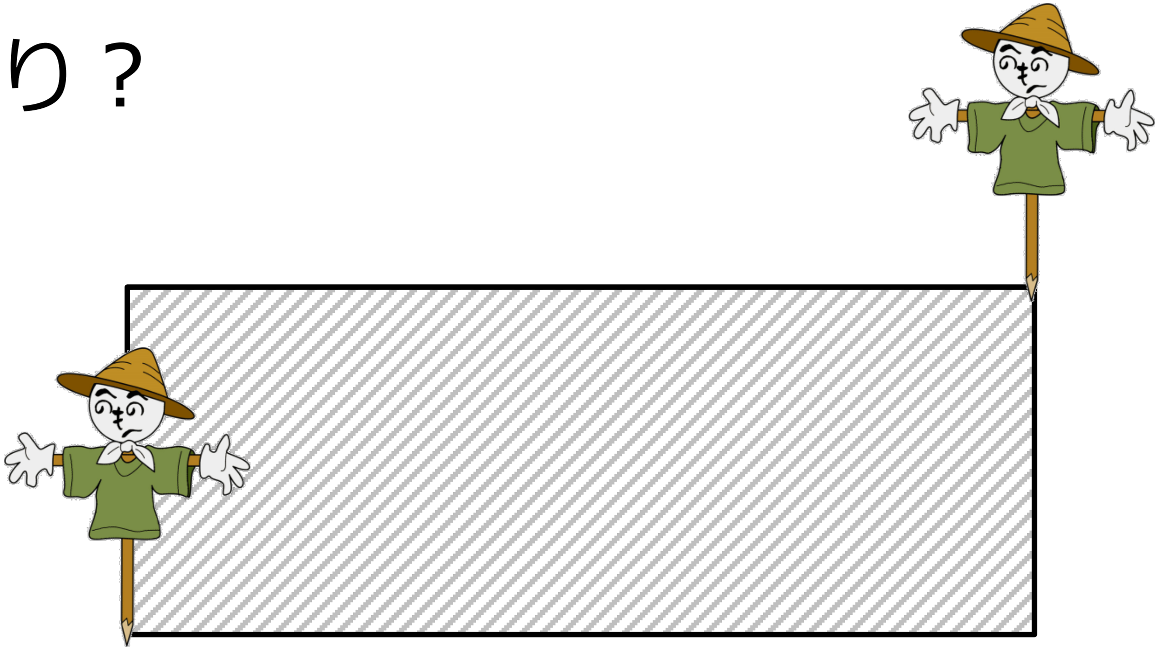
かかし (Scarecrows)

JOI 春合宿 2014 Day 3

解説： 保坂 和宏

問題概要

- N 個の点 ($N \leqq 200\,000$)
- 左下と右上の点を選んで長方形を作り，内部に他の点が入らないようにする
- 何通り？



小課題 1 (5 点)

- $N \leqq 400$
- 左下と右上の点を選んで長方形を作り，内部に他の点が入らないようにする
  - 左下を決める： $O(N)$
  - 右上を決める： $O(N)$
  - 他の点が内部に入っているか： $O(N)$

- $O(N^3)$ 時間

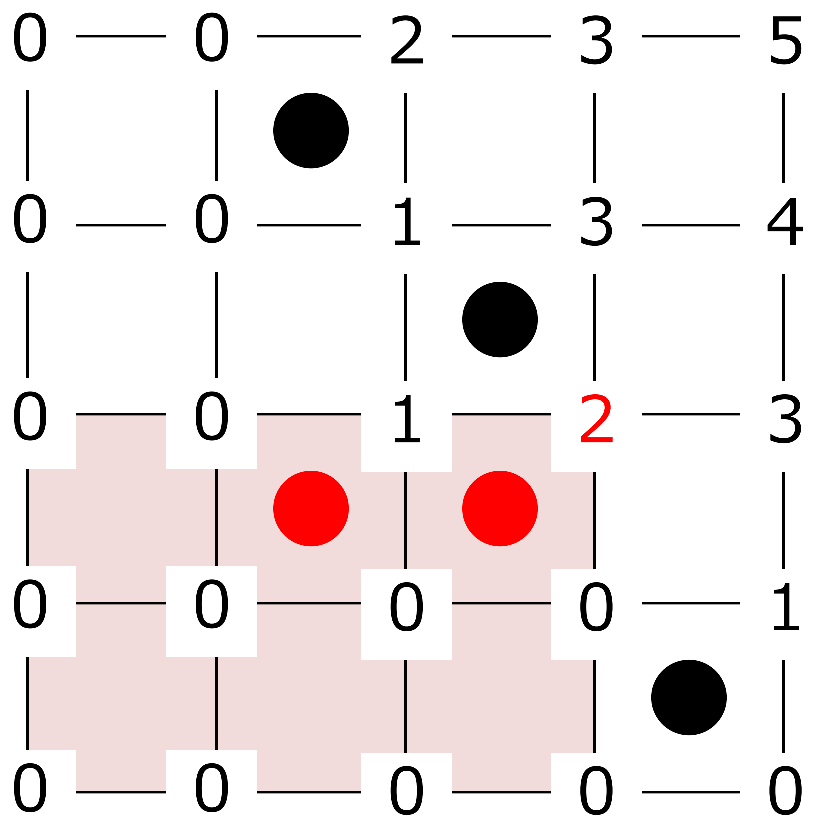
小課題 2 (10 点)

- $N \leqq 5\,000$
- 左下と右上の点を選んで長方形を作り，内部に他の点が入らないようにする
  - 左下を決める： $O(N)$
  - 右上を決める： $O(N)$
  - 他の点が内部に入っているか： $O(N)$
    - これを高速にできないか？

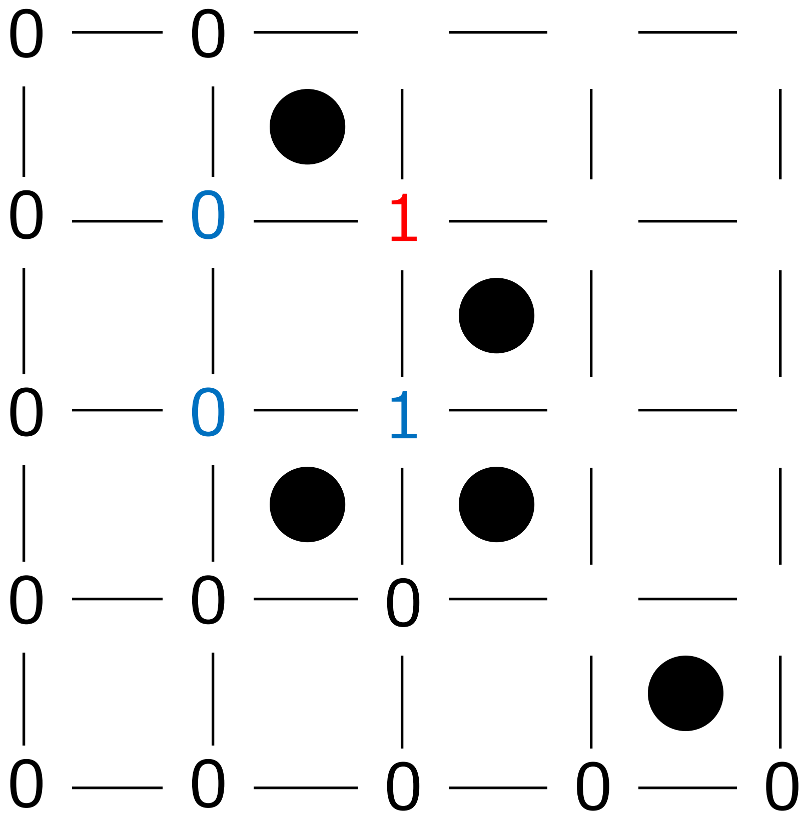
小課題 2 (10 点)

- 長方形を指定したとき，内部の点の個数を高速に数えたい

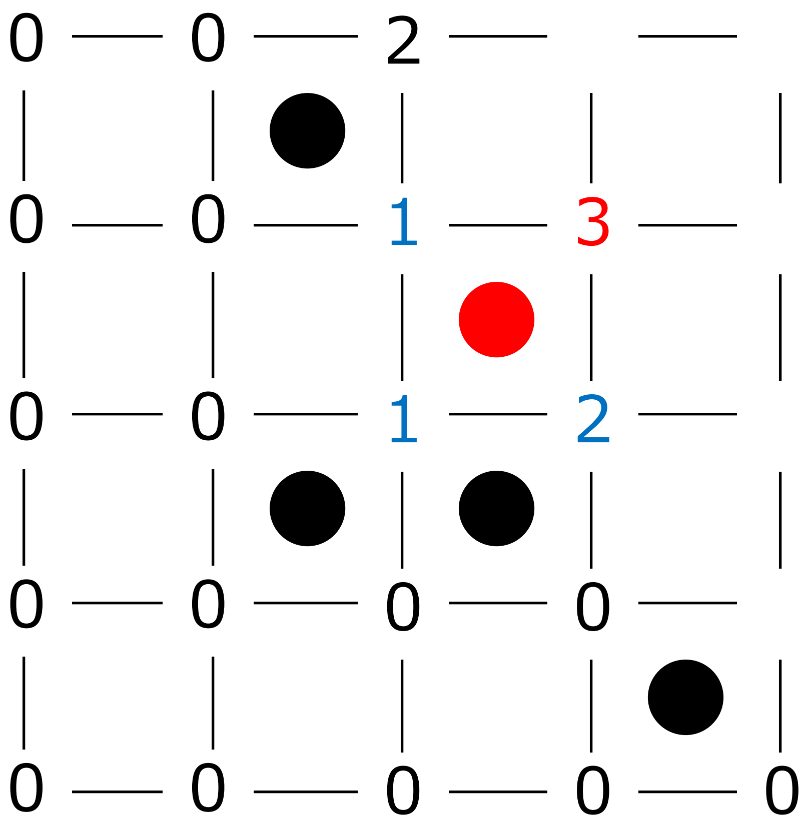
累積和



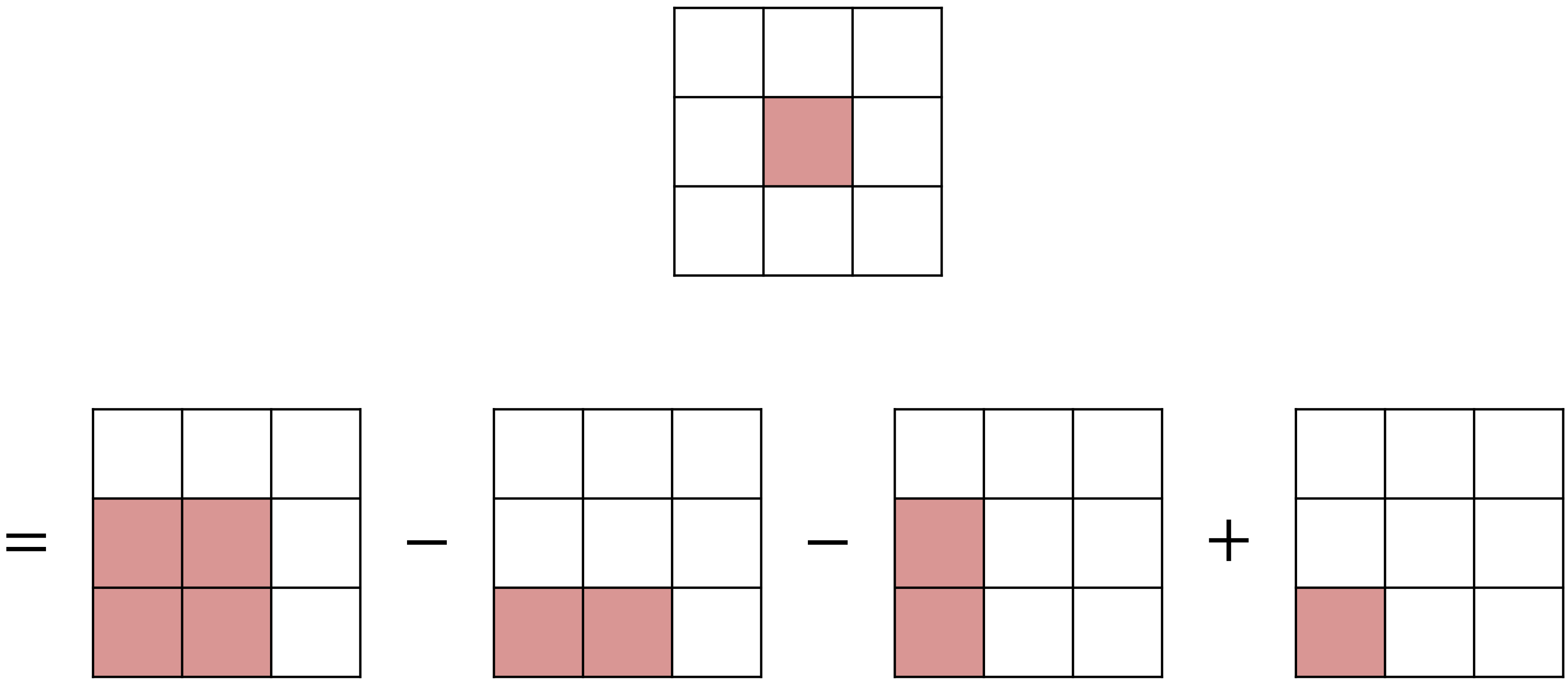
累積和



累積和



累積和



小課題 2 (10 点)

