购物 解题报告

南京师范大学附属中学 杨天祺

1 题目来源

题目改编加强自 Atcoder Grand Contest 018 Problem E. Sightseeing Plan[1]。原题为本题的 $n = 1$ 的版本的一个弱化版。

2 题目大意

在 $H \times W$ 的网格中有 $n+2$ 个矩形（标号为 $0 \ldots n+1$），矩形以 (x_{l_i}, y_{l_i}) 为左上角，(x_{r_i}, y_{r_i}) 为右下角，这 $n+2$ 个矩形的横坐标和纵坐标都递增，你要从第 0 个矩形中选一个点作为起点，第 $n+1$ 个矩形中选一个点作为终点，然后选一条从起点到终点的路径。

对于第 1 个到第 n 个矩形中的第 i 个矩形，如果路径经过第 i 个矩形中的 c_i 个格子，这条路径的权值就乘 $2^{c_i} - 1$（c 个餐馆中选出一个非空集合进入），即这条路径的权值为

$$\prod_{i=1}^{n} (2^{c_i} - 1)$$

求所有可行路径的权值和，答案对 998244353 取模。

3 数据规模

假设

$$AMX = \max_{0 \le i \le n+1} \{\max\{x_{r_i} - x_{l_i} + 1, y_{r_i} - y_{l_i} + 1\}\}$$

对于所有数据，有

- $2 \le H, W \le 2 \times 10^5$
- $0 \le n \le \min\{H, W\} - 2$
- 对于 $\forall\ 0 \le i \le n+1$，有 $1 \le x_{l_i} \le x_{r_i} \le H, 1 \le y_{l_i} \le y_{r_i} \le W$
- 对于 $\forall\ 0 \le i \le n$，有 $x_{r_i} < x_{l_{i+1}}, y_{r_i} < y_{l_{i+1}}$

[1]题目链接：https://beta.atcoder.jp/contests/agc018/tasks/agc018_e

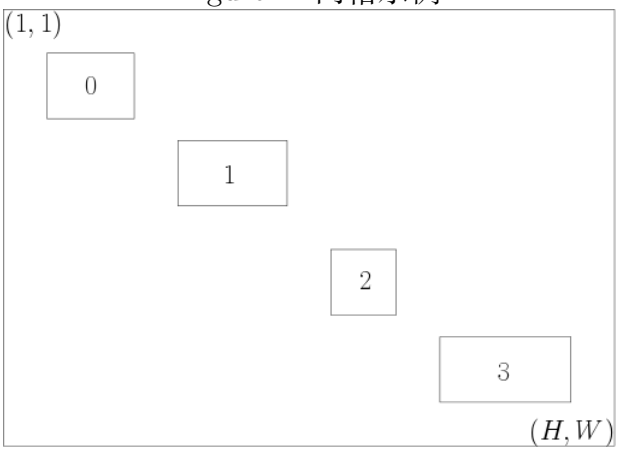
子任务：

子任务编号	分值	H, W	n	AMX
1	2	≤ 4	——	——
2	4	≤ 300	≤ 300	——
3	9	$\leq 2 \times 10^3$	——	——
4	4	——	——	$= 1$
5	7	——	$= 0$	$\leq 10^3$
6	10	——	$= 0$	——
7	5	——	$= 1$	$\leq 10^3$
8	24	——	$= 1$	——
9	35	——	——	——

4 约定

我们假设把 $H \times W$ 的网格画在平面上，遵循上北下南左西右东的方向，即左上角是 $(1,1)$，右下角是 (H,W)。例如下图所示：

Figure 1: 网格示例



假设 $belong_{i,j}$ 表示第 i 行第 j 列的点属于第几个矩形，如果不属于任何矩形，则令 $belong_{i,j} = -1$。

由于某些位置下标的原因，令 $xl_i = x_{l_i}$，$yl_i = y_{l_i}$，$xr_i = x_{r_i}$，$yr_i = y_{r_i}$。

