动态规划例题选讲

Dynamic Programming by Examples

清华大学计算机系　胡泽聪

目录

关于我

- 高中：湖南师大附中
- NOI2013 Au
- 现就读于THU计算机系

关于课程

通过例题学习动态规划。

关于课程

通过例题学习动态规划。

动态规划与其他知识点的区别：

- 网络流
- 数据结构

□

Little Elephant and Mouses [1]

有一个$n \times m$的网格。有一头大象，初始时在$(1, 1)$，要移动到(n, m)，每次只能向右或者向下走。有些格子中有老鼠，如果大象所在的格子和某个有老鼠的格子的曼哈顿距离≤ 1，大象就会被那只老鼠吓到。

求一条移动路径，使得吓到过大象的老鼠数量最少。$n, m \leq 100$。 □

[1] *https://www.codechef.com/JUNE13/problems/LEMOUSE*

Little Elephant and Mouses

如果多次经过一只老鼠旁边也会被吓多次，直接DP就好了。而这里只会被吓一次，我们需要往状态中加点东西。

Little Elephant and Mouses

如果多次经过一只老鼠旁边也会被吓多次，直接DP就好了。而这里只会被吓一次，我们需要往状态中加点东西。

注意到，一只老鼠最多能影响我们三步，因此只要记录上一步的走法即可。转移时分情况讨论。 □

Iahub and Permutations [2]

有一个长度为n的排列a，其中有一些位置被替换成了-1。你需要尝试恢复这个排列，将-1替换回数字。

求有多少种可行的替换方法，满足得到的是一个排列，且不存在$a_i = i$的位置。$n \le 2000$。 □

[2] http://codeforces.com/contest/341/problem/C

Iahub and Permutations

我们用一个$n \times n$的棋盘来表示一个排列，第i行第j列如果被标记，则代表数字i填在了第j个位置（$a_j = i$）。对于给定的排列，不为-1的位置已经被标记在棋盘上，而棋盘的主对角线上（$a_i = i$）不可以被标记。

Iahub and Permutations

我们用一个$n \times n$的棋盘来表示一个排列，第i行第j列如果被标记，则代表数字i填在了第j个位置（$a_j = i$）。对于给定的排列，不为-1的位置已经被标记在棋盘上，而棋盘的主对角线上（$a_i = i$）不可以被标记。

从棋盘中删去不为-1的位置的列，以及已经出现了的数字的行，记此时棋盘大小为N。不难发现，每列不可被标记的位置至多只有1个，每行也是同样。记这种位置的数量为M。

Iahub and Permutations

我们用一个$n \times n$的棋盘来表示一个排列，第i行第j列如果被标记，则代表数字i填在了第j个位置（$a_j = i$）。对于给定的排列，不为-1的位置已经被标记在棋盘上，而棋盘的主对角线上（$a_i = i$）不可以被标记。

从棋盘中删去不为-1的位置的列，以及已经出现了的数字的行，记此时棋盘大小为N。不难发现，每列不可被标记的位置至多只有1个，每行也是同样。记这种位置的数量为M。

令$f[N, M]$表示，在这样的棋盘上标记N个格子的方案数。转移方程为：

$$f[n, m] = f[n, m-1] - f[n-1, m-1]$$

边界为$f[i, 0] = i!$。

转移方程的含义为，相比起$f[n, m-1]$的状态，$f[n, m]$的状态要多一个不可标记的位置，而标记了这个位置的方案数为$f[n-1, m-1]$，因此从中减去。 □

EagleInZoo [3]

有一棵n个节点的有根树。现在依次飞来了k只鹰，想在树上休息。每只鹰初始都在根节点，然后采取如下操作：

1. 如果当前所在的节点是空的，那么占据这个节点休息；
2. 否则，如果该节点有儿子节点，那么等概率随机飞到一个儿子节点上；
3. 否则，这只鹰会放弃休息，直接飞走。

求第k只鹰最后留在树上的概率。$n \le 50$，$k \le 100$。 □

[3] https://community.topcoder.com/stat?c=problem_statement&pm=13117

EagleInZoo

首先注意到，这个问题有明显的子结构：当鹰飞到子树的树根时，整个问题和鹰初始时在根节点是一样的。

EagleInZoo

首先注意到，这个问题有明显的子结构：当鹰飞到子树的树根时，整个问题和鹰初始时在根节点是一样的。

因此我们直接令$p[x,k]$代表第k只鹰从节点x出发，最后能留在树上的概率。记节点x的儿子个数为c_x，边界情况有：

- $p[x,1]=1$；
- 如果$c_x=0$，那么对于$k>1$都有$p[x,k]=0$。

□

EagleInZoo

而对于一般情况，有下面的公式：

$$p[x,k] = \sum_{y \in child(x)} \frac{1}{c_x} \sum_{i=0}^{k-2} \binom{k-2}{i} \left(\frac{1}{c_x}\right)^i \left(1-\frac{1}{c_x}\right)^{k-2-i} p[y,i+1]$$