- 座標圧縮 ($O(N \log N)$)
- 累積和を計算 ($O(N^2)$)
- 左下と右上を決めて ($O(N^2)$) 内部の点の個数を数える ($O(1)$)

- $O(N^2)$ 時間

小課題 2 (10 点)

- 別解
  - 後述の満点解法をナイーブにするなどで $O(N^2)$ 時間解法がいくつか考えられる

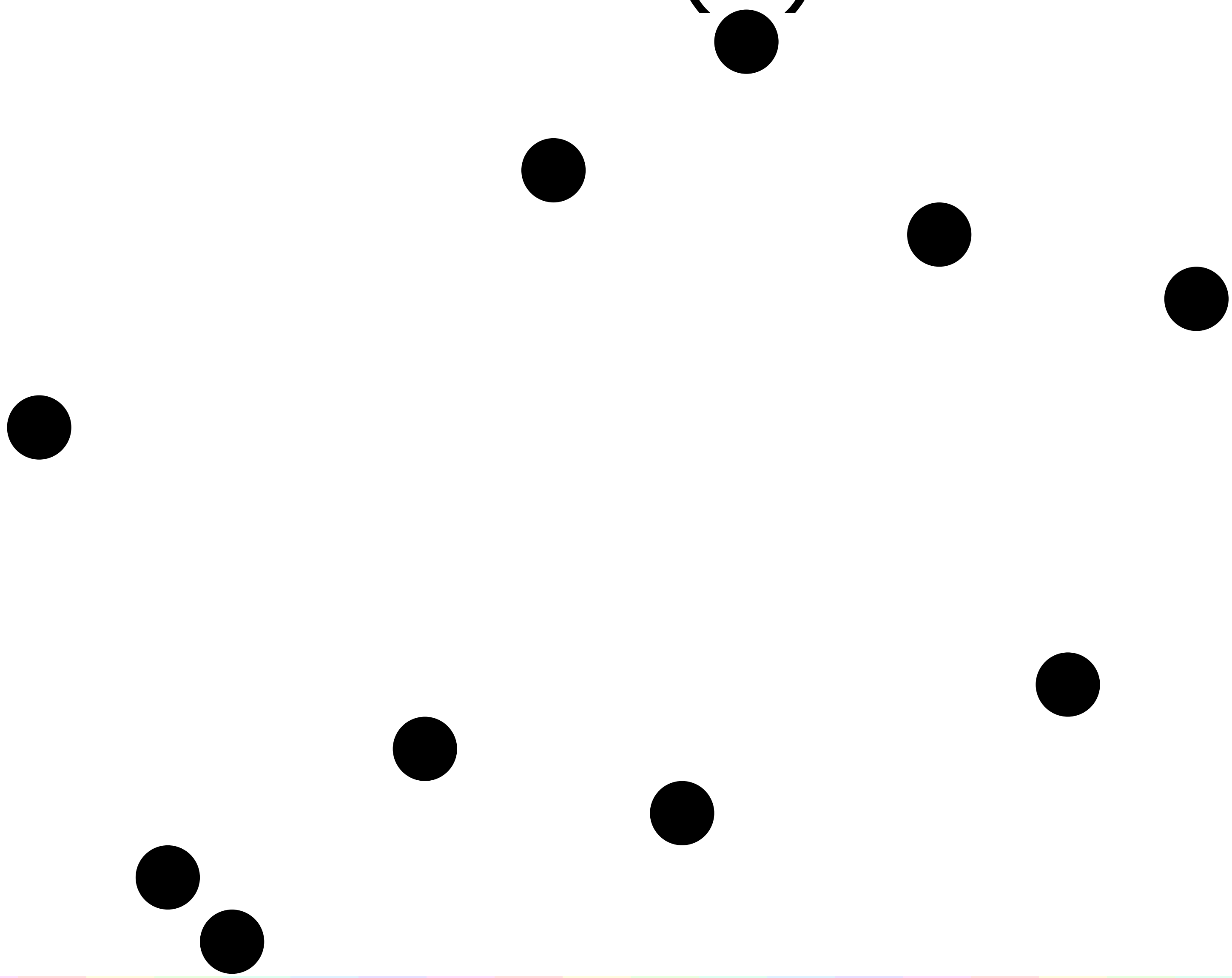
小課題 3 (85 点)

- $N \leqq 200\,000$
- 左下と右上の組をすべて試せない
  - 左下 (or 右上) の点のみ決めて，長方形が条件を満たすような右上 (or 左下) の点の個数を数える，ができるとよい

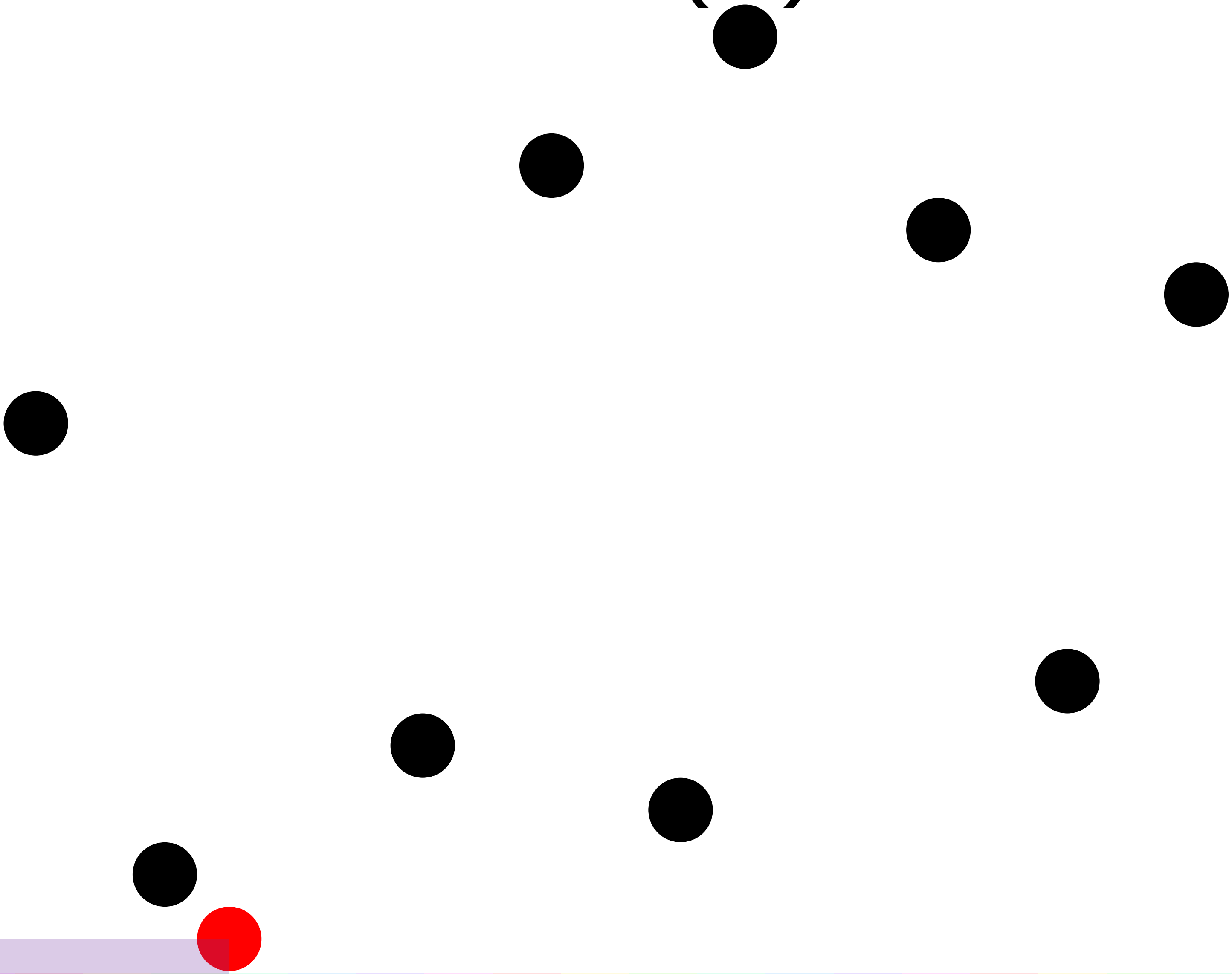
小課題 3 (85 点)

