cerc2015 解题报告

福建省福州第一中学 董克凡

1 试题来源

http://cerc.hsin.hr/tasks/cerc2015_problems.pdf

2 Frightful Formula

2.1 试题大意

给出一个$n*n$矩阵的第一行和第一列，其余的数通过如下公式推出：

$$F[i,j] = a*f[i,j-1] + b*f[i-1,j] + c$$

求$f[n,n]$对10^6+3（一个质数）取模的结果。

$n \le 2*10^5$

2.2 算法介绍

由于这个矩阵的递推式是线性的，故可以单独考虑每一项对答案的贡献。即$F[n,n]$最终可以表示为

$$F[n,n] = \sum_{i=2}^{n} F[1,i]X_i + F[i,1]Y_i + C$$

的形式。接下来分别考虑系数与常数项的求法。

2.2.1 系数的求法

考虑$F[1,i]$如何通过递推式贡献答案。从$F[1,i]$推到$F[n,n]$的每一个合法的转移方案可以与从$(1,i)$到(n,n)的一条只能向右下方移动的路线一一对

应（由于递推式的限制是$i, j \geq 2$，所以这里要求第一步只能向下走），那么$F[1,i]$在$F[x,y]$中的系数就是$a^x b^{y-i}\binom{x+y-i-1}{y-i}$。由于模数是一个质数，所以组合数可以用阶乘$O(1)$求出。所以这一部分的复杂度为$O(n)$

2.2.2 常数项的求法

同样的，常数项也可以单独考虑贡献答案。对于每一个$F[x,y](x,y \geq 2)$，除了从$F[x-1,y], F[x,y-1]$转移的一部分，还增加了一个常数c。那么这个c对答案的贡献就是$a^{n-x}b^{n-y}\binom{n-x+n-y}{n-x}$，所以答案中的常数项应该为：

$$C = c\sum_{x=2}^{n}\sum_{y=2}^{n} a^{n-x}b^{n-y}\binom{n-x+n-y}{n-x}$$

用$i = n-x, j = n-y$替换上式中的求和下标，得：

$$\begin{aligned} C &= c\sum_{i=0}^{n-2}\sum_{j=0}^{n-2} a^i b^j \binom{i+j}{i} \\ &= c\sum_{i=0}^{n-2}\sum_{j=0}^{n-2} a^i b^j \frac{(i+j)!}{i!\,j!} \end{aligned}$$

记

$$A(x) = \sum_{i=0}^{n-2}\frac{a^i}{i!}x^i, B(x) = \sum_{i\geq 0}^{n-2}\frac{b^i}{i!}x^i$$

并且

$$C(x) = A(x)B(x) = \sum_{i=0}^{n-2}\sum_{j=0}^{n-2}\frac{a^i b^j}{i!\,j!}x^{i+j}$$

那么

$$C = c\sum_{i=0}^{2n-4} i!(C(x)[i])$$

$C(x)$是两个多项式相乘的结果，可以用快速傅里叶变换求出，故这一部分的复杂度为$O(nlogn)$

那么总的时间复杂度就是$O(nlogn)$

3 Juice Junctions

3.1 试题大意

给出一张n个点m条边的无向图，每个点的度数最多为3。求任意两点间的最大流。

$n \leq 3000, m \leq 5000$

3.2 算法介绍

由于每个节点的度数最多为3，所以任意两点间的最大流至多为3。而最大流等于最小割，所以接下来按照答案的大小分类讨论。

当两个点不联通时，答案显然为0。

若两个点联通，那么当且仅当这两个点不在同一个边双联通分量中的时候，答案是1。

若两个点处在同一个边双联通分量中，那么两点之间的最大流可能为2，也可能为3。考虑最小割，如果两个点之间的最小割为3，那也就是说，当图中删去任意一条边的时候，两个点都处在同一个边双连通分量中。否则，如果删去边e的时候两个点不在同一个边双连通分量中，那么这两个点之间只需要再删去一条边就可以使他们两个不连通，所以这时这两个点之间的最小割为2。