我们这么理解：第一只鹰必然落在节点x上，假设第k只鹰最后落在x的儿子y的子树内，我们考虑有几只鹰也曾飞进了这棵子树。如果是i只，那么概率为$\binom{k-2}{i}\left(\frac{1}{c_x}\right)^i\left(1-\frac{1}{c_x}\right)^{k-2-i}$。

复杂度为$O(nk^2)$。 □

Cards, bags and coins [4]

n个数字，选出其一个子集。

求有多少子集满足其中数字之和是m的倍数。$n \le 100000$，$m \le 100$，最多90组数据。 $\square$

[4] https://www.codechef.com/APRIL14/problems/ANUCBC

Cards, bags and coins

有个显然的背包做法……只是会TLE。

Cards, bags and coins

有个显然的背包做法……只是会TLE。

注意到m只有100，也就是说可以当做有m种物品的多重背包做。求最大值的话可以用二进制拆分的方法，但求方案数不行。

Cards, bags and coins

有个显然的背包做法……只是会TLE。

注意到m只有100，也就是说可以当做有m种物品的多重背包做。求最大值的话可以用二进制拆分的方法，但求方案数不行。

我们依次考虑每个物品，令g_i为选出一些这个物品使得体积对m取模为i的方案数，这个就等于一些组合数之和。

那么现在又相当于m个物品了，做一遍背包即可。复杂度为$O(n+m^3)$。 □

Little Elephant and Movies [5]

对于一个序列，定义其“激动值”为序列中严格大于前面所有数的元素的个数。比如，$\{1,1,5,6,5\}$的激动值为3。

给定n个数$p_1,p_2,\ldots,p_n$，求这n个数的所有排列中，激动值不超过k的个数。$1 \le k \le n \le 200$，$1 \le p_i \le 200$。 □

[5] https://www.codechef.com/FEB14/problems/LEMOVIE

Little Elephant and Movies

先记录相同的数的个数，然后去重。

Little Elephant and Movies

先记录相同的数的个数，然后去重。

从大到小考虑每组相同的数。令$f[i,j]$代表已经插入前i大的数，且激动值为j的序列方案数。

Little Elephant and Movies

先记录相同的数的个数，然后去重。

从大到小考虑每组相同的数。令 $f[i, j]$ 代表已经插入前 i 大的数，且激动值为 j 的序列方案数。

考虑当前这组数插入在哪些位置。每一组数的贡献至多为1，有贡献当且仅当至少一个数被插在了最前面。无论插入在什么位置，已有的激动值不会减少。

Little Elephant and Movies

先记录相同的数的个数，然后去重。

从大到小考虑每组相同的数。令$f[i,j]$代表已经插入前i大的数，且激动值为j的序列方案数。

考虑当前这组数插入在哪些位置。每一组数的贡献至多为1，有贡献当且仅当至少一个数被插在了最前面。无论插入在什么位置，已有的激动值不会减少。

假设已经插入了x个数，当前这组有y个数，没有一个数被插在最前面的方案数为

$$y!\binom{x+y-1}{x-1}$$

至少一个被插在最前面的可以类似算。

复杂度$O(nk)$。 $\square$

Little Elephant and Movies

如果激动值的定义为，序列中相邻的且构成逆序对的数对呢？
（为啥会有这个问题，是因为我最开始就把题目看成这个了）

Little Elephant and Movies

如果激动值的定义为，序列中相邻的且构成逆序对的数对呢？
（为啥会有这个问题，是因为我最开始就把题目看成这个了）

这个问题实际上稍难于原问题。

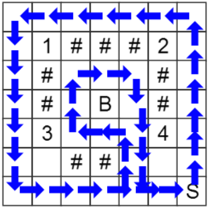
做法类似。从小到大考虑，此时插入一个数可能会抵消之前的激动值。枚举有几个数插在原来的非逆序对之间，可以类似地算出方案数。

复杂度$O(nk^2)$。 $\square$

Circling Round Treasures [6]

在一个$n \times m$的地图上，有一些障碍，还有a个宝箱和b个炸弹。你从(sx, sy)出发，走四连通的格子。你需要走一条闭合的路径，可以自交，且围出来的复杂多边形内不能包含任何炸弹。你围出来的复杂多边形中包含的宝箱的价值和就是你的收益。