5 暴力算法

5.1 算法 1

直接搜索所有合法的路径，求一下路径的权值大小，最后把所有路径求和就行了。复杂度取决于搜索实现的好坏，一般是能过第一个 Subtask 的。可以得

到 2 分。

- 时间复杂度：未知
- 空间复杂度：未知
- 期望通过子任务：1
- 期望得分：2 分

5.2 算法 2

考虑 DP。令 $dp_{i,j,k}$ 表示当前人在第 i 行第 j 列，上一个选的点（餐馆）是在第 k 个矩形中，那么我们就可以考虑从上边或者左边的格子转移。如果当前在某个矩形 $1 \dots n$ 内，就枚举当前是否进入餐馆。

$$dp_{i,j,k} = \begin{cases} 0, & i = 0 \text{ or } j = 0 \\ dp_{i-1,j,k} + dp_{i,j-1,k} + 1, & belong_{i,j} = 0 \text{ and } k = 0 \\ dp_{i-1,j,k} + dp_{i,j-1,k} + dp_{i-1,j,k-1} + dp_{i,j-1,k-1}, & belong_{i,j} = k \text{ and } k \le n \\ dp_{i-1,j,k} + dp_{i,j-1,k}, & \text{otherwise} \end{cases}$$

注意 k 应当 $\le n$，所以 $belong_{i,j} = n+1$ 的时候要单独判一下。然后按照这个式子转移即可。

- 时间复杂度：$\Theta(HWn)$
- 空间复杂度：$\Theta(HWn)$
- 期望通过子任务：1, 2
- 期望得分：7

5.3 算法 3

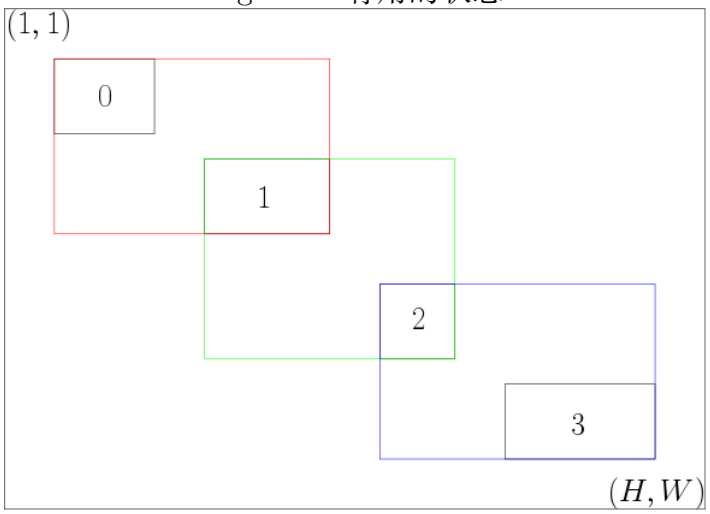
发现上面的算法其实有很多状态时没用的，其实可用的状态没多少。例如对于 $k = 0$，其实有用的 (i, j) 只有下图中红色矩形的部分，$k = 1$ 有用的只有绿色矩形的部分，$k = 2$ 有用的只有蓝色矩形的部分。

发现每个位置最多只会被两个矩形覆盖到，是它自己属于的矩形（如果有的话）和它左上方第一个矩形。那么只转移这些状态就行了。

实际实现时可以用 $dp_{i,j,k}$ 其中 $k = 0/1$ 表示状态，转移分几种情况讨论一下就行了。

- 时间复杂度：$\Theta(HW)$

Figure 2: 有用的状态



- 空间复杂度：$\Theta(HW)$
- 期望通过子任务：1, 2, 3
- 期望得分：18

6 发掘性质

6.1 算法 4

引理 6.1. *对于两个格子 (x_1, y_1)，(x_2, y_2)，如果 $x_1 \le x_2, y_1 \le y_2$，则从 (x_1, y_1) 到 (x_2, y_2)，每次向右或者向下走，不同路径的方案数为 $\binom{x_2-x_1+y_2-y_1}{x_2-x_1}$。*

