匹配和网络流小讲

武弘勋

July 3, 2017

传统匹配和建模

POJ chessboard

- $n*n$ 的棋盘上摆了 $n \le 10^5$ 个车，让他们两两不攻击

POJ chessboard

- $n*n$ 的棋盘上摆了 $n \leq 10^5$ 个车，让他们两两不攻击
- 第 i 个车必须摆在一个矩形里

POJ chessboard

- $n * n$ 的棋盘上摆了 $n \leq 10^5$ 个车，让他们两两不攻击
- 第 i 个车必须摆在一个矩形里
- 求哪些车的摆放位置是确定的?

POJ chessboard

- 行列独立

POJ chessboard

- 行列独立
- 二分图匹配

POJ chessboard

- 行列独立
- 二分图匹配
- 每个点匹配是否唯一?

POJ chessboard

- 行列独立
- 二分图匹配
- 每个点匹配是否唯一?
- 往右只保留匹配边，往左只保留未匹配边，它是不是在一个环中

POJ chessboard

- 行列独立
- 二分图匹配
- 每个点匹配是否唯一?
- 往右只保留匹配边，往左只保留未匹配边，它是不是在一个环中
- 缩强连通分量即可

- 边 i，边权 $+1$ 有代价 a_i,-1 有代价 b_i

bzoj3118

- 边 i，边权 $+1$ 有代价 a_i,-1 有代价 b_i
- 修改边权使得指定的生成树变为最小生成树，求最小代价

bzoj3118

- 边 i，边权 $+1$ 有代价 a_i,-1 有代价 b_i
- 修改边权使得指定的生成树变为最小生成树，求最小代价
- $n \leq 300$

bzoj3118

- 边 i，边权 $+1$ 有代价 a_i,-1 有代价 b_i
- 修改边权使得指定的生成树变为最小生成树，求最小代价
- $n \leq 300$
- 显然只会给生成树树边减，给非树边加，可以列出方程 $x_i + y_j \geq \delta$

bzoj3118

- 边 i，边权 $+1$ 有代价 a_i,-1 有代价 b_i
- 修改边权使得指定的生成树变为最小生成树，求最小代价
- $n \leq 300$
- 显然只会给生成树树边减，给非树边加，可以列出方程 $x_i + y_j \geq \delta$
- 最小顶标和，对偶一下，

CCPC E Problem Buyer

- 有 n 个题目的题库，每个题的难度为 c_i。从中出 m 个题目，第 i 题难度需要在 a_i 到 b_i 之间

CCPC E Problem Buyer

- 有 n 个题目的题库，每个题的难度为 c_i。从中出 m 个题目，第 i 题难度需要在 a_i 到 b_i 之间
- 求最小的 k 使得对于任意一个 k 个题目的子集都能凑出一套符合要求的题。

- 有 n 个题目的题库，每个题的难度为 c_i。从中出 m 个题目，第 i 题难度需要在 a_i 到 b_i 之间
- 求最小的 k 使得对于任意一个 k 个题目的子集都能凑出一套符合要求的题。
- $n, m \le 10^5$

CCPC E Problem Buyer

- 合法条件？每个子区间都满足霍尔定理

CCPC E Problem Buyer

- 合法条件？每个子区间都满足霍尔定理
- 扫描线 + 线段树维护

Delight for a cat

- 有一只喵，共 n 天，它每 k 天吃吃吃的天数要在 $[l, r]$ 之间，每天睡觉有个收益 s_i，吃猫粮有个收益 e_i，求最大收益。

Delight for a cat

- 有一只喵，共 n 天，它每 k 天吃吃吃的天数要在 $[l, r]$ 之间，每天睡觉有个收益 s_i，吃猫粮有个收益 e_i，求最大收益。
- $k, n \le 1000$

Delight for a cat

- 有一只喵，共 n 天，它每 k 天吃吃吃的天数要在 $[l, r]$ 之间，每天睡觉有个收益 s_i，吃猫粮有个收益 e_i，求最大收益。
- $k, n \leq 1000$
- 输出方案

