原题识别 (old) 解题报告

Claris

2018 年 5 月 13 日

原题识别 (old.c/cpp/pas)

给定一棵 n 个点的以 1 为根的树，每个点有一个颜色 a_i，m 次询问：

原题识别 (old.c/cpp/pas)

给定一棵 n 个点的以 1 为根的树，每个点有一个颜色 a_i，m 次询问：

$1\ x\ y$：询问 x 到 y 路径上本质不同的颜色数。

原题识别 (old.c/cpp/pas)

给定一棵 n 个点的以 1 为根的树，每个点有一个颜色 a_i，m 次询问：

$1\ x\ y$：询问 x 到 y 路径上本质不同的颜色数。

$2\ A\ B$：询问 x 在 A 到 1 路径上随机选，y 在 B 到 1 路径上随机选时，询问 1 答案的期望值。

原题识别 (old.c/cpp/pas)

给定一棵 n 个点的以 1 为根的树，每个点有一个颜色 a_i，m 次询问：

$1\ x\ y$：询问 x 到 y 路径上本质不同的颜色数。

$2\ A\ B$：询问 x 在 A 到 1 路径上随机选，y 在 B 到 1 路径上随机选时，询问 1 答案的期望值。

- 存在 30% 的数据，只包含第一类询问。

原题识别 (old.c/cpp/pas)

给定一棵 n 个点的以 1 为根的树，每个点有一个颜色 a_i，m 次询问：

$1\ x\ y$：询问 x 到 y 路径上本质不同的颜色数。

$2\ A\ B$：询问 x 在 A 到 1 路径上随机选，y 在 B 到 1 路径上随机选时，询问 1 答案的期望值。

- 存在 30% 的数据，只包含第一类询问。
- 存在 30% 的数据，树退化成链。

原题识别 (old.c/cpp/pas)

给定一棵 n 个点的以 1 为根的树，每个点有一个颜色 a_i，m 次询问：

$1\ x\ y$：询问 x 到 y 路径上本质不同的颜色数。

$2\ A\ B$：询问 x 在 A 到 1 路径上随机选，y 在 B 到 1 路径上随机选时，询问 1 答案的期望值。

- 存在 30% 的数据，只包含第一类询问。
- 存在 30% 的数据，树退化成链。
- 对于 100% 的数据，$n \leq 100000, m \leq 200000$，树形态和颜色按指定代码生成。

原题识别 (old.c/cpp/pas)

给定一棵 n 个点的以 1 为根的树，每个点有一个颜色 a_i，m 次询问：

$1\ x\ y$：询问 x 到 y 路径上本质不同的颜色数。

$2\ A\ B$：询问 x 在 A 到 1 路径上随机选，y 在 B 到 1 路径上随机选时，询问 1 答案的期望值。

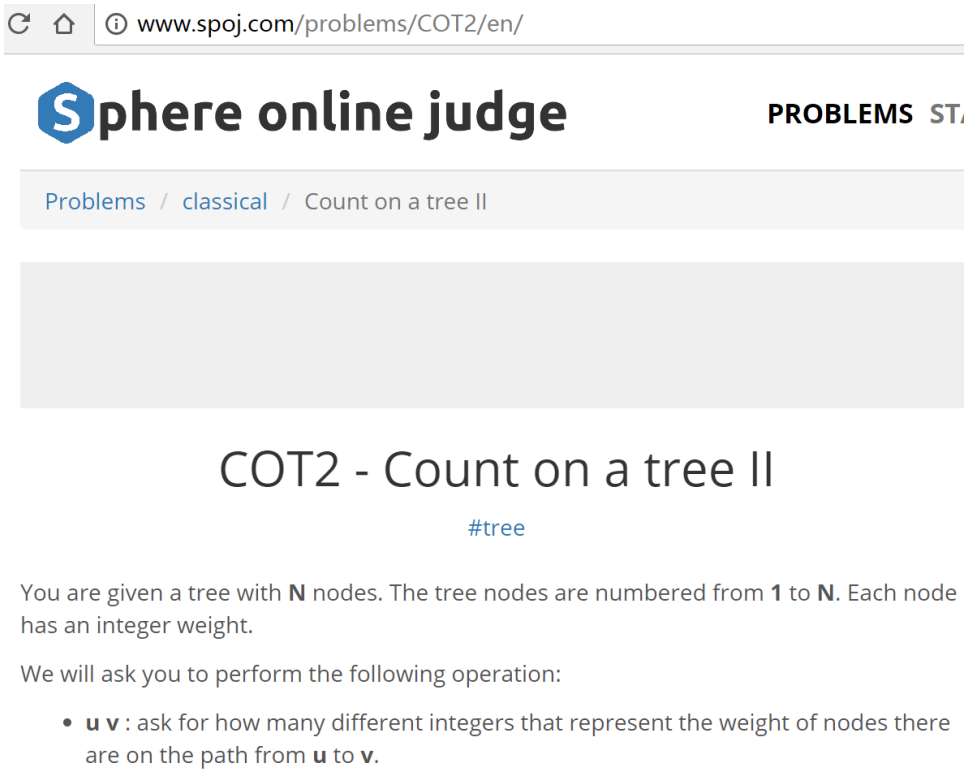
- 存在 30% 的数据，只包含第一类询问。
- 存在 30% 的数据，树退化成链。
- 对于 100% 的数据，$n \leq 100000, m \leq 200000$，树形态和颜色按指定代码生成。
- Shortest judge solution: 3776 bytes

30 分做法 1

- 存在 30% 的数据，只包含第一类询问。

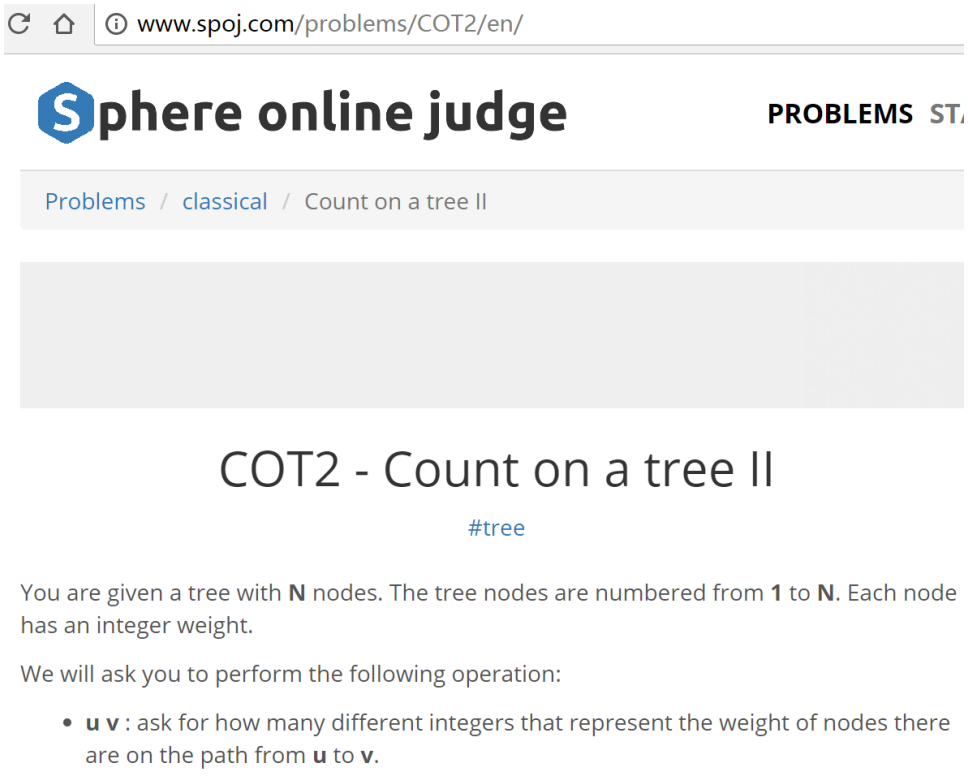
30 分做法 1

- 存在 30% 的数据，只包含第一类询问。



30 分做法 1

- 存在 30% 的数据，只包含第一类询问。



- ~~不知道大家有没有做过这个题呢？~~

30 分做法 1

- 对于每个点 x 求出 DFS 时入栈和出栈的时间 st_x, en_x。

30 分做法 1

- 对于每个点 x 求出 DFS 时入栈和出栈的时间 st_x, en_x。
- 对于询问 x, y，假设 $st_x \leq st_y$，则 $[st_x, st_y]$ 中每个点出现 1 到 2 次。

30 分做法 1

- 对于每个点 x 求出 DFS 时入栈和出栈的时间 st_x, en_x。
- 对于询问 x, y，假设 $st_x \leq st_y$，则 $[st_x, st_y]$ 中每个点出现 1 到 2 次。
- 可以发现出现 2 次的点不在 x 到 y 的路径上，而出现 1 次的点在路径上。

30 分做法 1

- 对于每个点 x 求出 DFS 时入栈和出栈的时间 st_x, en_x。
- 对于询问 x, y，假设 $st_x \leq st_y$，则 $[st_x, st_y]$ 中每个点出现 1 到 2 次。
- 可以发现出现 2 次的点不在 x 到 y 的路径上，而出现 1 次的点在路径上。
- x 和 y 的 LCA 是特例，需要特判。

30 分做法 1

- 对于每个点 x 求出 DFS 时入栈和出栈的时间 st_x, en_x。
- 对于询问 x, y，假设 $st_x \leq st_y$，则 $[st_x, st_y]$ 中每个点出现 1 到 2 次。
- 可以发现出现 2 次的点不在 x 到 y 的路径上，而出现 1 次的点在路径上。
- x 和 y 的 LCA 是特例，需要特判。
- 如此将树转化成序列后，使用莫队算法处理所有询问即可。

