《新生舞会》解题报告

南京外国语学校 蒋承佑

题目大意

给定一个有根树森林，两位玩家轮流操作，无法操作的人输。

每次操作当前玩家需要选择一个节点 u，并删去 u 所在的连通块的根 r 到节点 u 的唯一简单路径上的所有节点。

操作后新产生的连通块的根为该连通块中原先到 r 距离最小的节点。

你需要求出初始局面的 SG 值。

数据范围

每棵有根树的节点都由 1 到 n 编号，其中 n 为该树的节点数，节点 1 为根。

下文记 p_i 为节点 i 的父亲，$\sum n$ 为有根树森林的节点总数。

对于所有测试点，$1 \le \sum n \le 2 \times 10^6, 1 \le p_i < i$。

子任务编号	$\sum n \le$	特殊性质	分数
1	5000	无	10
2	2×10^6	p_i 在 $[1, i-1] \cap \mathbb{Z}$ 中均匀随机	10
3	2×10^5	无	30
4	2×10^6	无	50

单个测试点时间限制 3 秒，空间限制 128 MB。

解题过程

下文记 $son(u)$ 为 u 的儿子的点集，$sub(u)$ 为 u 的子树的点集，$path(u, v)$ 为 u 到 v 路径上的点集，$\operatorname{mex} S$ 为最小的不属于 S 的非负整数，运算 $\oplus$ 为按位异或。

Subtask 1 ~ 2

根据 SG 定理，我们只需要求出单有每棵有根树的局面的 SG 值并异或起来即可得到答案。

考虑如何求解某棵有根树的 SG 值。设 f_u 表示节点 u 的子树的 SG 值，目标为计算 f_1，由 SG 函数的定义有：

$$f_u = \operatorname{mex}\{\bigoplus_{p_w \in path(u,v)} f_w \mid v \in sub(u)\}$$

计算 f_u 时，不妨假设我们已经求出了所有满足 $v \in sub(u) \backslash \{u\}$ 的 f_v。

我们从 u 开始向子树进行深度优先搜索，设当前搜索到的节点为 v，不难在搜索过程中维护 $\bigoplus_{p_w \in path(u,v)} f_w$。

深度优先搜索时间复杂度为 $O(|sub(u)|)$，因此计算所有 f_u 的时间复杂度为 $O(\sum_u |sub(u)|)$。

该算法在 Subtask 1 中时间复杂度为 $O(n^2)$，空间复杂度为 $O(n)$。

该算法在 Subtask 2 中由于期望树高为 $O(\log n)$ 因此期望时间复杂度为 $O(n\log n)$，空间复杂度为 $O(n)$。

Subtask 3

记 $sum_u=\bigoplus_{v\in son(u)}f_v$，则上文的转移式可以变换为：

$$f_u=\operatorname{mex}\{sum_u\oplus\bigoplus_{w\in path(u,v)\setminus\{u\}}(f_w\oplus sum_w)\mid v\in sub(u)\}$$

解法 1

不妨使用 01-trie 合并优化转移。

具体地，每个节点处都维护一棵 01-trie，初始每棵 01-trie 都为空。

从 n 到 1 遍历每个节点，遍历到节点 u 时，假设我们已经求出了满足 $v>u$ 的 f_v，并且节点 u 处的 01-trie 维护的数集为 $\{\bigoplus_{w\in path(u,v)\setminus\{u\}}(f_w\oplus sum_w)\mid v\in sub(u)\setminus\{u\}\}$。

我们往节点 u 的 01-trie 中插入单个数 0，再整体异或上 sum_u，此时 f_u 即为该 01-trie 维护的数集的 mex，不难 $O(\log n)$ 求出。

求出 f_u 后，我们再把 01-trie 整体异或上 f_u，然后若 $u\neq 1$ 则把该 01-trie 合并到节点 p_u 维护的 01-trie 上。

不难归纳证明上述流程的正确性。

合并两棵 01-trie 的时间复杂度与「合并前两棵 01-trie 的节点数之和」减去「合并后 01-trie 的节点数」同阶，因此按照树形结构合并若干棵 01-trie 的总时间复杂度不超过初始节点数 $O(n\log n)$。