- すごいデータ構造……？
  - はい！ → 解法 (1) へ
  - いいえ → 解法 (2)(3) へ

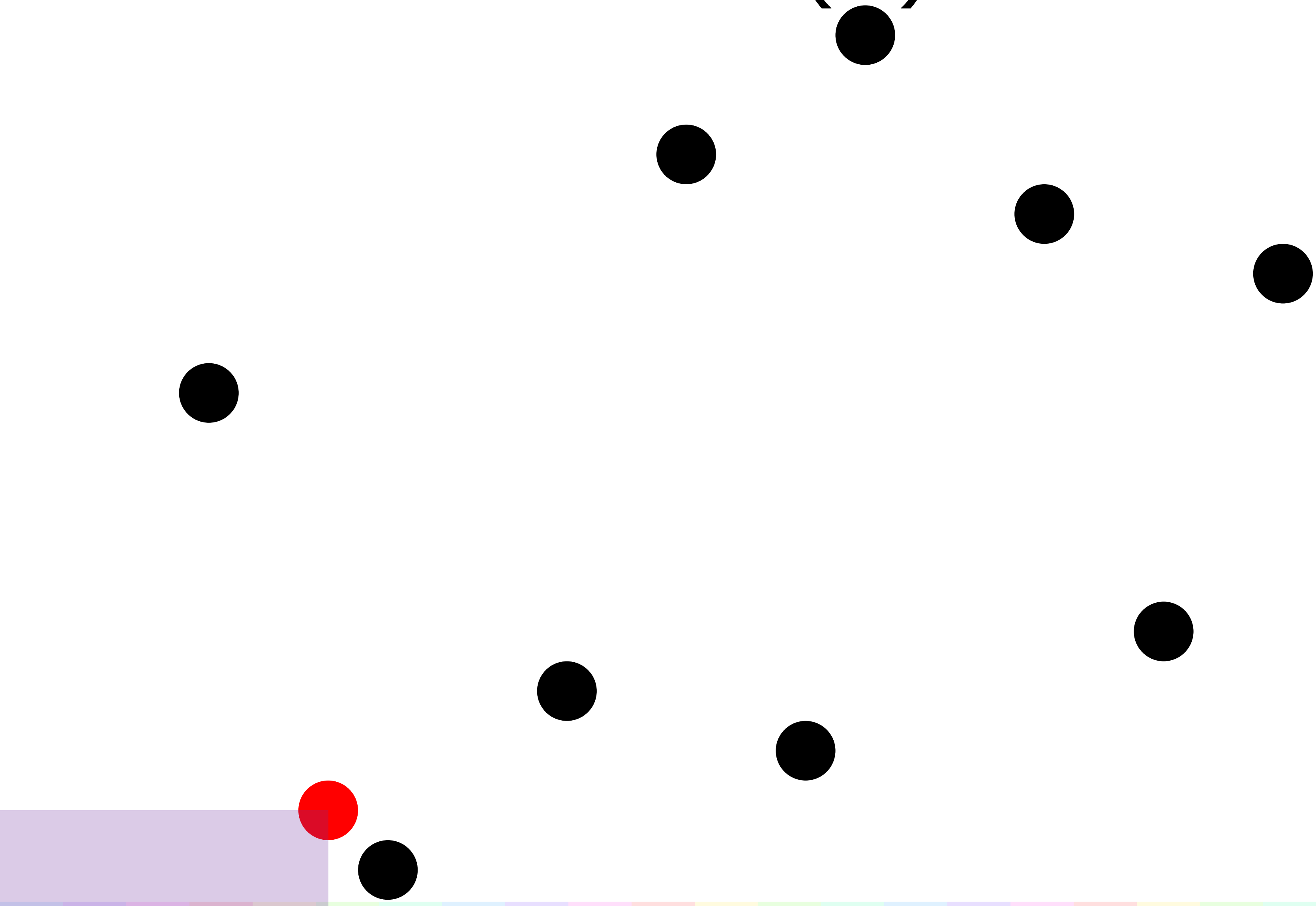
解法 (1)



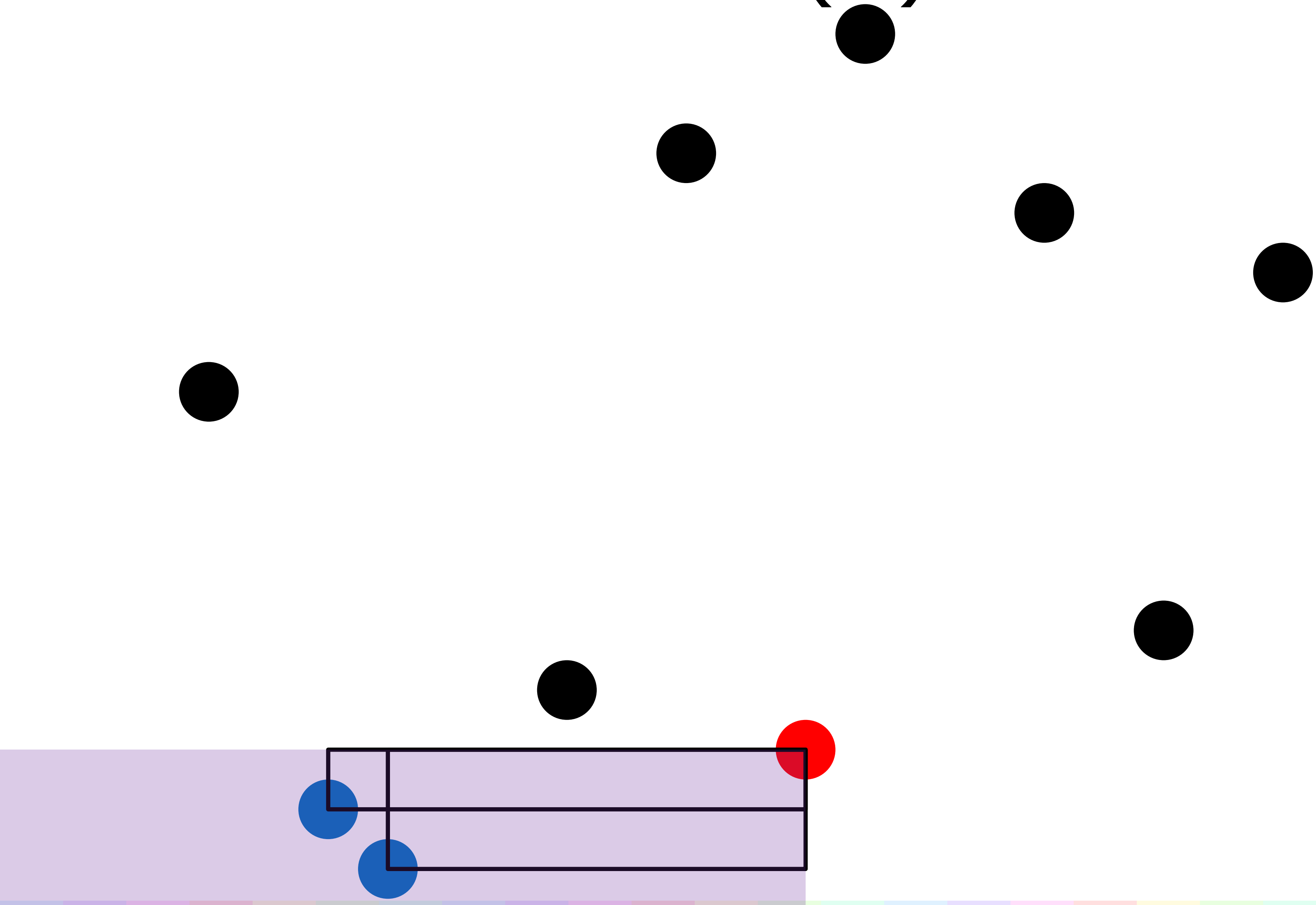
解法 (1)



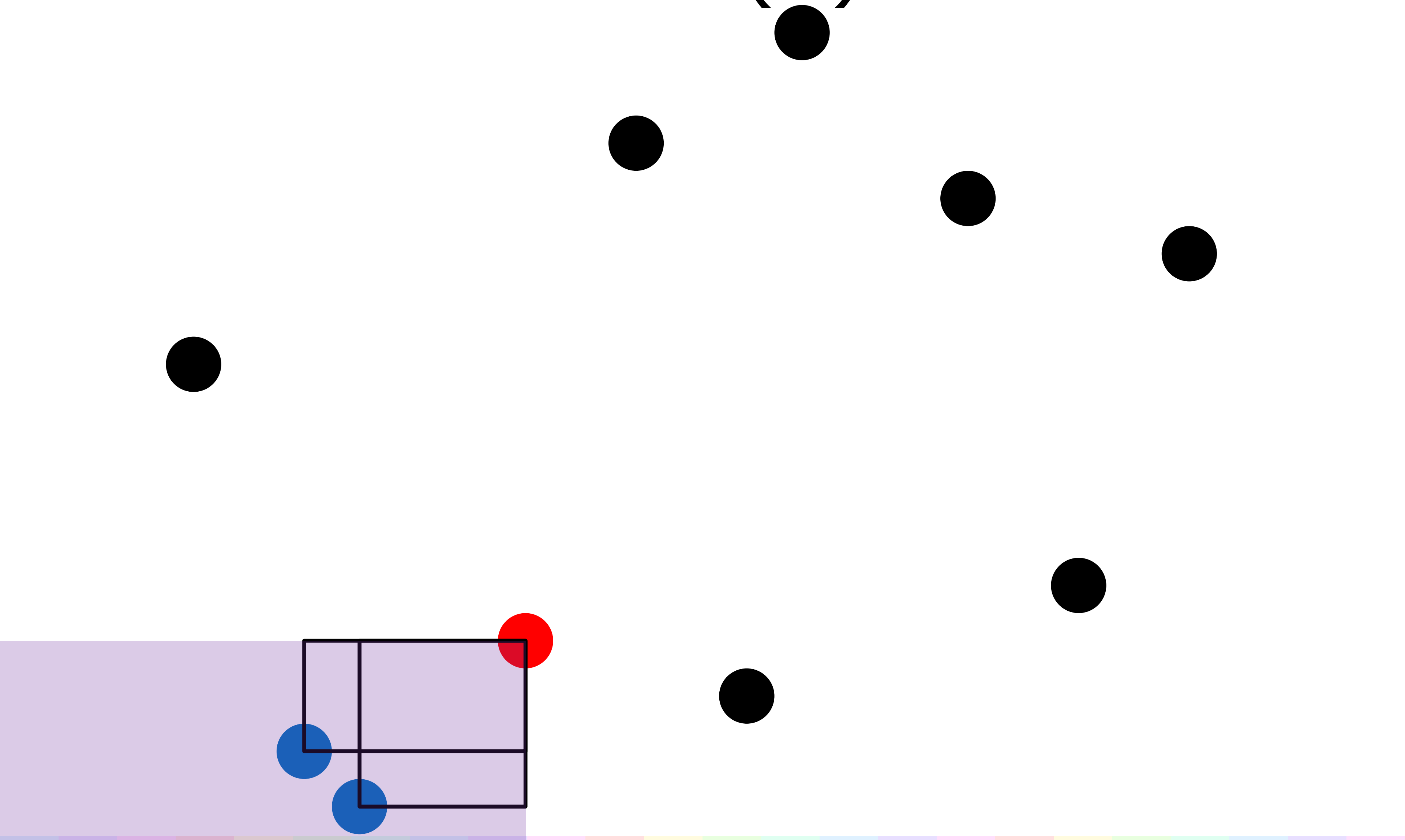
解法 (1)



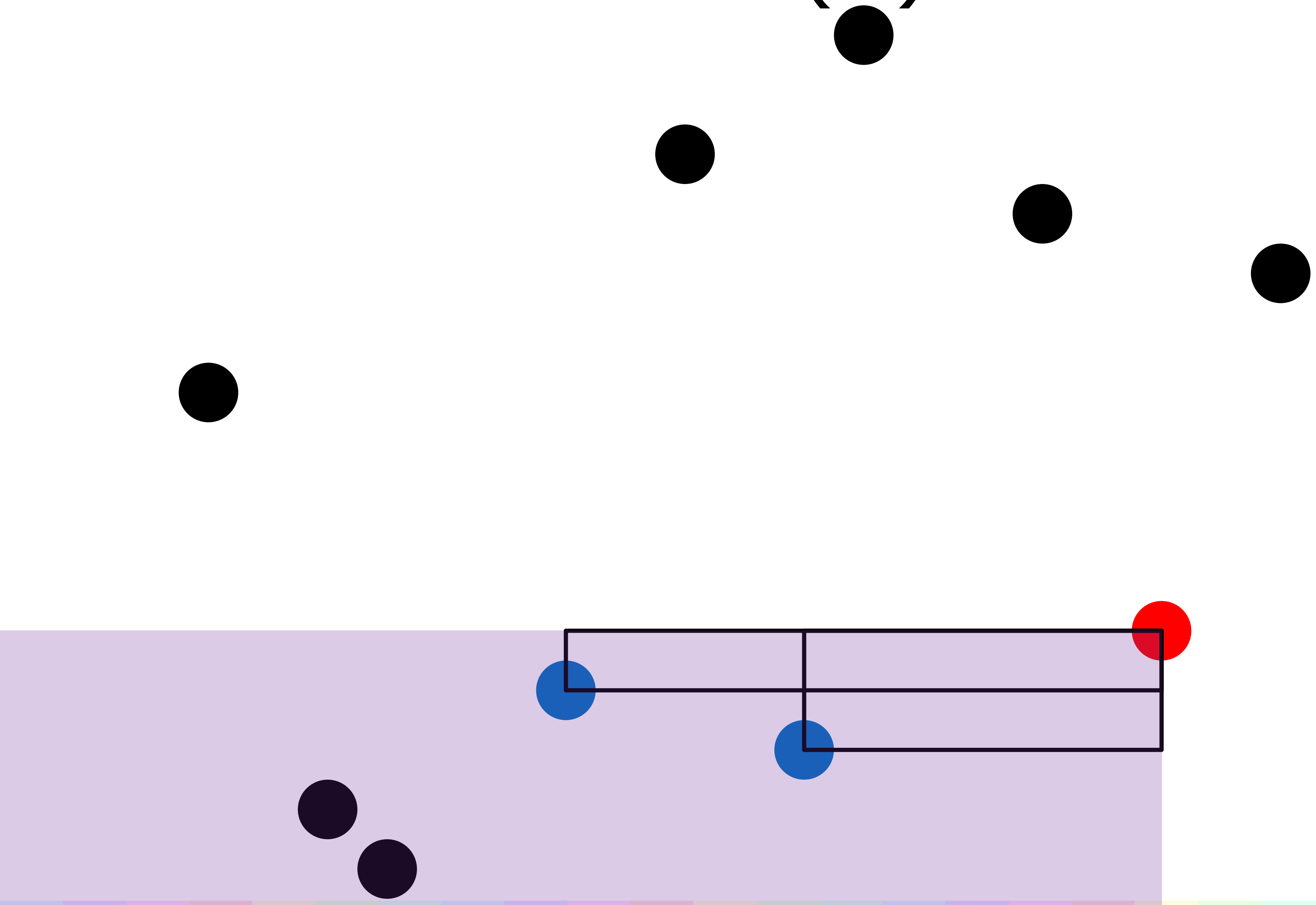
解法 (1)