30 分做法 1

- 将序列分成 $O(\sqrt{n})$ 块。

30 分做法 1

- 将序列分成 $O(\sqrt{n})$ 块。
- 将所有询问按照左端点为第一关键字，右端点为第二关键字排序。

30 分做法 1

- 将序列分成 $O(\sqrt{n})$ 块。
- 将所有询问按照左端点为第一关键字，右端点为第二关键字排序。
- 暴力从上一个询问挪动 l 和 r 转移到下一个询问。

30 分做法 1

- 将序列分成 $O(\sqrt{n})$ 块。
- 将所有询问按照左端点为第一关键字，右端点为第二关键字排序。
- 暴力从上一个询问挪动 l 和 r 转移到下一个询问。
- 利用数组记录每个颜色出现次数，从而实现 $O(1)$ 转移。

30 分做法 1

- 将序列分成 $O(\sqrt{n})$ 块。
- 将所有询问按照左端点为第一关键字，右端点为第二关键字排序。
- 暴力从上一个询问挪动 l 和 r 转移到下一个询问。
- 利用数组记录每个颜色出现次数，从而实现 $O(1)$ 转移。
- 左端点每次移动次数不超过 $O(\sqrt{n})$，右端点每块总计移动次数 $O(n)$。

30 分做法 1

- 将序列分成 $O(\sqrt{n})$ 块。
- 将所有询问按照左端点为第一关键字，右端点为第二关键字排序。
- 暴力从上一个询问挪动 l 和 r 转移到下一个询问。
- 利用数组记录每个颜色出现次数，从而实现 $O(1)$ 转移。
- 左端点每次移动次数不超过 $O(\sqrt{n})$，右端点每块总计移动次数 $O(n)$。
- 故总时间复杂度为 $O(n\sqrt{n})$。

30 分做法 2

- 存在 30% 的数据，树退化成链。

30 分做法 2

- 存在 30% 的数据，树退化成链。
- 首先考虑如何处理第一类询问，不妨假设 $x \leq y$。

30 分做法 2

- 存在 30% 的数据，树退化成链。
- 首先考虑如何处理第一类询问，不妨假设 $x \leq y$。

1878: [SDOI2009]HH的项链

Time Limit: 4 Sec Memory Limit: 64 MB
Submit: 5614 Solved: 2783
[Submit][Status][Discuss]

Description

HH有一串由各种漂亮的贝壳组成的项链。HH相信不同的贝壳会带来好运，所以每次散步 完后，他都会随意取出一段贝壳，思考它们所表达的含义。HH不断地收集新的贝壳，因此他的项链变得越来越长。有一天，他突然提出了一个问题：某一段贝壳中，包含了多少种不同的贝壳？这个问题很难回答。。。因为项链实在是太长了。于是，他只好求助睿智的你，来解决这个问题。

Input

第一行：一个整数N，表示项链的长度。
第二行：N个整数，表示依次表示项链中贝壳的编号（编号为0到1000000之间的整数）。
第三行：一个整数M，表示HH询问的个数。
接下来M行：每行两个整数，L和R（1 ≤ L ≤ R ≤ N），表示询问的区间。

- N ≤ 50000，M ≤ 200000。

30 分做法 2

- 存在 30% 的数据，树退化成链。
- 首先考虑如何处理第一类询问，不妨假设 $x \leq y$。

1878: [SDOI2009]HH的项链

Time Limit: 4 Sec Memory Limit: 64 MB
Submit: 5614 Solved: 2783
[Submit][Status][Discuss]

Description

HH有一串由各种漂亮的贝壳组成的项链。HH相信不同的贝壳会带来好运，所以每次散步 完后，他都会随意取出一段贝壳，思考它们所表达的含义。HH不断地收集新的贝壳，因此他的项链变得越来越长。有一天，他突然提出了一个问题：某一段贝壳中，包含了多少种不同的贝壳？这个问题很难回答。。。因为项链实在是太长了。于是，他只好求助睿智的你，来解决这个问题。

Input

第一行：一个整数N，表示项链的长度。
第二行：N个整数，表示依次表示项链中贝壳的编号（编号为0到1000000之间的整数）。
第三行：一个整数M，表示HH询问的个数。
接下来M行：每行两个整数，L和R（1 ≤ L ≤ R ≤ N），表示询问的区间。
N ≤ 50000，M ≤ 200000。

- ~~不知道大家有没有做过这个题呢？~~

30 分做法 2

- 从 1 到 n 依次考虑每个 r 的询问。

30 分做法 2

- 从 1 到 n 依次考虑每个 r 的询问。
- 设 ans_l 表示对应 $[l, r]$ 询问的答案，$last_x$ 表示颜色 x 上一次出现的位置。

30 分做法 2

- 从 1 到 n 依次考虑每个 r 的询问。
- 设 ans_l 表示对应 $[l, r]$ 询问的答案，$last_x$ 表示颜色 x 上一次出现的位置。
- 当 r 增加 1 时，$[last_{a_{r+1}} + 1, r + 1]$ 的 ans 都需要增加 1。

30 分做法 2

- 从 1 到 n 依次考虑每个 r 的询问。
- 设 ans_l 表示对应 $[l, r]$ 询问的答案，$last_x$ 表示颜色 x 上一次出现的位置。
- 当 r 增加 1 时，$[last_{a_{r+1}} + 1, r + 1]$ 的 ans 都需要增加 1。
- 线段树维护即可。

30 分做法 2

- 从 1 到 n 依次考虑每个 r 的询问。
- 设 ans_l 表示对应 $[l, r]$ 询问的答案，$last_x$ 表示颜色 x 上一次出现的位置。
- 当 r 增加 1 时，$[last_{a_{r+1}} + 1, r + 1]$ 的 ans 都需要增加 1。
- 线段树维护即可。
- 时间复杂度 $O(n \log n)$。

30 分做法 2

- 第二类询问 A, B 等价于求 $x \leq A, y \leq B$ 的答案之和。

30 分做法 2

- 第二类询问 A, B 等价于求 $x \le A, y \le B$ 的答案之和。
- 设 sum_l 表示 ans_l 历史的和。

30 分做法 2

- 第二类询问 A, B 等价于求 $x \leq A, y \leq B$ 的答案之和。
- 设 sum_l 表示 ans_l 历史的和。
- 当 r 增加 1 时，还需要将 sum_l 加上对应的 ans_l 的值。

30 分做法 2

- 第二类询问 A, B 等价于求 $x \leq A, y \leq B$ 的答案之和。
- 设 sum_l 表示 ans_l 历史的和。
- 当 r 增加 1 时，还需要将 sum_l 加上对应的 ans_l 的值。
- 询问时求出 sum 的区间和，还需要补上 $y < A, x > y$ 部分的贡献，可以用同样的方法预处理出。

30 分做法 2

- 第二类询问 A, B 等价于求 $x \le A, y \le B$ 的答案之和。
- 设 sum_l 表示 ans_l 历史的和。
- 当 r 增加 1 时，还需要将 sum_l 加上对应的 ans_l 的值。
- 询问时求出 sum 的区间和，还需要补上 $y < A, x > y$ 部分的贡献，可以用同样的方法预处理出。
- 修改操作是线性变换，线段树维护矩阵即可。

30 分做法 2

- 第二类询问 A, B 等价于求 $x \le A, y \le B$ 的答案之和。
- 设 sum_l 表示 ans_l 历史的和。
- 当 r 增加 1 时，还需要将 sum_l 加上对应的 ans_l 的值。
- 询问时求出 sum 的区间和，还需要补上 $y < A, x > y$ 部分的贡献，可以用同样的方法预处理出。
- 修改操作是线性变换，线段树维护矩阵即可。
- 时间复杂度 $O(n \log n)$，常数很大，本人抠了很久常数才从 60 秒抠到 4 秒。

30 分做法 2

- 第二类询问 A, B 等价于求 $x \le A, y \le B$ 的答案之和。
- 设 sum_l 表示 ans_l 历史的和。
- 当 r 增加 1 时，还需要将 sum_l 加上对应的 ans_l 的值。
- 询问时求出 sum 的区间和，还需要补上 $y < A, x > y$ 部分的贡献，可以用同样的方法预处理出。
- 修改操作是线性变换，线段树维护矩阵即可。
- 时间复杂度 $O(n \log n)$，常数很大，本人抠了很久常数才从 60 秒抠到 4 秒。
- 更好的方法是设计出类似的标记，比矩阵常数小很多。

100 分做法

- 如何求 A 到根的信息之和？

100 分做法

- 如何求 A 到根的信息之和？
- 对于每个点 x 求出 DFS 时入栈和出栈的时间 st_x, en_x。

100 分做法

- 如何求 A 到根的信息之和？
- 对于每个点 x 求出 DFS 时入栈和出栈的时间 st_x, en_x。
- 那么 x 点的子树可以表示为 $[st_x, en_x]$。

100 分做法

- 如何求 A 到根的信息之和？
- 对于每个点 x 求出 DFS 时入栈和出栈的时间 st_x, en_x。
- 那么 x 点的子树可以表示为 $[st_x, en_x]$。
- 将 st 权值设为 1，en 权值设为 -1，则 $[1, st_x]$ 的加权和对应 x 到根的信息之和。

