2018 年全国青少年信息学奥林匹克
浙江省队选拔赛第一试题解

竞赛时间：3 月 23 日 8:00 – 13:00

题目名称	线图	历史	迷宫
目录	line	history	maze
可执行文件名	line	history	maze
输入文件名	line.in	history.in	maze.in
输出文件名	line.out	history.out	maze.out
每个测试点时限	2s	2s	2s
内存限制	512MB	512MB	512MB
测试点数目	10	10	10
每个测试点分值	10	10	10
是否有部分分	否	否	否
题目类型	传统型	传统型	传统型
是否有附加文件	是	是	是

提交源程序必须加后缀

对于 C++ 语言	line.cpp	history.cpp	maze.cpp
对于 C 语言	line.c	history.c	maze.c
对于 Pascal 语言	line.pas	history.pas	maze.pas

编译开关

对于 C++ 语言	-O2 -lm	-O2 -lm	-O2 -lm
对于 C 语言	-O2 -lm	-O2 -lm	-O2 -lm
对于 Pascal 语言	-O2	-O2	-O2

1 线图

考虑 $L^k(G)$ 中的每一个点在 G 中代表的形状。其中：

- $L(G)$ 中的每一个点代表的是 G 中的一条边
- $L^2(G)$ 中的点代表的是 G 中一个两条边的链
- $L^3(G)$ 中的点代表的是 G 中的一个三元环或者一条三条边的链。

以此类推不难发现，在 $L^k(G)$ 中的每一个点对应的是 G 中的一个不超过 k 条边的联通导出子图，其中要注意的是：

- 可能有很多个点对应同一个导出子图，例如令 G 是一个三元环，那么 $L^3(G)$ 也是一个三元环，这时每一个点都对应着 G 本身。
- 可能有一些点对应小于 k 条边的联通导出子图，同样用 G 为三元环举例，$L^4(G)$ 是一个三元环，而每一个点对应的仍然是 G，G 只有三条边。

这题的特殊性在于，原图是一棵树 T，因此 $L^k(T)$ 中的每一个点对应的是一个不超过 k 条边，即 $k+1$ 个点的树。一个简单的结论是：两个结构相同（即同构）的导出子图，它们在 $L^k(G)$ 中对应的节点个数一定也是相同的。因此就有了一个初步的算法：

- 枚举不超过 $k+1$ 个点的所有不同构的有根树 T_i（当然枚举不同构的无根树也行，但是无根树的在树上的嵌入可能还是要转化成有根树来做）。
- 计算 T_i 在 $L^k(T)$ 中对应的节点个数 w_i 以及在 T 中的出现次数 t_i。
- $\sum_{T_i} w_i t_i$ 就是答案。

枚举不同构的有根树可以枚举括号序列然后用树哈希来去重，因此主要考虑给定 T_i，如何求解 w_i 和 t_i。

1.1 求解 w_i

我们可以求出 $L^k(T_i)$ 节点数，每一个节点都对应着 T_i 的一个联通导出子图。可以直接 $O(2^{|T_i|})$ 枚举所有导出子图然后容斥出对应 T_i 的节点数，即 w_i。

容斥的复杂度大约是 $O(S2^k k)$，其中 S 为不同构的有根树个数，并不是瓶颈。关键在于如何求 $L^k(T_i)$ 的节点数，直接暴力 L^k 的复杂度大约是 $O(k^k)$，需要进行一定的优化。

对于一张 n 个点 m 条边的无向图 G，我们可以 $O(m)$ 计算出 $L(G), L^2(G), L^3(G)$ 以及 L^4G 的点数：

- $L(G)$ 的点数为 m。
- $L^2(G)$ 的点数为 $\sum_{i\ inV} \binom{d_i}{2}$，其中 d_i 是第 i 个点的度数。
- $L^3(G)$ 的点数为 $\sum_{(u_i,v_i)\in E}(d_{u_i}-1)(d_{v_i}-1)+\sum_{i\in V}\binom{d_i}{3}$。

- $L^4(G)$ 的点数要稍微难算一些。考虑到 $L^4(G) = L^3(G)$，因此可以用 $L^3(G)$ 的式子来计算 $L^4(G)$。首先可以计算出每一条边在 $L(G)$ 中的度数，即 $d_{u_i} + d_{v_i} - 1$，接着计算出每一个点的所有边在 $L(G)$ 中的度数和 D_i，代入 $L^3(G)$ 的和式就能 $O(m)$ 计算了。