该算法时间复杂度为 $O(n\log n)$，空间复杂度为 $O(n\log n)$。

解法 2

不妨使用树上启发式合并优化转移。

具体地，先进行轻重链剖分，全局维护一棵 01-trie。

我们希望计算 f_u 前 01-trie 为空，计算 f_u 后 01-trie 维护的数集为 $\{\bigoplus_{w\in path(u,v)}(f_w\oplus sum_w)\mid v\in sub(u)\}$。

计算 f_u 时，我们先遍历 u 的所有轻儿子 v，对于每个 v 递归计算 f_v，递归返回后清空 01-trie。

接着递归计算 u 的重儿子的 SG 值，递归返回后再次遍历 u 的所有轻儿子 v，通过从 v 向子树进行深度优先搜索向 01-trie 插入数集 $\{\bigoplus_{p\in path(v,w)}(f_p\oplus sum_p)\mid w\in sub(v)\}$。

然后向 01-trie 中插入单个数 0，再把 01-trie 整体异或上 sum_u，此时 f_u 即为该 01-trie 维护的数集的 mex，不难 $O(\log n)$ 求出。

求出 f_u 后，再把 01-trie 整体异或上 f_u，即可返回。

不难归纳证明上述流程的正确性。

由树上启发式合并的分析过程，该算法共进行了 $O(n\log n)$ 次向 01-trie 插入单个数，时间复杂度为 $O(n\log^2 n)$。

由 f_u 的转移式不难发现 $f_u\leq|sub(u)|$，因此全局维护的 01-trie 值域仅为 $O(n)$，不动态开点空间复杂度为 $O(n)$。

Subtask 4

解法 1

不妨把 Subtask 3 解法 1 中的 01-trie 改为压缩 01-trie。

具体地，压缩 01-trie 维护了压缩前 01-trie 所有叶节点与根的虚树，因此压缩 01-trie 的节点数不超过叶节点数两倍。

在 std 的实现方法中，压缩 01-trie 的每个节点维护了四个 32 位整型 ls, rs, up, tag，分别表示该节点的左儿子、右儿子、从父亲走到该点的二进制表示、以及该节点子树整体异或的标记。特别地，我们用 tag 的最高非 0 位表示该节点的深度，用 $ls = rs = 0$ 表示该点子树是满的。

不难发现压缩 01-trie 也支持整体异或上某个数，与合并两个压缩 01-trie。

该算法时间复杂度为 $O(n \log n)$，合并时回收废弃节点，空间复杂度为 $O(n)$。

解法 2

不妨用深度优先搜索实现 Subtask 3 解法 1，**并在向子树递归时首先递归到重儿子。**

注意到包含了 a 个数的 01-trie 大小为 $O(a(\log n - \log a + 1))$，这是因为前 $O(\log a)$ 层最多包含 $O(a)$ 个节点，剩下的层最多包含 $O(a(\log n - \log a))$ 个节点。

把一棵 01-trie 合并到另一棵 01-trie 上后，前者就被清空了。因此在深度优先搜索的过程中，节点 u 上的 01-trie 非空当且仅当当前节点为 u 或当前节点在 u 的某个轻儿子子树内。

进而从当前节点开始往根走，每次经过的非空 01-trie 所在节点的子树大小总不小于上一次经过的两倍，即 01-trie 每时每刻总节点数不超过 $\sum_{i=0}^{\lfloor \log_2 n \rfloor} \frac{n}{2^i}(\log n - \log \frac{n}{2^i} + 1) = O(n)$。

因此该算法时间复杂度为 $O(n \log n)$，合并时回收废弃节点，空间复杂度为 $O(n)$。

空间限制较小，应通过精细实现避免递归深度达到 $O(n)$。

致谢

感谢欧阳达晟同学、屠佳铭同学以及陈哲章同学在命题过程中与本人的交流与讨论。

参考资料

1. Entropy Increaser - 随机父节点树的期望树高
2. 国家集训队 2022 论文集 - 陈知轩《浅谈信息学竞赛中的空间优化问题》
3. pp-orange - 线段树合并特殊情况下的线性空间做法