100 分做法

- 建立 $2n \times 2n$ 的矩阵，两维分别表示路径两端点的 DFS 序。

100 分做法

- 建立 $2n \times 2n$ 的矩阵，两维分别表示路径两端点的 DFS 序。
- 对于路径 x, y，答案为 k，则对应平面上 4 个点：

  (st_x, st_y)，权值 k。

  (st_x, en_y)，权值 $-k$。

  (en_x, st_y)，权值 $-k$。

  (en_x, en_y)，权值 k。

100 分做法

- 建立 $2n \times 2n$ 的矩阵，两维分别表示路径两端点的 DFS 序。
- 对于路径 x, y，答案为 k，则对应平面上 4 个点：

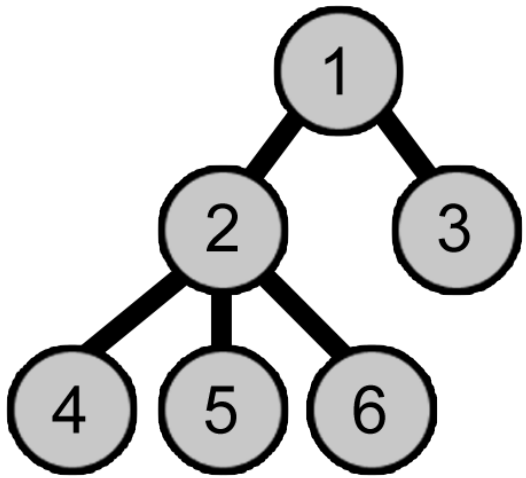
  (st_x, st_y)，权值 k。

  (st_x, en_y)，权值 $-k$。

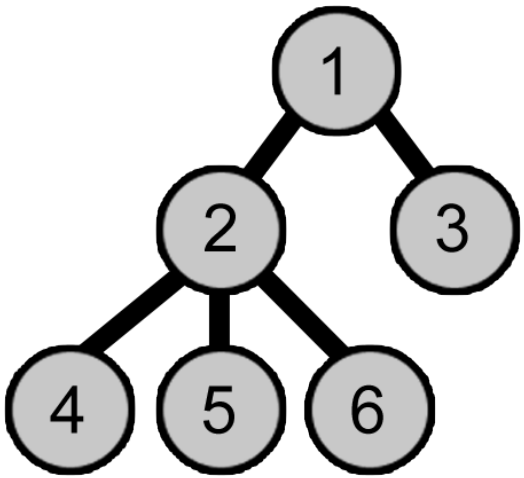
  (en_x, st_y)，权值 $-k$。

  (en_x, en_y)，权值 k。
- 则询问 A, B 的答案即为 (st_A, st_B) 左下角所有点的权值和。

100 分做法



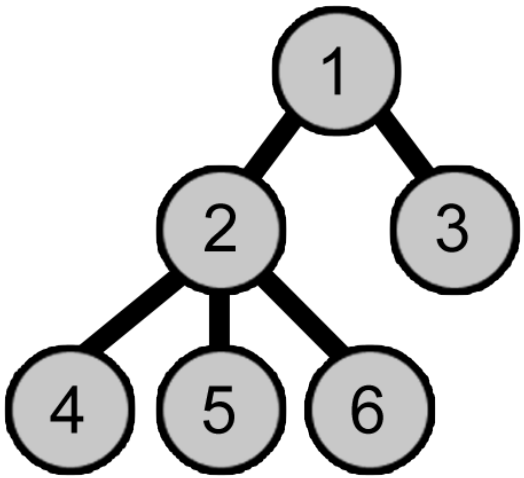
100 分做法