于是我们可以枚举每一条边，将这条边删掉，然后求出这张图的边双联通分量，这样对于每一个点就得到了m个值来表示这个点每一次所在的边双连通分量的编号。要判断两个点是否在所有删边方案中都处在同一个边双联通分量里面，就等价于判断这两个点的m个值是否全部相等。这里可以使用hash判断，也可以将所有的点以这m个值为权插入一棵字典树中，使用字典树来判断。

时间复杂度：$O(nm)$

4 Looping Labyrinth

4.1 试题大意

在一个二维平面上有一个无限大的迷宫，这个迷宫是有一块$n * m$的区域无限复制形成的。给出这一块迷宫，需要处理q个询问，每次询问输入x, y，回答从原点能否走到点(x, y)

$n, m \le 100, q \le 10^5, |x|, |y| \le 10^9$

4.2 算法介绍

首先排除从$(0,0)$出发能到达的点是有限多的情况。

虽然这个迷宫是无限大的，但是整个迷宫必然是循环的（也就是说存在一组整数(dx, dy)，使得如果能够达到点(x, y)，那么一定能够到达点$(x+dx, y+dy)$）。这是因为如果从一个格子出发能够到达另一块区域的相应位置，那么就找到了一个循环节，而区域内的点是有限的，能够到达的点是无限的，所以迷宫必然是循环的。

由于这个迷宫是有一块区域复制而成的，那么这块区域中于边界相连的部分（也就是出去这块区域中的封闭部分）的个数不会超过$n+m$。这是因为边界上的点一共只有$2(n+m)$个，两个部分之间需要至少一个位置将其分开。由于在最小循环节中不会经过同一块区域超过两次（否则就可以找到一个更小的循环节），所以最小循环节（即$|dx|+|dy|$ 的大小）不会超过$n+m$。也就是说，只需要经过不超过$(n+m)^3$ 个点，就能遍历整个循环节里能到达的所有点。所以，可以从$(0,0)$ 出发进行bfs，直到$(0,0)$这个点的等效点[1]被经过$n+m$ 次，那么对于每一个能够达到的节点，我们就遍历了它的至少一个等效点。这一部分的复杂度为$O\left((n+m)^3\right)$

接下来，将经过的$n+m$次原点的等效点两两作差，得到一些二维向量，这些二维向量就代表着一些不同的循环节的大小。若这些向量张成的空间维数为2，那么也就是说从$(0,0)$出发可以到达任意一个$(0,0)$ 的等效点（这是因为这样的一条路径必然属于这个向量空间，也就是说我们可以经过循环节的线性组合得到这条路径），那么要一个点是否可达，就等价于这个点的任意一个等效点是否可达，而我们在第一步中已经遍历了所有节点的等效点至少一次，所以这种情况可以直接判断。

如果循环节所构成的向量空间的维数为1，那么可以到达的原点的等效点就一定是$(k*dx, k*dy)$的形式。那么可以先将重点沿着$(-dx, -dy)$的方向移动到在曼哈顿距离下离原点最近的一个等效点，然后就等价于判断这个等效点在bfs中是否经过，这一部分也可以直接判断。

时间复杂度：$O\left((n+m)^3+q\right)$，这个复杂度已经可以通过这一题了。

[1]这里等效点是说这个点以及在另一份复制中与这个点位置相同的节点

但是这个复杂度仍然可以继续优化。考虑将同一个迷宫块内的联通块缩起来，这样最终能与外界（这个迷宫块的边界）联通的块的个数不会超过$n+m$个。将这些块之间跨越迷宫块的可达情况处理出来，形成一张新的图，由于在边界上的一个点最多形成一条边，所以这张图的边数是$n+m$的，那么由于在同一个联通块的点能够在不跨越迷宫块的情况下互相到达，所以这些块里面的点是等价的，所以上面的分析对这些块仍然适用，所以我们只需要在这张新的图上bfs，要求每一个块最多经过$n+m$次，那么由于图的总边数不会超过$n+m$条，所以总的bfs过程的复杂度为$O\left((n+m)^2\right)$，之后的判断与上面的做法相同。那么总复杂度为$O\left((n+m)^2+q\right)$，可以通过$n,m\le 1000$的数据。