求最大收益。$n, m \le 20$，$a + b \le 8$。



[6] http://codeforces.com/contest/375/problem/C

Circling Round Treasures

先考虑这个问题：如何判断一个点是否被复杂多边形所包含（不考虑在边上）？

Circling Round Treasures

先考虑这个问题：如何判断一个点是否被复杂多边形所包含（不考虑在边上）？

射线法。找一条由该点发出且不经过多边形上点的射线，然后看这条射线与多边形交了几次。如果交了奇数次，则在多边形内，否则在多边形外。

是否可以把这一方法用到这道题中？ □

Circling Round Treasures

其实是可以的。我们对于每个宝箱和炸弹维护一条射线，然后维护这些射线中哪些穿过了路径奇数次。

Circling Round Treasures

其实是可以的。我们对于每个宝箱和炸弹维护一条射线，然后维护这些射线中哪些穿过了路径奇数次。

令$F[x, y, k]$表示，当前在(x, y)，且集合k中的射线穿过了路径奇数次，这样的状态是否可行。

转移时枚举下一步，然后看两步之间的线段穿过了哪些射线，并更新集合k。

最后枚举所有集合k，要求炸弹的射线穿过路径偶数次，且$F[sx, sy, k] = \mathtt{true}$。用集合中宝箱的价值和更新答案。

复杂度$O(nm(a+b)2^{a+b})$。 □

<Source Unknown>

Parallel

给定一个有N个点的图，每个点都向外连出一条有向边，边长均为1。现要求在图中添加尽量少的从1号点出发的边，使得1号节点到每个节点的距离都不超过K。

求添加的最少边数。$N \le 100000$。 □

Parallel

不难发现，这个图是一个环套树。从1号点向某个节点连边就相当于把那个节点的距离设为1。

Parallel

不难发现，这个图是一个环套树。从1号点向某个节点连边就相当于把那个节点的距离设为1。

一个显然的结论是，我们需要对每个入度为0的节点添加一条边，因为不然这些节点根本不能从1号点到达。而对于树的部分，我们可以使用自底向上的树形DP简单处理。问题在于环的部分。 □

Parallel

一个朴素的$O(N^2)$的处理环的方法是枚举断开的地方（即把环视为从某个节点开始的链），然后按照在树上的方法$O(N)$做一遍，取所有断开处答案的最小值。

Parallel

一个朴素的$O(N^2)$的处理环的方法是枚举断开的地方（即把环视为从某个节点开始的链），然后按照在树上的方法$O(N)$做一遍，取所有断开处答案的最小值。

可以发现，对于环上每一个距离超过K的点，在该点的前$K-1$个点或该点之中至少存在一个点要从1连边。那么我们只需要断开K个地方。

Parallel

一个朴素的$O(N^2)$的处理环的方法是枚举断开的地方（即把环视为从某个节点开始的链），然后按照在树上的方法$O(N)$做一遍，取所有断开处答案的最小值。

可以发现，对于环上每一个距离超过K的点，在该点的前$K-1$个点或该点之中至少存在一个点要从1连边。那么我们只需要断开K个地方。

同时我们可以处理出数组$D[i]$，表示链上的i号点之后最近的距离大于K的点的编号，如果i号点本身的距离就大于K，则$D[i]=i$。这样我们访问完x号点时，就可以直接访问$D[x]$，而不用每次访问下一个节点了；而更新到一个需要连边的点之后，只需要往后跳K个节点再访问D数组即可。

这样在环上做一遍的复杂度就是$O(N/K)$的了，总的复杂度也就变成了$O(K \cdot N/K = N)$。 □

Pandora [7]

k个DFA，选出尽量多的自动机$a_0, a_1, a_2, \ldots, a_t$，使得$a_1$包含$a_0$、$a_2$包含$a_1$，以此类推。$k \le 50$。

DFA的字符集为$\{0, 1\}$，节点数$n \le 50$。 $\square$

[7] http://www.lydsy.com/JudgeOnline/problem.php?id=1194

Pandora

请先听我口头介绍什么是DFA。

Pandora

请先听我口头介绍什么是DFA。

用二元组(x, y)表示当前状态处在A的x点与B的y点上，初始为$(0, 0)$。转移时x和y一起走到加0和加1后的位置，即$(px0, py0)$和$(px1, py1)$，如果已经访问过该状态就不转移。假设在某个状态中x为输出元而y不是，那么B就不能输出这个A能输出的字符串，即B不包含A。如果不存在这种状态，则B包含A。

Pandora

请先听我口头介绍什么是DFA。

用二元组(x, y)表示当前状态处在A的x点与B的y点上，初始为$(0, 0)$。转移时x和y一起走到加0和加1后的位置，即$(px0, py0)$和$(px1, py1)$，如果已经访问过该状态就不转移。假设在某个状态中x为输出元而y不是，那么B就不能输出这个A能输出的字符串，即B不包含A。如果不存在这种状态，则B包含A。

用上面的方法对每两个自动机都进行判断然后建图。建完之后缩强连通分量，得到的会是一个拓扑图。在这个拓扑图上DP即可。 □

TwoSidedCards [8]