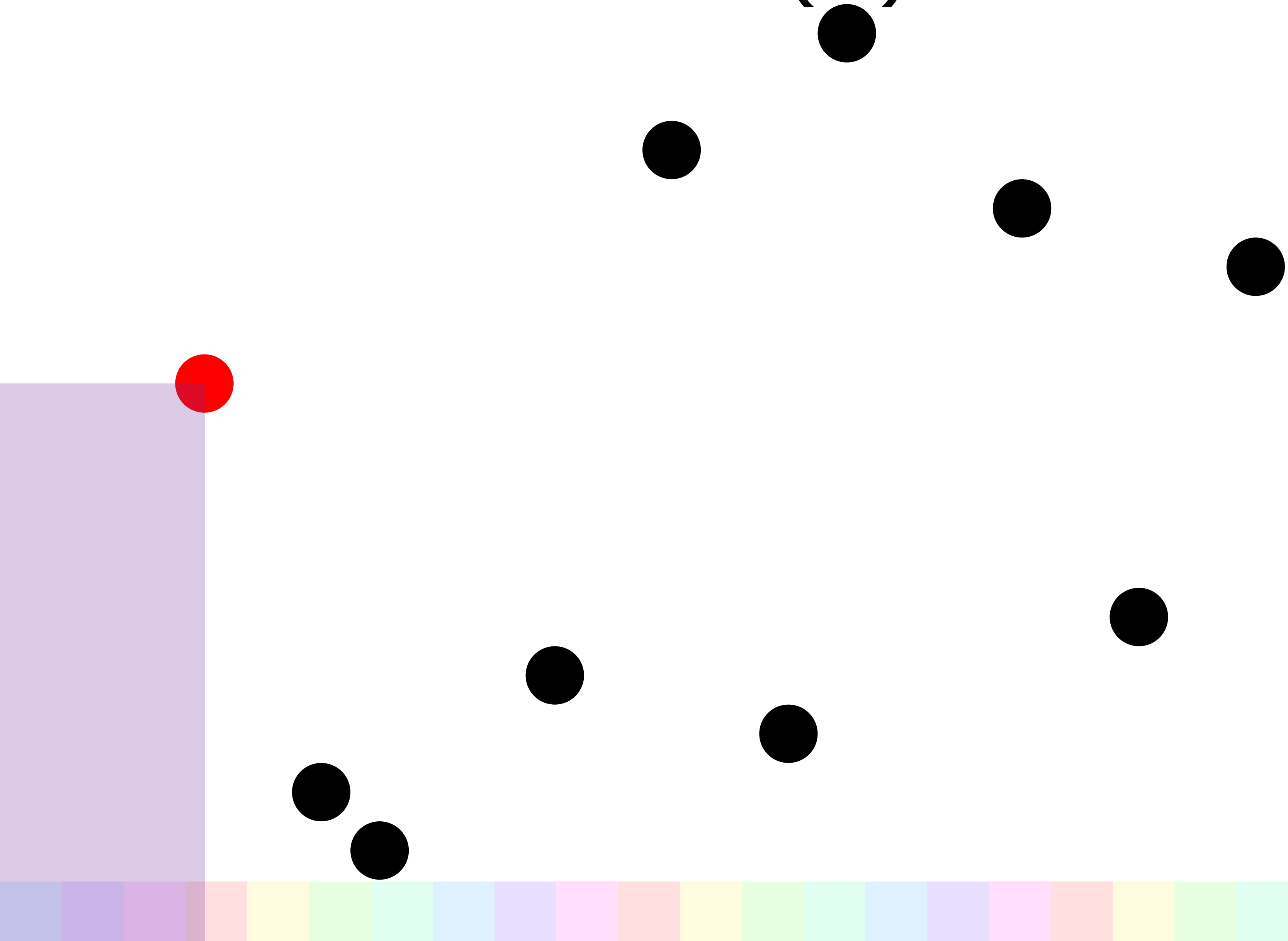
解法 (1)



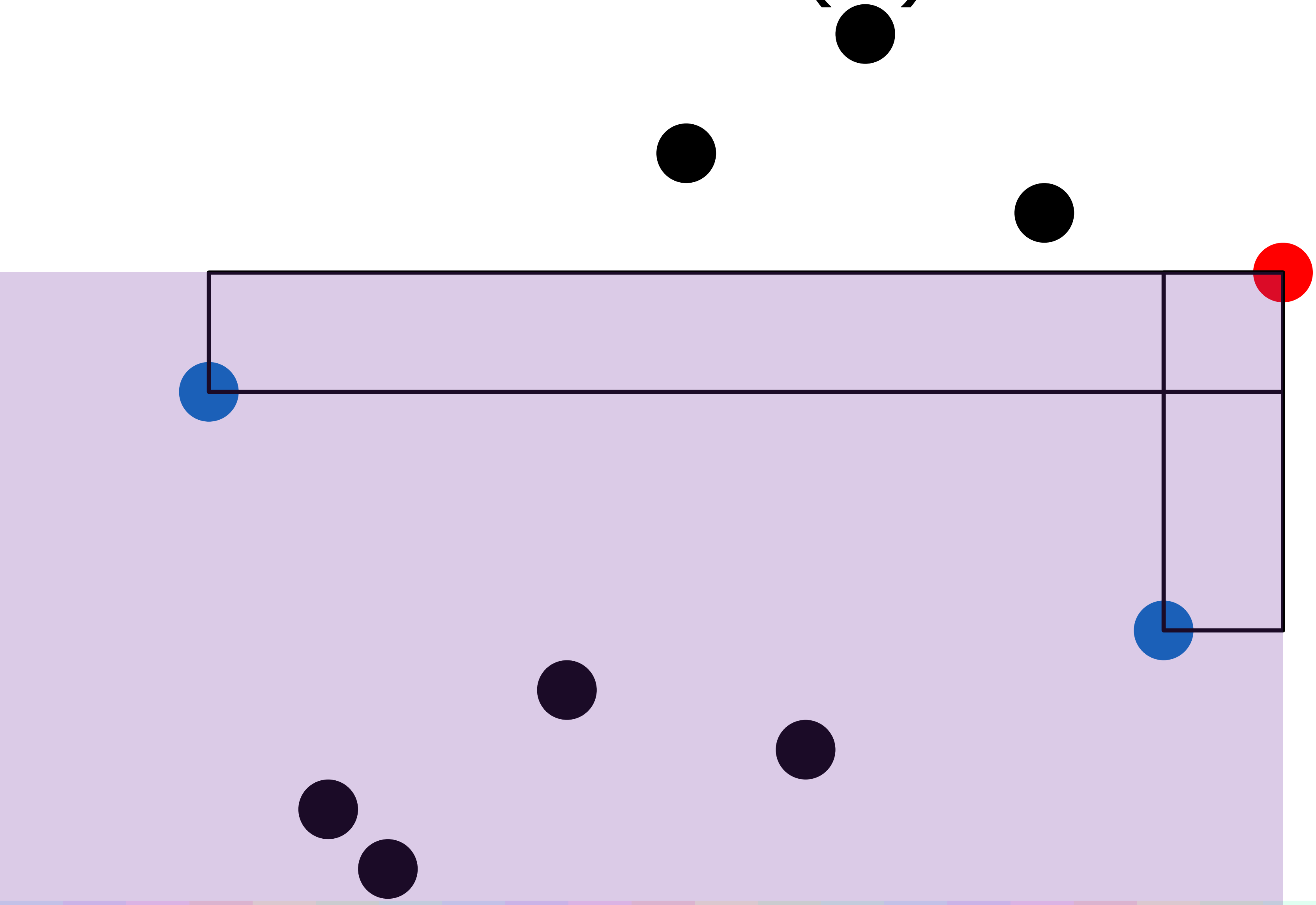
解法 (1)



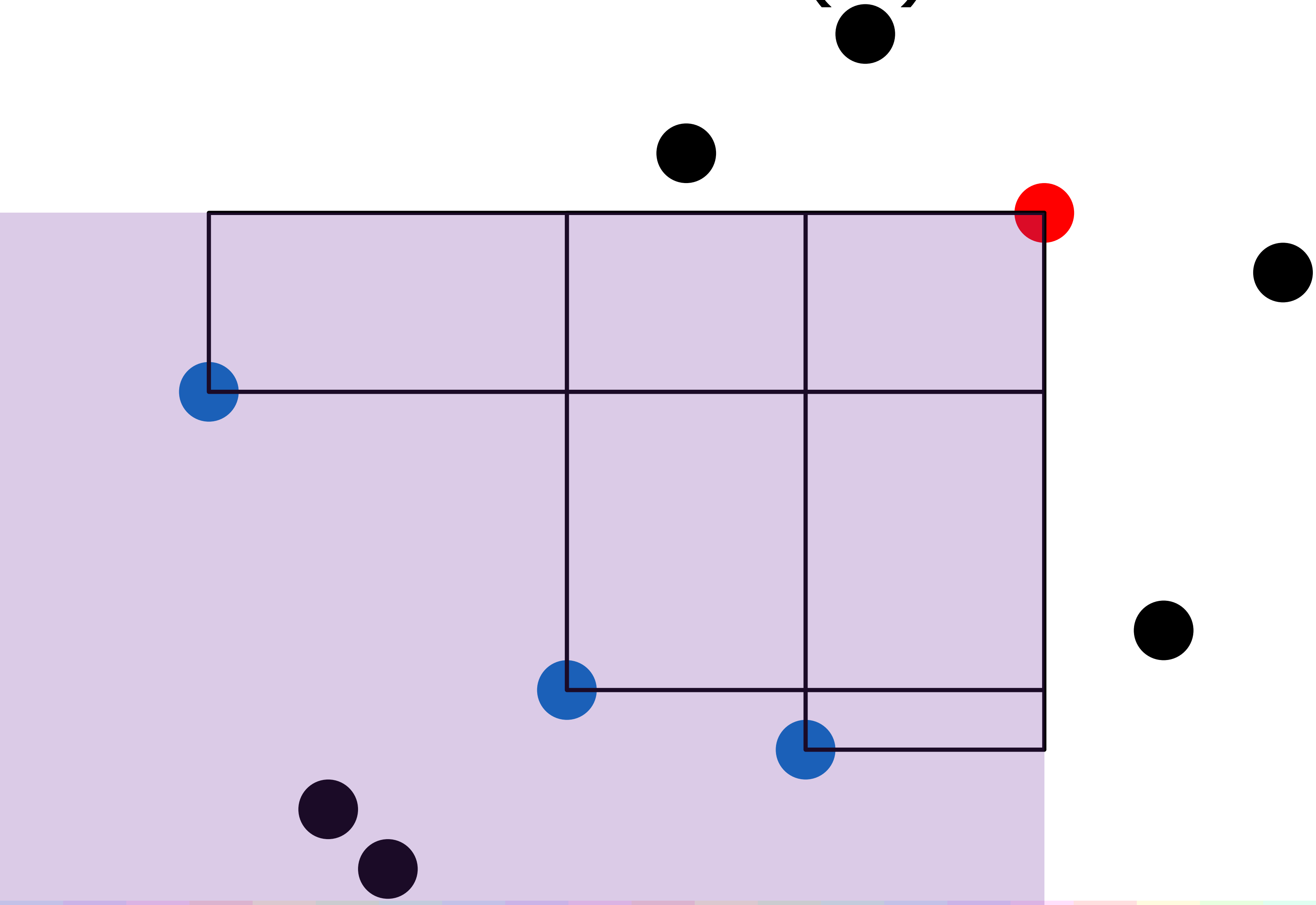
解法 (1)



