希望 解题报告

出题人：清华大学 计算机系 杨天祺

验题人：清华大学 计算机系 李嘉图

清华大学 计算机系 刘承奥

清华大学 计算机系 王之栋

题目大意

- 给一棵 n 个点的树，求有多少方案选 k 个联通块，使得存在一个中心点，k个联通块中所有点到这个中心点的距离都 $\leq L$
- $1 \leq n \leq 10^6$，$1 \leq k \leq 10$

吐槽&讨论

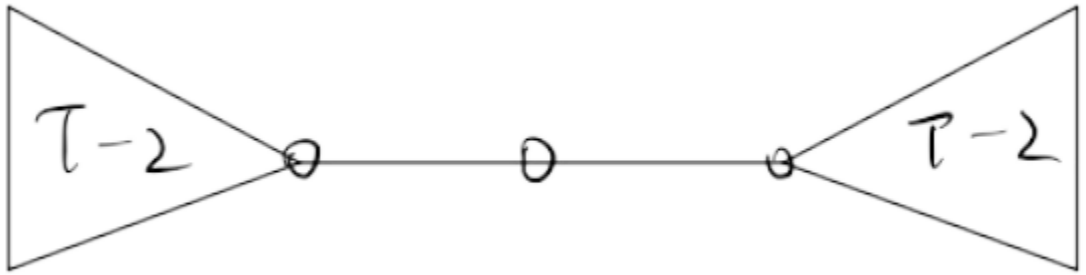
算法1

- 2^{nk}枚举一 k 个联通块判断是否可行

- 复杂度：$O\left(2^{nk} * poly(n,k)\right)$
- 期望得分：8分

引理1

- 一个重要的发现：
- **引理1.** 对于一个联通块 S，我们设满足在 S 中且到 S 中任意点距离都 $\leq L$ 的点集为 T，则 T 也是一个联通块
- **证明.** 对于 T 中的任意两点 u 和 v，一定有 u 到 v 路径上的任意点也在 T 中



推论2

- 我们还可以把结论推广到 k 个点的情形：
- **推论2.** 对于一个选 k 个联通块方案 $S_1, S_2, S_3, \ldots, S_k$，我们假设其满足条件的中心点集合为 T，则 T 也是树上的一个联通块

算法2

- 枚举推论2中的中心点联通块 T，检查其它每个集合是否能够满足这个联通块

- 复杂度：$O(3^n)$ 或 $O(2^n)$
- 期望得分：16分

引理3

- 注意到对于一个树上的联通块，我们设它的点集为 S，我们设 S 的导出子图的边集为 T，则有 $|S|-|T|=1$
- 我们定义 x 可以控制联通块 S，当且仅当 x 可以是 S 的中心点
- **引理3.** 若我们假设$f(v)$表示点 v 控制的联通块数，$g(e)$表示边 e 的两个端点都控制的联通块数，则答案就为

$$\sum_{v\in V} f(v)^k - \sum_{e\in E} g(e)^k$$

- 因为 k 个联通块同时被 v 控制的方案数是 $f(v)^k$

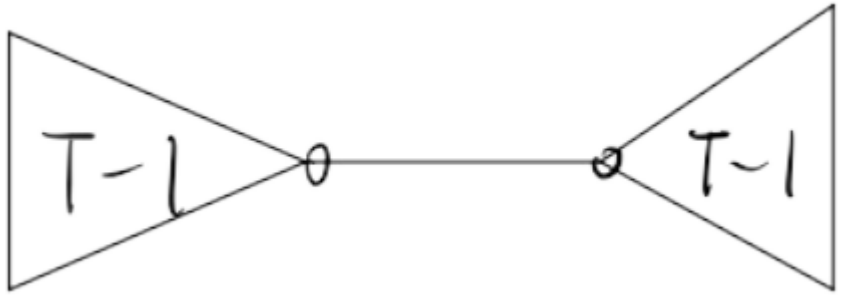
算法3

- 有了引理3，我们就可以考虑多项式做法了。一个点控制一个联通块当且仅当联通块中每个点到它距离都 $\leq L$，且这个点在这个联通块内。因此若我们假设 T_i 表示所有到 i 距离 $\leq L$ 的所有点构成的联通子树，则 v 控制的联通块数等于 T_v 中 v 包含 v 的联通块数。
- 我们考虑以 v 为根做树形DP。$dp_i - 1$ 表示以 i 为根的子树中，包含 i 的联通块数，则转移可以写成：

$$dp_i = \prod_{v \in ch(i)} dp_v + 1$$

- $f(v)$ 就等于 dp_v，因为这个时候我们假定了 v 是根，那么以 v 为根的子树就是整棵树。$g(e)$ 也可以类似的求。$g(e)$ 的要求是两端点对应的子树深度都 $\leq L-1$

