

连通性状压 DP 选讲

Claris

Hangzhou Dianzi University

2016 年 2 月 7 日

在一张 $n \times n$ 的网格图上每个格子都有一只蚂蚁。每只蚂蚁将会同时往上下左右 4 个方向中的某个方向前进一步。

在一张 $n \times n$ 的网格图上每个格子都有一只蚂蚁。每只蚂蚁将会同时往上下左右 4 个方向中的某个方向前进一步。

一个合法的方案是指所有蚂蚁移动完后仍然在网格内，且每个格子仍然有且仅有一只蚂蚁。

在一张 $n \times n$ 的网格图上每个格子都有一只蚂蚁。每只蚂蚁将会同时往上下左右 4 个方向中的某个方向前进一步。

一个合法的方案是指所有蚂蚁移动完后仍然在网格内，且每个格子仍然有且仅有一只蚂蚁。

请统计出所有合法的方案的总数。

在一张 $n \times n$ 的网格图上每个格子都有一只蚂蚁。每只蚂蚁将会同时往上下左右 4 个方向中的某个方向前进一步。

一个合法的方案是指所有蚂蚁移动完后仍然在网格内，且每个格子仍然有且仅有一只蚂蚁。

请统计出所有合法的方案的总数。

- $2 \leq n \leq 10$ 。

在一张 $n \times n$ 的网格图上每个格子都有一只蚂蚁。每只蚂蚁将会同时往上下左右 4 个方向中的某个方向前进一步。

一个合法的方案是指所有蚂蚁移动完后仍然在网格内，且每个格子仍然有且仅有一只蚂蚁。

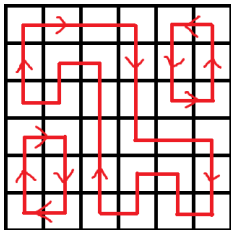
请统计出所有合法的方案的总数。

- $2 \leq n \leq 10$ 。
- Source: 2015 年吉林省大学生程序设计竞赛

- 合法的方案等价于用若干个互不相交的顺时针或逆时针的回路恰好填满整张网格图。

- 合法的方案等价于用若干个互不相交的顺时针或逆时针的回路恰好填满整张网格图。
- 例如下图是 $n = 6$ 的一组解：


- 合法的方案等价于用若干个互不相交的顺时针或逆时针的回路恰好填满整张网格图。
- 例如下图是 $n = 6$ 的一组解:





- 考虑动态规划，从上到下，从左到右逐格递推。

- 考虑动态规划，从上到下，从左到右逐格递推。

- 定义三种插头：

0 表示空插头： 

1 表示外插头： 

2 表示内插头： 

- 考虑动态规划，从上到下，从左到右逐格递推。

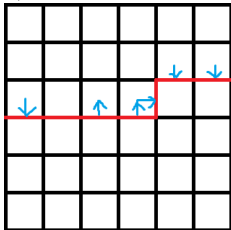
- 定义三种插头：

0 表示空插头：□

1 表示外插头：↓

2 表示内插头：↑

- 定义轮廓线表示已决策格子和未决策格子的分界线，并用一个 3 进制的数表示轮廓线上从左到右的 $n+1$ 个插头：



- 设 $f(i, j, S)$ 表示考虑到了 (i, j) 这个格子，轮廓线上插头状态为 S 的方案数。

- 设 $f(i, j, S)$ 表示考虑到了 (i, j) 这个格子，轮廓线上插头状态为 S 的方案数。
- 初始条件 $f(1, 1, S_0) = f(1, 1, S_1) = 1$, S_0, S_1 分别为以下两种情况：



- 设 $f(i, j, S)$ 表示考虑到了 (i, j) 这个格子，轮廓线上插头状态为 S 的方案数。
- 初始条件 $f(1, 1, S_0) = f(1, 1, S_1) = 1$, S_0, S_1 分别为以下两种情况：



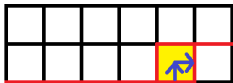
- $ans = f(n, n, 0)$, 即轮廓线上 $n + 1$ 个插头均为空插头：



- 对于状态的转移，从第 i 行转移到第 $i+1$ 行时，第 $n+1$ 个插头必须为空插头，然后在最前面插入一个空插头。从 $(i, j-1)$ 转移到 (i, j) 时，需要分类讨论，下图中黄色格子表示 (i, j) ，设 x 为 $(i, j-1)$ 轮廓线上竖着部分的插头， y 为 x 右侧的插头，类似地设 p 为 (i, j) 轮廓线上竖着部分的插头， q 为 p 右侧的插头。

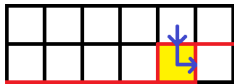
- 对于状态的转移，从第 i 行转移到第 $i+1$ 行时，第 $n+1$ 个插头必须为空插头，然后在最前面插入一个空插头。从 $(i, j-1)$ 转移到 (i, j) 时，需要分类讨论，下图中黄色格子表示 (i, j) ，设 x 为 $(i, j-1)$ 轮廓线上竖着部分的插头， y 为 x 右侧的插头，类似地设 p 为 (i, j) 轮廓线上竖着部分的插头， q 为 p 右侧的插头。
- 第 1 种情况， $x=0, y=0$ ，那么 $p=1, q=2$ 或

$p=2, q=1$:



- 第 2 种情况, $x = 0, y = 1$, 那么 $p = 1, q = 0$ 或

$p = 0, q = 1$:



- 第 2 种情况, $x = 0, y = 1$, 那么 $p = 1, q = 0$ 或

$p = 0, q = 1$:



- 第 3 种情况, $x = 0, y = 2$, 那么 $p = 2, q = 0$ 或

$p = 0, q = 2$:



- 第 2 种情况, $x = 0, y = 1$, 那么 $p = 1, q = 0$ 或

$p = 0, q = 1$:



- 第 3 种情况, $x = 0, y = 2$, 那么 $p = 2, q = 0$ 或

$p = 0, q = 2$:



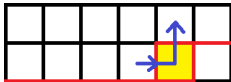
- 第 4 种情况, $x = 1, y = 0$, 那么 $p = 1, q = 0$ 或

$p = 0, q = 1$:

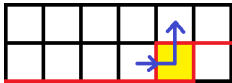


- 第 5 种情况, $x = 1, y = 1$, 那么无解, 直接抛弃这个状态。

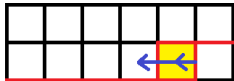
- 第 5 种情况, $x = 1, y = 1$, 那么无解, 直接抛弃这个状态。
- 第 6 种情况, $x = 1, y = 2$, 那么 $p = 0, q = 0$:



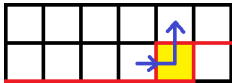
- 第 5 种情况, $x = 1, y = 1$, 那么无解, 直接抛弃这个状态。
- 第 6 种情况, $x = 1, y = 2$, 那么 $p = 0, q = 0$:



- 第 7 种情况, $x = 2, y = 0$, 那么 $p = 2, q = 0$ 或 $p = 0, q = 2$:



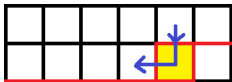
- 第 5 种情况, $x = 1, y = 1$, 那么无解, 直接抛弃这个状态。
- 第 6 种情况, $x = 1, y = 2$, 那么 $p = 0, q = 0$:



- 第 7 种情况, $x = 2, y = 0$, 那么 $p = 2, q = 0$ 或 $p = 0, q = 2$:



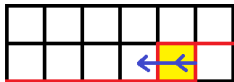
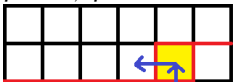
- 第 8 种情况, $x = 2, y = 1$, 那么 $p = 0, q = 0$:



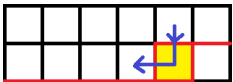
- 第 5 种情况, $x = 1, y = 1$, 那么无解, 直接抛弃这个状态。
- 第 6 种情况, $x = 1, y = 2$, 那么 $p = 0, q = 0$:



- 第 7 种情况, $x = 2, y = 0$, 那么 $p = 2, q = 0$ 或 $p = 0, q = 2$:



- 第 8 种情况, $x = 2, y = 1$, 那么 $p = 0, q = 0$:



- 第 9 种情况, $x = 2, y = 2$, 那么无解, 直接抛弃这个状态。

- 对于状态的保存，可以考虑使用 Hash 表，每次只从有效状态开始转移。

- 对于状态的保存, 可以考虑使用 Hash 表, 每次只从有效状态开始转移。
- 时间复杂度 $O(n^2 3^n)$ 。

给定一张 $n \times m$ 的网格图，有些格子只能左右通过，有些格子只能上下通过，有些格子不能通过，求把所有可以通过的块都经过且只经过一次并回到原地的方案数，其中顺时针和逆时针算同一种方案。

给定一张 $n \times m$ 的网格图，有些格子只能左右通过，有些格子只能上下通过，有些格子不能通过，求把所有可以通过的块都经过且只经过一次并回到原地的方案数，其中顺时针和逆时针算同一种方案。

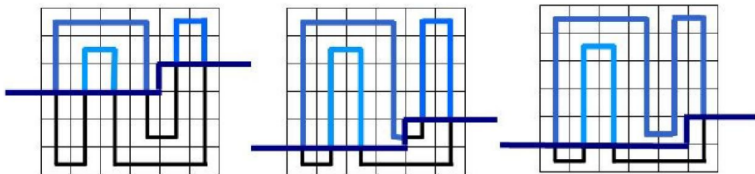
- $1 \leq n, m \leq 12$ 。

给定一张 $n \times m$ 的网格图，有些格子只能左右通过，有些格子只能上下通过，有些格子不能通过，求把所有可以通过的块都经过且只经过一次并回到原地的方案数，其中顺时针和逆时针算同一种方案。

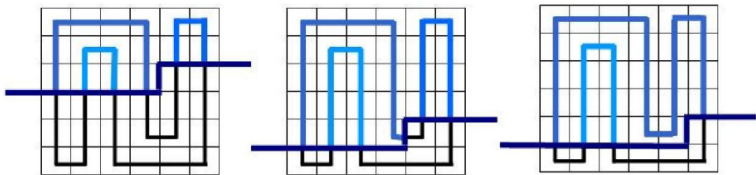
- $1 \leq n, m \leq 12$ 。
- Source: BZOJ 3125