Proof. 考虑走的序列，每次要么向右，要么向下，一共要走 $x_2 - x_1 + y_2 - y_1$ 次，其中恰好有 $x_2 - x_1$ 次向下走。因此共有 $\binom{x_2-x_1+y_2-y_1}{x_2-x_1}$ 种不同的方式选取走的序列，即不同的路径数。 □

有了引理 6.1，我们就可以做 Subtask4 了，由于 $AMX = 1$，因此每个矩形都是 1×1 的，因此起点固定，终点固定，每个矩形都必须经过，且每个矩形对应的那个点必须选。因此就是依次经过每个矩形的方案数。即答案为

$$\prod_{i=0}^{n} \binom{x_{l_{i+1}} - x_{l_i} + y_{l_{i+1}} - y_{l_i}}{x_{l_{i+1}} - x_{l_i}}$$

组合数可以通过预处理阶乘和阶乘逆元实现，逆元可以用费马小定理求出，预处理前 n 个数的阶乘和阶乘的逆元的复杂度是 $\Theta(n)$。然后用 $\binom{n}{m} = \frac{n!}{m!(n-m)!}$ 计算即可，计算的时间复杂度是 $\Theta(1)$ 的。

- 时间复杂度：$\Theta(n+H+W)$
- 空间复杂度：$\Theta(n+H+W)$
- 期望通过子任务：4
- 期望得分：4

6.2 算法 5

考虑 $n=0$ 怎么做。这个时候没有餐馆，只是从一个矩形中任意一个起点到一共矩形中任意一个终点的方案数。我们考虑枚举终点，考虑从所有起点到这一个终点 (x,y) 的方案数，这应该等于

$$\sum_{i=x_{l_0}}^{x_{r_0}}\sum_{j=y_{l_0}}^{y_{r_0}}\binom{x-i+y-j}{x-i} \tag{1}$$

考虑下面这个结论

引理 6.2. *对于给定的 n，m，有*

$$\sum_{i=0}^{n}\binom{i}{m}=\binom{n+1}{m+1}$$

Proof. 考虑对 n 做数学归纳法。

- 当 $n=0$ 时，结论显然成立。
- 当 $n>0$ 时，有

$$\begin{aligned}\sum_{i=0}^{n}\binom{i}{m}&=\sum_{i=0}^{n-1}\binom{i}{m}+\binom{n}{m}\\&=\binom{n}{m+1}+\binom{n}{m}\\&=\binom{n+1}{m+1}\end{aligned}$$

□

利用这个结论，我们就可以推出：

引理 6.3. *对于给定的常数 a，b，有*

$$\sum_{i=0}^{n-1}\sum_{j=0}^{m-1}\binom{i+a+j+b}{i+a}=\binom{a+n+b+m}{a+n}-\binom{a+m+b}{a}-\binom{a+n+b}{b}+\binom{a+b}{a}$$

Proof.

$$
\begin{aligned}
&\sum_{i=0}^{n-1}\sum_{j=0}^{m-1}\binom{i+a+j+b}{i+a}\\
=&\sum_{i=0}^{n-1}\left(\sum_{j=0}^{b+m-1}\binom{i+a+j}{i+a}-\sum_{j=0}^{b-1}\binom{i+a+j}{i+a}\right)\\
=&\sum_{i=0}^{n-1}\left(\binom{i+a+b+m}{b+m-1}-\binom{i+a+b}{b-1}\right)\\
=&\sum_{i=0}^{n-1}\binom{b+m-1+i+a+1}{b+m-1}-\sum_{i=0}^{n-1}\binom{b-1+i+a+1}{b-1}\\
=&\left(\sum_{i=0}^{a+n}\binom{b+m-1+i}{b+m-1}-\sum_{i=0}^{a}\binom{b+m-1+i}{b+m-1}\right)-\left(\sum_{i=0}^{a+n}\binom{b-1+i}{b-1}-\sum_{i=0}^{a}\binom{b-1+i}{b-1}\right)\\
=&\binom{a+n+b+m}{a+n}-\binom{a+m+b}{a}-\binom{a+n+b}{b}+\binom{a+b}{a}
\end{aligned}
$$