- 

- DFS 序 (+ 表示入栈，-表示出栈)：

1	2	3	4	5	6	7	8	9	10	11	12
+1	+2	+4	-4	+5	-5	+6	-6	-2	+3	-3	-1

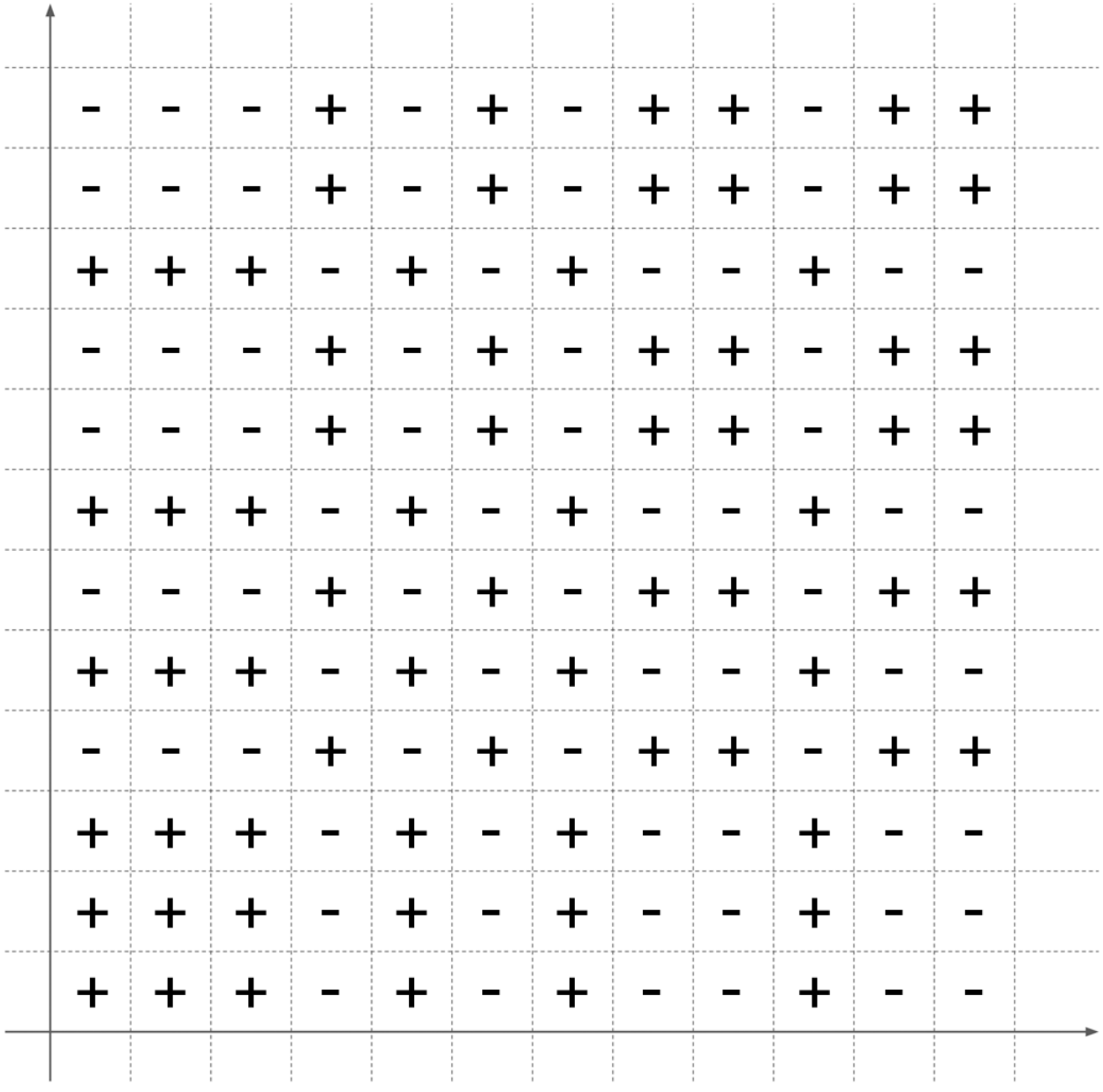
100 分做法

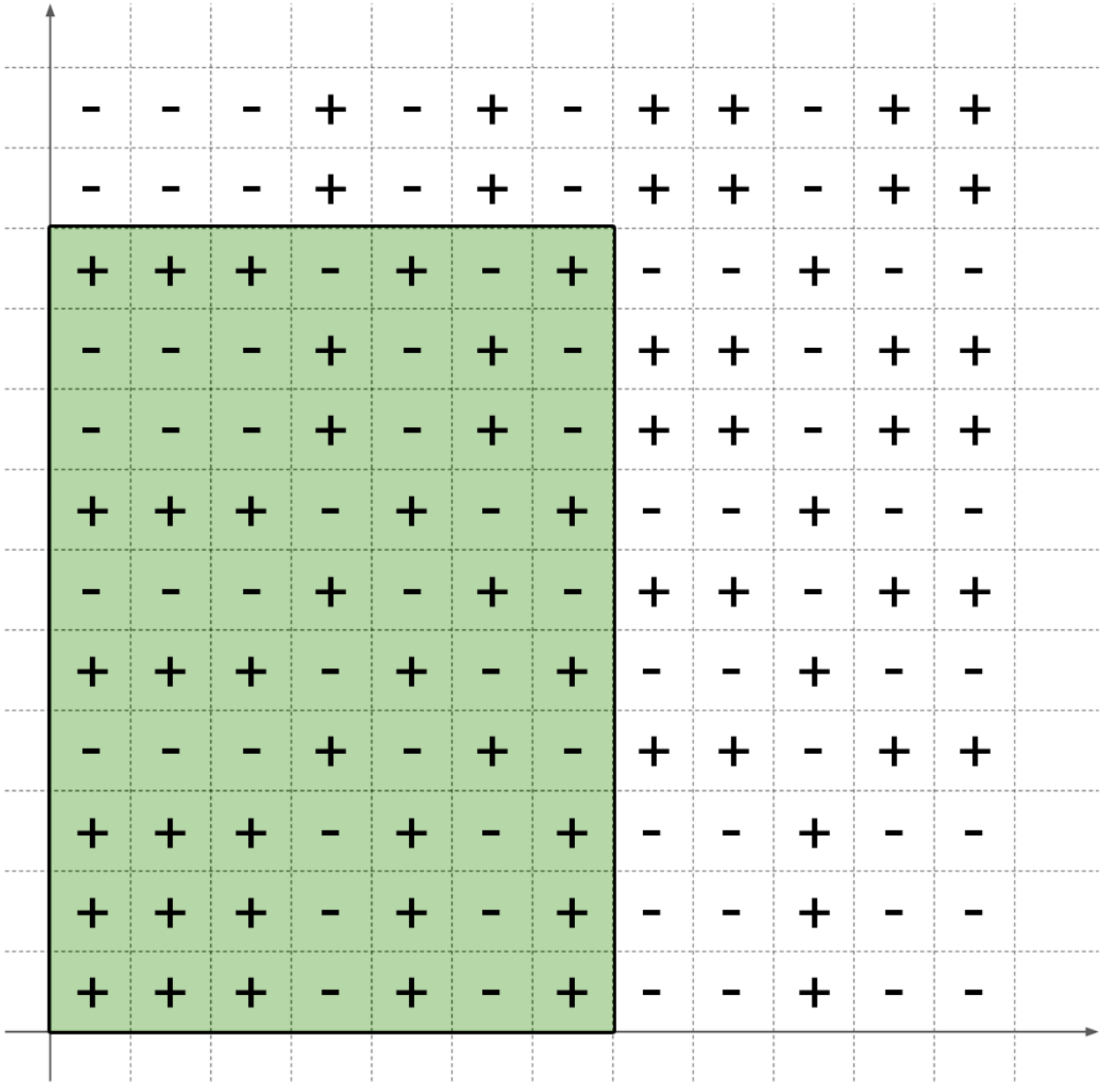
- 

- DFS 序 (+ 表示入栈，-表示出栈)：

1	2	3	4	5	6	7	8	9	10	11	12
+1	+2	+4	-4	+5	-5	+6	-6	-2	+3	-3	-1

- 询问 $A = 6, B = 3$ 对应平面上点 $(st_6, st_3) = (7, 10)$。





100 分做法

- 每种颜色对答案的贡献独立，分别处理。

100 分做法

- 每种颜色对答案的贡献独立，分别处理。
- 对于点 x，经过 x 的路径可以表示为：

100 分做法

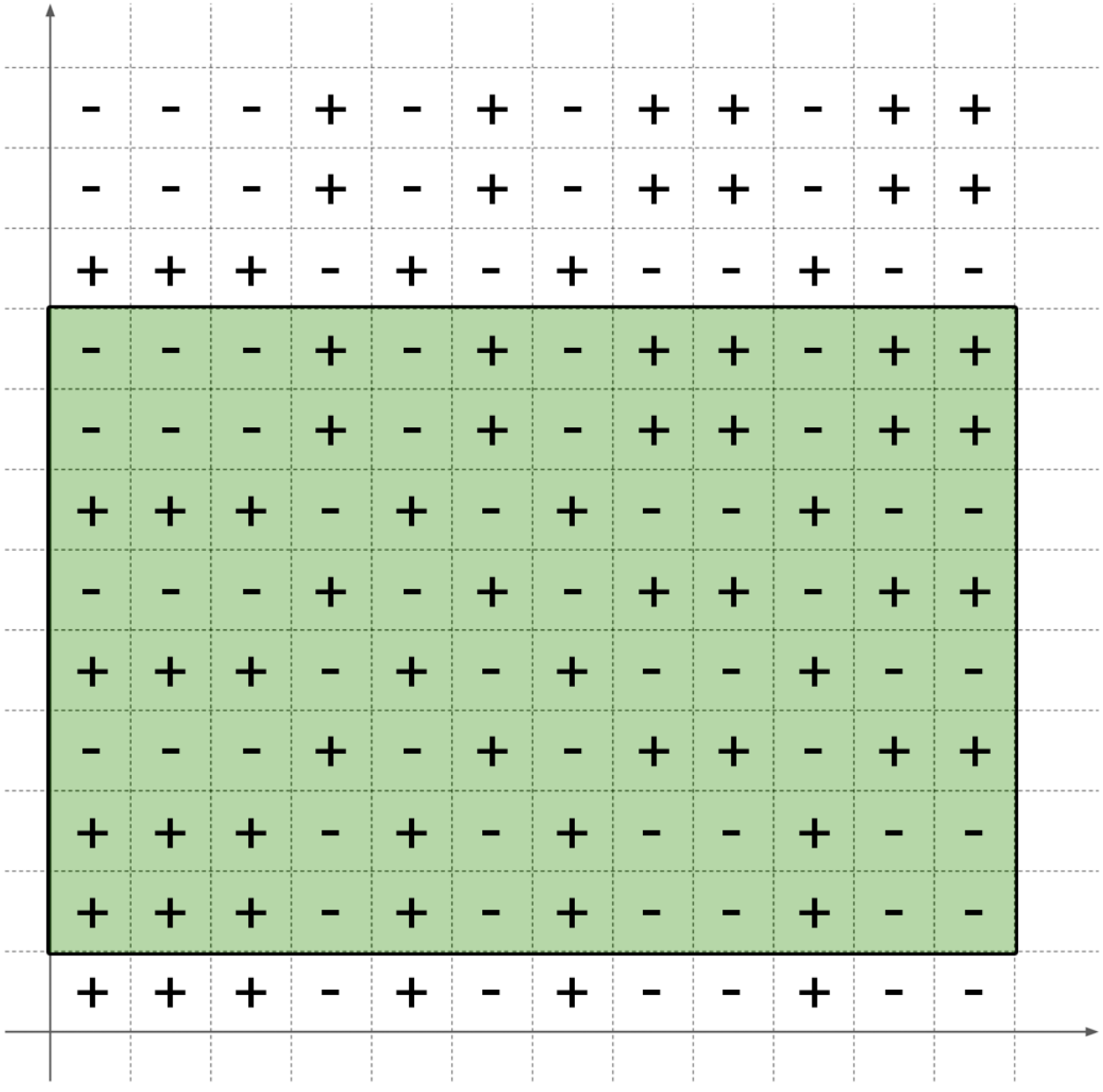
- 每种颜色对答案的贡献独立，分别处理。
- 对于点 x，经过 x 的路径可以表示为：
- 1. 至少有一个端点在 x 子树内部，即 $[1, 2n] \times [st_x, en_x]$ 和 $[st_x, en_x] \times [1, 2n]$。