5 Cow Confinement

5.1 题目大意

在一个平面中有n只奶牛，m朵花以及F个围栏。每个围栏都是一个边平行于坐标轴的矩形。保证围栏不交（但是可能会有嵌套的情况）。每一只奶牛只能向右或者向下移动，求每只奶牛在不穿越围栏的情况下可能会碰到多少朵花。

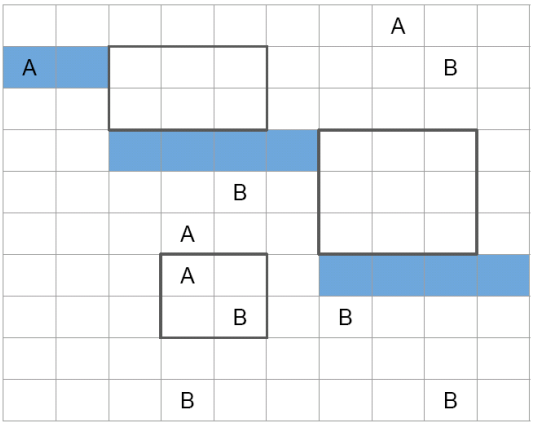
$n,m,F\le 2*10^5$

5.2 算法介绍

由于每个奶牛只能向下走或者向右走，这就导致奶牛能够到达的地方是不规则的，难以直接用数据结构维护。

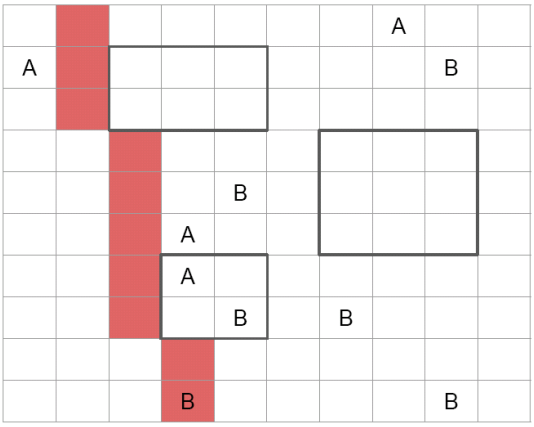
当奶牛A可以到达花B的时候，A到B的路径可能有很多条，在这里不妨定义一条特殊路径方便统计。这条路径需要满足以下的性质：如果当前A向右走合法并且向右走之后仍能满足A能够到达B，那么就规定A要继续向右走，否则A向下走。这样若A能到达B，那么就存在唯一的一条A到B的特殊路径，反之，若存在这样一条路径，那么A可以到达B。那么接下来就只需要考虑A沿着特殊路径的规则可以到达多少朵花。

这样的特殊路径可以分为两个部分，第一部分为A尽量向右走，另一部分为A需要考虑向下走来碰到一朵花，我们考虑在两部分的分界点统计答案。首先考虑第一部分路径：



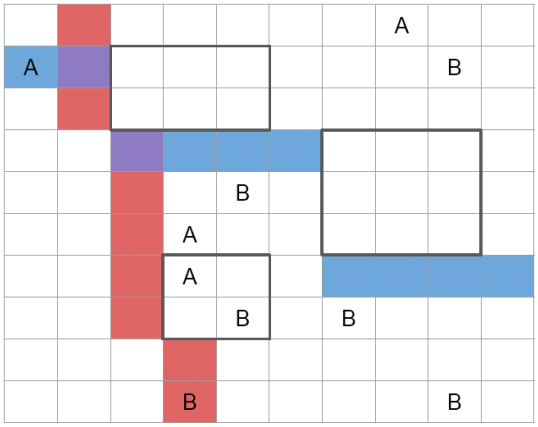
例如对于图片左上角的奶牛（用A表示）的第一部分路径就是用蓝色表示的这些格子，即出发后，尽量向右行走，直到遇到一个矩形的边界。若这个边界是矩形的左侧边界，那么只能向下绕过这个矩形，从矩形左下角的下方格子开始继续向右行走（由于在向下绕过矩形的过程中不会产生两种路径的分界点，所以在这里不标为蓝色）。若这个边界是矩形的右侧边界，那么这只奶牛就困在了这个围栏中，那么这条路径就在此结束。