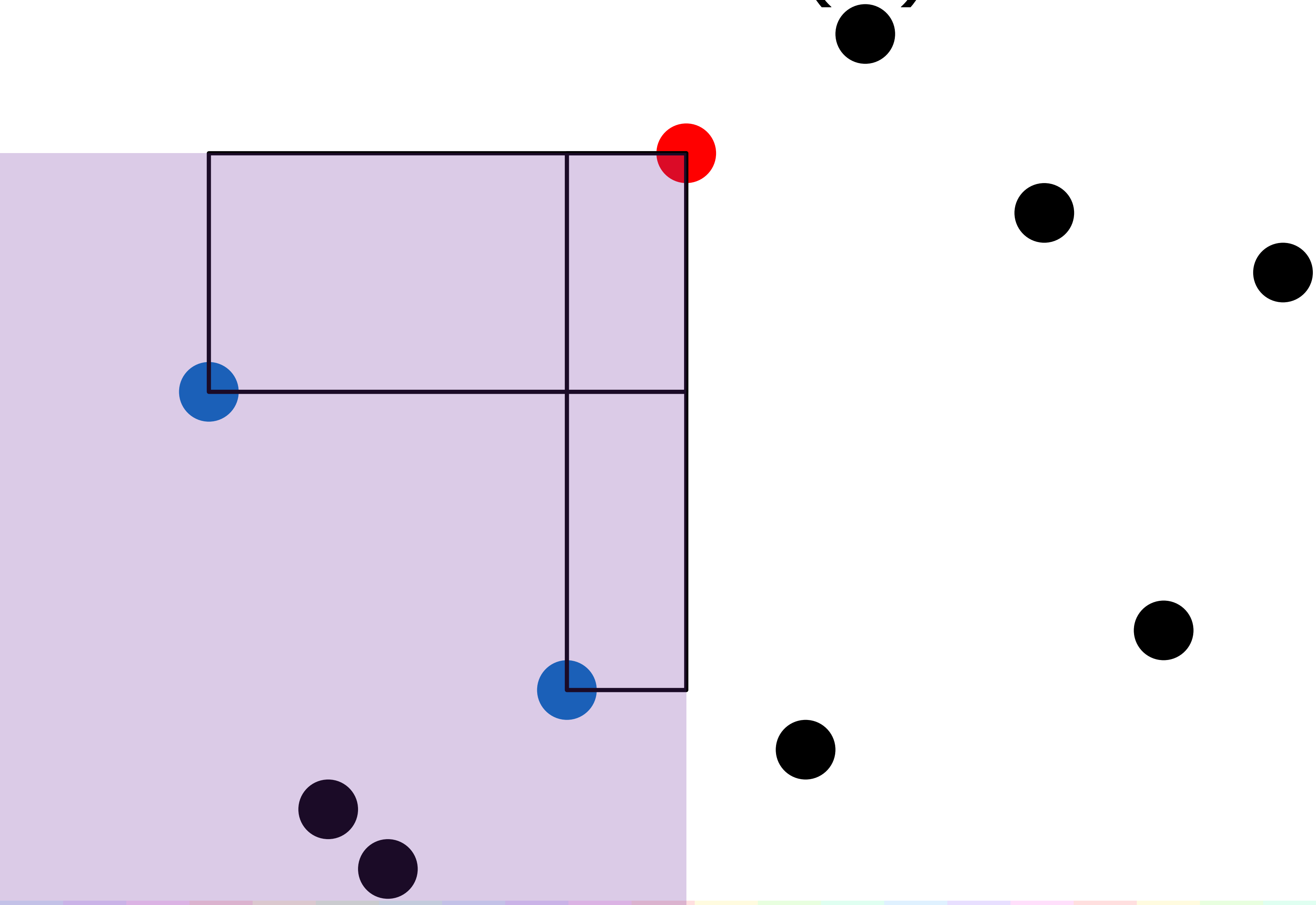
解法 (1)



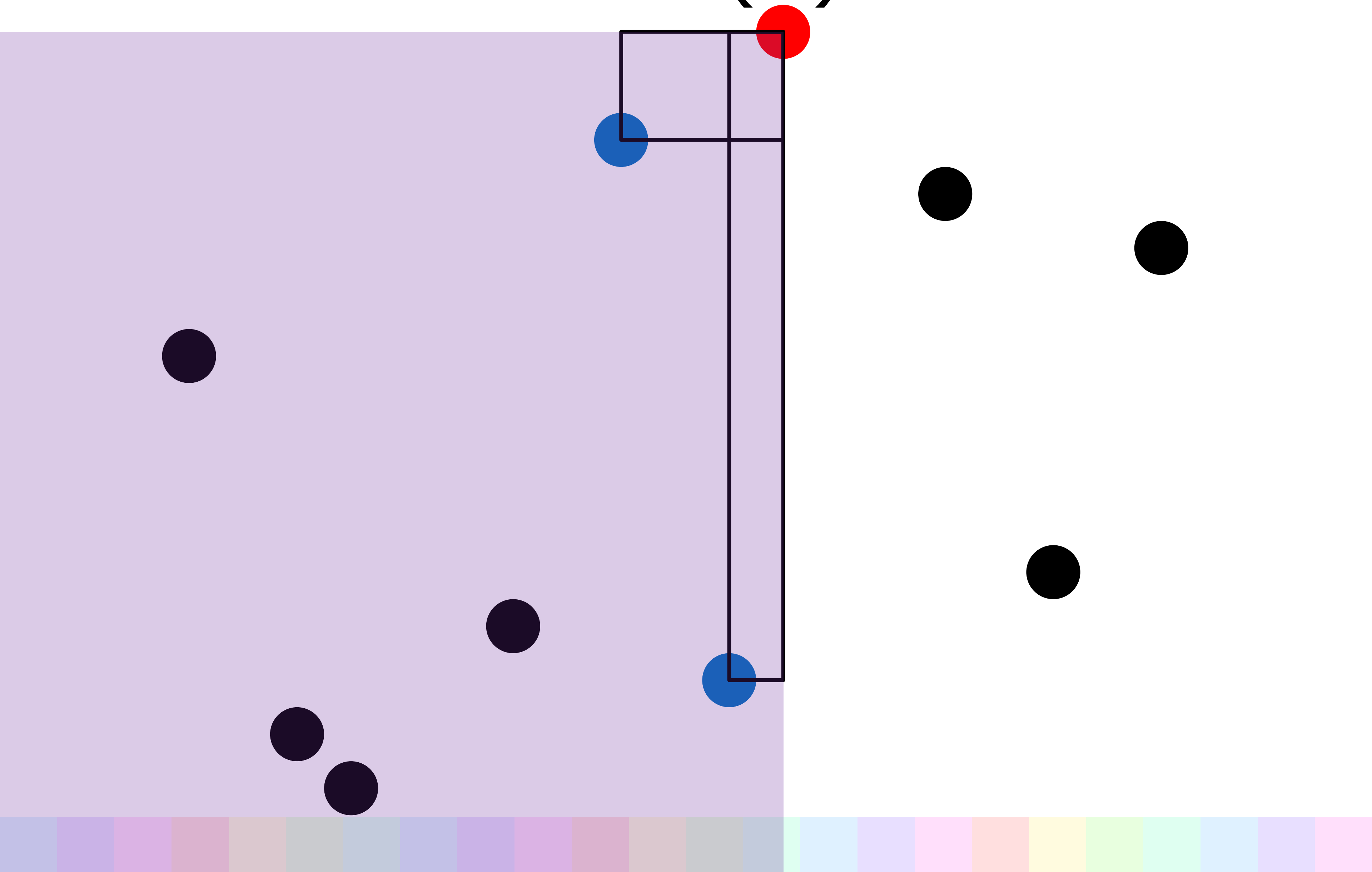
解法 (1)



解法 (1)



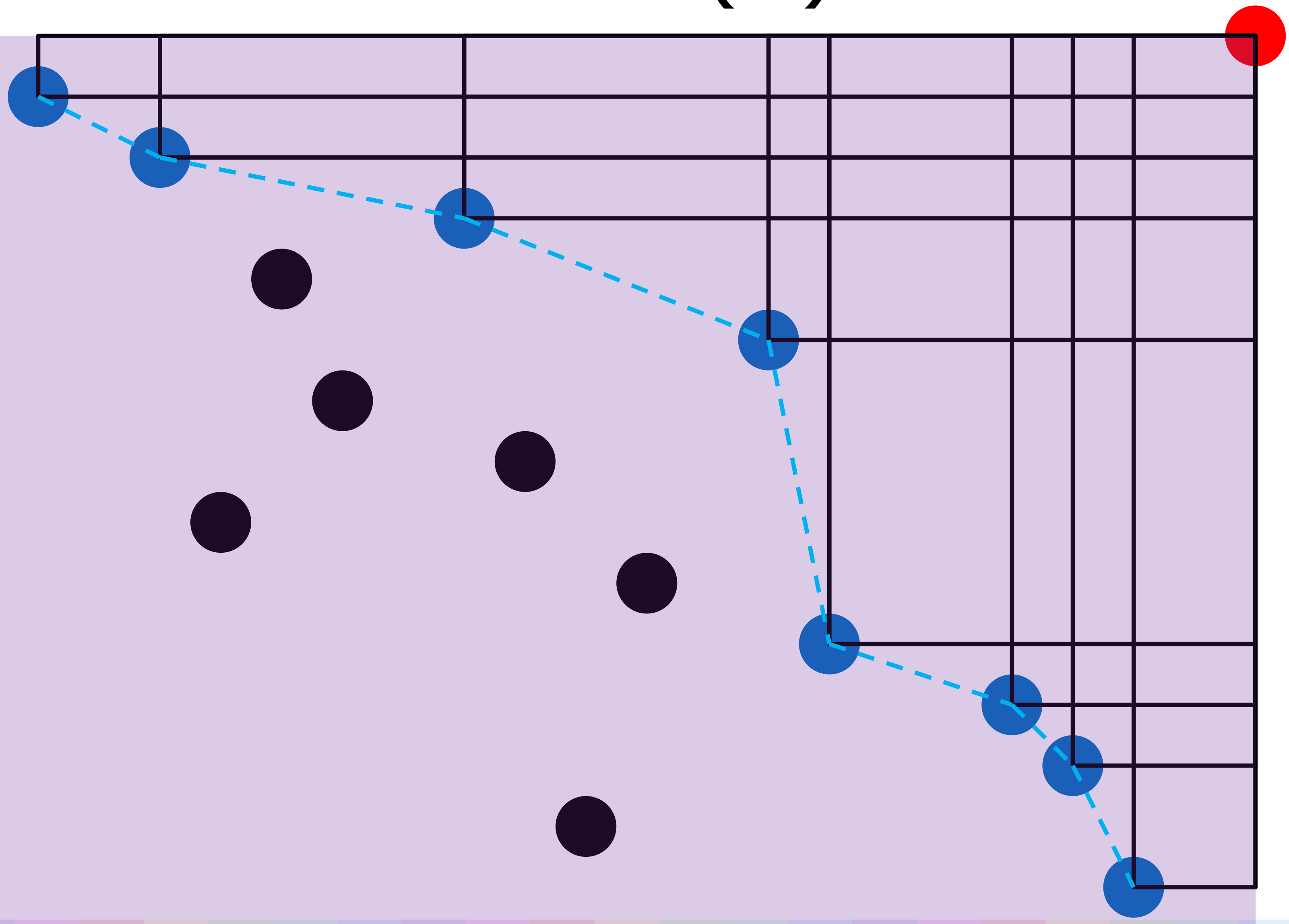
解法 (1)