- 解法 1：每个限制的天数都是连续的，可以对偶差分

线性规划转网络流

- 解法 1：每个限制的天数都是连续的，可以对偶差分
- 解法 2: 每天在限制上也是连续的，可以直接差分不对偶

负权环建图

- 考虑每天对于限制是一个连续区间

负权环建图

- 考虑每天对于限制是一个连续区间
- 我们从 $i+k+1$ 连到 i，代价为 e_i-s_i，然后 i 往 $i+1$ 连上下界为 $[l,r]$ 的边

负权环建图

- 考虑每天对于限制是一个连续区间
- 我们从 $i+k+1$ 连到 i，代价为 $e_i - s_i$，然后 i 往 $i+1$ 连上下界为 $[l, r]$ 的边
- 这是个非常容易拓展出题的 trick，解决了直接从 S 和 T 连不匹配的问题

负权环建图

- 考虑每天对于限制是一个连续区间
- 我们从 $i+k+1$ 连到 i，代价为 $e_i - s_i$，然后 i 往 $i+1$ 连上下界为 $[l, r]$ 的边
- 这是个非常容易拓展出题的 trick，解决了直接从 S 和 T 连不匹配的问题
- 比如 DAG 上选若干路径，最大化他们的并的权值和减去他们的权值和

负权建图技巧

- 预先流满

负权建图技巧

- 预先流满
- 缺陷?

- 每个格子里填一个拐弯的管道

ChinaFianl J Mr.Panda and TubeMaster

- 每个格子里填一个拐弯的管道
- 管道里

Primal-Dual cost flow

- 用 dijkstra 跑费用流

Primal-Dual cost flow

- 用 dijkstra 跑费用流
- 当它遇到负权建图技巧

Primal-Dual cost flow

- 用 dijkstra 跑费用流
- 当它遇到负权建图技巧
- Boom!

线代方法

线代方法

- 偶环覆盖

- 偶环覆盖
- 反对称矩阵的列极大线性无关方程组和行极大线性无关方程组

CERC16B

- 给你一个二分图，每侧 n 个点

CERC16B

- 给你一个二分图，每侧 n 个点
- 多少个点的子集存在完美匹配

CERC16B

- 给你一个二分图，每侧 n 个点
- 多少个点的子集存在完美匹配
- $n \leq 20$

CERC16B

- 回忆一下线代方法的证明，反对称矩阵的性质

CERC16B

- 回忆一下线代方法的证明，反对称矩阵的性质
- 不难猜想存在完美匹配的冲要条件是左边和右边的子集分别满足霍尔定理

CERC16B

- 回忆一下线代方法的证明，反对称矩阵的性质
- 不难猜想存在完美匹配的冲要条件是左边和右边的子集分别满足霍尔定理
- check 霍尔定理？ mask dp

线性规划的对偶原理

- $max\{c^Tx|Ax \leq b\} = min\{b^Ty|A^Ty \geq c\}$

对偶原理

- $max\{c^T x | Ax \leq b\} = min\{b^T y | A^T y \geq c\}$
- why？我们来直观的理解一下

对偶原理

- $max\{c^Tx|Ax \le b\} = min\{b^Ty|A^Ty \ge c\}$
- why？我们来直观的理解一下
- 假设你是工厂主，每个产品的收益为 c, 每个产品需要的材料是 A，你有的原材料个数是 b

对偶原理

- $max\{c^Tx|Ax \leq b\} = min\{b^Ty|A^Ty \geq c\}$
- why？我们来直观的理解一下
- 假设你是工厂主，每个产品的收益为 c, 每个产品需要的材料是 A，你有的原材料个数是 b
- 决策生产个数 x 最大化收益（忽略加工费之类的）

- $max\{c^T x | Ax \le b\} = min\{b^T y | A^T y \ge c\}$
- why？我们来直观的理解一下
- 假设你是工厂主，每个产品的收益为 c, 每个产品需要的材料是 A，你有的原材料个数是 b
- 决策生产个数 x 最大化收益（忽略加工费之类的）
- 反过来你要卖给工厂主这些材料，决策原材料单价 y，你至少卖给他多贵才不会亏

