《Axium Crisis》解题报告

杭州学军中学教育集团文渊中学 叶开

引言

本题原本为我给为我们联考 NOI 模拟赛投的 T2，结果最后因为太难被胡伟栋老师放到了 T3。

模拟赛时本题最高分来自冯政玮同学，97pts，是一个乱搞的贪心随机化，当时甚至直接过了所有样例，只在 $o=4$ 数据点上卡了 3 个（当时并没有绑 subtask）。

后来因为要投集训队互测，我向他要了一份代码，把这种做法构造攻击掉了，使得现在的数据是较强的。

但还有一种来自刘海峰同学的乱搞做法现在还是能过这题的……而且在现行数据范围内（$n \leq 18$）大概率没法卡……

我将在解题过程的最后介绍一下这个算法，欢迎同学们赛后去 hack 这种做法。

一开始的题面中边权均是确定的，也就是 $0 \leq w \leq 1$。对于这个，周康阳同学给出了一个 $O(2^n \mathrm{poly}(n))$ 的做法。

原本就想直接放出来，结果我仔细思考了一下这个做法，发现可以直接推广到现在这个问题并通过一些优化做到相同复杂度。

于是觉得好像这个复杂度分析还蛮有意思的，然后就把这个题目出出来了。希望大家会喜欢这道题目。

赛时为了避免卡常，并没有往正式数据里全塞完全跑满的（链）。

题目大意

给定一颗 n 个节点的树，节点编号 $0 \sim n-1$。

边有边权，边权一般为 0 或者 1；但有的边的边权还未确定。

你要给每条未被确定边权的边确定一个 0 或者 1 的边权，然后从树上取出若干条有向路径，使得这些链两两之间满足边不相交。

然后你会把这些路径插入一颗 0/1-Trie，你希望最大化这颗 0/1-Trie 上的节点数。

需要构造方案。

数据范围

时间限制：10s。

空间限制：2GB。

本题使用 Special Judge。

本题每个测试点内部有 T 组测试数据。

对于所有的数据，保证 $2 \le n \le 18$，$1 \le T \le 3000$。

同时我们采用参数 o 表示该组数据保证一定的特殊性质，$0 \le o \le 4$。即，我们设输入的边权为 w，那么有如下特殊性质：

- $o = 0$ 时，不保证特殊性质。
- $o = 1$ 时，保证输入中 $w = 0$。
- $o = 2$ 时，保证输入中 $w = 2$。
- $o = 3$ 时，保证输入中 $w = 0$ 或 $w = 1$。
- $o = 4$ 时，保证输入中 $w = 0$ 或 $w = 2$。

对于单个测试点，你可以选择均输出方案 / 均不输出方案，不输出方案则获得当前测试点 80% 的分数。

具体的数据规模分布可以见下表，各子任务等分，即满分均为 5pts。其中形如 l, r 的一列对应的数据表示 $l \le n \le r$ 的数据组数，留空表示无额外限制。

各子任务捆绑评测，其分数为该子任务各测试点分数最小值。子任务依赖意味着只有所依赖的子任务分数均非 0 才会评测当前子任务，且分数与所依赖的子任务也取最小值。o 的含义将在之后注明。

子任务	$2,4$	$5,6$	$7,8$	$9,11$	$12,14$	$15,17$	$18,18$	o	子任务依赖
1	$\leq 10^3$	$=0$	$=0$	$=0$	$=0$	$=0$	$=0$	$=0$	无
2		≤ 15	$=0$	$=0$	$=0$	$=0$	$=0$	$=0$	1
3		≤ 500	≤ 10	$=0$	$=0$	$=0$	$=0$	$=0$	2
4		$\leq 10^3$	≤ 50	≤ 10	$=0$	$=0$	$=0$	$=2$	无
5		$\leq 10^3$	≤ 50	≤ 10	$=0$	$=0$	$=0$	$=3$	无
6		$\leq 10^3$	≤ 50	≤ 10	$=0$	$=0$	$=0$	$=4$	4
7		$\leq 10^3$	≤ 50	≤ 10	$=0$	$=0$	$=0$	$=0$	$3,5,6$
8			$\leq 10^3$	≤ 60	≤ 10	$=0$	$=0$	$=2$	4
9			$\leq 10^3$	≤ 60	≤ 10	$=0$	$=0$	$=3$	5
10			$\leq 10^3$	≤ 60	≤ 10	$=0$	$=0$	$=4$	$6,8$
11			$\leq 10^3$	≤ 60	≤ 10	$=0$	$=0$	$=0$	$7,9,10$
12				≤ 300	≤ 30	≤ 10	$=0$	$=2$	8
13				≤ 300	≤ 30	≤ 10	$=0$	$=3$	9
14				≤ 300	≤ 30	≤ 10	$=0$	$=4$	$10,12$
15				≤ 300	≤ 30	≤ 10	$=0$	$=0$	$11,13,14$
16				≤ 500	≤ 40	≤ 15	≤ 5	$=1$	无
17				≤ 500	≤ 40	≤ 15	≤ 5	$=2$	12
18				≤ 500	≤ 40	≤ 15	≤ 5	$=3$	$13,16$
19				≤ 500	≤ 40	≤ 15	≤ 5	$=4$	$14,16,17$
20				≤ 500	≤ 40	≤ 15	≤ 5	$=0$	$15,18,19$