解法 (1)

- ある点から左下の領域について，「右上のほうに並んでいる点たち」を高速に数えたい
  - 正確には，その領域内では右上に他の点がないような点たち

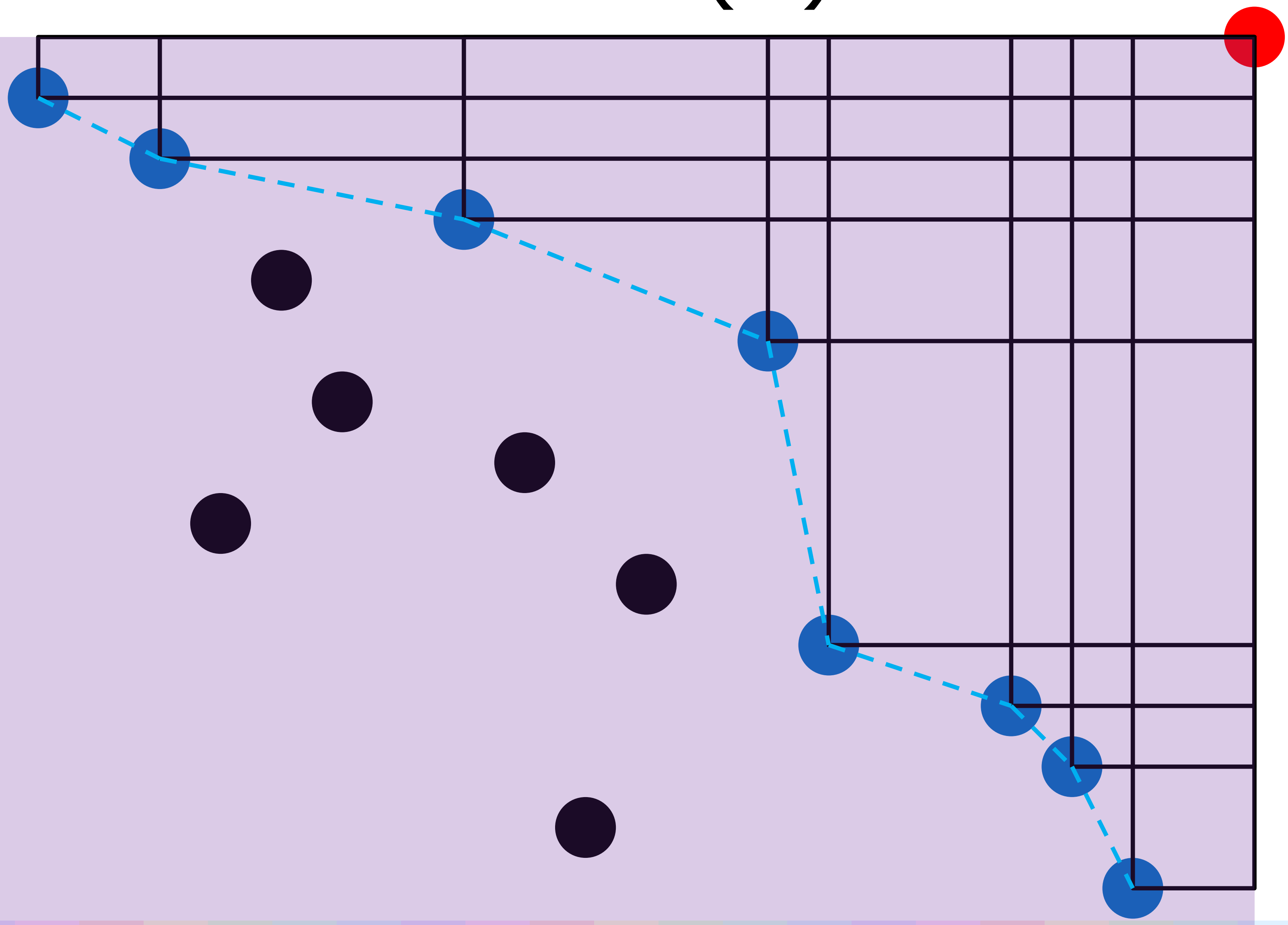
解法 (1)



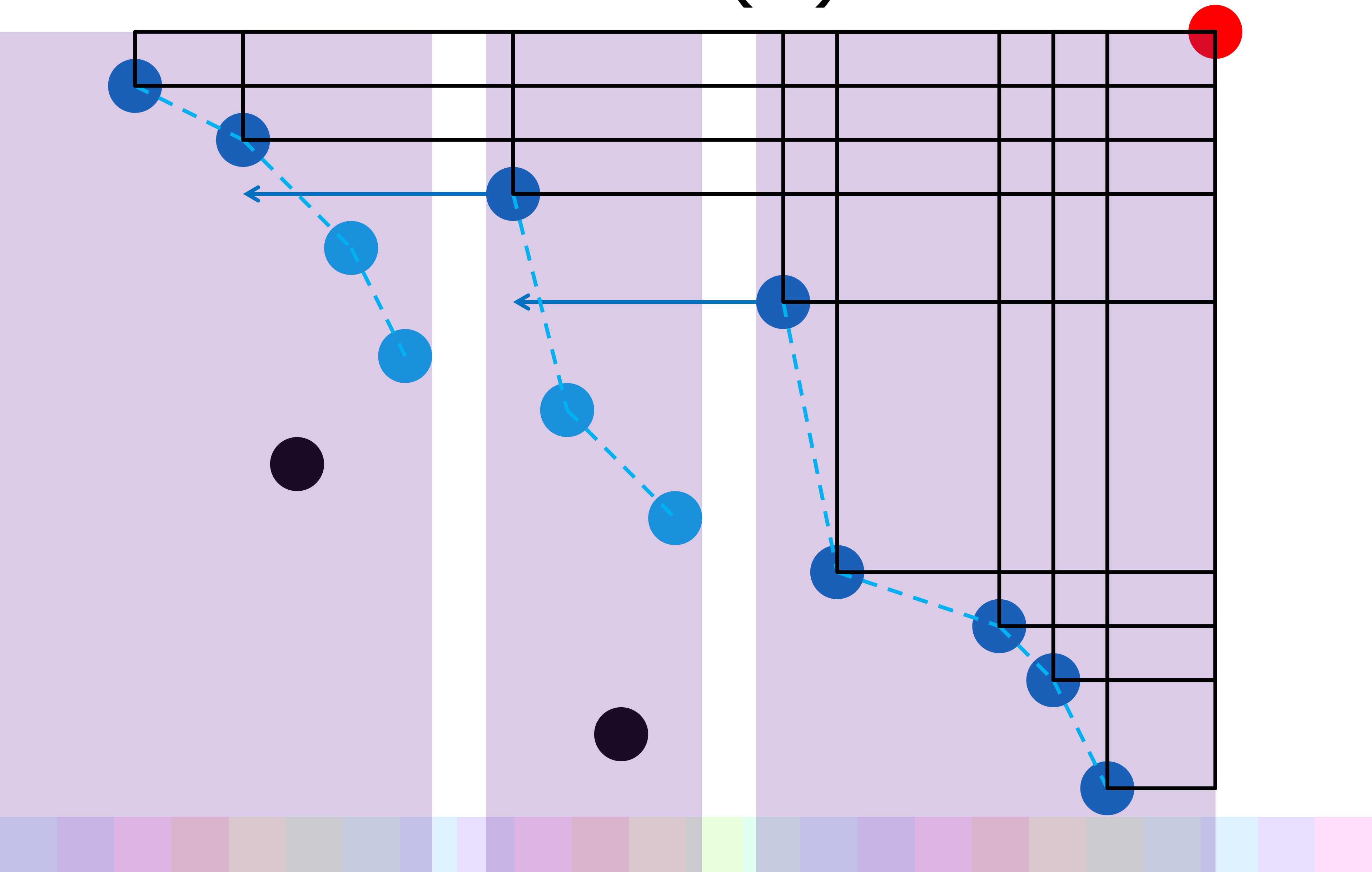
解法 (1)

- ある点から左下の領域について，「右上のほうに並んでいる点たち」を高速に数えたい
  - 正確には，その領域内では右上に他の点がないような点たち
- y 座標が小さい点から順番に追加していくことにする
  - y 座標の範囲を気にしなくてよくなる

解法 (1)



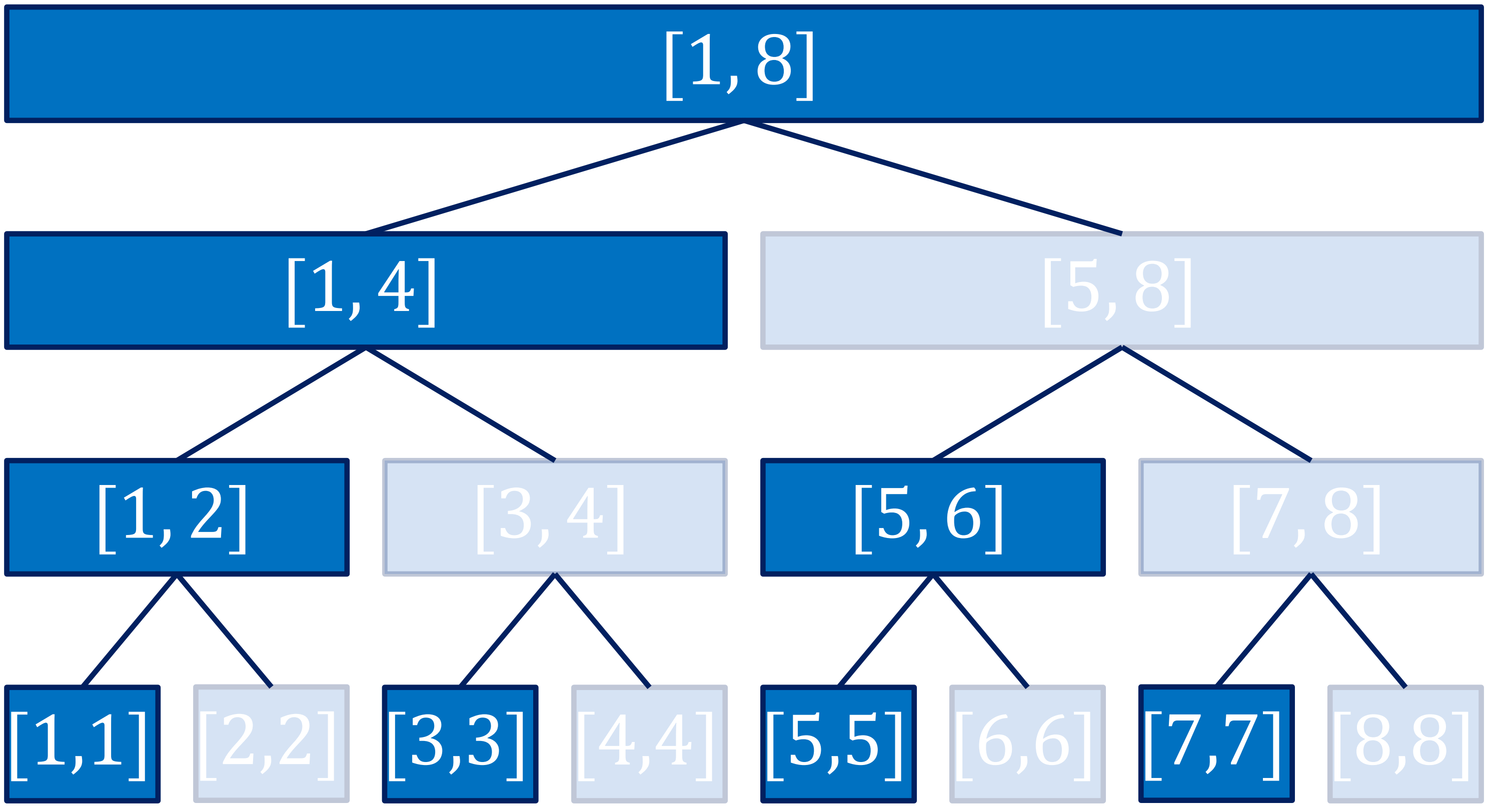
解法 (1)