- 与上一题不同，本题只能存在一条回路，因此状态的表示和转移略有区别。

- 与上一题不同，本题只能存在一条回路，因此状态的表示和转移略有区别。
- 观察一下简单回路有什么特点：



- 与上一题不同，本题只能存在一条回路，因此状态的表示和转移略有区别。
- 观察一下简单回路有什么特点：



- 轮廓线上方的路径互不相交，且每条路径在轮廓线上恰好对应了两个插头，而且这些插头两两匹配，互不交叉，所以可以用合法的括号序列来表示轮廓线上插头的情况。

- 重新定义三种插头：

0 表示空插头，1 表示左括号，2 表示右括号。

那么轮廓线上插头的状态仍然可以用一个 $m+1$ 位 3 进制数表示。

- 重新定义三种插头：

0 表示空插头，1 表示左括号，2 表示右括号。

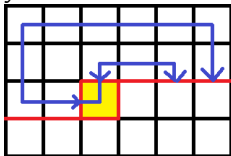
那么轮廓线上插头的状态仍然可以用一个 $m+1$ 位 3 进制数表示。

- 为了减少状态数，可以在一开始预处理出所有合法的括号序列，将这些状态重标号，并将里面与每个括号相匹配的另一个括号的位置记录下来，方便后面状态转移。

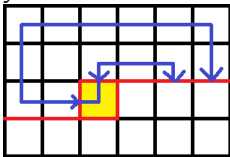
- 对于状态的转移，从第 i 行转移到第 $i+1$ 行时，第 $n+1$ 个插头必须为空插头，然后在最前面插入一个空插头。从 $(i, j-1)$ 转移到 (i, j) 时，需要分类讨论，下图中黄色格子表示 (i, j) ，设 x 为 $(i, j-1)$ 轮廓线上竖着部分的插头， y 为 x 右侧的插头，类似地设 p 为 (i, j) 轮廓线上竖着部分的插头， q 为 p 右侧的插头。

- 对于状态的转移，从第 i 行转移到第 $i+1$ 行时，第 $n+1$ 个插头必须为空插头，然后在最前面插入一个空插头。从 $(i, j-1)$ 转移到 (i, j) 时，需要分类讨论，下图中黄色格子表示 (i, j) ，设 x 为 $(i, j-1)$ 轮廓线上竖着部分的插头， y 为 x 右侧的插头，类似地设 p 为 (i, j) 轮廓线上竖着部分的插头， q 为 p 右侧的插头。
- 第 1 种情况， $x=0, y=0$ ，那么 $p=1, q=2$ ，表示新建了一个连通分量。

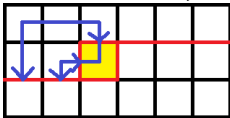
- 第 2 种情况, $x = 1, y = 1$, 那么 $p = 0, q = 0$, 同时需要把 y 对应的右括号改成左括号, 表示合并两个连通分量。



- 第 2 种情况, $x = 1, y = 1$, 那么 $p = 0, q = 0$, 同时需要把 y 对应的右括号改成左括号, 表示合并两个连通分量。



- 第 3 种情况, $x = 2, y = 2$, 那么 $p = 0, q = 0$, 同时需要把 x 对应的左括号改成右括号, 表示合并两个连通分量。



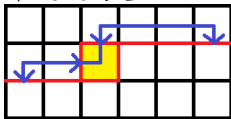
- 第 4 种情况, $x = 1, y = 2$, 那么 $p = 0, q = 0$, 表示得到一条回路, 那么这种转移只能发生在最后一个格子。



- 第 4 种情况, $x = 1, y = 2$, 那么 $p = 0, q = 0$, 表示得到一条回路, 那么这种转移只能发生在最后一个格子。



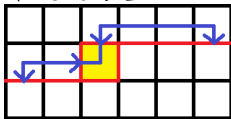
- 第 5 种情况, $x = 2, y = 1$, 那么 $p = 0, q = 0$, 表示合并两个连通分量。



- 第 4 种情况, $x = 1, y = 2$, 那么 $p = 0, q = 0$, 表示得到一条回路, 那么这种转移只能发生在最后一个格子。



- 第 5 种情况, $x = 2, y = 1$, 那么 $p = 0, q = 0$, 表示合并两个连通分量。

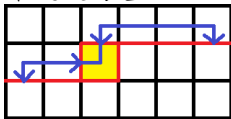


- 第 6 种情况, x, y 中有一个为 0, 那么 $p = 0, q = x + y$, 或 $p = x + y, q = 0$, 表示插头延伸一格。

- 第 4 种情况, $x = 1, y = 2$, 那么 $p = 0, q = 0$, 表示得到一条回路, 那么这种转移只能发生在最后一个格子。



- 第 5 种情况, $x = 2, y = 1$, 那么 $p = 0, q = 0$, 表示合并两个连通分量。



- 第 6 种情况, x, y 中有一个为 0, 那么 $p = 0, q = x + y$, 或 $p = x + y, q = 0$, 表示插头延伸一格。
- 对于本题, 还需要考虑格子本身的通行限制来进行转移, 这个 DP 的时间复杂度为 $O(nm3^m)$ 。


给定一个 $n \times m$ 的矩阵, (i, j) 格子的权值为 $V_{i,j}$ (可能为负), 找一条简单路径, 使得每个点最多经过一次, 并且经过的点权值之和最大。