有N张牌，正反都有数字。所有正面和所有反面的数字各构成一个排列。将牌排成一排，每张牌可以正面朝上或者反面朝上，这样可以构成一个序列。

给定N张牌正面和反面的数字，求：所有可能的序列方案数。$N \le 50$。

比如：3张牌，正面分别为1、2、3，反面分别为1、3、2。那么一共有12种序列：

123、132、213、231、312、321、
133、313、331、122、212、221。

□

[8] https://community.topcoder.com/stat?c=problem_statement&pm=10947

TwoSidedCards

假设N足够小，可以让我们枚举哪些牌正面朝上。

那么可以发现，可能会有一些数字重复出现，而且重复出现次数最多为2。

记重复出现的数字个数为k，那么根据多重集的排列公式，这些牌的不同排列数为：

$$\frac{N!}{2^k}$$

不妨枚举k。现在我们要求的就是，对于任意合法的k，在多少种牌的正反方案中有k个数出现了2次。 □

TwoSidedCards

我们把所有牌按正面数值$1 \sim N$的顺序摆成一排。

这样我们就可以把反面数值视为一个置换，把这张牌翻面就相当于置换这一个位置。

可以发现，置换的不同循环之间是不影响的。

现在我们就来考虑一个循环。记这个循环的长度为L，显然有$k \leq \lfloor L/2 \rfloor$。 $\square$

TwoSidedCards

建立图论模型。
将每张牌正面的数字视为一个节点，连向其反面数字对应的节点。
可以发现，整张图会是若干个环，一个循环就对应图中的一个环。
有了这张图我们可以干什么呢？ □

TwoSidedCards

一张牌可以视为图中的一条有向边，从其正面的数字指向反面的数字。如果将牌翻面，就相当于将边反向。

出现两次的数也即入度为2的节点，这样的节点的一定是和入度为0的节点个数相等的，而且在同一个环上交替出现。

TwoSidedCards

一张牌可以视为图中的一条有向边，从其正面的数字指向反面的数字。如果将牌翻面，就相当于将边反向。

出现两次的数也即入度为2的节点，这样的节点的一定是和入度为0的节点个数相等的，而且在同一个环上交替出现。

由此可以得到公式：长度为L的循环中有k个数出现两次时，牌的正反的方案数为

$$2 \cdot \binom{L}{2k}$$

即从L个数字中各选出k个作为出现2次和0次的，而这2k个数内部有2种方案（第一个数出现2次还是0次）。

特别地，$k = 0$时方案数为1（无论全部正面还是反面，朝上的数的集合相同）。 □

TwoSidedCards

不过刚才只考虑了一个循环内牌的正反，如何回到原问题？

TwoSidedCards

不过刚才只考虑了一个循环内牌的正反，如何回到原问题？

用背包求出对于整个序列出现两次的数为k个的方案数。最后用最开始的公式计算排列即可。 □

CentaurCompany [9]

n个节点的树，每个节点被随机染成黑白两色之一。现在要使同种颜色的点连通，因此需要加边。每个节点最多免费加一条边，如果还要再加，那么每加一条边都需要付出1的代价（注意是针对节点而言的）。

求期望最小代价。$n \le 36$。 □

[9] https://community.topcoder.com/stat?c=problem_statement&pm=12428

CentaurCompany

先考虑单个的问题：给定一种染色方案，如何求其最小代价？

CentaurCompany

先考虑单个的问题：给定一种染色方案，如何求其最小代价？

对于染成同一种颜色的n个点，如果它们分布在c个连通块中，那么产生的费用为$\max(0, 2c-n-2)$。

CentaurCompany

先考虑单个的问题：给定一种染色方案，如何求其最小代价？

对于染成同一种颜色的n个点，如果它们分布在c个连通块中，那么产生的费用为$\max(0, 2c - n - 2)$。

那么令$f[col, p, n_1, c_1, n_2, c_2]$代表$p$的颜色为$col$，而且以$p$为根的子树中被染上颜色1和2的点有$n_1$和$n_2$个，分成$c_1$和$c_2$个连通块，这样的状态的个数。注意$n_1 + n_2 = size[p]$，因此状态数为$O(n^4)$。

CentaurCompany

先考虑单个的问题：给定一种染色方案，如何求其最小代价？

对于染成同一种颜色的n个点，如果它们分布在c个连通块中，那么产生的费用为$\max(0, 2c-n-2)$。

那么令$f[col, p, n_1, c_1, n_2, c_2]$代表$p$的颜色为$col$，而且以$p$为根的子树中被染上颜色$1$和$2$的点有$n_1$和$n_2$个，分成$c_1$和$c_2$个连通块，这样的状态的个数。注意$n_1+n_2=size[p]$，因此状态数为$O(n^4)$。

转移时枚举儿子颜色，如果儿子的颜色和父亲相同则不产生新的连通块，否则会产生一个新的连通块。再枚举儿子子树中颜色1的点数，用类似背包的方法求解。转移复杂度为$O(n^3)$。

总复杂度高达$O(n^7)$，需要优化。 $\square$

CentaurCompany

优化其实也很简单，注意到计算答案只需要$2c_1 - n_1$和$2c_2 - n_2$的值，因此我们直接维护它。

CentaurCompany

优化其实也很简单，注意到计算答案只需要$2c_1-n_1$和$2c_2-n_2$的值，因此我们直接维护它。

再进一步，由期望的线性性，我们知道

$$E((2c_1-n_1)+(2c_2-n_2))=E(2c_1-n_1)+E(2c_2-n_2)$$

而由于对称性，这两个东西是相等的。因此我们只要知道一个颜色的$2c-n$即可，复杂度优化到$O(n^3)$。 $\square$

AppleTrees [10]