□

则我们可以在常数时间内求出 (1) 式。那么我们枚举终点就可以在 $\Theta(HW)$ 的复杂度内求出 $n=0$ 的解。

- 时间复杂度：$O(AMX^2)$
- 空间复杂度：$O(AMX^2)$
- 期望通过子任务：5
- 期望得分：8

6.3 算法 6

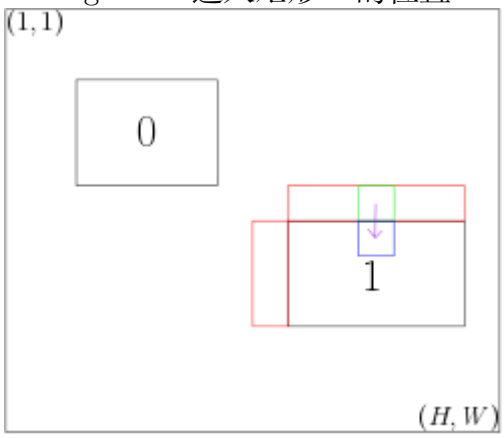
我们考虑如何优化算法 5。算法 5 的瓶颈在于枚举每一个终点，而引理 6.3 我们只对一个方向使用了。考虑如何对两个方向都使用引理 6.3。

我们取矩形 1 的左上边框，即下图中红色标出的那些点，那么每条路径是从某一个红色的格子**进入**矩形 1，例如从绿色的那个格子进入矩形 1，到达蓝色的格子，如紫色的箭头所示。

发现这个进入矩形 1 的格子一定是唯一的，因此可以通过枚举某一个红色的格子，计算有多少从当前红色格子进入矩形 1 的路径个数。那么根据乘法原理，这个值就等于这个格子到矩形 0 任意一个格子的路径数乘上这个格子到矩形 1 中任意能到的格子的路径数。

由于矩形的边框长度是 $O(H+W)$ 的，所以总复杂度就是 $O(H+W)$ 的。

Figure 3: 进入矩形 1 的位置

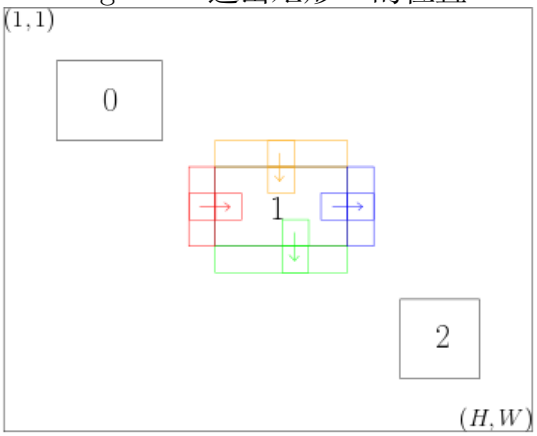


- 时间复杂度：$O(H+W)$
- 空间复杂度：$O(H+W)$
- 期望通过子任务：5, 6
- 期望得分：23

6.4 算法 7

考虑 $n=1$ 怎么办。考虑把算法 6 扩展到 $n=1$ 的情况。因为要知道经过矩形 1 的路径长度，没法直接用算法 6。考虑枚举矩形 1 是从哪里进从哪里出的，如下图，进一定是在左上边界（见下图中红色和橙色格子），出一定是在右下边界（见下图中绿色和蓝色格子）。考虑路径的进入点（如红色和橙色箭头）和离开点（如绿色和蓝色箭头）。