给定一个 $n \times m$ 的矩阵, (i, j) 格子的权值为 $V_{i,j}$ (可能为负), 找一条简单路径, 使得每个点最多经过一次, 并且经过的点权值之和最大。


- $1 \leq n \leq 100, 1 \leq m \leq 8$ 。


给定一个 $n \times m$ 的矩阵, (i, j) 格子的权值为 $V_{i,j}$ (可能为负), 找一条简单路径, 使得每个点最多经过一次, 并且经过的点权值之和最大。

- $1 \leq n \leq 100, 1 \leq m \leq 8$ 。
- Source: BZOJ 2310

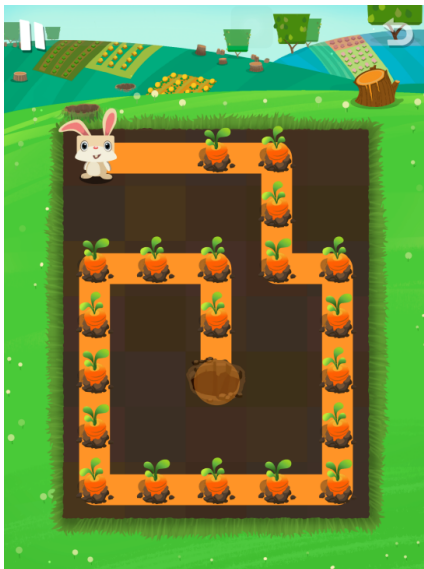
- 与上一个问题不同，本题是简单路径，所以需要新增一种插头，在这里我们用 3 来表示独立插头：

- 与上一个问题不同，本题是简单路径，所以需要新增一种插头，在这里我们用 3 来表示独立插头：
- 现在轮廓线上插头的状态仍然可以用一个 $m+1$ 位 4 进制数表示。按照上题方法预处理出所有有效状态，实际运行中状态数不超过 8000 个。

- 与上一个问题不同，本题是简单路径，所以需要新增一种插头，在这里我们用 3 来表示独立插头：
- 现在轮廓线上插头的状态仍然可以用一个 $m+1$ 位 4 进制数表示。按照上题方法预处理出所有有效状态，实际运行中状态数不超过 8000 个。
- 对于状态转移，与括号表示法的转移类似，对于独立插头，要考虑在连接两个连通分量的时候将某个左右括号对应的左右括号修改为独立插头，具体细节都可以通过简单的分类讨论得到。

- 与上一个问题不同，本题是简单路径，所以需要新增一种插头，在这里我们用 3 来表示独立插头：
- 现在轮廓线上插头的状态仍然可以用一个 $m+1$ 位 4 进制数表示。按照上题方法预处理出所有有效状态，实际运行中状态数不超过 8000 个。
- 对于状态转移，与括号表示法的转移类似，对于独立插头，要考虑在连接两个连通分量的时候将某个左右括号对应的左右括号修改为独立插头，具体细节都可以通过简单的分类讨论得到。
- 时间复杂度 $O(nm4^m)$ 。

Patchmania I



在一个 $n \times m$ 的网格图农田上有一些萝卜，一只兔子要吃完所有萝卜并跳到洞里，当开始吃萝卜时，如果还有萝卜那么下一步一定要吃到萝卜。吃完萝卜时下一步一定要跳到洞里。兔子每一步只能选择上下左右四个方向行走一个格子。

在一个 $n \times m$ 的网格图农田上有一些萝卜，一只兔子要吃完所有萝卜并跳到洞里，当开始吃萝卜时，如果还有萝卜那么下一步一定要吃到萝卜。吃完萝卜时下一步一定要跳到洞里。兔子每一步只能选择上下左右四个方向行走一个格子。

兔子一开始的位置和唯一的洞的位置都是给定的。求最短距离和该距离下的方案数。

在一个 $n \times m$ 的网格图农田上有一些萝卜，一只兔子要吃完所有萝卜并跳到洞里，当开始吃萝卜时，如果还有萝卜那么下一步一定要吃到萝卜。吃完萝卜时下一步一定要跳到洞里。兔子每一步只能选择上下左右四个方向行走一个格子。

兔子一开始的位置和唯一的洞的位置都是给定的。求最短距离和该距离下的方案数。

- $2 \leq n, m \leq 8$ 。

在一个 $n \times m$ 的网格图农田上有一些萝卜，一只兔子要吃
完所有萝卜并跳到洞里，当开始吃萝卜时，如果还有萝卜那么下
一步一定要吃到萝卜。吃完萝卜时下一步一定要跳到洞里。兔子
每一步只能选择上下左右四个方向行走一个格子。

兔子一开始的位置和唯一的洞的位置都是给定的。求最短距
离和该距离下的方案数。

- $2 \leq n, m \leq 8$ 。
- Source: FZU 2199

- 从兔子所在位置开始 BFS，求出到每个萝卜的最短距离以及该距离下的方案数。

- 从兔子所在位置开始 BFS，求出到每个萝卜的最短距离以及该距离下的方案数。
- 枚举第一个吃到的萝卜，那么接下来要走的就是一条起点为该萝卜，终点为洞的简单路径，且除了终点外一路上都是萝卜。可以通过带独立插头的轮廓线 DP 解决。