解法 (1)

- x 座標の範囲をいくつかの区間に分割する
- 各区間で「右上のほうに並んでいる点たち」のどこから見ればいいか
  - 二分探索でわかる
- どう分割するか？

解法 (1)

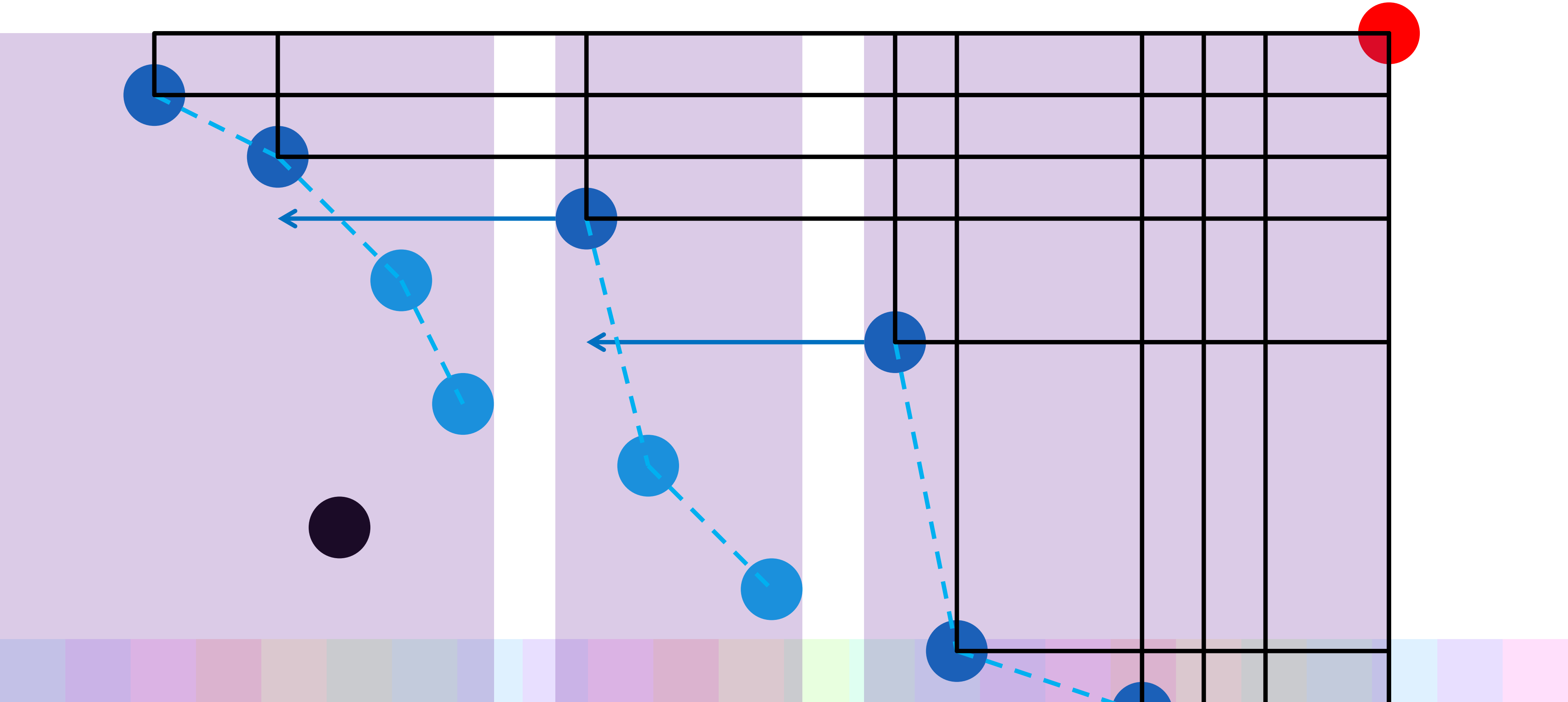


解法 (1)

- x 座標を座標圧縮して 1 から N に
- Segment Tree あるいは Binary Indexed Tree を考え，各区間に対して，「右上のほうに並んでいる点たち」を管理する
  - 点たちを座標でソートした列としてもつ
  - $O(N \log N)$ メモリ

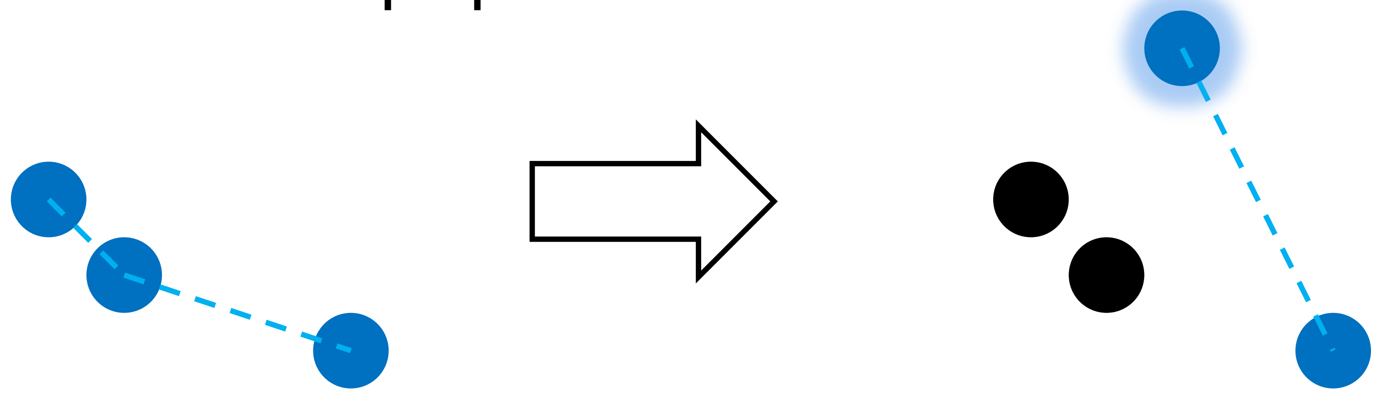
解法 (1)

- 数える
  - $O(\log N)$ 個に分割された区間を右から辿る



解法 (1)

- 点を追加する
  - 全体でみると，どの点も高々 1 回追加・削除されるだけ
  - y 座標が小さい順に追加するので，配列から必要なだけ pop してから追加



解法 (1)

- 必要な操作
  - 末尾から削除・末尾に追加
  - 二分探索
  - 個数を数える

- 全体で $O(N(\log N)^2)$ 時間
  - 実装：ちょっと大変

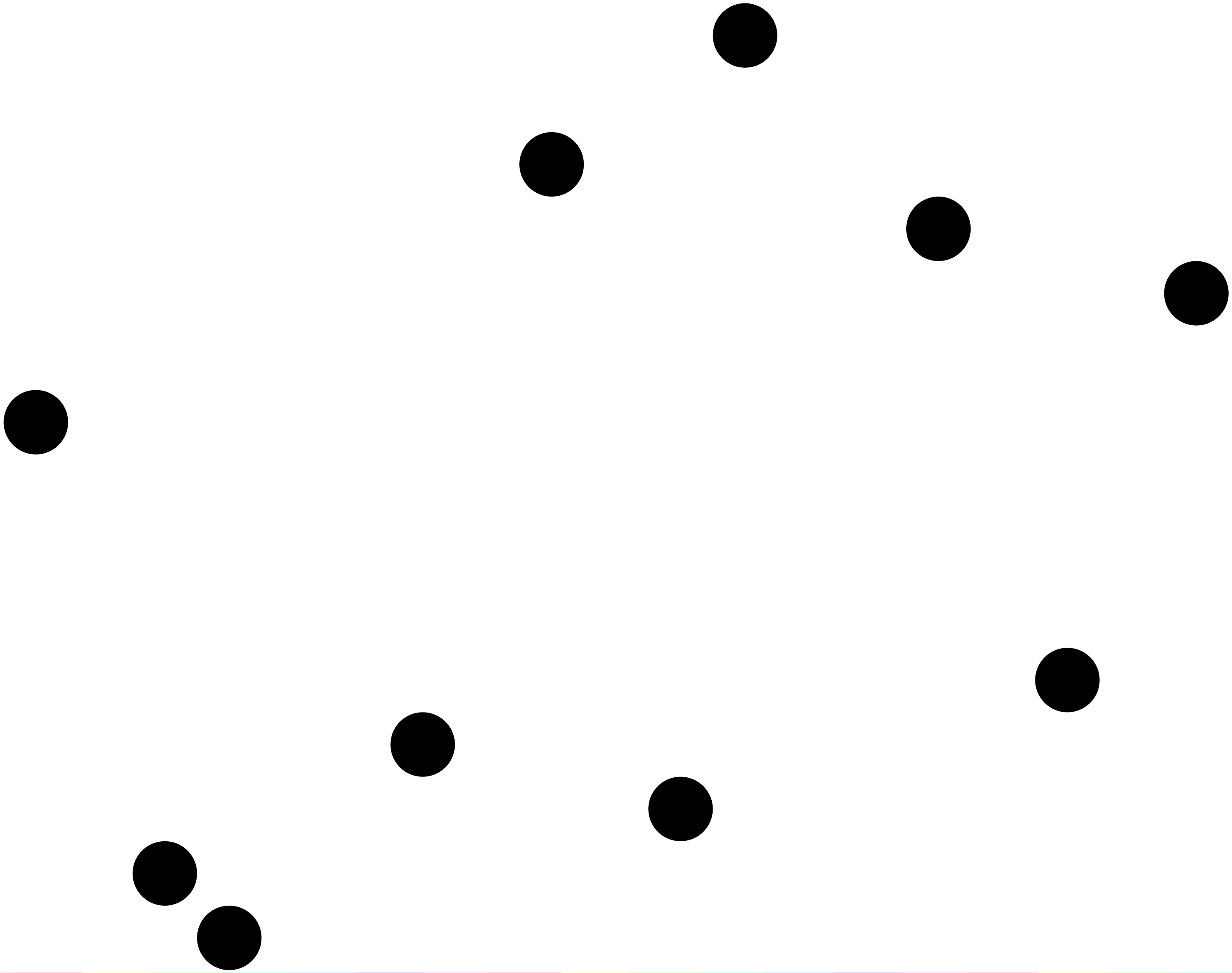
小課題 3 (85 点)