实际互测时由于数据资源限制，部分子任务被去除了。

解题过程

我们分各档部分分介绍过去。

$n \leq 4$

供手玩的部分分。

对于链的情况，答案显然就是 n。

对于 $n = 4$ 的菊花，当三条边都是 0 或者都是 1 时答案为 3，否则为 4。

输出方案也是容易的。

$n \leq 6/n \leq 8$

写个暴力搜一搜，把剪枝拉满，然后再松一松，就过了！

一些乱搞也可以过这档分。

如果足够松可能也可以过 $n \leq 11$ 的几个点，毕竟 $9 \leq n \leq 11$ 此时只有不超过 10 个。

$w = 0$

答案即为直径上的点数。

$w = 2$

答案只和剖出来的链长集合有关。

最后的方案一定形如存在一个 d，使得长度 $\geq d$ 的路径 $\geq 2^d$ 条，长度 $> d$ 的路径 $< 2^{d+1}$ 条。

而对应的答案即为将长度 $\geq d$ 的路径把长度减去 d 之后的长度之和，再加上 $2^{d+1} - 1$。

枚举 d，然后 dp 一下，可以做到小常数 $O(\text{poly}(n))$。

一些乱搞也可以过这档分。

$w \in \{0, 1\}$

也就是本题最初的题面。

容易去想各种贪心，但是你仔细想想就会发现这个东西直接贪根本贪不出来。

我们把所有 $n(n-1)$ 条路径都提取出来，把路径上的字符串按字典序全部排序，然后求出相邻两项的 lcp 长度。

一个答案集合的方案合法等价于任意两条对应的边集不相交，其权值为这些串的总长减去字典序上相邻串的 lcp 长度。

直接按字典序状压 dp，状态里记录上一条选择的路径，以及当前边集。

每次选择一条边时暴力更新当前边集的超集的答案。

容易做到 $O^*(2^n)$。

具体的，直接暴力 dp 可以做到时间复杂度 $O(n^4 2^n)$。应该可以通过测试点 $25 \sim 26$，常数如果足够好甚至可以直接通过测试点 $35 \sim 36$。

不过对这个东西，一个菊花即可卡满复杂度。怎么继续优化？

考虑直接在状态中记录下上个被选的串和当前串的 lcp 长度，每次转移前再把 lcp 长度和当前相邻两个串的 lcp 长度取个 min，这样每次转移复杂度就降下来了，暴力实现的复杂度显然不超过 $O(n^3 2^n)$。

到目前为止，输出方案的部分是比较简单的，故按下不表。

一些乱搞也可以过这档分。

拼上前面的所有暴力，再写个输出方案，可以获得 60 分。

$w \in \{0, 2\}$

这档分实际上没有找到什么比较好的做法，主要是用来卡错解用的。

说不定有什么乱搞能在这组数据上有比较良好的表现。

$w \in \{0, 1, 2\}, c = 0$

类似于 $w \in \{0, 1\}$ 时的做法，我们提取出所有路径，暴力枚举边集上所有可能的边权情况，由于边集不重所以必然不会引发矛盾。

这样总的边集数目达到了 $O(n^2 2^n)$ 级别！

但真的只能分析出 $O(n^2 2^n)$ 吗？

注意到我们每次删除一个叶子，并且统计经过该叶子的路径的贡献，那么每次统计了不超过 2^{n+1} 种选法，总选法数不超过 2^{n+2}。

即，我们分析出了 $O(2^n)$ 的上界，且这个在链上卡得最满。

直接调用前面的第一个算法，复杂度 $O(8^n)$；第二个做法，复杂度 $O(n4^n)$。

然后，我们注意到，一段长度为 j 的路径，其边集的超集个数是 2^{n-j-1} 的，而其本身各边边权的选法数是 $O(2^j)$ 的，所以其花在转移上的总复杂度会是 $O(n2^n)$ 的！