这样计算 $L^k(T_i)$ 的值就只需要计算 $L^{k-4}(T_i)$，时间上可以接受。

1.2 求解 t_i

有根树的嵌入是一个树形 DP 问题。令 $f_{i,j}$ 为第 i 个点为第 j 种有根树的根的嵌入方案数。

考虑第 j 种有根树的所有孩子，它的每一个孩子也是有根树，记为 a_i 至 a_m，那么在嵌入时，i 的孩子必须也有 m 个点分别对应这 m 棵子树，简单的状压 DP 即可。要注意重复计数，直接 DP 可能需要除去第 j 种树的自同构个数。

直接 DP 的时间复杂度约为 $O(Sn2^k)$，复杂度略高（虽然实际上并不会满）。一个简单的优化是，嵌入时不考虑叶子节点，只考虑大小大于 1 的子树，在嵌入结束后用组合数计算叶子节点的对应，这样复杂度就降为了 $O(Sn2^{\frac{k}{2}})$。

最后将 $\sum_{T_i} w_i t_i$ 累加起来就是答案了。

2 历史

问题的本质是，给出一棵树，给定每一个点的 access 次数，计算轻重链切换次数的最大值。

考虑没有修改的时候如何计算答案，考虑在第 i 个点处的轻重链切换次数的最大值，设它自己的 access 次数为 A_0，它的每一个孩子的子树内的总 access 次数为 A_i。

观察在什么时候 i 向下的边会发生轻重链的切换，影响 i 的只有 i 子树内的节点的 access，因此我们只考虑这些 access 之间的相对顺序 x_1 至 x_k。对于相邻的两次 access x_j 和 x_{j+1}，如果他们都在 i 的同一个子树内或者都是 i，那么这时 i 处不会发生轻重链切换，否则 i 处会发生一次切换。

因此为了最大化 i 处的轻重链切换次数，问题转化成了，有 $m+1$ 种颜色的小球，第 i 种颜色有 A_{i-1} 个，要求把所有小球摆成一列，最大化左右小球的颜色不同的间隔数。这是一个 trivial 的问题，令 $t = \sum_{i=0}^{m} A_i, h = \max_{i=0}^{m} A_i$，那么答案就是 $\min(t-1, 2(t-h))$，即 $2h \geq t+1$ 时，答案就是 $2(t-h)$，否则答案就是 $t-1$。

不难发现每一个节点是独立的，因此只要把每一个点的切换次数的最大值累加，就是答案了。这样就可以在 $O(n)$ 的时间内计算出没有修改时的答案。

考虑如何处理修改，令 f_i 为节点 i 子树中的 access 总次数，考虑 i 到它的父亲 F 之间的边，如果 $2f_i \geq f_F + 1$，那么我们把这条边标记为实边，其余边标记为虚边，不难发现每一个点往下最多只有一条实边，即所有实边在树上是若干条链。

现在我们把 a_j 加上 k，这时会影响 j 到根路径上所有点的切换次数以及边的虚实关系。不难发现所有实边依然还是实边，且对于一条实边 (i, F) 来说，如果 i 子树内的 access 总数增加了，F 的最大切换次数是不变的（因为 $t-h$）不变。因此我们只需要考虑路径上的虚边就行了。

一个结论是，任何一个点 i 到根路径上，最多只有 $O(\log \sum a_i)$ 条虚边（以后简记为 $O(\log n)$），因为对于一条虚边 (i, F)，f_F 至少是 f_i 是两倍，而 $f_1 = \sum a_i$，因此经过的虚边数

为 $O(\log \sum a_i)$。因此我们可以枚举每一条虚边，然后暴力修改它的虚实以及它对答案的贡献。

唯一的问题变成了如何在树上找到所有的虚边，一种方法是用树链剖分来维护所有边，然后每一次查找时就在线段树/树状数组上二分，这样的时间复杂度是 $O(n \log^2 n)$，常数较好可能可以通过此题。