一排D个坑中要种下n棵树，每棵树有半径r_i，表示其左右r_i的范围内不能种其他树。

求合法的方案数。$D \le 100000$，$n, r_i \le 40$。 $\square$

[10] https://community.topcoder.com/stat?c=problem_statement&pm=11213

AppleTrees

假设我们已经知道树的相对顺序，我们可以先把它们尽量紧凑地排在一起，然后在树之间插入空的坑。

AppleTrees

假设我们已经知道树的相对顺序，我们可以先把它们尽量紧凑地排在一起，然后在树之间插入空的坑。

如何计算给定顺序时，两端的两棵树之间的最短距离？

AppleTrees

假设我们已经知道树的相对顺序，我们可以先把它们尽量紧凑地排在一起，然后在树之间插入空的坑。

如何计算给定顺序时，两端的两棵树之间的最短距离？相邻两棵树的距离为这两棵树半径的较大值。

AppleTrees

假设我们已经知道树的相对顺序，我们可以先把它们尽量紧凑地排在一起，然后在树之间插入空的坑。

如何计算给定顺序时，两端的两棵树之间的最短距离？相邻两棵树的距离为这两棵树半径的较大值。

我们考虑按某种顺序一棵一棵种下这些树。比如按照r_i从小到大。

一个问题是，只有当我们确定两棵树之间不会再插入新的树的时候，我们才能确定这两棵树之间的距离是多少。 □

AppleTrees

那么在状态中记录：目前两端的两棵树之间的距离k，以及还有多少对相邻的树之间还会插入别的（r_i更大的）树。称后者中满足条件的两棵树之间存在“缝隙”。

当我们确定两棵树之间不会再有别的树的时候，我们才将其距离计入k。缝隙的距离不计入k。

AppleTrees

那么在状态中记录：目前两端的两棵树之间的距离k，以及还有多少对相邻的树之间还会插入别的（r_i更大的）树。称后者中满足条件的两棵树之间存在"缝隙"。

当我们确定两棵树之间不会再有别的树的时候，我们才将其距离计入k。缝隙的距离不计入k。

具体的DP方案如下：用$f[i,j,k]$表示：前i棵树目前有j个缝隙，最两端树之间的距离为k（不算缝隙）的方案数。

转移时枚举下一棵树放哪：缝隙中还是两端，同时讨论新放的树与左右的树之间是否留缝隙。

最终答案$\sum f[N,0,k] \times \binom{D-k+N}{D-k}$。复杂度$O(N^3 r_i)$。 $\square$

Sereja and Game [11]

n个数（可能存在相同的数），双方轮流取数。如果在一方选取之后，所有已选取数字的GCD变为1，则此方输。

问：

1. 若双方均采取最优策略，先手是否必胜？
2. 若双方随机取数，先手获胜的概率为多少？

$n, a_i \leq 100$。 □

[11] https://www.codechef.com/MAY14/problems/SEAGM

Sereja and Game

令 $f[i,j]$表示GCD为i，未选而且是i的倍数的数有j个的状态。

Sereja and Game

令 $f[i,j]$ 表示 GCD 为 i，未选而且是 i 的倍数的数有 j 个的状态。

转移时，要么是转移到 $f[i,j-1]$，要么是枚举一个令 GCD 减小的数转移。在 GCD 减小后，原来是倍数的还是倍数，此外还可能有之前不是倍数的，现在变成了倍数，统计一下即可。

通过预处理倍数可以做到 $O(n)$ 的转移。而两问都可以用这个方法解决。

Sereja and Game

令 $f[i,j]$ 表示 GCD 为 i，未选而且是 i 的倍数的数有 j 个的状态。

转移时，要么是转移到 $f[i,j-1]$，要么是枚举一个令 GCD 减小的数转移。在 GCD 减小后，原来是倍数的还是倍数，此外还可能有之前不是倍数的，现在变成了倍数，统计一下即可。

通过预处理倍数可以做到 $O(n)$ 的转移。而两问都可以用这个方法解决。

这个方法的核心在于找到了一个合理的考虑牌的顺序，使得我们能划分阶段。 $\square$

RooksParty [12]

有N种颜色的车（国际象棋），每种Col_i个，放置在$R \times C$的棋盘上。如果有两个不同颜色的车处于同行或同列，它们就会互相攻击。

求使得所有车不互相攻击的摆放方案数。$R, C \leq 30$，$N \leq 10$。 □

[12] https://community.topcoder.com/stat?c=problem_statement&pm=10980

RooksParty

每行和每列的所有格子要么是同一种颜色的车，要么为空。那么给每行和每列标上一种颜色。

一种颜色的车能放置的格子为，该行和该列的颜色都为车的颜色的格子。也即每种颜色之间无关。

RooksParty

每行和每列的所有格子要么是同一种颜色的车，要么为空。那么给每行和每列标上一种颜色。

一种颜色的车能放置的格子为，该行和该列的颜色都为车的颜色的格子。也即每种颜色之间无关。

令$F[t,x,y]$表示颜色$1 \sim t$的车占据x行和y列的方案数。

$$F[t,x,y]=\sum_{r=1}^{x}\sum_{c=1}^{y}F[t-1,x-r,y-c]\binom{x}{r}\binom{y}{c}S[Col_t,r,c]$$