我们考虑**滚动数组**，那么每次就不用把属于的 dp 数组复制一遍，因此转移复杂度总不会超过 $O(n^3 2^n)$！

那么现在的复杂度瓶颈在哪？

注意到在和 lcp 取 min 的过程中，我们每次要把 dp 数组的一整行下传，所以复杂度退化成了 $O(4^n)$！

考虑**不直接下传一整行**；根据我们之前的分析，dp 数组中有效的位置总数实际上是很少的。

我们直接暴力枚举前一个转移串，将其超集的位置全部下传转移。

由于一个每个串最多被更新 n 次 lcp，该部分下传时，枚举的复杂度是 $O(n^3 2^n)$ 的。复杂度分析和上一部分分析是类似的。

这样就做到了 $O(n^3 2^n)$ 的总时间复杂度，空间复杂度 $O(n2^n)$，直接实现可以获得 80 分，拼上前面的各档暴力可以则多于 90 分。

$w \in \{0, 1, 2\}, c = 1$

刚刚的滚动数组 dp 怎么输出方案？

由于现在虽然有效状态数仍然很少，但总状态数依旧很多，我们不能简单地直接记录是从谁转移来的。

一种方法是哈希表，但是无疑时空常数巨大。更何况这么大数据下冲突是显著的。

考虑使用「可回退化数据结构」，也即一个栈，来记录当前所有已有的 dp 转移。然后反推方案时直接回溯到那个时刻即可。

考虑分析一下空间，我们会发现下传 lcp 部分的修改次数还没有被分析出优于 $O(n^3 2^n)$。

接下来证明该部分修改的位置数是 $O(n^2 2^n)$ 的。

考虑对于每一条长为 j 的路径分析下传次数。

观察到，对于同样的路径集合，在保持相同 lcp 时，其超集只会被下传一次；且之后会始终保持 lcp 相同。

对于每种路径，每次下传相当于是把其中最大的若干种 lcp 降到相同值且带来改变的 lcp 种类数的贡献。

每次可能会加入一个 $=j$ 的路径，最多加入 2^j 次。最坏情况下其 Trie 为一颗深度为 j 的满二叉树。

由于深度为 j 的满二叉树除叶子外只有 2^j-1 个节点，因此最多下传 2^j-1 轮。

而单轮下传会带来 2^{n-j-1} 次下传单个位置的贡献，因此会下传不超过 2^{n-1} 个位置。

由于本质不同的路径集合只有 $n(n-1)/2$ 种，因此总共会下传不超过 $n(n-1)2^{n-2}$ 个位置。

每次修改会下传两个位置，即下传前的位置与下传到的位置，因此修改不超过 $n(n-1)2^{n-1}$ 个位置，也即 $O(n^2 2^n)$。

然后对于前面更新超集答案的部分，我们也可以分析出不超过 $n(n-1)2^{n-1}$ 次修改。

因此，总修改位置数不会超过 $n(n-1)2^{n-1}$，也就是可回退化栈大约要开 8×10^7 左右，仔细算一下会发现空间完全够用。

不过实际上经测试极限数据下栈里只有 2×10^7 左右个元素。我也不知道能不能构造更高的结果。

最终时间复杂度 $O(n^3 2^n)$，空间复杂度 $O(n^2 2^n)$，可以获得 100 分。

一种乱搞做法

考虑在提取路径时，只对前 3 条边中的 2 暴力枚举选 0 还是选 1，后面的边直接随机一个 0/1（事实上直接钦定为 0 似乎也可以），然后直接暴力状压 dp。

这个玩意可以过所有正式数据，跑得还很快。甚至把 3 改成 2 都行。

经过长达 3 天的思考与构造，最后得到的结果是：这玩意完全卡不掉。

bonus：本题的一些扩展

对于特殊性质 $3,4$ 有没有什么多项式复杂度做法？
本题有没有什么常数较小的 $O(n^{2.5}2^n)$ 做法？（注：$\binom{n}{\lfloor n/2 \rfloor} = \Theta(2^n/\sqrt{n})$）
听周康阳同学说本题可以直接在虚树上 dp，做到 $O(n^2 2^n)$ 的总复杂度，是真的吗？

参考资料

感谢周康阳同学对本题 $o = 3$ 的做法的提供。
感谢冯政玮同学提供了一种乱搞做法，使得本题数据大增强。
感谢刘海峰同学提供了卡不掉的乱搞做法。