复杂度更低的做法是用类似 LCT 的方法，对于每一条全是实边的链，用一棵 splay 去维护它，然后用类似 LCT access 的方法来进行一次修改，唯一的不同是在 access 的过程中经过的虚边不会全变成实边。

因为在整个过程中，splay 的总次数是 $O(n \log n)$ 的，因此单旋的次数为 $O(n \log n)$，套用 LCT 的时间复杂度证明可以得到总的时间复杂度为 $O(n \log n)$，可以轻易的通过此题。

3 迷宫

问题的本质是，计算能识别所有 m 进制下 K 的倍数的确定性有限状态自动机（DFA）的最小点数。当然这题并不了解自动机相关的知识也可以做。

首先可以轻易的得到答案的一个上界 K。我们建 K 个节点（编号 0 到 $K-1$）依次代表模 K 余 i，那么点 i 的第 j 条出边连向 $im+j \mod K$。不难发现这样的自动机一定满足题目的条件。

这个自动机是有冗余的，如果我们能把所有冗余的点给去掉，那么就能得到答案了。去掉冗余的过程相当于计算上述自动机的等价类个数：两个点 i, j 是等价的当且仅当 $\forall s$，从 i 出发能接受 $s \iff$ 从 j 出发能接受 s。

定义 $f(i)$ 为从 i 出发，能接受的最短串的长度，同时定义 $g(i)$ 为 $i \times m^{f(i)} \mod K$。举例来讲，当 $m = 2, K = 10$ 的时候，0 到 7 的 $f(i)$ 与 $g(i)$ 如下：

i	0	1	2	3	4	5	6	7	8	9
$f(i)$	0	3	2	4	4	1	3	2	4	4
$g(i)$	0	8	8	8	4	0	8	8	8	4

不难发现两个点 i, j 等价的充要条件是 $f(i) = f(j)$ 且 $g(i) = g(j)$。考虑如何利用这一点来计算等价类的个数：

首先如果 $(m, K) = 1$，那么显然答案为 K，我们来考虑不互质的情况：

- 最开始 0 肯定是一个独立的等价类，我们手上还保留数 1 至 $K-1$。
- 把这些数都在模 K 意义下乘 m，如果两个数乘 m 模 K 相同，那么这两个数一定处于同一个等价类，因此把这些数进行去重。令 $d = (K, m)$，这时所有数一定都是 d 的倍数。
- 在剩下的数中，每一个大于 $K - m + 1$ 的数以及 0 a_i 意味着一个等价类，因为它们的 f_i 等于 1 且 $g_i = a$。把这些数加入答案并删去。
- 接着对于剩下的数，我们需要再对它们乘 m 模 K，去重后删去大于 $K - m^2 + 1$ 的数以及 0，以此类推。

我们要做的就是快速的模拟这个过程，令函数 $f(L,m,K)$ 表示当前保留数 1 到 L，进行上述过程得到的答案：

- 如果 $(m,K)=1$，那么答案就是 L，否则令 $d=(m,K)$。
- 状态 L 意味着在上一轮中，我们删掉了 (L,K) 中的数，因此在这一轮中，我们要删掉所有乘 $(K-m(K-L),K)\cup\{0\}$ 中的数。考虑如何计算这时删掉的数的个数：
  - 考虑函数 $h(i)=im \mod K$，$h(i)$ 的取值是 $[0,K)$ 中 d 的倍数且模 $\frac{K}{d}$ 循环。
  - 因此若 $L>\frac{K}{d}$，则 $h(1)$ 至 $h(L)$ 能取遍所有 $\frac{K}{d}$ 的值，因此剩下的是 $(0,K-m(K-L)]$ 中 d 的倍数，删掉了 $\frac{m(K-L)}{d}$ 个数。
  - 若 $L\le\frac{K}{d}$，则必有 $K-m(K-L)\le 0$，因此这时 $h(1)$ 至 $h(L)$ 两两不等，因此什么数都没剩下，删掉了 L 个数。
- 若此时还有数剩下，那么剩下的是 $(0,K-m(K-L)]$ 中 d 的倍数，把所有数除以 d，递归子问题 $f(\frac{K-m(K-L)}{d},m,\frac{K}{d})$。

把删掉的数全累加起来就是答案，时间复杂度 $O(T\log K)$。