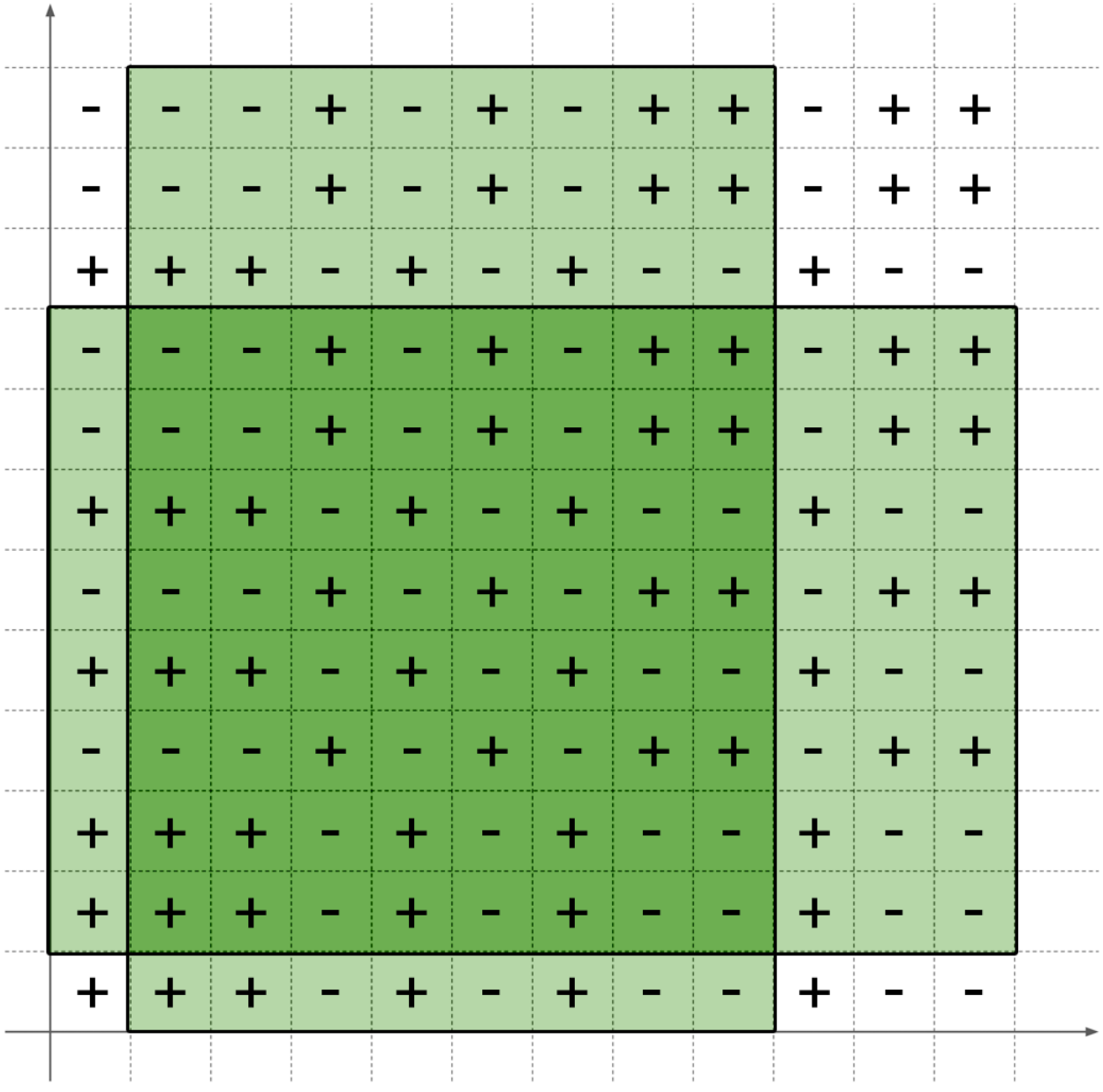
100 分做法

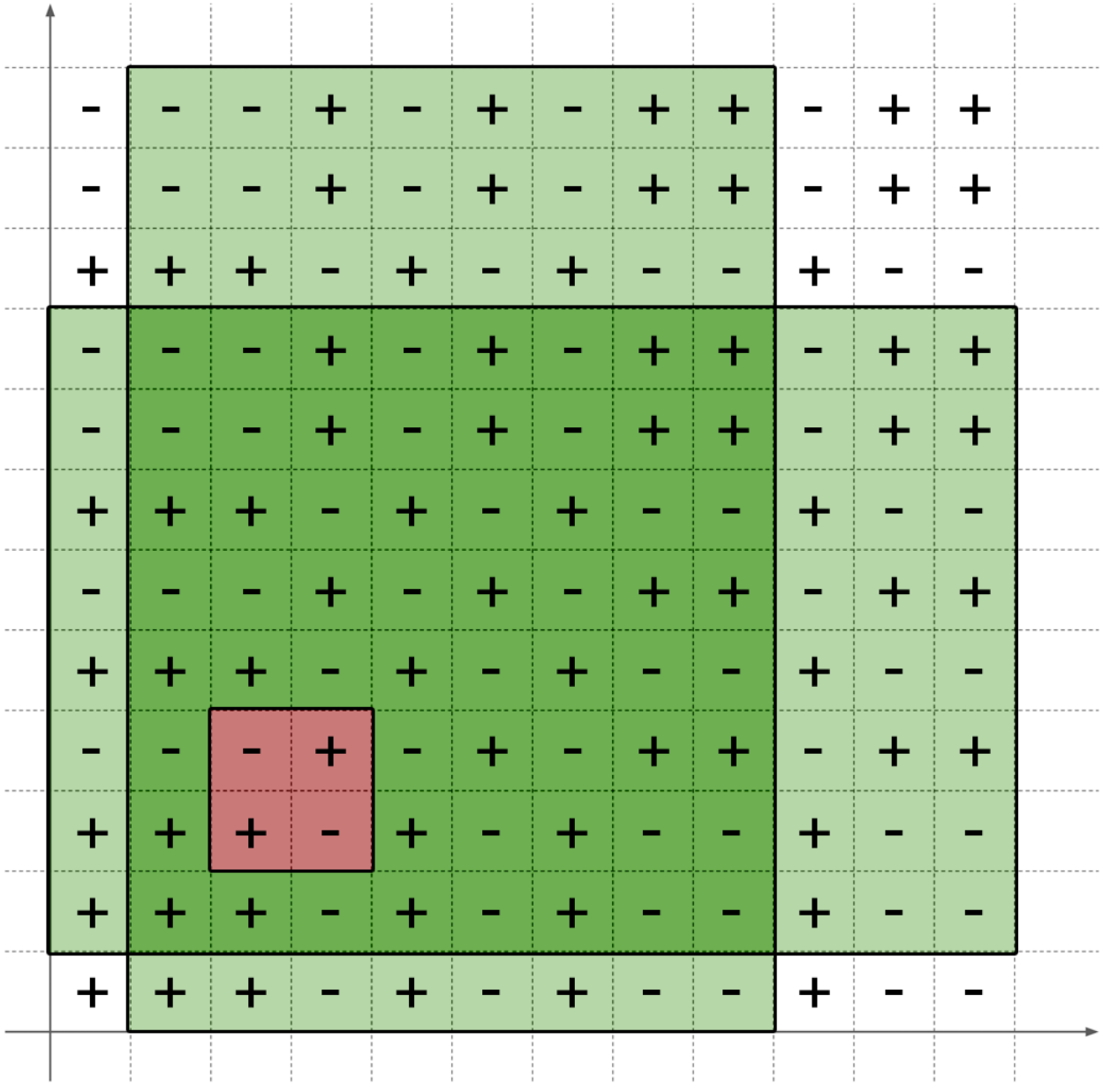
- 每种颜色对答案的贡献独立，分别处理。
- 对于点 x，经过 x 的路径可以表示为：
- 1. 至少有一个端点在 x 子树内部，即 $[1, 2n] \times [st_x, en_x]$ 和 $[st_x, en_x] \times [1, 2n]$。
- 2. 两个端点不能同时在 x 的任何一个儿子 y 内部，即不能在 $[st_y, en_y] \times [st_y, en_y]$。