- 复杂度：$O(n^2)$
- 期望得分：24分



算法4

- 我们考虑 $L = n$ 的点。这个时候一个点可以控制所有包含它的联通块。L 的限制没有意义了。
- 那么我们就是要数有多少个联通块包含 v。还是考虑树形DP。我们设 $dp_i - 1$ 表示以 i 为根的子树中包含 i 的联通块数，up_i表示包含 i 且不包含以 i 为根的子树内其它点的联通块数，则可以直接转移

- 复杂度：$O(n)$
- 期望得分：20分
- 结合算法3可得36分

算法5

- 当然我们可以直接把算法4的做法直接推广到 L 小的时候。我们修改一下DP的定义：
- 我们设 $dp_{i,j}-1$ 表示以 i 为根的子树中包含 i 且深度不超过 j 的联通块数，$up_{i,j}$ 表示包含 i 且不包含以 i 为根的子树内其它点的深度不超过 j 的联通块数，转移：

$$dp_{i,j}=\prod_{v\in ch(i)} dp_{v,j-1}+1$$

$$up_{i,j}=up_{fa_i,j-1}\prod_{v\in ch(i)} dp_{v,j-2}+1$$

- DFS 两遍，第一遍算 dp，第二遍算 up 即可。

- 复杂度：$O(nL)$
- 期望得分：36分
- 结合算法4可得48分

算法6

- 一条链的时候我们可以直接手算出 $f(v)$ 和 $g(e)$

- 期望得分：4分
- 集合算法5可得52分

算法7 - dp

- 我们考虑算法5的瓶颈在于DP的维数是 $O(nL)$ 的，但是我们发现转移类似一个点积，因此我们可以考虑使用长链剖分优化
- 对于第一部分 dp 的计算，我们只把轻边向重链合并
- 但是这里有一个问题，我们维护的是不超过 L 的个数，因此我们假设当且考虑DFS到点 u 对应的子树，在轻儿子 v 向重链合并的时候，需要将比轻子树深度 l 大的深度的 dp 值全部乘上 $dp_{v,l}$，因为对于比 l 大的深度，轻儿子 v 对答案的贡献一定都是 $dp_{v,l}$。
- 因为我们需要一个可以区间加、区间乘的数据结构，可以用可持久化线段树维护。这里区间加是因为 dp 值最后有个 $+1$，可持久化的原因是在第二遍DFS算 up 的时候还需要用到每个时刻的 dp 值

算法7 - up

- 关键问题是算 up 的时候如何优化复杂度。我们考虑长剖均摊的关键是转移 i 的时候复杂度要是轻儿子深度和的。我们分转移重链和转移轻边考虑
- 1. 重链：这个时候把所有轻儿子的 dp 值合并到 up 上就行了，复杂度是所有轻儿子深度和的，但一个点只会有一个轻儿子，因此这部分只会做一次
- 2. 轻边：我们考虑如果一个轻子树的深度是 dep_i，则对于这个子树的里面的所有点，他们需要用到的 $up_{i,j}$ 的 j 的范围只会是 $[j-dep_i,j]$，因为注意到计算 $f(v)$ 我们只需要用到 $up_{v,L}$ 的值。这部分的复杂度是当前轻子树深度的，因此所有轻边加起来复杂度就是所有轻儿子深度和
- 因此两部分这样在点 i 转移 up 的时候复杂度就是所有轻儿子深度和的了。这样这部分复杂度就可以类似长剖地证明了。

算法7 - 逆元

- 如何转移轻边？
- 方案1：先将 up 和所有儿子的 dp 乘起来，求的时候再把对应轻儿子的值除掉
- 真的一定存在逆元吗？正好是模数的倍数怎么办？
- 取决于算法会被卡12-20分
- 方案2：求出前缀和和后缀和，然后把对应的前缀和和后缀和乘一下就行了