- $max\{c^Tx|Ax \leq b\} = min\{b^Ty|A^Ty \geq c\}$
- why？我们来直观的理解一下
- 假设你是工厂主，每个产品的收益为 c, 每个产品需要的材料是 A，你有的原材料个数是 b
- 决策生产个数 x 最大化收益（忽略加工费之类的）
- 反过来你要卖给工厂主这些材料，决策原材料单价 y，你至少卖给他多贵才不会亏
- y 称为『影子价格』是对价格的估计，显然因为忽略了工厂主的社会必要劳动时间，所以这两个问题的答案应该是相等的（成本 = 收益）

cf352D

- 一个有 n 个点的树，树上的边已经全部损坏了。现在有 m 个工人，每个工人可以修复从 u_i 到 v_i (保证 v_i 是 u_i 到 $root$ 上的一个点) 上的所有边，花费为 c_i。问把所有边修好的最小花费为多少?

cf352D

- 一个有 n 个点的树，树上的边已经全部损坏了。现在有 m 个工人，每个工人可以修复从 u_i 到 v_i (保证 v_i 是 u_i 到 $root$ 上的一个点) 上的所有边，花费为 c_i。问把所有边修好的最小花费为多少?
- $n \leq 3 * 10^5$

cf352D

- 一个有 n 个点的树，树上的边已经全部损坏了。现在有 m 个工人，每个工人可以修复从 u_i 到 v_i (保证 v_i 是 u_i 到 $root$ 上的一个点) 上的所有边，花费为 c_i。问把所有边修好的最小花费为多少?
- $n \leq 3 * 10^5$
- Feel the magic of dual!

cf352D

- 传统做法大概 dp 一下

cf352D

- 传统做法大概 dp 一下
- f_i 表示完成 fa_i 到 i 的边和子树的最小代价，转移在回溯的时候维护每个末端点的代价

cf352D

- 传统做法大概 dp 一下
- f_i 表示完成 fa_i 到 i 的边和子树的最小代价，转移在回溯的时候维护每个末端点的代价
- 相信大家在 OI 中已经对这样的做法审美疲劳了

cf352D

- 传统做法大概 dp 一下
- f_i 表示完成 fa_i 到 i 的边和子树的最小代价，转移在回溯的时候维护每个末端点的代价
- 相信大家在 OI 中已经对这样的做法审美疲劳了
- 对偶做法: 考虑对偶问题是『选最多的边，使得每个工人可以修复的范围内少于 c_i 个』

cf352D

- 传统做法大概 dp 一下
- f_i 表示完成 fa_i 到 i 的边和子树的最小代价，转移在回溯的时候维护每个末端点的代价
- 相信大家在 OI 中已经对这样的做法审美疲劳了
- 对偶做法: 考虑对偶问题是『选最多的边，使得每个工人可以修复的范围内少于 c_i 个』
- magical greedy!

cf352D

- 传统做法大概 dp 一下
- f_i 表示完成 fa_i 到 i 的边和子树的最小代价，转移在回溯的时候维护每个末端点的代价
- 相信大家在 OI 中已经对这样的做法审美疲劳了
- 对偶做法: 考虑对偶问题是『选最多的边，使得每个工人可以修复的范围内少于 c_i 个』
- magical greedy!
- 可以选的时候一定是选了更优

cf352D

- 传统做法大概 dp 一下
- f_i 表示完成 fa_i 到 i 的边和子树的最小代价，转移在回溯的时候维护每个末端点的代价
- 相信大家在 OI 中已经对这样的做法审美疲劳了
- 对偶做法: 考虑对偶问题是『选最多的边，使得每个工人可以修复的范围内少于 c_i 个』
- magical greedy!
- 可以选的时候一定是选了更优
- 判断可不可以选可以 set 维护一下剩余多少个，启发式合并

- 有一个技能 DAG，你要点满所有技能点，每个技能点需要时间 t

SRM676 L3

- 有一个技能 DAG，你要点满所有技能点，每个技能点需要时间 t
- 你有 C 元，每次可以氪金 c_i 元来给一个技能 $-1s$，求你点完所有技能点的最小时间（可以同时点多个技能点，但是这多个之间不能有依赖关系）

SRM676 L3

- 有一个技能 DAG，你要点满所有技能点，每个技能点需要时间 t
- 你有 C 元，每次可以氪金 c_i 元来给一个技能 $-1s$，求你点完所有技能点的最小时间（可以同时点多个技能点，但是这多个之间不能有依赖关系）
- $n \leq 50$