- 从兔子所在位置开始 BFS，求出到每个萝卜的最短距离以及该距离下的方案数。
- 枚举第一个吃到的萝卜，那么接下来要走的就是一条起点为该萝卜，终点为洞的简单路径，且除了终点外一路上都是萝卜。可以通过带独立插头的轮廓线 DP 解决。
- 时间复杂度 $O(n^2 m^2 4^m)$ 。

Locally Linked Sequences

一个序列 a_1, a_2, \dots, a_m 被称为 linked 当且仅当存在 $i < j < k < l$, 使得 $a_i = a_k, a_j = a_l, a_i \neq a_j$ 。

Locally Linked Sequences

一个序列 a_1, a_2, \dots, a_m 被称为 linked 当且仅当存在 $i < j < k < l$, 使得 $a_i = a_k, a_j = a_l, a_i \neq a_j$ 。

对于一个长度为 n 的序列, 如果它任意一段长度为 m 的连续子序列都是 linked, 那么它就是满足条件的。

Locally Linked Sequences

一个序列 a_1, a_2, \dots, a_m 被称为 linked 当且仅当存在 $i < j < k < l$, 使得 $a_i = a_k, a_j = a_l, a_i \neq a_j$ 。

对于一个长度为 n 的序列, 如果它任意一段长度为 m 的连续子序列都是 linked, 那么它就是满足条件的。

给定 n, m, r , 请统计出所有长度为 n 的满足条件的序列的个数。

Locally Linked Sequences

一个序列 a_1, a_2, \dots, a_m 被称为 linked 当且仅当存在 $i < j < k < l$, 使得 $a_i = a_k, a_j = a_l, a_i \neq a_j$ 。

对于一个长度为 n 的序列, 如果它任意一段长度为 m 的连续子序列都是 linked, 那么它就是满足条件的。

给定 n, m, r , 请统计出所有长度为 n 的满足条件的序列的个数。

- $4 \leq m \leq 8, m \leq n \leq 50, 2 \leq r \leq 50$ 。

Locally Linked Sequences

一个序列 a_1, a_2, \dots, a_m 被称为 linked 当且仅当存在 $i < j < k < l$, 使得 $a_i = a_k, a_j = a_l, a_i \neq a_j$ 。

对于一个长度为 n 的序列, 如果它任意一段长度为 m 的连续子序列都是 linked, 那么它就是满足条件的。

给定 n, m, r , 请统计出所有长度为 n 的满足条件的序列的个数。

- $4 \leq m \leq 8, m \leq n \leq 50, 2 \leq r \leq 50$ 。
- Source: Andrew Stankevich Contest 27

Locally Linked Sequences

- 对于一个长度为 m 的序列，如果 $a_i = a_j$ ，那么就认为它们属于同一个连通分量，可以用所在连通分量的标号集合来表示一个序列的连通性，如 $(0, 1, 1, 0, 2)$ 。

Locally Linked Sequences

- 对于一个长度为 m 的序列，如果 $a_i = a_j$ ，那么就认为它们属于同一个连通分量，可以用所在连通分量的标号集合来表示一个序列的连通性，如 $(0, 1, 1, 0, 2)$ 。
- 注意到 $(0, 1, 1, 0, 2)$ 与 $(6, 3, 3, 1, 4)$ 是等价的，所以对于一个状态，可以通过 $O(m)$ 的线性扫描将其修改为最小表示。

Locally Linked Sequences

- 对于一个长度为 m 的序列，如果 $a_i = a_j$ ，那么就认为它们属于同一个连通分量，可以用所在连通分量的标号集合来表示一个序列的连通性，如 $(0, 1, 1, 0, 2)$ 。
- 注意到 $(0, 1, 1, 0, 2)$ 与 $(6, 3, 3, 1, 4)$ 是等价的，所以对于一个状态，可以通过 $O(m)$ 的线性扫描将其修改为最小表示。
- 具体方法为，从左到右扫描每个元素，如果一个数没有出现过则给这种数一个最小的没用过的标号，然后将每个数用标号替代。

Locally Linked Sequences

- 设 $f(i, S)$ 表示长度为 i 的序列，最后 m 个数的连通性为 S 的方案数。

Locally Linked Sequences

- 设 $f(i, S)$ 表示长度为 i 的序列，最后 m 个数的连通性为 S 的方案数。
- 考虑从 $f(i, S)$ 转移到 $f(i+1, T)$ ，设 S 中的数为 $0..x-1$ ，枚举 $i+1$ 的位置的数 y ，首先判断加入后该序列是否 linked，如果合法，那么如果 $0 \leq y \leq x-1$ ，则 $f(i+1, T) += f(i, S)$ ，如果 $y = x$ ，则 $f(i+1, T) += f(i, S) \times (r - x)$ 。

Locally Linked Sequences

- 设 $f(i, S)$ 表示长度为 i 的序列，最后 m 个数的连通性为 S 的方案数。
- 考虑从 $f(i, S)$ 转移到 $f(i+1, T)$ ，设 S 中的数为 $0..x-1$ ，枚举 $i+1$ 的位置的数 y ，首先判断加入后该序列是否 linked，如果合法，那么如果 $0 \leq y \leq x-1$ ，则 $f(i+1, T) += f(i, S)$ ，如果 $y = x$ ，则 $f(i+1, T) += f(i, S) \times (r - x)$ 。
- 设总状态数为 $|S|$ ，则时间复杂度为 $O(nmr|S|)$ 。