算法7

- 结合之前所述，即可将复杂度优化到 $O(n\log n)$，$\log n$ 是来自可持久化线段树维护 dp 和 up

- 复杂度：$O(n\log n)$
- 期望得分：80分
- 结合算法6即可得到84分

算法8 - 可回退化

- 我们考虑如何把复杂度优化到线性。
- 算法7复杂度的瓶颈是可持久化线段树，我们考虑如何去掉这个。
- 首先我们发现这里不需要可持久化，只需要可回退化就行了，因为我们可以按第一次 DFS 的倒序进行第二次 DFS，每次把一个 dp 的版本（第一次DFS的顺序）往回退一个就行了。而在 up 的时候我们本身就是在一个DFS的过程上维护的，因此只需要版本栈和DFS的栈保持一致就可以了
- 这样我们就可以用可回退化线段树代替可持久化线段树，但这并不能优化复杂度

算法8 - 线段树？数组！

- 我们考虑为什么要用线段树维护，因为我们要支持区间加和区间乘。
- 但我们仔细考虑一下这个区间加和区间乘的过程，我们发现实际上是一个加只会整体加，乘只会后缀乘，因此我们考虑如何借助这部分优化复杂度
- 我们考虑给全局维护一个 $ax+b$ 的标记 a 和 b，然后在后缀乘的时候把前面部分除回去。因为需要除的部分是转移的时候的子树的深度，因此复杂度也是对的。但是这里也需要处理后缀乘 0 的问题。我们再维护一个后缀赋值标记，对于要把后缀乘 0 的时候，我们先把标记向后推的乘 0 的位置，然后打上一个后缀赋值为 $-\frac{b}{a}$的标记即可。
- 将 dp、up、前缀和和后缀和都用这么个结构维护即可
- 这个结构上修改、查询都是 $O(1)$ 的，因此复杂度就变成 $O(n)$ 了

算法8 - 逆元

- 还有最后一个问题：在后缀乘 a 的时候需要把前面部分除以 a，这里需要求 a 的逆元 a^{-1}
- 这里我们仔细分析一下乘的 a 会是啥，我们发现它一定是 $dp_{v,dep(v)}$ 或者一些 $dp_{v,dep(v)}$ 的乘积。而 $dp_{v,dep(v)}$ 就是我们在算法4中算的不考虑 L 的限制的联通块个数。因此我们可以先 $O(n)$ 扫一遍算出所有 $dp_{v,dep(v)}$，然后用预处理 $1\sim n$ 的前缀和的方法预处理所有 $dp_{v,dep(v)}$ 的逆元，复杂度 $O(n+\log p)$。
- 然后每次维护 $ax+b$ 的标记 a 和 b 的同时维护 a^{-1}，即可将这部分做成线性。

算法8 - 线性预处理逆元

- 我们考虑如何线性预处理 $a_1, a_2, a_3, \ldots, a_n$ 的逆元。我们记

$$b_i = \prod_{i=1}^{i-1} a_i$$

- 则有 $b_i = b_{i-1} \times a_i$，$b_i^{-1} = b_{i+1}^{-1} \times a_{i+1}$。所以只需要算 b_n^{-1}，即可线性推出其它的。
- 最后再用 $a_i^{-1} = b_i \times a_{i-1}$ 计算 a_i^{-1} 即可。

算法8

- 综合以上所述，就可以得到一个线性的做法。

- 复杂度：$O(n)$
- 期望得分：100分

算法9

- 关于 $k=1$ 的情况，有一个特殊的做法。
- 我们假设以 1 为根，考虑在每个联通块深度最低的位置统计答案。设 $dp_{i,j}$ 表示以 i 为根的子树，深度 $\leq j$ 且不存在超过 $2L-1$ 的直径的联通块个数，然后类似前面的用长链剖分优化，用数据结构维护转移即可。

- 期望得分：40分

花絮

题目背景杂谈

- 背景中的两句话来自鲁迅的《记谈话》
  - 我们所可以自慰的，想来想去，也还是所谓对于将来的希望。
  - 希望是附丽于存在的，有存在，便有希望，有希望，便是光明。
- 后面的一段话是这样的：
  - 黑暗只能附丽于渐就灭亡的事物，一灭亡，黑暗也就一同灭亡了，它不永久。
  - 然而将来是永远要有的，并且总要光明起来；
  - 只要不做黑暗的附着物，为光明而灭亡，则我们一定有悠久的将来，而且一定是光明的将来。
- 无论省选对于你意味着什么，都希望你能拥有光明的将来。
- 也祝愿OI能有光明的将来。

谢谢大家