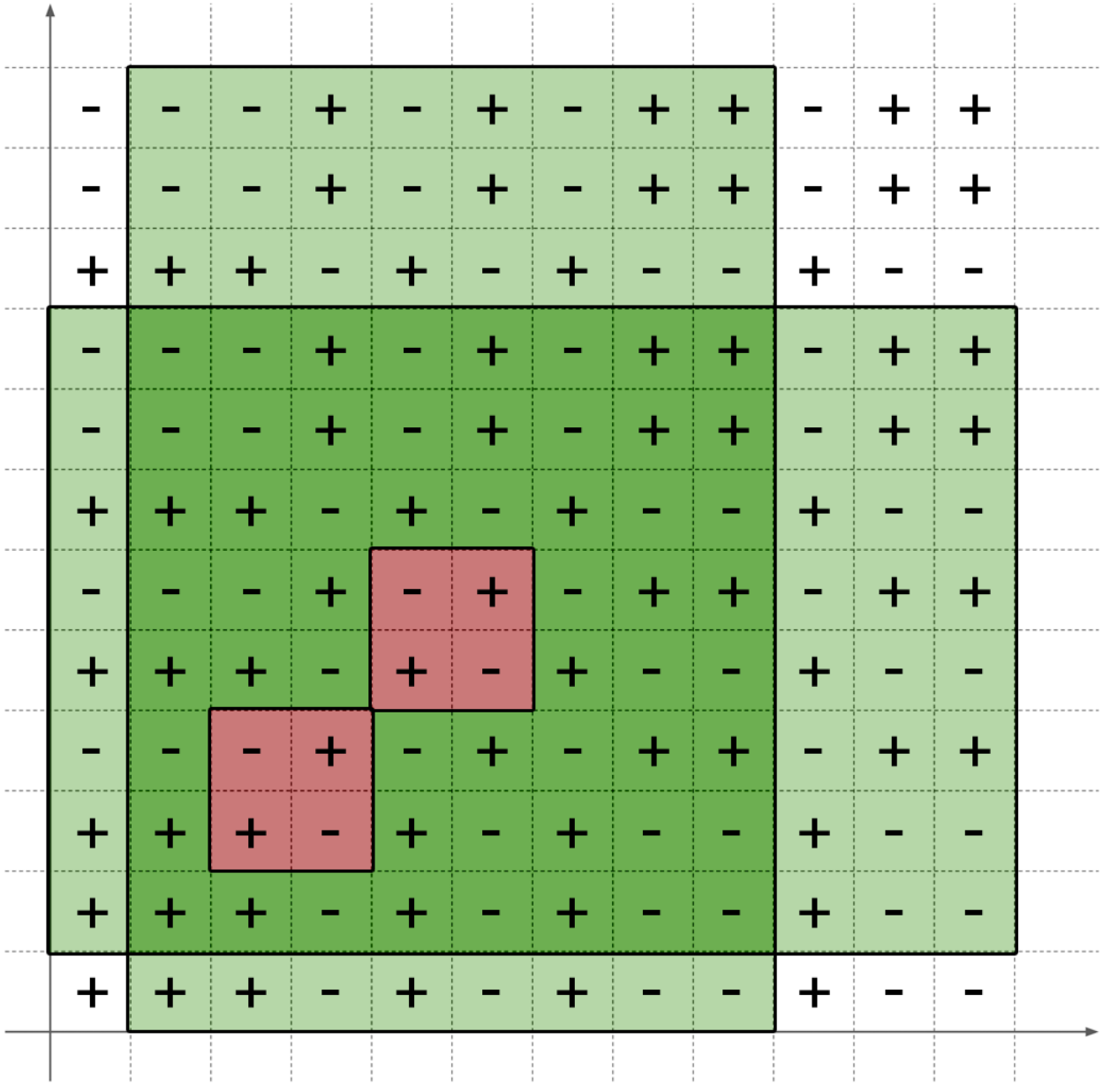
100 分做法

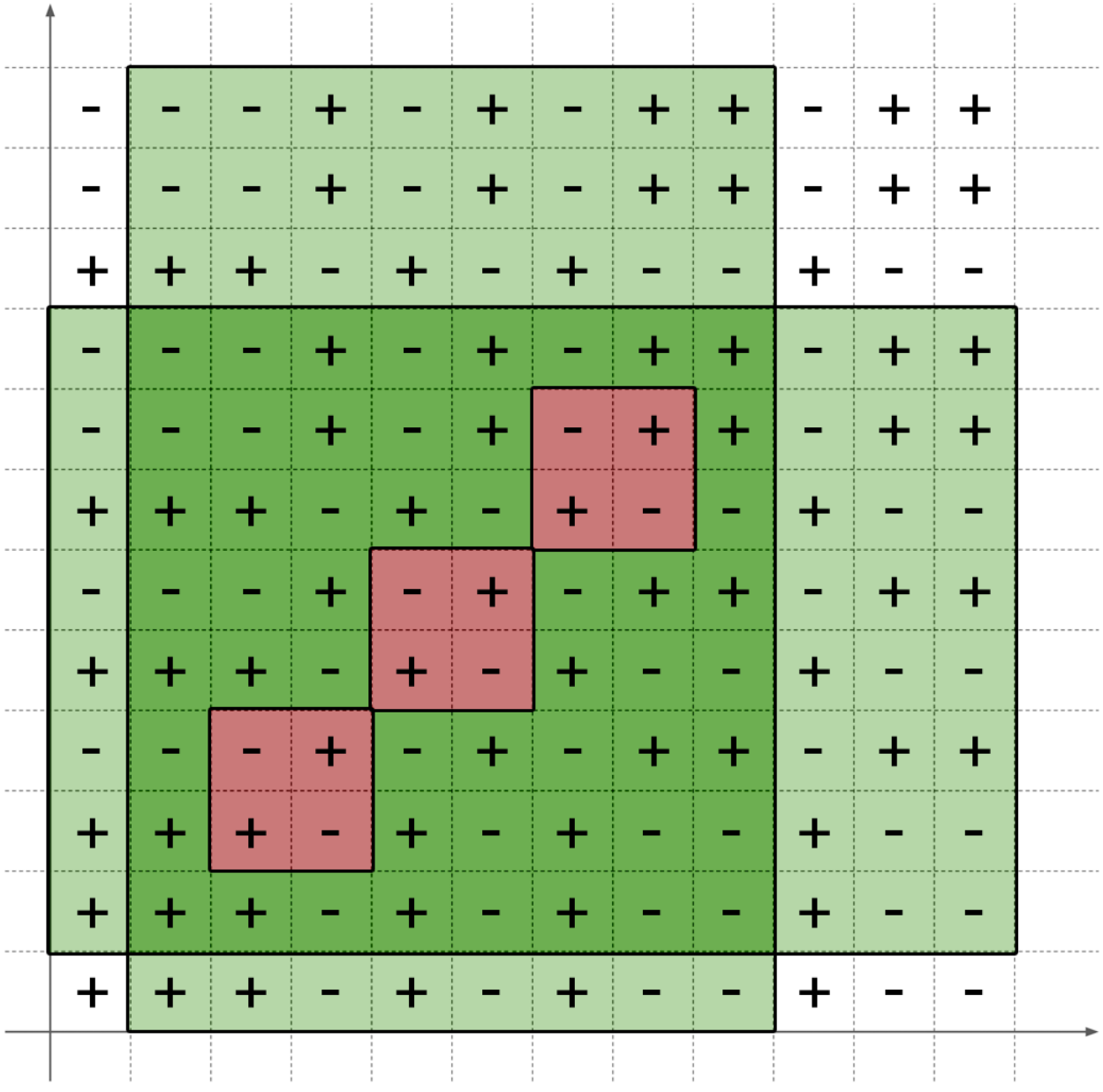
- 每种颜色对答案的贡献独立，分别处理。
- 对于点 x，经过 x 的路径可以表示为：
- 1. 至少有一个端点在 x 子树内部，即 $[1,2n]\times[st_x,en_x]$ 和 $[st_x,en_x]\times[1,2n]$。
- 2. 两个端点不能同时在 x 的任何一个儿子 y 内部，即不能在 $[st_y,en_y]\times[st_y,en_y]$。
- 例：经过 2 号点的路径对应 2 个可行区域和 3 个禁止区域。











100 分做法

- 将可行区域权值设为 1，禁止区域权值设为 -2。

100 分做法

- 将可行区域权值设为 1，禁止区域权值设为 -2。
- 若经过一个格子的权值和非 0，则说明该格子对应的路径上出现了当前颜色。

100 分做法

- 将可行区域权值设为 1，禁止区域权值设为 -2。
- 若经过一个格子的权值和非 0，则说明该格子对应的路径上出现了当前颜色。
- 将所有矩形的边界坐标离散化，通过二维差分前缀和完成矩形加操作。

100 分做法

- 将可行区域权值设为 1，禁止区域权值设为 -2。
- 若经过一个格子的权值和非 0，则说明该格子对应的路径上出现了当前颜色。
- 将所有矩形的边界坐标离散化，通过二维差分前缀和完成矩形加操作。
- 再遍历每个格子，若被加的权值和非 0，则对应原矩阵中一个矩形内答案全部加 1。

100 分做法

- 问题转化为若干次矩形加后，m 次询问矩形和。

100 分做法

- 问题转化为若干次矩形加后，m 次询问矩形和。
- 对于一个矩形加操作，可以拆成 4 个形如将 (x, y) 右上角格子全部加上 p 的操作。

100 分做法

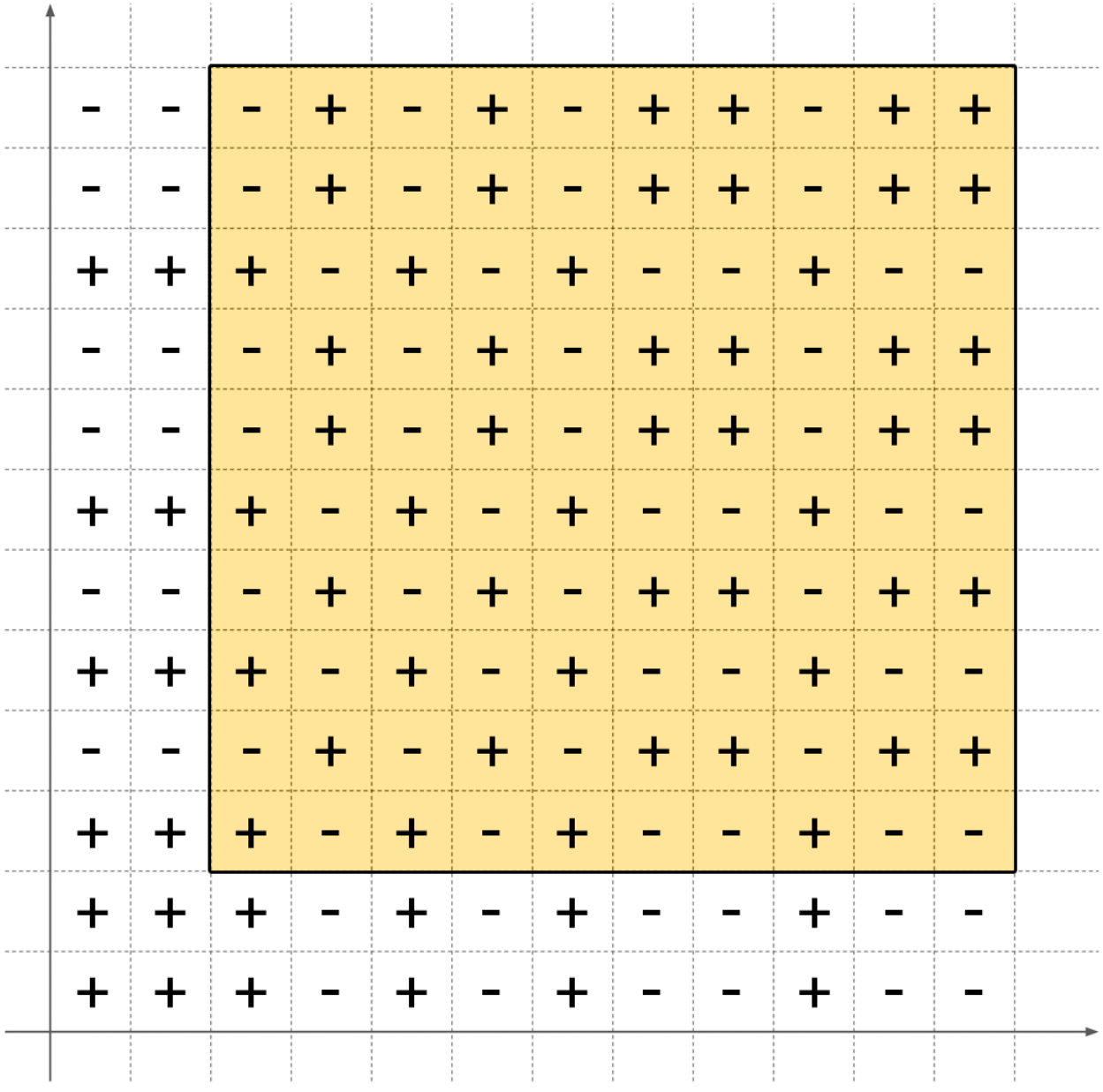
- 问题转化为若干次矩形加后，m 次询问矩形和。
- 对于一个矩形加操作，可以拆成 4 个形如将 (x, y) 右上角格子全部加上 p 的操作。
- 考虑一个修改 (x, y, p) 对一个询问 (A, B) 的贡献。