给定一个 $n \times m$ 的网格图，以及相邻格子之间边的边权。

给定一个 $n \times m$ 的网格图，以及相邻格子之间边的边权。
对于命题 x ，如果它为真，那么 $[x] = 1$ ，否则 $[x] = 0$ 。

给定一个 $n \times m$ 的网格图，以及相邻格子之间边的边权。

对于命题 x ，如果它为真，那么 $[x] = 1$ ，否则 $[x] = 0$ 。

对于一棵生成树，每个点的贡献为 $1 + [\text{有一条连向上边的边}] + [\text{有一条连向左边的边}]$ 。这棵树的贡献为所有点的贡献的乘积。

给定一个 $n \times m$ 的网格图，以及相邻格子之间边的边权。

对于命题 x ，如果它为真，那么 $[x] = 1$ ，否则 $[x] = 0$ 。

对于一棵生成树，每个点的贡献为 $1 + [\text{有一条连向上边的边}] + [\text{有一条连向左边的边}]$ 。这棵树的贡献为所有点的贡献的乘积。

求这个网格图所有最小生成树的贡献之和。

给定一个 $n \times m$ 的网格图，以及相邻格子之间边的边权。

对于命题 x ，如果它为真，那么 $[x] = 1$ ，否则 $[x] = 0$ 。

对于一棵生成树，每个点的贡献为 $1 + [\text{有一条连向上边的边}] + [\text{有一条连向左边的边}]$ 。这棵树的贡献为所有点的贡献的乘积。

求这个网格图所有最小生成树的贡献之和。

- $1 \leq n \leq 800, 1 \leq m \leq 7$ 。

给定一个 $n \times m$ 的网格图，以及相邻格子之间边的边权。

对于命题 x ，如果它为真，那么 $[x] = 1$ ，否则 $[x] = 0$ 。

对于一棵生成树，每个点的贡献为 $1 + [\text{有一条连向上边的边}] + [\text{有一条连向左边的边}]$ 。这棵树的贡献为所有点的贡献的乘积。

求这个网格图所有最小生成树的贡献之和。

- $1 \leq n \leq 800, 1 \leq m \leq 7$ 。
- Source: 2015 ACM/ICPC 亚洲区沈阳站

- 考虑轮廓线 DP，设 $f(i, j, S)$ 表示考虑到了 (i, j) 这个格子，轮廓线上 m 个点的连通性的最小表示为 S 的最小边权和， $g(i, j, S)$ 表示该情况下的贡献和。

- 考虑轮廓线 DP，设 $f(i, j, S)$ 表示考虑到了 (i, j) 这个格子，轮廓线上 m 个点的连通性的最小表示为 S 的最小边权和， $g(i, j, S)$ 表示该情况下的贡献和。
- 逐格转移的时候，只考虑这个点往上连的边和往左连的边，要么都不选，要么选一条，要么都选，需要根据 S 的情况分类讨论来处理。

- 考虑轮廓线 DP，设 $f(i, j, S)$ 表示考虑到了 (i, j) 这个格子，轮廓线上 m 个点的连通性的最小表示为 S 的最小边权和， $g(i, j, S)$ 表示该情况下的贡献和。
- 逐格转移的时候，只考虑这个点往上连的边和往左连的边，要么都不选，要么选一条，要么都选，需要根据 S 的情况分类讨论来处理。
- 时间复杂度 $O(nm^2|S|)$ 。

Black and White

一个 $n \times m$ 的棋盘，有的格子已经染上黑色或者白色，现在要求将所有未染色的格子染上黑色或白色，满足以下两个条件：

Black and White

一个 $n \times m$ 的棋盘，有的格子已经染上黑色或者白色，现在要求将所有未染色的格子染上黑色或白色，满足以下两个条件：

(1) 所有黑色的格子四连通，所有白色的格子也四连通。

Black and White

一个 $n \times m$ 的棋盘，有的格子已经染上黑色或者白色，现在要求将所有未染色的格子染上黑色或白色，满足以下两个条件：

- (1) 所有黑色的格子四连通，所有白色的格子也四连通。
- (2) 不存在一个 2×2 的子矩阵满足 4 个格子的颜色都相同。

Black and White

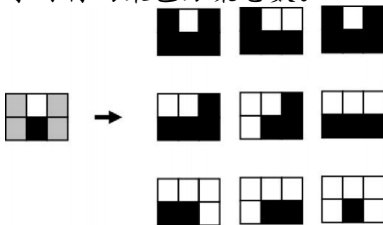
一个 $n \times m$ 的棋盘，有的格子已经染上黑色或者白色，现在要求将所有未染色的格子染上黑色或白色，满足以下两个条件：

- (1) 所有黑色的格子四连通，所有白色的格子也四连通。
 - (2) 不存在一个 2×2 的子矩阵满足 4 个格子的颜色都相同。
- 求可行的染色方案总数。

Black and White

一个 $n \times m$ 的棋盘，有的格子已经染上黑色或者白色，现在要求将所有未染色的格子染上黑色或白色，满足以下两个条件：

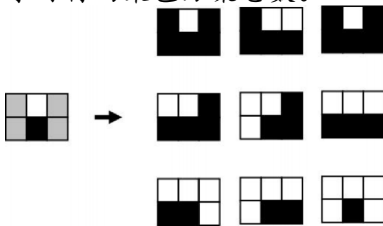
- (1) 所有黑色的格子四连通，所有白色的格子也四连通。
 - (2) 不存在一个 2×2 的子矩阵满足 4 个格子的颜色都相同。
- 求可行的染色方案总数。