其中$S[n,r,c]$为将n个车摆放在$r\times c$的棋盘上，使得**每行每列都有至少一个车**的方案数（不考虑互相攻击，因为同色嘛）。

公式很好理解：考虑当前颜色的车占据了哪几行哪几列，乘上摆放方案数，再递归考虑剩下的。

注意边界和转移的合法性。

TopCoder SRM473 D1L3

RooksParty

怎么求$S[n, r, c]$呢？

RooksParty

怎么求$S[n,r,c]$呢？

先只考虑列。令$R[n,r,c]$代表满足了每列有至少一个车的摆放方案数。

RooksParty

怎么求$S[n,r,c]$呢？

先只考虑列。令$R[n,r,c]$代表满足了每列有至少一个车的摆放方案数。

运用容斥原理：

$$R[n,r,c]=\sum_{i=0}^{c}(-1)^i\binom{c}{i}\binom{r(c-i)}{n}$$

其中i代表有i列为空，和式的该项为有至少i列为空的摆放方案数。

RooksParty

怎么求$S[n,r,c]$呢？

先只考虑列。令$R[n,r,c]$代表满足了每列有至少一个车的摆放方案数。

运用容斥原理：

$$R[n,r,c]=\sum_{i=0}^{c}(-1)^i\binom{c}{i}\binom{r(c-i)}{n}$$

其中i代表有i列为空，和式的该项为有至少i列为空的摆放方案数。

在把行也加入考虑，类似可以得到：

$$S[n,r,c]=\sum_{j=0}^{r}(-1)^j\binom{r}{j}R[n,r-j,c]$$

其中j代表有j行为空。 □

RooksParty

我们分析一下时间复杂度。

- 计算f的复杂度：状态$O(nRC)$，转移$O(RC)$，总复杂度$O(nR^2C^2)$。
- 计算S的复杂度：状态$O(Col_xRC)$，转移$O(R+C)$，总复杂度为$O(nRC(R+C))$。

算法的总复杂度为$O(nR^2C^2)$。 □

TopCoder SRM473 D1L3

$S[n,r,c]$还有另外一种直接考虑二维问题的求法：

$$S[n,r,c]=\binom{r\cdot c}{n}-\sum_{i=1}^{r}\sum_{j=1}^{c}\binom{r}{i}\binom{c}{j}S[n,r-i,c-j]$$

即从所有方案中减去有i行j列为空的方案。

这个方法的复杂度为$O(nR^2C^2)$，可以稍作优化：

$$S[n,r,c]=\binom{r\cdot c}{n}-\sum_{i=1}^{r}\binom{r}{r-1}\sum_{j=1}^{c}\binom{c}{c-j}S[n,r-i,c-j]$$

维护$G[k]=\sum_{j=1}^{c}\binom{c}{c-j}S[n,k,c-j]$，可以优化转移。总复杂度降为$O(nR^2C)$。 □

Arthur [13]

玩家有一套共n张卡牌，每张卡牌都有一个技能。第i张卡牌的技能发动概率为p_i $(0 < p_i < 1)$，如果成功发动，则会对敌方造成d_i点伤害。

一局游戏一共有r轮，在一轮中，系统将按照顺序依次考虑每张卡牌。对于一张卡牌，判定如下：

1. 如果这张卡牌已经发动过技能，则跳过之；
2. 否则，设这张卡牌为第i张，那么其将以p_i的概率发动技能。如果技能发动，则对敌方造成d_i点伤害，并结束这一轮；
3. 否则，考虑下一张卡牌。特别地，如果这张卡牌已经是最后一张（即第n张），则这一轮不发动技能，并结束这一轮。