- 二分答案 mid

SRM676 L3

- 二分答案 mid
- 对偶之后变成了选最若干个路径，每个边至多选 c_i 次，最大化每条路径 $-mid$ 的权值和

SRM676 L3

- 二分答案 mid
- 对偶之后变成了选最若干个路径，每个边至多选 c_i 次，最大化每条路径 $-mid$ 的权值和
- DAG 上的费用流问题

- 二分答案 mid
- 对偶之后变成了选最若干个路径，每个边至多选 c_i 次，最大化每条路径 $-mid$ 的权值和
- DAG 上的费用流问题
- 当然可以拆点最小割，但是没有对偶优雅简洁。

- 一个 DAG，满足 $u->v$ 的边中 $u<v$，并且满足边只有包含没有相交（顶点处相交不算）

TCO SemiFinal Colorful Tree

- 一个 DAG，满足 $u->v$ 的边中 $u<v$，并且满足边只有包含没有相交（顶点处相交不算）
- 点有颜色，求一条 1 到 n 的路径

TCO SemiFinal Colorful Tree

- 一个 DAG，满足 $u->v$ 的边中 $u<v$，并且满足边只有包含没有相交（顶点处相交不算）
- 点有颜色，求一条 1 到 n 的路径
- 每个颜色的点要么全部经过要么全部不经过，边有边权，求最短的这条路径

TCO SemiFinal Colorful Tree

- 一个 DAG，满足 $u->v$ 的边中 $u<v$，并且满足边只有包含没有相交（顶点处相交不算）
- 点有颜色，求一条 1 到 n 的路径
- 每个颜色的点要么全部经过要么全部不经过，边有边权，求最短的这条路径
- $n \leq 50$

TCO SemiFinal Colorful Tree

- 边只有包含没有相交其实构成了一棵树

TCO SemiFinal Colorful Tree

- 边只有包含没有相交其实构成了一棵树
- 这个图的最短路也就是对偶图（树）的最小割

TCO SemiFinal Colorful Tree

- 边只有包含没有相交其实构成了一棵树
- 这个图的最短路也就是对偶图（树）的最小割
- 可以连 inf 边来限制两个点同时在 S/T 集

TCO SemiFinal Colorful Tree

- 边只有包含没有相交其实构成了一棵树
- 这个图的最短路也就是对偶图（树）的最小割
- 可以连 inf 边来限制两个点同时在 S/T 集
- 连 inf 边会影响形态，每个点要往父亲连 inf 边确保每个路径只割一刀

对偶原理如何出解

- 互补松弛定理

对偶原理如何出解

- 互补松弛定理
- 让我们回到工厂，现在如果一个东西是亏的（右边不等式没有取到等号)，那么你还会去买吗?

对偶原理如何出解

- 互补松弛定理
- 让我们回到工厂，现在如果一个东西是亏的（右边不等式没有取到等号)，那么你还会去买吗?
- 对偶问题里不等式取不到等号，x_i 为 0。逆否命题，x_i 大于 0，一定取等号。

对偶原理如何出解

- 互补松弛定理
- 让我们回到工厂，现在如果一个东西是亏的（右边不等式没有取到等号），那么你还会去买吗？
- 对偶问题里不等式取不到等号，x_i 为 0。逆否命题，x_i 大于 0，一定取等号。
- 我们现在是知道了 y 求 x，对偶一下

- 互补松弛定理
- 让我们回到工厂，现在如果一个东西是亏的（右边不等式没有取到等号），那么你还会去买吗？
- 对偶问题里不等式取不到等号，x_i 为 0。逆否命题，x_i 大于 0，一定取等号。
- 我们现在是知道了 y 求 x，对偶一下
- 影子价格不为 0 当且仅当对应的对偶问题不等式取等号

对偶原理如何出解

- 互补松弛定理
- 让我们回到工厂，现在如果一个东西是亏的（右边不等式没有取到等号），那么你还会去买吗？
- 对偶问题里不等式取不到等号，x_i 为 0。逆否命题，x_i 大于 0，一定取等号。
- 我们现在是知道了 y 求 x，对偶一下
- 影子价格不为 0 当且仅当对应的对偶问题不等式取等号
- $x^T(c - A^Ty) = 0, y^T(b - Ax) = 0$，高斯消元

flow simulation

USACO Landscape

- N 个花圃，第 i 个花圃有 A_i 个泥土

USACO Landscape

- N 个花圃，第 i 个花圃有 A_i 个泥土
- 使每个花圃最后有 B_i 个泥土。$A_i, B_i \in [0, 10]$

USACO Landscape

- N 个花圃，第 i 个花圃有 A_i 个泥土
- 使每个花圃最后有 B_i 个泥土。$A_i, B_i \in [0, 10]$
- 可以购买一个单位的泥土，并将它放在他选择的花圃中，用 X 单位的钱。