Black and White

一个 $n \times m$ 的棋盘，有的格子已经染上黑色或者白色，现在要求将所有未染色的格子染上黑色或白色，满足以下两个条件：

- (1) 所有黑色的格子四连通，所有白色的格子也四连通。
 - (2) 不存在一个 2×2 的子矩阵满足 4 个格子的颜色都相同。
- 求可行的染色方案总数。

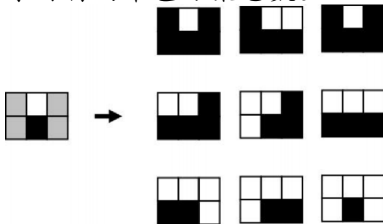


- $2 \leq n, m \leq 8$ 。

Black and White

一个 $n \times m$ 的棋盘，有的格子已经染上黑色或者白色，现在要求将所有未染色的格子染上黑色或白色，满足以下两个条件：

- (1) 所有黑色的格子四连通，所有白色的格子也四连通。
 - (2) 不存在一个 2×2 的子矩阵满足 4 个格子的颜色都相同。
- 求可行的染色方案总数。



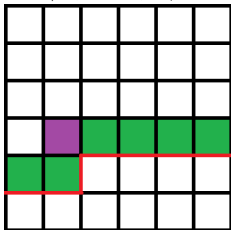
- $2 \leq n, m \leq 8$ 。
- Source: Uva 10572

Black and White

- 考虑轮廓线 DP，设 $f(i, j, A, B, C)$ 表示考虑到了 (i, j) 这个格子，轮廓线上 m 个点的连通性的最小表示为 A ，染色情况为 B ，转折点颜色为 C 的方案数。

Black and White

- 考虑轮廓线 DP，设 $f(i, j, A, B, C)$ 表示考虑到了 (i, j) 这个格子，轮廓线上 m 个点的连通性的最小表示为 A ，染色情况为 B ，转折点颜色为 C 的方案数。
- 如下图，红色部分为轮廓线，绿色部分为轮廓线上的点，紫色部分为转折点：



Black and White

- 转移的时候，枚举当前格子要染的颜色 X ，设左边颜色为 L ，上面颜色为 U ，左上角颜色为 C ，那么如果 $X = L = U = C$ ，它就不满足条件 (2)，要舍弃这个状态。

- 转移的时候，枚举当前格子要染的颜色 X ，设左边颜色为 L ，上面颜色为 U ，左上角颜色为 C ，那么如果 $X = L = U = C$ ，它就不满足条件 (2)，要舍弃这个状态。
- 考虑条件 (1)，如果 $X \neq U$ ，且 U 是孤立的一个连通块，那么会造成 U 这一块的封闭。

Black and White

- 转移的时候，枚举当前格子要染的颜色 X ，设左边颜色为 L ，上面颜色为 U ，左上角颜色为 C ，那么如果 $X = L = U = C$ ，它就不满足条件 (2)，要舍弃这个状态。
- 考虑条件 (1)，如果 $X \neq U$ ，且 U 是孤立的一个连通块，那么会造成 U 这一块的封闭。
- 如果 U 封闭，且轮廓线上颜色为 U 的格子多于 1 个，那么肯定不满足条件 (1)。

Black and White

- 转移的时候，枚举当前格子要染的颜色 X ，设左边颜色为 L ，上面颜色为 U ，左上角颜色为 C ，那么如果 $X = L = U = C$ ，它就不满足条件 (2)，要舍弃这个状态。
- 考虑条件 (1)，如果 $X \neq U$ ，且 U 是孤立的一个连通块，那么会造成 U 这一块的封闭。
- 如果 U 封闭，且轮廓线上颜色为 U 的格子多于 1 个，那么肯定不满足条件 (1)。
- 如果 U 封闭，且轮廓线上颜色为 U 的格子只有 1 个，那么如果当前不是 $(n, m - 1)$ 或 (n, m) ，那么肯定不能同时满足条件 (1) 和条件 (2)。

- 在 (n, m) 处更新答案时，需要检验轮廓线上是否只有不超过 2 个连通块。

Black and White

- 在 (n, m) 处更新答案时，需要检验轮廓线上是否只有不超过 2 个连通块。
- 对于下面这种特殊情况，即 $X = C, X \neq L, X \neq U$ 时，也不满足条件 (1)，不能更新答案：



Black and White

- 在 (n, m) 处更新答案时，需要检验轮廓线上是否只有不超过 2 个连通块。
- 对于下面这种特殊情况，即 $X = C, X \neq L, X \neq U$ 时，也不满足条件 (1)，不能更新答案：