求出一局游戏中造成的伤害的期望值。$n \leq 220$，$r \leq 132$，最多444组测试数据。 □

[13] http://www.lydsy.com/JudgeOnline/problem.php?id=4008

Arthur

我们先来看一道简单的题：

- 求有多少$1 \sim n$的排列，其逆序对数为k。

Arthur

我们先来看一道简单的题：

- 求有多少$1 \sim n$的排列，其逆序对数为k。

做法非常简单，令$f[i,j]$为$1 \sim i$的排列中逆序对数为j的个数。

Arthur

我们先来看一道简单的题：

- 求有多少$1 \sim n$的排列，其逆序对数为k。

做法非常简单，令$f[i,j]$为$1 \sim i$的排列中逆序对数为j的个数。
转移方程如下：

$$f[i,j] = \sum_{t=\max(1,j-i)}^{j} f[i-1,t]$$

利用前缀和可以优化到$O(nk)$。 □

Arthur

设没有发动技能的一轮为“空轮”。

我们可以把一种方案抽象为一个带空格的部分排列，空格为空轮，其他的为该轮发动的卡片。这样的抽象和原方案是一一对应的。

Arthur

设没有发动技能的一轮为“空轮”。

我们可以把一种方案抽象为一个带空格的部分排列，空格为空轮，其他的为该轮发动的卡片。这样的抽象和原方案是一一对应的。

先假设没有空轮。不妨考虑到最后发动了技能的卡片的集合，设为S。要求得到这一集合的概率，如果暴力做，应当枚举其所有排列。

一个排列的概率即为，每轮的概率之积。而一轮的概率又是发动技能的卡片的p_i乘上i之前所有的$1-p_j$，再除去所有已发动卡片的$1-p_k$。

Arthur

设没有发动技能的一轮为“空轮”。

我们可以把一种方案抽象为一个带空格的部分排列，空格为空轮，其他的为该轮发动的卡片。这样的抽象和原方案是一一对应的。

先假设没有空轮。不妨考虑到最后发动了技能的卡片的集合，设为S。要求得到这一集合的概率，如果暴力做，应当枚举其所有排列。

一个排列的概率即为，每轮的概率之积。而一轮的概率又是发动技能的卡片的p_i乘上i之前所有的$1-p_j$，再除去所有已发动卡片的$1-p_k$。

那么先不考虑已发动技能，之后再将其除去。设第i张卡牌在第j轮发动，那么对于第j轮之后发动的编号大于i的卡牌，都需要除一次$1-p_j$。为此，我们需要知道每个位置的顺序对数。

直接套用开始的算法，按照$n\sim 1$的顺序枚举即可。复杂度为$O(nr^2)$。

Arthur

设没有发动技能的一轮为“空轮”。

我们可以把一种方案抽象为一个带空格的部分排列，空格为空轮，其他的为该轮发动的卡片。这样的抽象和原方案是一一对应的。

先假设没有空轮。不妨考虑到最后发动了技能的卡片的集合，设为S。要求得到这一集合的概率，如果暴力做，应当枚举其所有排列。

一个排列的概率即为，每轮的概率之积。而一轮的概率又是发动技能的卡片的p_i乘上i之前所有的$1-p_j$，再除去所有已发动卡片的$1-p_k$。

那么先不考虑已发动技能，之后再将其除去。设第i张卡牌在第j轮发动，那么对于第j轮之后发动的编号大于i的卡牌，都需要除一次$1-p_j$。为此，我们需要知道每个位置的顺序对数。

直接套用开始的算法，按照$n \sim 1$的顺序枚举即可。复杂度为$O(nr^2)$。

进而，我们还可以直接在DP的时候选择集合。复杂度仍为$O(nr^2)$。 □

Arthur

如果考虑空轮呢？现在要处理的问题就是，确定了排列后，还需要向其中插入空格。同时一张卡片还需要对其之后的每个空格除一次$1-p_i$。

Arthur

如果考虑空轮呢？现在要处理的问题就是，确定了排列后，还需要向其中插入空格。同时一张卡片还需要对其之后的每个空格除一次$1-p_i$。

那么不妨把空格当做第$n+1$张牌，先插入若干个空格，然后再用上面的算法。复杂度仍为$O(nr^2)$。

当然，还得额外维护期望。

Arthur

如果考虑空轮呢？现在要处理的问题就是，确定了排列后，还需要向其中插入空格。同时一张卡片还需要对其之后的每个空格除一次$1-p_i$。

那么不妨把空格当做第$n+1$张牌，先插入若干个空格，然后再用上面的算法。复杂度仍为$O(nr^2)$。

当然，还得额外维护期望。

另外注意到，复杂度的$O(nr^2)$来自于$O(nr)$的状态和$O(r)$的转移，而转移可以很简单地用前缀和优化成$O(1)$，这样总复杂度就是$O(nr)$了。

注意使用`long double`。 □

Arthur

这题最核心的部分就在于找到这么一个合适的顺序。本质上和之前的 SEAGM 是类似的。 □

Tighten Up! [14]

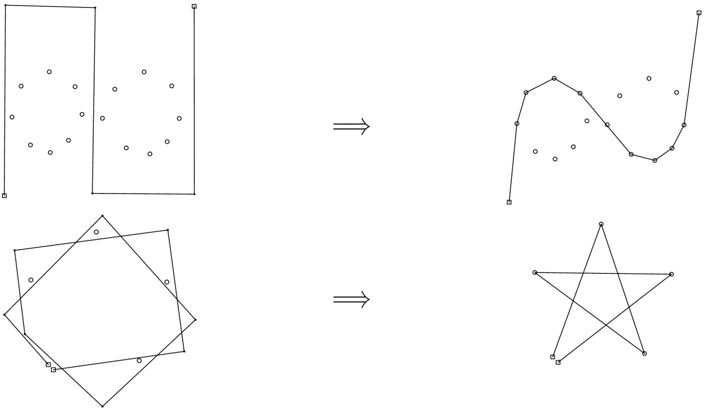
平面上有两个洞和n个固定在平面上的柱子。有一根绳子从洞中穿过，在平面上绕来绕去，经过平面上m个点的位置，然后从另一个洞中穿出。

