20231114题解

seal

source: POI2015 PIE

拿一个数组，记录一下印章上其他的点和第一个点的相对位置。

然后我们从上往下，从左往右扫那张纸，第一个出现的点一定是最上最左的点，然后遍历可以印到的点，判断是否是'x'，是的话将'x'变为'.'（因为说了只能印一次），如果不是或者印出去等情况的话说明是不合法的，跳出就好啦。

时间复杂度$O(nm)$。

deadlock

source: ARC105E

好思维，直接研究初末状态找结论。

考察在游戏结束的时候，整个图是什么样子。不难发现其实就是分别以 1 和 n 为中心裂成了两部分，两部分都是完全图。这样下一次再连的时候，就只能把这两部分连起来，导致不合法。

不妨假定最后有 x 个点与 1 联通，有 $y=n-x$ 个点与 n 联通，那从初状态到达末状态需要经历的边数即是 $\frac{n(n-1)}{2}-m-x\times y$。显然，输赢之和这个式子的奇偶性有关。奇数则先手赢，偶数则后手赢。而其中 x 是唯一的变量，试想 x 奇偶性的变化会对原式产生什么影响。

注意到，如果 n 为奇数，x 不能影响原式的奇偶性，故只考虑 n 为偶数的情况。先手的优势在于可以通过连边主导 x 的奇偶性变化。思考这个过程，若一开始有 x' 个点和 1 连通，y' 和 n 连通。

- x' 是我们想要的目标奇偶，那么我们需要在之后的过程中保证与 x' 连接的奇数大小的连通块个数为偶数，又因为 n 为偶数，故当且仅当 x' 与 y' 奇偶相同时满足。在此情况下，如果 x' 不是我们想要的目标奇偶，也无法通过先手优势去改变 x' 的奇偶。故而，当 x' 与 y' 奇偶相同时，最终的奇偶性只又最初的奇偶性决定。
- 自然讨论当 x' 与 y' 奇偶不同时。同样的思路，利用先手优势改变 x' 的奇偶，不难发现所有对手做出的逆向改变，你都恰有更多的符合需要的连通块来抵消这样的改变。即此时先手必胜。

tree

source: CEOI 2013 Board

subtask 1

总点数不超过2^D-1，可以暴力保留所有的点和边，然后BFS。

复杂度$O(2^D+|S|)$。

subtask 2,3,4

对于两个点 u,v，设它们的一条路径上深度最小的点为 w。

设 u 的祖先中与 w 同深度的点为 x，$P:x\to w$ 是深度不变为 d 的一条路径，上面有 n 个点。则在 $u\to w$ 的路径上，纵向的边需要走至少 dep_u-dep_w-1 条，且无论怎么走不会改变这个点深度为 d 的祖先（其深度至少为 d。而所有 P 上的点为根的子树都至少经过一次，故横向边至少经过 $n-1$ 条。
而显然可以通过先纵向走到 x 再横向走到 w 来取到最小值，因此可以求出 u 到 w 的最短路长。

同理可求出 v 到 w 的最短路长，而在深度为 d 的一层中，可以取 w 为 u 或 v 的祖先来避免多余的横向边，因此可以通过枚举深度最小点的深度求出最短路。

枚举最浅点的深度 d，则纵向边的条数为 dep_u+dep_v-2d，横向边的条数为 $|u \bmod 2^d-v \bmod 2^d|$。

subtask 2,3

可以用`long long`或者数组模拟二进制或者bitset存下到达的点的变化，枚举最浅点深度可以完成。时间复杂度$O(N^2+|S|)$或者$O(\frac{N^2}{\omega}+|S|)$。

subtask 4

考虑如何用给定字符串求出其与根最短路（只通过纵向边）。

我们需要维护一个初始值为 1 的数：

1. 乘以 2 （并加上 1）；
2. 除以 2 并下取整；
3. 加减 1。

可维护一个数组 $a_1,\cdots,a_n$，表示当前数为 $\sum_{i=1}^n a_i2^{n-i}$。

对于加减法，只要改动最后一位；对于乘法，n 加上 1 并改动 a_n；对于除法，先进行**进位**操作：通过改动 a_{n-1} 和 a_n 使得维护的数不变，且 $a_n\in\{0,1\}$，此时可以将 n 减去 1。在整个字符串处理完
后，从后向前进位使得每一位都是 0 或 1。通过这种 lazy 操作，可以在 $O(|S|)$ 时间内求出。总时间复杂度 $O(N+|S|)$。

machine

$30pts$

首先注意到S长啥样不重要，重要的只是出现的字符集合。同时每一步操作要么字符减少1，要么不变，所以我们只要最小化减少的字符数量即可。

保证每个点入度最多为1，那么图退化成基环树森林，每棵基环树可以单独考虑。

先考虑树的情况，对于一个儿子均为叶子的点，若所有儿子全都出现在集合中，且自身也出现在集合中，那么必定会造成至少1的字符损失。否则可以通过一步转换成某个没出现过的儿子，然后在自身不存在的情况下将所有通向其他儿子的步骤全部执行并完全不造成损失。