- 时间复杂度 $O(nm^2|S|)$ 。

Czarnoksieznicy okraglego stolu

n 个人的编号依次为 1 到 n ，他们围着圆桌坐成一圈。座位相邻的两个人，其编号之差的绝对值不可以超过 p 。

Czarnoksieznicy okraglego stolu

n 个人的编号依次为 1 到 n ，他们围着圆桌坐成一圈。座位相邻的两个人，其编号之差的绝对值不可以超过 p 。

他们之中有些人不喜欢别人，一共有 k 对这种关系。如果 a 不喜欢 b ，那么 b 不能坐在 a 右边的那一个位置上。

Czarnoksieznicy okraglego stolu

n 个人的编号依次为 1 到 n ，他们围着圆桌坐成一圈。座位相邻的两个人，其编号之差的绝对值不可以超过 p 。

他们之中有些人不喜欢别人，一共有 k 对这种关系。如果 a 不喜欢 b ，那么 b 不能坐在 a 右边的那一个位置上。

现在，假设第 1 个人的座位已经固定，要给剩下的人安排座位，求合法方案的总数。

Czarnoksieznicy okraglego stolu

n 个人的编号依次为 1 到 n , 他们围着圆桌坐成一圈。座位相邻的两个人, 其编号之差的绝对值不可以超过 p 。

他们之中有些人不喜欢别人, 一共有 k 对这种关系。如果 a 不喜欢 b , 那么 b 不能坐在 a 右边的那一个位置上。

现在, 假设第 1 个人的座位已经固定, 要给剩下的人安排座位, 求合法方案的总数。

- $1 \leq n \leq 1000000, 0 \leq k \leq 1000000, 0 \leq p \leq 3$ 。

Czarnoksieznicy okraglego stolu

n 个人的编号依次为 1 到 n , 他们围着圆桌坐成一圈。座位相邻的两个人, 其编号之差的绝对值不可以超过 p 。

他们之中有些人不喜欢别人, 一共有 k 对这种关系。如果 a 不喜欢 b , 那么 b 不能坐在 a 右边的那一个位置上。

现在, 假设第 1 个人的座位已经固定, 要给剩下的人安排座位, 求合法方案的总数。

- $1 \leq n \leq 1000000, 0 \leq k \leq 1000000, 0 \leq p \leq 3$ 。
- Source: POI 2015

Czarnoksieznicy okraglego stolu

- $p = 0$ 或 $p = 1$, 特判即可。

Czarnoksieznicy okraglego stolu

- $p = 0$ 或 $p = 1$, 特判即可。
- $p = 2$, 只有如下两种可能的合法方案, 分别检验即可:

1 3 5 7 8 6 4 2

1 2 4 6 8 7 5 3

Czarnoksieznicy okraglego stolu

- $p = 0$ 或 $p = 1$, 特判即可。
- $p = 2$, 只有如下两种可能的合法方案, 分别检验即可:
1 3 5 7 8 6 4 2
1 2 4 6 8 7 5 3
- 下面重点讨论 $p = 3$ 的情况。

Czarnoksieznicy okrągłego stołu

- 首先可以假设 1 的位置也不固定，最后答案再除以 n 即可。

Czarnoksieznicy okrągłego stołu

- 首先可以假设 1 的位置也不固定，最后答案再除以 n 即可。
- 考虑将每个人依次插入序列，可以发现对于 $i+1$ 的位置安排，我们只关心 $i-2, i-1, i$ 的相对顺序、它们的相邻情况以及它们是否在边界上。

Czarnoksieznicy okraglego stolu

- 首先可以假设 1 的位置也不固定，最后答案再除以 n 即可。
- 考虑将每个人依次插入序列，可以发现对于 $i+1$ 的位置安排，我们只关心 $i-2, i-1, i$ 的相对顺序、它们的相邻情况以及它们是否在边界上。
- 设 $f(i, S_0, S_1, S_2)$ 表示已经安排了前 i 个人的座位， $i-2, i-1, i$ 的顺序为 S_0 ，边界情况为 S_1 ，相邻情况为 S_2 的方案数。

Czarnoksieznicy okraglego stolu

- 首先可以假设 1 的位置也不固定，最后答案再除以 n 即可。
- 考虑将每个人依次插入序列，可以发现对于 $i+1$ 的位置安排，我们只关心 $i-2, i-1, i$ 的相对顺序、它们的相邻情况以及它们是否在边界上。
- 设 $f(i, S_0, S_1, S_2)$ 表示已经安排了前 i 个人的座位， $i-2, i-1, i$ 的顺序为 S_0 ，边界情况为 S_1 ，相邻情况为 S_2 的方案数。
- 因为状态转移并不好写，所以在一开始通过枚举和模拟先预处理出所有转移。DP 的时候再对 k 对关系作判断即可。

Czarnoksieznicy okraglego stolu

- 首先可以假设 1 的位置也不固定，最后答案再除以 n 即可。
- 考虑将每个人依次插入序列，可以发现对于 $i+1$ 的位置安排，我们只关心 $i-2, i-1, i$ 的相对顺序、它们的相邻情况以及它们是否在边界上。
- 设 $f(i, S_0, S_1, S_2)$ 表示已经安排了前 i 个人的座位， $i-2, i-1, i$ 的顺序为 S_0 ，边界情况为 S_1 ，相邻情况为 S_2 的方案数。
- 因为状态转移并不好写，所以在一开始通过枚举和模拟先预处理出所有转移。DP 的时候再对 k 对关系作判断即可。
- 时间复杂度 $O(3!2^6n)$ 。

Thank you for your attention

Thank you!