USACO Landscape

- N 个花圃，第 i 个花圃有 A_i 个泥土
- 使每个花圃最后有 B_i 个泥土。$A_i, B_i \in [0, 10]$
- 可以购买一个单位的泥土，并将它放在他选择的花圃中，用 X 单位的钱。
- 可以从花圃上清除一块泥土，并用 Y 单位的钱运出去。

- N 个花圃，第 i 个花圃有 A_i 个泥土
- 使每个花圃最后有 B_i 个泥土。$A_i, B_i \in [0, 10]$
- 可以购买一个单位的泥土，并将它放在他选择的花圃中，用 X 单位的钱。
- 可以从花圃上清除一块泥土，并用 Y 单位的钱运出去。
- 还可以用 $Z * |i - j|$ 的花费将一单位的泥土从花圃 i 运输到花圃 j，求最小成本。

USACO Landscape

- 匹配模型，设 $d_i = B_i - A_i$，多的一定和少的匹配

USACO Landscape

- 匹配模型，设 $d_i = B_i - A_i$，多的一定和少的匹配
- 我们放入一个 X 之后，后面如果有个人和它匹配那么他的贡献是 $-z * i - X$

USACO Landscape

- 匹配模型，设 $d_i = B_i - A_i$，多的一定和少的匹配
- 我们放入一个 X 之后，后面如果有个人和它匹配那么他的贡献是 $-z*i-X$
- 同样的的如果它和之前的匹配，贡献是 c，然后后面要和他匹配，那么贡献就是 $-z*i-c$

- 匹配模型，设 $d_i = B_i - A_i$，多的一定和少的匹配
- 我们放入一个 X 之后，后面如果有个人和它匹配那么他的贡献是 $-z * i - X$
- 同样的的如果它和之前的匹配，贡献是 c，然后后面要和他匹配，那么贡献就是 $-z * i - c$
- 我们等于是一个个加入每个节点，维护之前的每个增广路的贡献

- 匹配模型，设 $d_i = B_i - A_i$，多的一定和少的匹配
- 我们放入一个 X 之后，后面如果有个人和它匹配那么他的贡献是 $-z * i - X$
- 同样的的如果它和之前的匹配，贡献是 c，然后后面要和他匹配，那么贡献就是 $-z * i - c$
- 我们等于是一个个加入每个节点，维护之前的每个增广路的贡献
- 用两个堆维护两种增广路即可

最大 k 不相交子段和

- 相信大家都会线段树

最大 **k** 不相交子段和

- 相信大家都会线段树
- 怎么线性呢?

最大 k 不相交子段和

- 相信大家都会线段树
- 怎么线性呢?
- 考虑前缀和之后的序列，我们每次就是要加入一对贡献最大的括号

最大 k 不相交子段和

- 相信大家都会线段树
- 怎么线性呢?
- 考虑前缀和之后的序列，我们每次就是要加入一对贡献最大的括号
- 类似于区间最大值的笛卡尔树，用单调栈解决

bzoj3593 Control

- 有 n 个病人, 每个病人的病情可以用 b_i 表示。每过一个时刻，病人的病情会增加 a_i 救治一个病情严重程度为 x 的病人，小 x 需要消耗 x 点体力

bzoj3593 Control

- 有 n 个病人, 每个病人的病情可以用 b_i 表示。每过一个时刻，病人的病情会增加 a_i 救治一个病情严重程度为 x 的病人，小 x 需要消耗 x 点体力
- 有些时候，小 x 会带来一瓶超级药水，在他到达这个城市的时候可以给某个病人服下（不消耗时间，也不一定要最先救治这个病人)，那么这个病人的 b_i 值会变成 0。

bzoj3593 Control

- 有 n 个病人, 每个病人的病情可以用 b_i 表示。每过一个时刻，病人的病情会增加 a_i 救治一个病情严重程度为 x 的病人，小 x 需要消耗 x 点体力
- 有些时候，小 x 会带来一瓶超级药水，在他到达这个城市的时候可以给某个病人服下（不消耗时间，也不一定要最先救治这个病人)，那么这个病人的 b_i 值会变成 0。
- 只要成功救治了 p 个病人，就可以认为病情稳定，小 x 就可以休息了。问小 x 最少需要消耗多少体力。