现在拉紧绳子的两端，问绳子留在平面上的部分有多长。$m, n \leq 100$。 □

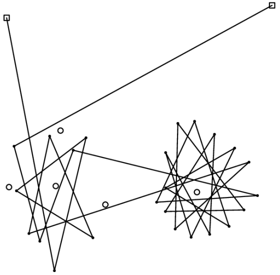
[14] http://judge.u-aizu.ac.jp/onlinejudge/description.jsp?id=1164

ACM/ICPC Japan Domestic Contest 2009: Problem F

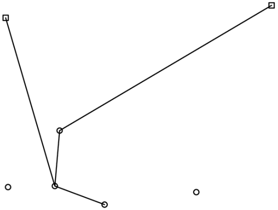
Tighten Up!



Tighten Up!



$\Longrightarrow$



□

Tighten Up!

我们先看一道相对简单的题目：

- 平面上有一些固定的杆子。给定平面上一段闭合的绳圈，问能否将其收缩至无穷小的面积？

Tighten Up!

我们先看一道相对简单的题目：

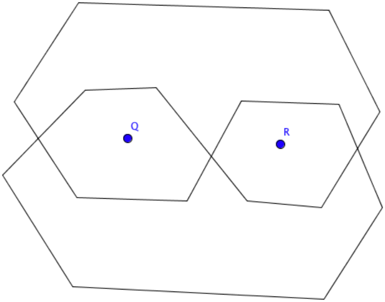
- 平面上有一些固定的杆子。给定平面上一段闭合的绳圈，问能否将其收缩至无穷小的面积？

一个直观的想法是，如果有一个点在绳圈的“内部”，那么就无法收缩至无穷小，否则可以。判断内部可以求出点与每条边的有向角度之和，如果为0则不是内部。 □

ACM/ICPC Japan Domestic Contest 2009: Problem F

Tighten Up!

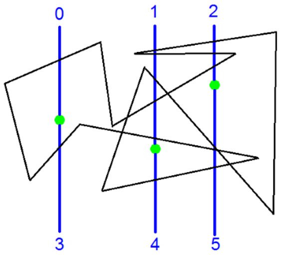
但这个做法其实是错的，反例如下：



Tighten Up!

正确做法如下：假设任意两点的横坐标都不同，作n条竖线穿过每个点，并对这2n条射线标号。我们记录绳圈穿过这些射线的顺序，比如下面的绳圈会对应序列

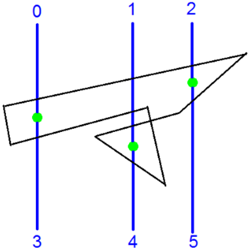
$$0 \to 1 \to 2 \to 2 \to 1 \to 1 \to 2 \to 5 \to 1 \to 4 \to 5 \to 5 \to 1 \to 3$$



Tighten Up!

而很显然，如果序列中出现了两个相邻且相同的数字，就意味着绳圈来回穿过了这条线，在将其“拉直”后肯定不会穿过。因此我们每次可以删去序列中两个相邻且相同的数字。对上面的序列处理完之后如下

$$0 \rightarrow 1 \rightarrow 2 \rightarrow 5 \rightarrow 1 \rightarrow 4 \rightarrow 1 \rightarrow 3$$



Tighten Up!

那对于这道题而言，我们只需判断能否通过这一操作将序列删空即可。

我们将这一思路带回原题，同样对原题求出一个序列。我们现在要求的是，以给定的两点为端点，且满足这一序列的最短（因为被拉紧了）绳子长度。

Tighten Up!

那对于这道题而言，我们只需判断能否通过这一操作将序列删空即可。

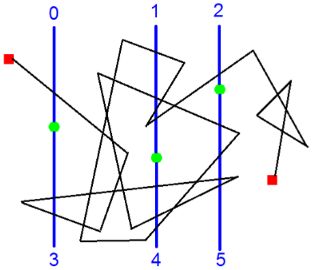
我们将这一思路带回原题，同样对原题求出一个序列。我们现在要求的是，以给定的两点为端点，且满足这一序列的最短（因为被拉紧了）绳子长度。

这其实是一个DP问题。令$f[i]$表示绳子到序列第i位为止的长度，此时绳子的一段必然在序列这一位的射线对应的柱子上。我们枚举这一段绳子绕过的上一个节点是哪个，并判断是否可行。

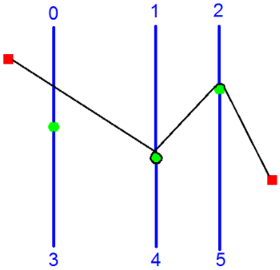
复杂度$O((nm)^2)$，因为理论上序列的长度可以达到$O(nm)$。 $\square$

ACM/ICPC Japan Domestic Contest 2009: Problem F

Tighten Up!



$\Longrightarrow$



□

Fin.

谢谢大家！欢迎课后交流。