Figure 4: 进出矩形 1 的位置



那么每条路径应该是存在唯一的进入点和离开点。而且通过进入点和离开点，我们可以唯一的确定这条路径在矩形 1 中的长度。这样我们就可以直接计算这个进入点和这个离开点对应的路径个数和路径权值了。

具体的，应当分四种情况讨论：

下面为了书写方便，我们定义：

$$x_1, x_2, x_3, x_4, x_5, x_6 = x_{l_0}, x_{r_0}, x_{l_1}, x_{r_1}, x_{l_2}, x_{r_2}$$

$$y_1, y_2, y_3, y_4, y_5, y_6 = y_{l_0}, y_{r_0}, y_{l_1}, y_{r_1}, y_{l_2}, y_{r_2}$$

$$n = x_4 - x_3 + 1, m = y_4 - y_3 + 1$$

我们设任意一个点到起点方案为 $f_S(x,y)$ ，到终点方案为 $f_T(x,y)$，这可以直接通过引理 6.3 求出。

6.4.1 上边界 → 下边界

考虑从橙色格子进入，绿色格子离开的路径的权值总和。假设进入是从 $(x_i - 1, y_i) \to (x_i, y_i)$，离开是从 $(x_o, y_o) \to (x_o + 1, y_o)$，则这一部分的答案是：

$$ans_1 = \sum_{y_i=y_3}^{y_4} \sum_{y_o=y_i}^{y_4} f_S(x_i - 1, y_i) * f_T(x_o + 1, y_o) * \binom{x_o - x_i + y_o - y_i}{x_o - x_i} * 2^{x_o - x_i + y_o - y_i + 1} \quad (2)$$

6.4.2 上边界 → 右边界

考虑从橙色格子进入，蓝色格子离开的路径的权值总和。假设进入是从 $(x_i - 1, y_i) \to (x_i, y_i)$，离开是从 $(x_o, y_o) \to (x_0, y_o + 1)$。则这一部分的答案是：

$$ans_2 = \sum_{y_i=y_3}^{y_4} \sum_{x_o=x_3}^{x_4} f_S(x_i - 1, y_i) * f_T(x_o, y_o + 1) * \binom{x_o - x_i + y_o - y_i}{x_o - x_i} * 2^{x_o - x_i + y_o - y_i + 1} \quad (3)$$

6.4.3 剩下两种情况

左边界进入的两种情况分别和上边界进入的两种情况类似，这里就不列举了。

由于进入点和离开点各有 $O(AMX)$ 个，故

- 时间复杂度：$O(AMX^2)$
- 空间复杂度：$O(AMX^2)$
- 期望通过子任务：7
- 期望得分：9

6.5 算法 8

我们考虑把算法 7 的式子具体化简。

6.5.1 上边界 → 下边界

考虑优化式 (2)。则有

$$\begin{aligned}
ans_1 &= \sum_{y_i=y_3}^{y_4}\sum_{y_o=y_i}^{y_4} f_S(x_i-1,y_i) * f_T(x_o+1,y_o) * \binom{x_o-x_i+y_o-y_i}{x_o-x_i} * 2^{x_o-x_i+y_o-y_i+1} \\
&= \sum_{y_i=y_3}^{y_4}\sum_{y_o=y_i}^{y_4} f_S(x_3-1,y_i) * f_T(x_4+1,y_o) * \binom{n-1+y_o-y_i}{n-1} * 2^{n+y_o-y_i} \\
&= \sum_{i=0}^{m-1}\sum_{j=i}^{m-1} f_S(x_3-1,y_3+i) * f_T(x_4+1,y_3+j) * \binom{n-1+y_3+j-y_3-i}{n-1} * 2^{n+y_3+j-y_3-i} \\
&= \sum_{i=0}^{m-1}\sum_{j=0}^{m-1-i} f_S(x_3-1,y_3+i) * f_T(x_4+1,y_4-j) * \binom{n-1+y_4-j-y_3-i}{n-1} * 2^{n+y_4-j-y_3-i} \\
&= \sum_{i=0}^{m-1}\sum_{j=0}^{m-1-i} f_S(x_3-1,y_3+i) * f_T(x_4+1,y_4-j) * \binom{n+m-j-i-2}{n-1} * 2^{n+m-j-i-1}
\end{aligned}$$