bzoj3593 Control

- 有 n 个病人, 每个病人的病情可以用 b_i 表示。每过一个时刻，病人的病情会增加 a_i 救治一个病情严重程度为 x 的病人，小 x 需要消耗 x 点体力
- 有些时候，小 x 会带来一瓶超级药水，在他到达这个城市的时候可以给某个病人服下（不消耗时间，也不一定要最先救治这个病人)，那么这个病人的 b_i 值会变成 0。
- 只要成功救治了 p 个病人，就可以认为病情稳定，小 x 就可以休息了。问小 x 最少需要消耗多少体力。
- $n \leq 10^5$

bzoj3593 Control

- 首先假设我们已经会做这个题了，怎么处理膜法药水?

bzoj3593 Control

- 首先假设我们已经会做这个题了，怎么处理膜法药水?
- 如果在最优解集合中，那么就给集合中 b_i 最大的点服用即可

bzoj3593 Control

- 首先假设我们已经会做这个题了，怎么处理膜法药水?
- 如果在最优解集合中，那么就给集合中 b_i 最大的点服用即可
- 如果不在最优解集合中，那么贪心的一定会给 a_i 最小的点服用

bzoj3593 Control

- 首先假设我们已经会做这个题了，怎么处理膜法药水?
- 如果在最优解集合中，那么就给集合中 b_i 最大的点服用即可
- 如果不在最优解集合中，那么贪心的一定会给 a_i 最小的点服用
- 再跑一边最优解即可

bzoj3593 Control

- 这个问题本质上是一个二分图匹配，边权是 $e_{i,j} = a_i + b_i * j$

bzoj3593 Control

- 这个问题本质上是一个二分图匹配，边权是 $e_{i,j} = a_i + b_i * j$
- 二分图匹配退流不会退掉 S 和 T 有关的点，具有最优子结构！

bzoj3593 Control

- 这个问题本质上是一个二分图匹配，边权是 $e_{i,j} = a_i + b_i * j$
- 二分图匹配退流不会退掉 S 和 T 有关的点，具有最优子结构！
- 我们肯定会按照 a_i 从大到小救治，按这个顺序排序

bzoj3593 Control

- 这个问题本质上是一个二分图匹配，边权是 $e_{i,j} = a_i + b_i * j$
- 二分图匹配退流不会退掉 S 和 T 有关的点，具有最优子结构！
- 我们肯定会按照 a_i 从大到小救治，按这个顺序排序
- 那么我们每次选一个点的贡献是右边选了的点的 a_i 之和加上左边选了的点的个数乘上它自己的 a_i

bzoj3593 Control

- 这个问题本质上是一个二分图匹配，边权是 $e_{i,j} = a_i + b_i * j$
- 二分图匹配退流不会退掉 S 和 T 有关的点，具有最优子结构!
- 我们肯定会按照 a_i 从大到小救治，按这个顺序排序
- 那么我们每次选一个点的贡献是右边选了的点的 a_i 之和加上左边选了的点的个数乘上它自己的 a_i
- 分块凸包维护最小贡献,$O(n\sqrt{n})$

谢谢大家

bzoj3593 Control

- 你以为结束了？ naive

- 你以为结束了？naive
- naive approach：dp，$f_{i,j}$ 表示前 i 个选了 j 个

bzoj3593 Control

- 你以为结束了？ naive
- naive approach：dp，$f_{i,j}$ 表示前 i 个选了 j 个
- 因为最优子结构，所以每次更新的 j 是一个后缀！

bzoj3593 Control

- 你以为结束了？ naive
- naive approach：dp，$f_{i,j}$ 表示前 i 个选了 j 个
- 因为最优子结构，所以每次更新的 j 是一个后缀！
- 用 treap 可以比较方便的维护，每次二分一下更新的位置

bzoj3593 Control

- 你以为结束了？ naive
- naive approach：dp，$f_{i,j}$ 表示前 i 个选了 j 个
- 因为最优子结构，所以每次更新的 j 是一个后缀！
- 用 treap 可以比较方便的维护，每次二分一下更新的位置
- 复杂度 $O(nlogn)$