路径的第二种部分就是从某个点出发向下行走碰到一朵花，并且要求这个点的右边一个点出发不能碰到这朵花。那么也就等价于从每朵花出发，只能向左上走，并且优先选择向上方行走，如图中的红色格子：

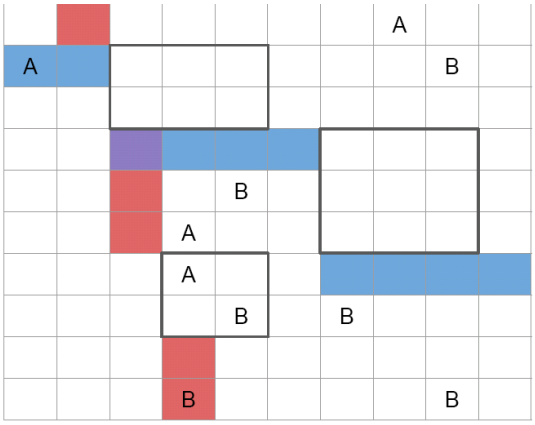


那么，如果以上两种颜色的格子有交叉，那么交叉点就是两种路径的转折点。所以对于每个奶牛，只需要求出这个奶牛对应的蓝色路径与多少朵花的红色路径有交点即可。

直接这样求交会导致某一朵花与奶牛的边界相交两次，例如下图的情况，这种情况只会在在沿着矩形边界走的时候发生，这是因为沿着矩形边界走的一整段都可以被认为是分界点：



为了避免这种情况，需要将某一个边界的初始位置更改，例如将红色边界的初始位置改为矩形左上角的左上的格子，也就是强行将分界点设为离开矩形边界的那一个格子：



经过这样的修改，每一朵花最多只会与已知奶牛的边界相交一次，所以我们只需要统计相交次数即可。当一个奶牛的出发碰到第一个矩形边界之后，剩下的边界就只与这个矩形的位置有关，与奶牛的具体位置无关，所以蓝色的路径一共不会超过$n+F$条。同理红色路径不会超过$m+F$条。经过这样的合并，红色路径就变成了有权的（权值为这条红色路径属于的花的数量）；蓝色路径形成了一个树形结构，每个奶牛是一个叶子节点，而矩形是非叶子结点，从一个奶

牛出发经过一段水平的蓝色路径会到达一个矩形边界，这个矩形就是这个奶牛对应的叶节点在树形结构中的父亲，从矩形出发同理。

接下来考虑如何求出这些路径。两种路径类似，所以这里只考虑如何处理蓝色路径。所有的蓝色路径都是一条平行于x轴的线段，而且线段的左端点已知，所以考虑如何求出每一条线段的右端点。

使用扫描线的方法，把所有的垂直于x轴的边界的上下端点以及所有的询问（也就是已知线段的左端点）按照y坐标排序，同时用一棵平衡树维护所有的垂直边界的x坐标，那么需要进行的操作就是询问大于当前询问点x坐标中x最小的垂直边界，以及插入或删除一个垂直边界，这些操作可以用STL中的set维护，以减小代码量。

那么对于树上的每一个节点，它的权就是与这个节点对应的蓝色路径相交的红色路径的权值和。由于所有的红色路径平行于y轴，所有蓝色路径平行于x轴，而且可以离线处理，故可以用扫描线加树状数组求出每个蓝色路径的权值，然后每一个奶牛能够到达的花的数量就是这个奶牛对应的叶节点到根的点权和。

令$s = n + m + F$，则时间复杂度为：$O(slogs)$