我们设 $f_i = f_S(x_3-1,y_3+i),\ g_i = f_T(x_4+1,y_4-i)$ ，则有

$$ans_1 = \sum_{0\le i+j\le m-1} f_i * g_j * \binom{n+m-j-i-2}{n-1} * 2^{n+m-j-i-1}$$

发现这正好是一个卷积的形式。若令 $h = f * g$ ，就有

$$ans_1 = \sum_{i=0}^{m-1} h_i * \binom{n+m-i-2}{n-1} * 2^{n+m-i-1}$$

于是如果我们用 FFT 优化计算 h ，则可以在 $\Theta(n\log n)$ 的时间复杂度内求解 ans_1。由于这题模数 $998244353 = 119 * 2^{23} + 1$，因此可以用 NTT 来解决。

6.5.2 上边界 → 右边界

考虑优化式 (3)。则有

$$\begin{aligned} ans_2 &= \sum_{y_i=y_3}^{y_4}\sum_{x_o=x_3}^{x_4} f_S(x_i-1,y_i) * f_T(x_o,y_o+1) * \binom{x_o-x_i+y_o-y_i}{x_o-x_i} * 2^{x_o-x_i+y_o-y_i+1} \\ &= \sum_{y_i=y_3}^{y_4}\sum_{x_o=x_3}^{x_4} f_S(x_3-1,y_i) * f_T(x_o,y_4+1) * \binom{x_o-x_3+y_4-y_i}{x_o-x_3} * 2^{x_o-x_3+y_4-y_i+1} \\ &= \sum_{i=0}^{m-1}\sum_{j=0}^{n-1} f_S(x_3-1,y_4-i) * f_T(x_3+j,y_4+1) * \binom{i+j}{i} * 2^{i+j+1} \end{aligned}$$

同样我们设 $f_i = f_S(x_3-1,y_4-i)$, $g_i = f_T(x_3+i,y_4+1)$ ，则有

$$ans_2 = \sum_{i=0}^{m-1}\sum_{j=0}^{n-1} f_i * g_j * \binom{i+j}{i} * 2^{i+j+1}$$

于是我们令 $h_n = \sum_{i+j=n}\binom{n}{i} * f_i * g_j$ ，我们发现后面就是一个指数型生成函数卷积的形式，于是我们强行把 f 和 g 写出指数型生成函数 f' 和 g' ，于是就有 h 的指数型生成函数 $h' = f' * g'$ 。于是就可以用 FFT 来优化了。其实就是

$$\frac{f_n}{n!} = \sum_{i+j=n}\frac{f_i}{i!} * \frac{g_j}{j!}$$

于是就有

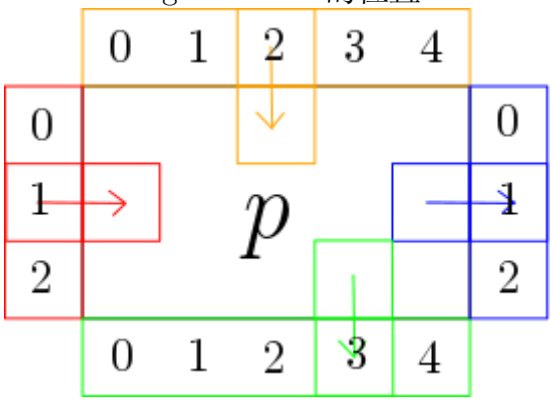
$$ans_2 = \sum_{i=0}^{n+m-2} h_i * 2^{i+1}$$

