形一下，弄成卷积形式，NTT 优化即可，时间复杂度 $O(nk \log k)$