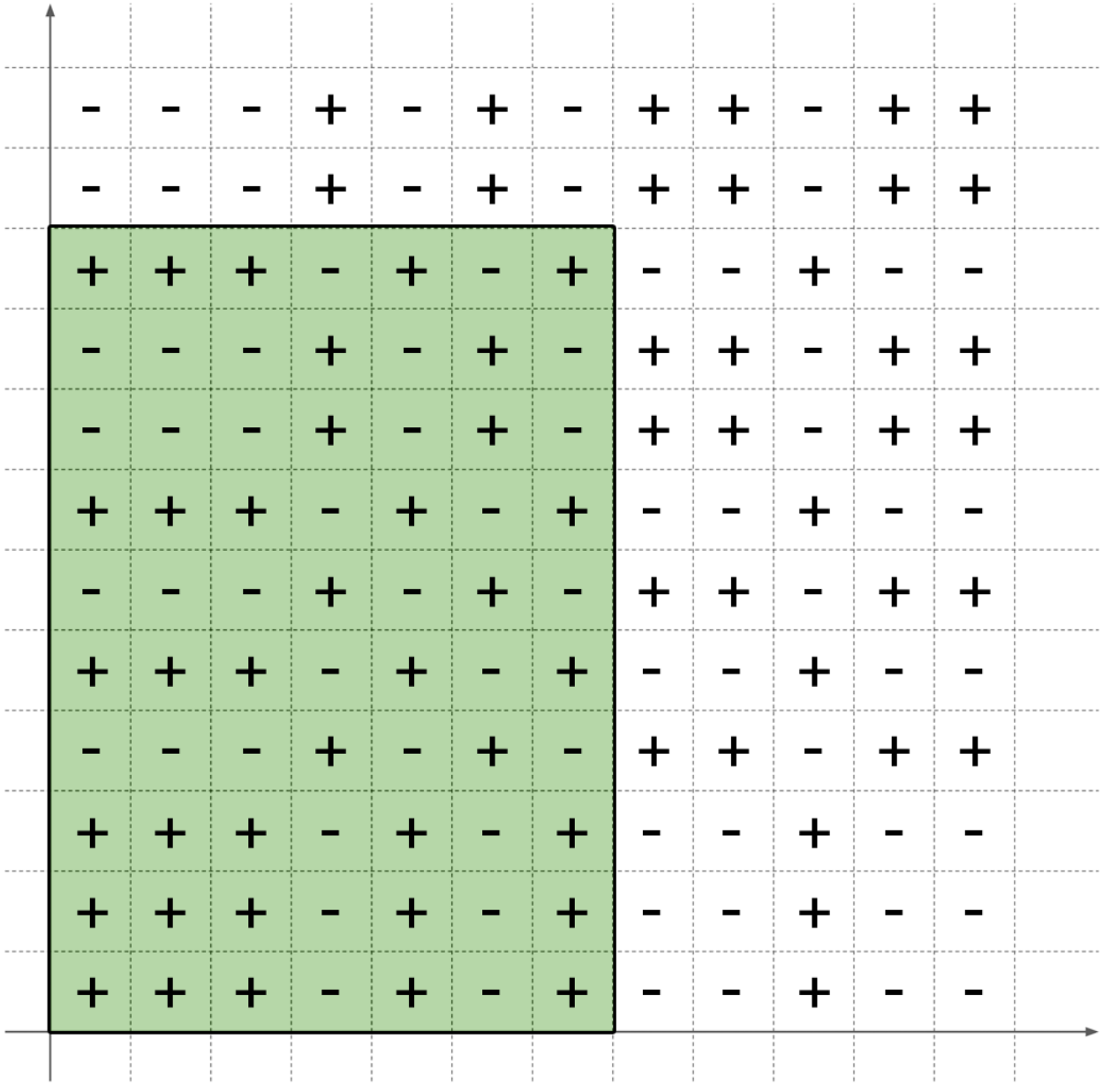
100 分做法

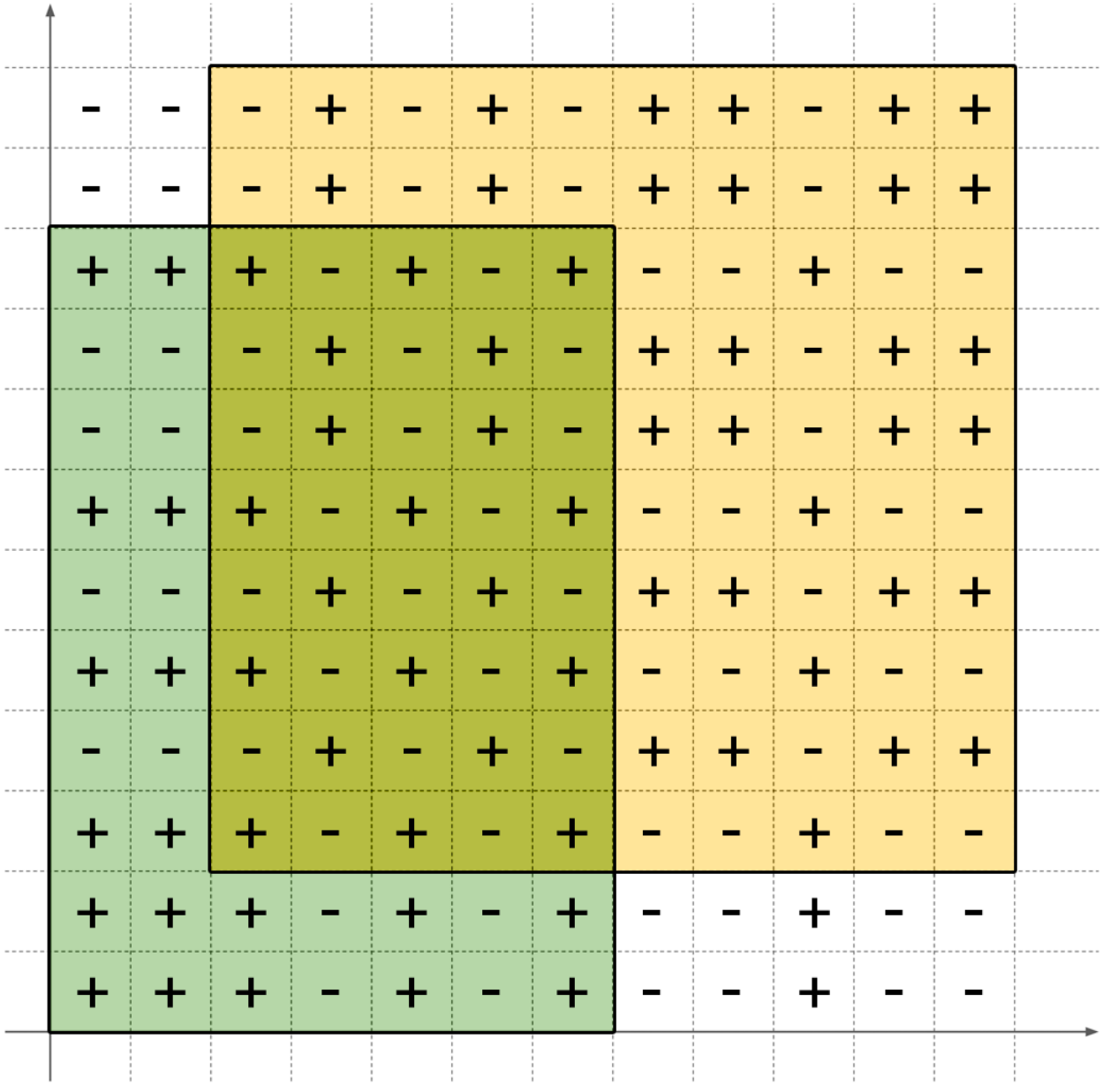
- 问题转化为若干次矩形加后，m 次询问矩形和。
- 对于一个矩形加操作，可以拆成 4 个形如将 (x, y) 右上角格子全部加上 p 的操作。
- 考虑一个修改 (x, y, p) 对一个询问 (A, B) 的贡献。
- 首先要满足 $x \leq A$ 且 $y \leq B$。

100 分做法

- 问题转化为若干次矩形加后，m 次询问矩形和。
- 对于一个矩形加操作，可以拆成 4 个形如将 (x, y) 右上角格子全部加上 p 的操作。
- 考虑一个修改 (x, y, p) 对一个询问 (A, B) 的贡献。
- 首先要满足 $x \leq A$ 且 $y \leq B$。
- 设 s_x 表示入栈出栈序前 x 项的答案系数之和，则贡献为 $(s_A - s_{x-1})(s_B - s_{y-1})p$。







100 分做法

- $\sum (s_A - s_{x-1})(s_B - s_{y-1})p$

100 分做法

- $\sum (s_A - s_{x-1})(s_B - s_{y-1})p$
- $= \sum (s_A s_B - s_B s_{x-1} - s_A s_{y-1} + s_{x-1} s_{y-1})p$

100 分做法

- $\sum (s_A - s_{x-1})(s_B - s_{y-1})p$
- $= \sum (s_A s_B - s_B s_{x-1} - s_A s_{y-1} + s_{x-1} s_{y-1})p$
- $= s_A s_B \sum p - s_B \sum s_{x-1} p - s_A \sum s_{y-1} p + \sum s_{x-1} s_{y-1} p$

100 分做法

- $\sum (s_A - s_{x-1})(s_B - s_{y-1})p$
- $= \sum (s_A s_B - s_B s_{x-1} - s_A s_{y-1} + s_{x-1} s_{y-1})p$
- $= s_A s_B \sum p - s_B \sum s_{x-1} p - s_A \sum s_{y-1} p + \sum s_{x-1} s_{y-1} p$
- 按 x 从左往右处理每个事件，在 y 方向用树状数组维护 4 个前缀和即可。