这样这一部分就也可以用 FFT 优化做到 $\Theta(n\log n)$。

剩下两种情况也可以类似的优化，于是可以算出答案了。

- 时间复杂度：$O((H+W)\log(H+W))$
- 空间复杂度：$O(H+W)$
- 期望通过子任务：7, 8
- 期望得分：30

Figure 5: DP 的位置



7 标准算法

7.1 算法 9

我们考虑如何做 $n \le 2$ 的情况。

显然不能像算法 6-8 那样枚举一个分界线。我们回到算法 3，考虑那个 DP 算法。受算法 8 的启发，我们可以考虑只维护每个矩形边界的 DP 值。

假设当前在考虑第 p 个矩形。设 $dpU_{p,i}$ 表示上边界上从左往右第 i 个位置的 dp 值，这里 i 从 0 开始（如上图所示），形式化地，令 $dpU_{p,i} = dp_{xl_p-1,yl_p+i,p-1}$。类似地，定义

- $dpU_{p,i} = dp_{xl_p-1,yl_p+i,p-1}$
- $dpD_{p,i} = dp_{xr_p+1,yl_p+i,p}$
- $dpL_{p,i} = dp_{xl_p+i,yl_p-1,p-1}$
- $dpR_{p,i} = dp_{xl_p+i,yr_p+1,p}$

考虑转移。一共有 8 种转移方式（前 4 种是矩形内部的路径转移，后 4 种是从第 p 个矩形走到第 $p+1$ 个矩形的路径转移）：

1. $dpU_p \to dpD_p$
2. $dpU_p \to dpR_p$
3. $dpL_p \to dpD_p$
4. $dpL_p \to dpR_p$
5. $dpD_p \to dpU_{p+1}$
6. $dpD_p \to dpL_{p+1}$

7. $dpR_p \to dpU_{p+1}$

8. $dpR_p \to dpL_{p+1}$

7.1.1 转移方式 1：$dpU_p \to dpD_p$

令 h_i 为由 dpU_p 转移到 $dpD_{p,i}$ 的和，$f_i = dpU_{p,i}$，$n = yr_p - yl_p$，$m = xr_p - xl_p$。则有

$$h_i = \sum_{j=0}^{i} \binom{i-j+m}{m} f_j \left(2^{i-j+m+1} - 1\right)$$

令 $g_i = \binom{i+m-1}{m-1}\left(2^{i+m} - 1\right)$，则有 $h = f * g$，直接 FFT 优化即可。

7.1.2 转移方式 2：$dpU_p \to dpR_p$

令 h_i 为由 dpU_p 转移到 $dpR_{p,i}$ 的和，$f_i = dpU_{p,i}$，$n = yr_p - yl_p$，$m = xr_p - xl_p$，则有

$$h_i = \sum_{j=0}^{n} \binom{i+n-j}{i} f_j \left(2^{i+n-j+1} - 1\right)$$

这东西不能直接做，我们对它变个形：

$$\begin{aligned} h_i &= \sum_{j=0}^{n} \binom{i+n-j}{i} f_j \left(2^{i+n-j+1} - 1\right) \\ &= \sum_{j=0}^{n} \frac{(i+n-j)!}{i!(n-j)!} f_j \left(2^{i+n-j+1} - 1\right) \\ i! \cdot h_i &= \sum_{j=0}^{n} \frac{(i+n-j)!}{(n-j)!} f_j \left(2^{i+n-j+1} - 1\right) \end{aligned}$$

我们若令 $f'_i = \frac{f_j}{(n-j)!}$，$g_i = i!\left(2^{i+1} - 1\right)$，$h'_i = (i-n)! \cdot h_{i-n}$，则 $h'_i = f'_i * g_i$，则可用 FFT 优化。