而如果存在一个不为叶子的儿子，那么可以先对该儿子执行操作，此时儿子一定不再出现，那么执行自身到该儿子的边，不会造成任何损失。

因此树的情况带来的损失等于“所有儿子都是叶子，且自身和儿子们全都出现过”的点的数量。

考虑环的情况，如果环上所有点都出现过，那么第一步一定会造成1损失，但之后可以不造成任何损失；否则，可以完全不造成损失。

那么考虑基环树，若环上有点没有出现过，那么执行环上的边可以做到不带来损失。而执行时每个位置都会存在为空的时刻，此时执行该位置的子树，答案也一定是最优。

否则，环上所有点都出现过。若环上某个点具有空的儿子或非叶儿子，则可以优先执行该棵子树，之后该点上将不再出现，再使用上述做法。

否则，环上的边必然会带来至少1损失，损失后环上将出现空的点，接下来同样使用上述做法。

$100pts$

接下来考虑一般图的情况。

注意到每个点一旦执行了一条出边，那么立刻执行剩下的出边无需任何代价。而如果一个点为空，则一开始就可以执行其所有出边。因此可以把条件放宽为，执行每个出现在字符串中的点的至少1条出边（除非没有出边）。注意依然可以使用其他边，只是不强制要求。

我们定义图中的路径集合L的权值W为$|L|-s$，其中s为“在字符串中未出现，且是L中至少一条路径的末尾”的点数。并称一个路径集合L覆盖了一个点p，当且仅当p在至少一条路径中作为非结尾节点出现。

我们可以证明：

1. 若存在一个覆盖了所有出现在字符串中的点的路径集合L的权值为X，我们可以找到一个造成损失不超过X的执行方案。
2. 若存在一个造成X损失的执行方案，我们可以找到一个路径集合L覆盖了所有出现在字符串中的点，且权值不超过X。

证明1：

在路径集合中可以找到$|L|-X$条路径，他们的结尾没有在字符串中出现过，且两两不同。

对于一条路径，我们将其在所有没出现在字符串中的点处断开，断成k条子路径。

对于最后一条子路径，我们从末尾到开头依次执行边，由于末尾没有出现在字符串中，因此效果等价于整体平移，此时开头将会变为空。

考虑倒数第二条子路径，我们执行这条子路径到最后一条子路径之间的所有边（中间的点均为空），将末尾的点移动到最后一条子路径的开头。

此时这条子路径的末尾也为空，如此反复操作即可。

最终效果即为，路径上所有边全部被执行，除了最开头的点移动到了最末尾，其他点没有任何变化。

因此对于这$|L|-X$条路径，由于末尾没有出现过，且两两不同，执行后不会带来任何损失。

剩下的X条路径，他们的结尾有可能不为空，同样使用上述做法后，每条路径最多带来1次损失，因此损失不超过X。

由于L覆盖了所有出现在字符串中的点，所以执行完这些操作后每个点至少用掉了1条出边，是一个合法的方案。

证明2：

考虑按顺序执行该方案的边，设当前为$x_i\to y_i$，初始时路径集合为空。

1. x_i不存在当前的字符串中，什么都不做
2. x_i存在，且当前路径集合中存在一条以y_i开头的路径，那么将$x_i\to y_i$接在该路径的开头。
3. x_i存在，且当前路径集合中不存在y_i开头的路径，那么新增一条$x_i\to y_i$的路径。

1、2操作不会更改路径集合的权值。

3操作可能会使路径集合的权值增加1，此时说明y_i一开始在字符串中出现过或已经有路径的结尾为y_i。不论哪种情况，此时y_i一定存在字符串中，会产生1损失。

因此每次权值增加1时，一定伴随着1损失，那么权值一定不会超过损失量。

每个出现在字符串中的x_i第一次被执行的时候，一定会被加入到某条路径的开头，因此所有这些x_i都在非末尾位置出现过，即被构造的路径集合覆盖了。

证明完上述两个结论后，可以发现最小的损失量即等于覆盖所有出现在字符串中的点的路径集合的最小权值。

注意到一条路径如果经过了一个点，可以访问这个点所在的强连通分量内所有点并回到该点，且不会增加路径的权值。因此每个存在需要覆盖的点的强连通分量只需要访问一个代表点即可，这样可以将路径变为简单路径。

于是就是经典问题了。

设二分图左部A，其中包含了所有需要覆盖的强连通分量的代表点。

右部B，同样包含了强联通分量的代表点，但额外包含了所有初始未出现在字符串中的点。

A中的点u可以匹配v，当且仅当u能到达v。于是这个二分图的一个匹配即对应了一个路径集合方案。左部的未匹配点的数量即为$|L| - X$。

求该图的最大匹配即可，数据很小，各种算法都可以接受。