100 分做法

- $\sum (s_A - s_{x-1})(s_B - s_{y-1})p$
- $= \sum (s_A s_B - s_B s_{x-1} - s_A s_{y-1} + s_{x-1} s_{y-1})p$
- $= s_A s_B \sum p - s_B \sum s_{x-1} p - s_A \sum s_{y-1} p + \sum s_{x-1} s_{y-1} p$
- 按 x 从左往右处理每个事件，在 y 方向用树状数组维护 4 个前缀和即可。
- 设矩形数量为 e，则时间复杂度为 $O(e \log e)$。

100 分做法

- 那么矩形数量到底有多少呢？

100 分做法

- 那么矩形数量到底有多少呢？
- 对于某种颜色，设 deg_x 表示 x 的度数，则每个点会贡献 $2deg_x$ 个边界坐标。

100 分做法

- 那么矩形数量到底有多少呢？
- 对于某种颜色，设 deg_x 表示 x 的度数，则每个点会贡献 $2deg_x$ 个边界坐标。
- 矩形数量显然不超过边界坐标个数的平方，即 $(1+2\sum deg_x)^2$。

100 分做法

- 注意到每个点颜色在 $[1, n]$ 随机，$[1, p]$ 长链之后每个点 x 的父亲在 $[1, x-1]$ 随机。

100 分做法

- 注意到每个点颜色在 $[1, n]$ 随机，$[1, p]$ 长链之后每个点 x 的父亲在 $[1, x-1]$ 随机。
- 先考虑计算 $(\sum deg_i)^2$ 对颜色求和的期望。

100 分做法

- 注意到每个点颜色在 $[1, n]$ 随机，$[1, p]$ 长链之后每个点 x 的父亲在 $[1, x-1]$ 随机。
- 先考虑计算 $(\sum deg_i)^2$ 对颜色求和的期望。
- 这相当于枚举两个点 i 和 j，当且仅当这两个点颜色相同的时候对矩形数量有 $deg_i \times deg_j$ 的贡献。

100 分做法

- 注意到每个点颜色在 $[1, n]$ 随机，$[1, p]$ 长链之后每个点 x 的父亲在 $[1, x-1]$ 随机。
- 先考虑计算 $(\sum deg_i)^2$ 对颜色求和的期望。
- 这相当于枚举两个点 i 和 j，当且仅当这两个点颜色相同的时候对矩形数量有 $deg_i \times deg_j$ 的贡献。
- 如果 $i \neq j$，颜色相同的概率是 $\frac{1}{n}$，否则是 1。

100 分做法

- 注意到每个点颜色在 $[1, n]$ 随机，$[1, p]$ 长链之后每个点 x 的父亲在 $[1, x-1]$ 随机。
- 先考虑计算 $(\sum deg_i)^2$ 对颜色求和的期望。
- 这相当于枚举两个点 i 和 j，当且仅当这两个点颜色相同的时候对矩形数量有 $deg_i \times deg_j$ 的贡献。
- 如果 $i \neq j$，颜色相同的概率是 $\frac{1}{n}$，否则是 1。
- 因此期望是 $\frac{1}{n} \sum_{i \neq j} deg_i \times deg_j + \sum deg_i^2$。

100 分做法

- 注意到每个点颜色在 $[1, n]$ 随机，$[1, p]$ 长链之后每个点 x 的父亲在 $[1, x-1]$ 随机。
- 先考虑计算 $(\sum deg_i)^2$ 对颜色求和的期望。
- 这相当于枚举两个点 i 和 j，当且仅当这两个点颜色相同的时候对矩形数量有 $deg_i \times deg_j$ 的贡献。
- 如果 $i \neq j$，颜色相同的概率是 $\frac{1}{n}$，否则是 1。
- 因此期望是 $\frac{1}{n}\sum_{i\neq j} deg_i \times deg_j + \sum deg_i^2$。
- 对于前一部分，

  $\sum_{i\neq j} deg_i \times deg_j = (\sum deg_i)^2 - \sum deg_i^2 = (2n-2)^2 - \sum deg_i^2$。

100 分做法

- 对于后一部分，先计算 $\sum(deg_i - 1)^2$ 的期望，为了方便讨论，下设 $p = 2$。

100 分做法

- 对于后一部分，先计算 $\sum(deg_i - 1)^2$ 的期望，为了方便讨论，下设 $p = 2$。
- 考虑这样一个模型，有标号为 $2, 3, \ldots, n-1$ 的 $n-2$ 个球，标号为 $1, 2, \ldots, n-1$ 的 $n-1$ 个桶，标号为 i 的球会等概率放进标号为 $1, 2, \ldots, i$ 的桶内，那么所有桶内球数的平方和的期望即为所求。

100 分做法

- 对于后一部分，先计算 $\sum(deg_i - 1)^2$ 的期望，为了方便讨论，下设 $p = 2$。
- 考虑这样一个模型，有标号为 $2, 3, \ldots, n-1$ 的 $n-2$ 个球，标号为 $1, 2, \ldots, n-1$ 的 $n-1$ 个桶，标号为 i 的球会等概率放进标号为 $1, 2, \ldots, i$ 的桶内，那么所有桶内球数的平方和的期望即为所求。
- 这相当于枚举两个球 i 和 j，当且仅当两个球放入同一个桶时有 1 的贡献。

100 分做法

- 如果 $i \neq j$，放入同一个桶的概率是 $\frac{1}{\max(i,j)}$，否则是 1。

100 分做法

- 如果 $i \neq j$，放入同一个桶的概率是 $\frac{1}{\max(i,j)}$，否则是 1。
- 因此期望是 $(n-2) + \sum_{i \neq j} \frac{1}{\max(i,j)} = 3n - 4 - 2H_{n-1}$。

100 分做法

- 如果 $i \neq j$，放入同一个桶的概率是 $\frac{1}{\max(i,j)}$，否则是 1。
- 因此期望是 $(n-2) + \sum_{i \neq j} \frac{1}{\max(i,j)} = 3n - 4 - 2H_{n-1}$。
- 于是有

  $$\sum deg_i^2 = \sum (deg_i - 1)^2 + 2\sum deg_i - n = 6n - 8 - 2H_{n-1}。$$

100 分做法

- 如果 $i \neq j$，放入同一个桶的概率是 $\frac{1}{\max(i,j)}$，否则是 1。
- 因此期望是 $(n-2)+\sum_{i\neq j}\frac{1}{\max(i,j)}=3n-4-2H_{n-1}$。
- 于是有

  $\sum deg_i^2=\sum(deg_i-1)^2+2\sum deg_i-n=6n-8-2H_{n-1}$。
- 代入原式整理可得当 $p=2$ 时所求期望是 $O(n)$，蕴含一个 48 的常数，并且 p 越大这个常数越小，当 $p=n$ 时是 40。

100 分做法

- 如果 $i \neq j$，放入同一个桶的概率是 $\frac{1}{\max(i,j)}$，否则是 1。
- 因此期望是 $(n-2)+\sum_{i\neq j}\frac{1}{\max(i,j)}=3n-4-2H_{n-1}$。
- 于是有
  $\sum deg_i^2=\sum(deg_i-1)^2+2\sum deg_i-n=6n-8-2H_{n-1}$。
- 代入原式整理可得当 $p=2$ 时所求期望是 $O(n)$，蕴含一个 48 的常数，并且 p 越大这个常数越小，当 $p=n$ 时是 40。
- 故本算法总时间复杂度为 $O((n+m)\log n)$。

100 分做法

- 如果 $i \neq j$，放入同一个桶的概率是 $\frac{1}{\max(i,j)}$，否则是 1。
- 因此期望是 $(n-2)+\sum_{i\neq j}\frac{1}{\max(i,j)}=3n-4-2H_{n-1}$。
- 于是有
  $\sum deg_i^2=\sum(deg_i-1)^2+2\sum deg_i-n=6n-8-2H_{n-1}$。
- 代入原式整理可得当 $p=2$ 时所求期望是 $O(n)$，蕴含一个 48 的常数，并且 p 越大这个常数越小，当 $p=n$ 时是 40。
- 故本算法总时间复杂度为 $O((n+m)\log n)$。
- 实践证明当 $n=100000$ 时，非空有效矩形数量大约为 3×10^6 到 4×10^6 个。

一个小优化

- 当数据越趋向于一条链 (即 p 越接近 n) 时，非空有效矩形数量越多。

一个小优化

- 当数据越趋向于一条链 (即 p 越接近 n) 时，非空有效矩形数量越多。
- 这是因为链的情况下矩形之间都是嵌套关系，使得有效矩形数增加。

一个小优化

- 当数据越趋向于一条链 (即 p 越接近 n) 时，非空有效矩形数量越多。
- 这是因为链的情况下矩形之间都是嵌套关系，使得有效矩形数增加。
- 设 lim 为最后一个 DFS 到的点的 st，那么忽略 lim 之后的格子不会影响答案。

一个小优化

- 当数据越趋向于一条链 (即 p 越接近 n) 时，非空有效矩形数量越多。
- 这是因为链的情况下矩形之间都是嵌套关系，使得有效矩形数增加。
- 设 lim 为最后一个 DFS 到的点的 st，那么忽略 lim 之后的格子不会影响答案。
- 实践证明如此优化后，链数据中矩形数量降低为约 10^6 个。

Thank you!