剩下数种情况都就可以类似上面两种情况优化，于是所有 8 种转移都可以优化了。注意转移 5 和 8 的时候，如果两个的对应坐标相邻，则是不能转移的，即转移 5 中，当 $xl_{i+1} - xr_i = 1$ 时，以及转移 8 中，当 $yl_{i+1} - yr_i = 1$ 时，是不能转移的。

DP 初始情况可以用引理 6.3 计算，答案就直接用 dpU_{n+1} 和 dpL_{n+1} 的值乘上引理 6.3 的系数就行了。

由于 $\sum_{p=0}^{n+1} dpU_{p,i} \le H$，$\sum_{p=0}^{n+1} dpD_{p,i} \le H$，$\sum_{p=0}^{n+1} dpL_{p,i} \le W$，$\sum_{p=0}^{n+1} dpR_{p,i} \le W$，因此总复杂度是 $\Theta((H+W)\log(H+W))$ 的。

- 时间复杂度：$O((H+W)\log(H+W))$
- 空间复杂度：$O((H+W)\log(H+W))$
- 期望通过子任务：1, 2, 3, 4, 5, 6, 7, 8, 9
- 期望得分: 100

8 扩展问题

8.1 任意模数

本题可以直接扩展到任意模数，任意模数情况下的 FFT 可参见参考文献 [2]。

8.2 更大范围的 H, W

这题的标准算法如果转移情况 5、6、7、8 实现较为精细的话，复杂度可以写成 $\Theta(S\log S)$。其中，

$$S=\sum_{i=0}^{n+1}(xr_i-xl_i+1)+(yr_i-yl_i+1)$$

因此其实本题可以去掉 $\forall\ 0\le i\le n$，$x_{r_i}<x_{l_{i+1}}, y_{r_i}<y_{l_{i+1}}$ 这一限制。但是要多讨论几种情况。

当然，这样 H 和 W 就可以做到更大，此时瓶颈在于求组合数，在模数小的时候可以用 Lucas 定理，模数大的时候可以分块打表。

9 总结

考虑到选手的考场策略等因素，由于本题是本场较简单的题，选手得分应相对较高。

- 对于子任务 1,2,4，难度较低，预计约有 90% 的选手能通过。
- 对于子任务 3，可以通过对子任务 2 的 DP 进行简单优化得到，预计约有 80% 的选手能通过。
- 对于子任务 5，需要一点简单的数学能力，预计也约有 80% 的选手能通过。
- 对于子任务 6，需要发掘本题最核心的性质，即取一个分割线然后考虑分割线到两端的情况，思维难度提升。预计约有 60% 的选手能通过。
- 对于子任务 7，需要进一步利用这一性质，考虑矩形的四周边界。预计 50% 选手能通过。

- 对于子任务 8，需要用 FFT 优化这一性质，代码难度提升，预计大约 40% 的选手能通过。
- 对于子任务 9，需要结合子任务 3 和子任务 8，将 FFT 直接优化答案变成优化 DP 的转移，需讨论一系列情况，思维难度和代码难度都较高，预计大约 30% 的选手能通过。

总体而言，本题涉及动态规划、组合计数、公式推导等知识点，需要选手能够灵活运用其它工具（FFT）优化这个动态规划。有一定的代码难度，不仅考察选手动态规划的能力，还考察选手通过组合计数进行推导的、分析性质、运用 FFT 对这一动态规划的转移进行优化的能力。代码难度中等，需要选手灵活运用 8 种转移之间的相似关系减少代码中不同的转移数量，降低代码难度。在本场中，定位为较简单的题，难度略低于近年 NOI 的 T2 难度。

References

[1] Atcoder Grand Contest 018 Editorial：https://atcoder.jp/img/agc018/editorial.pdf

[2] 毛啸. 再探快速傅里叶变换. 2016 年信息学奥林匹克中国国家队候选队论文.

[3] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein. Introduction to Algorithms. MIT Press, 1990.