- すごいデータ構造なしでいったい？？

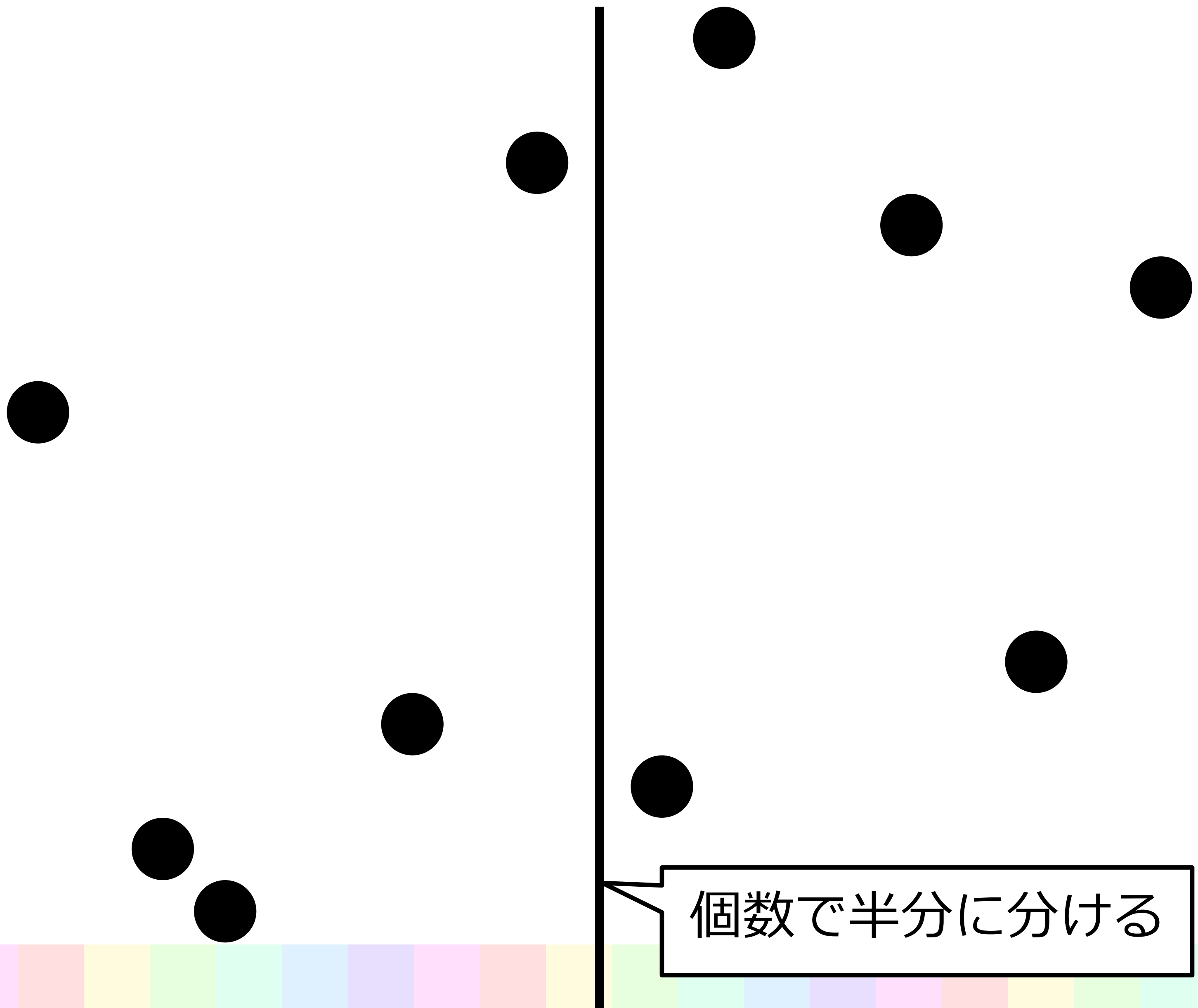
分割統治法

-Divide and Conquer-

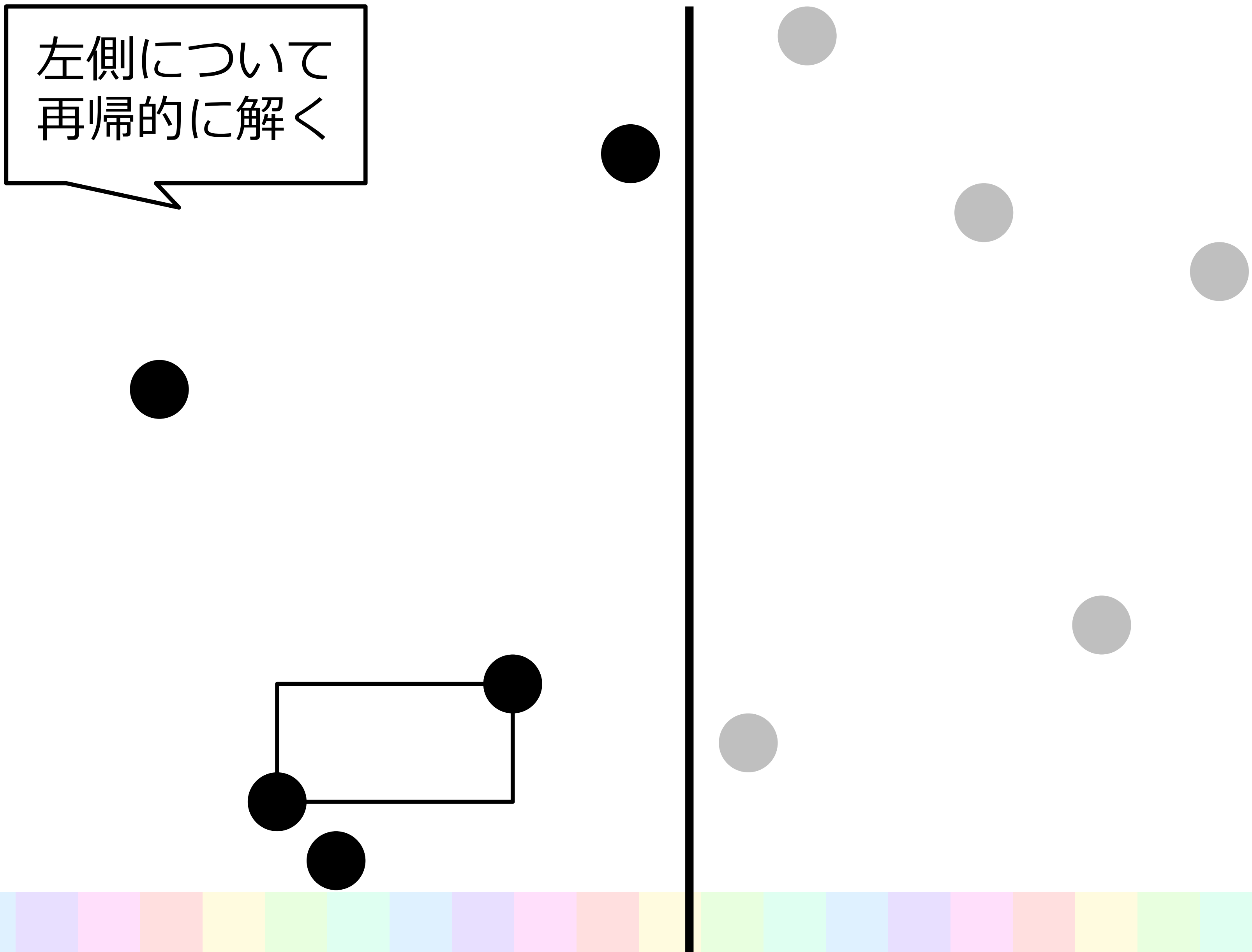
分割統治法



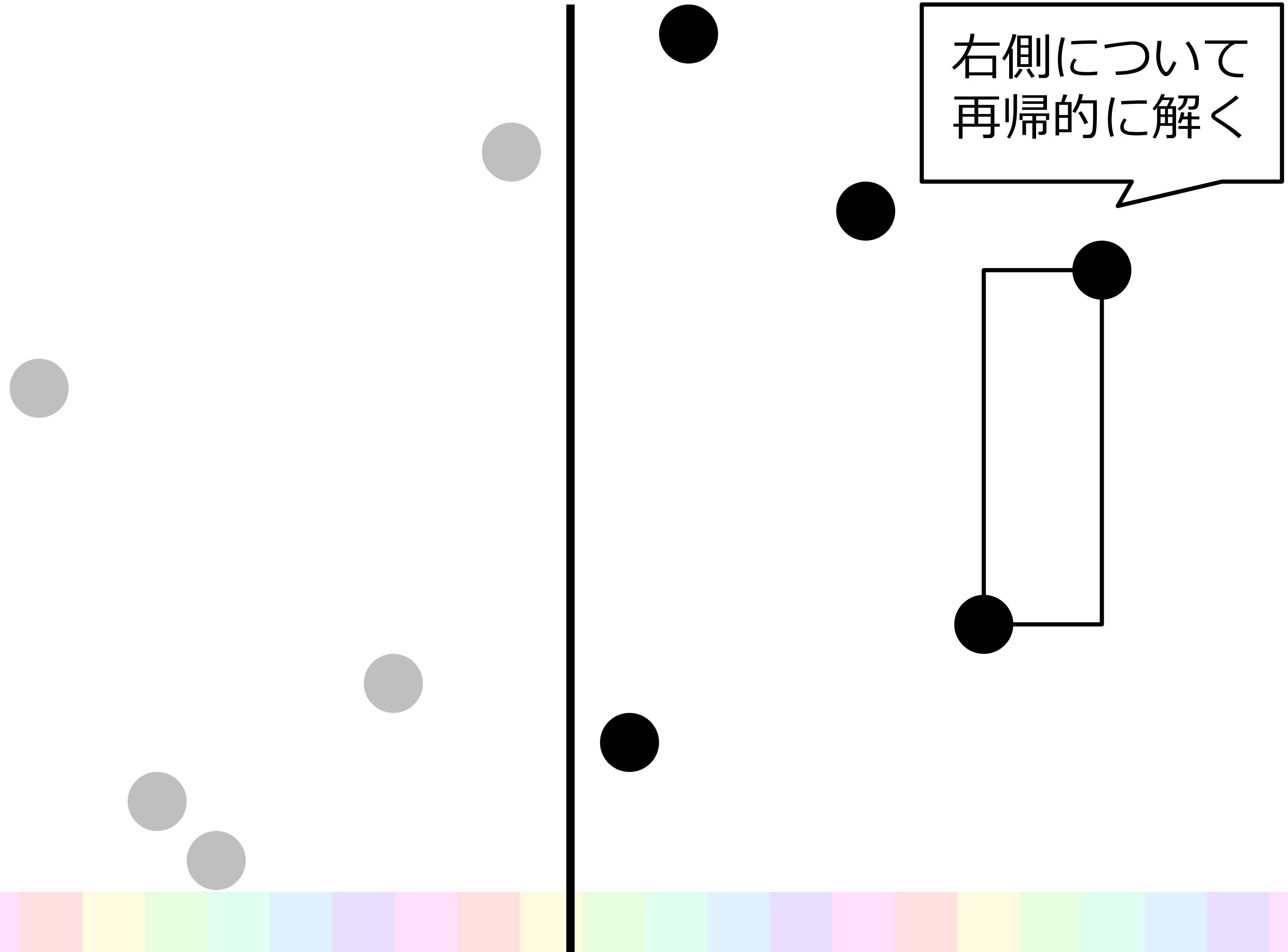
分割統治法



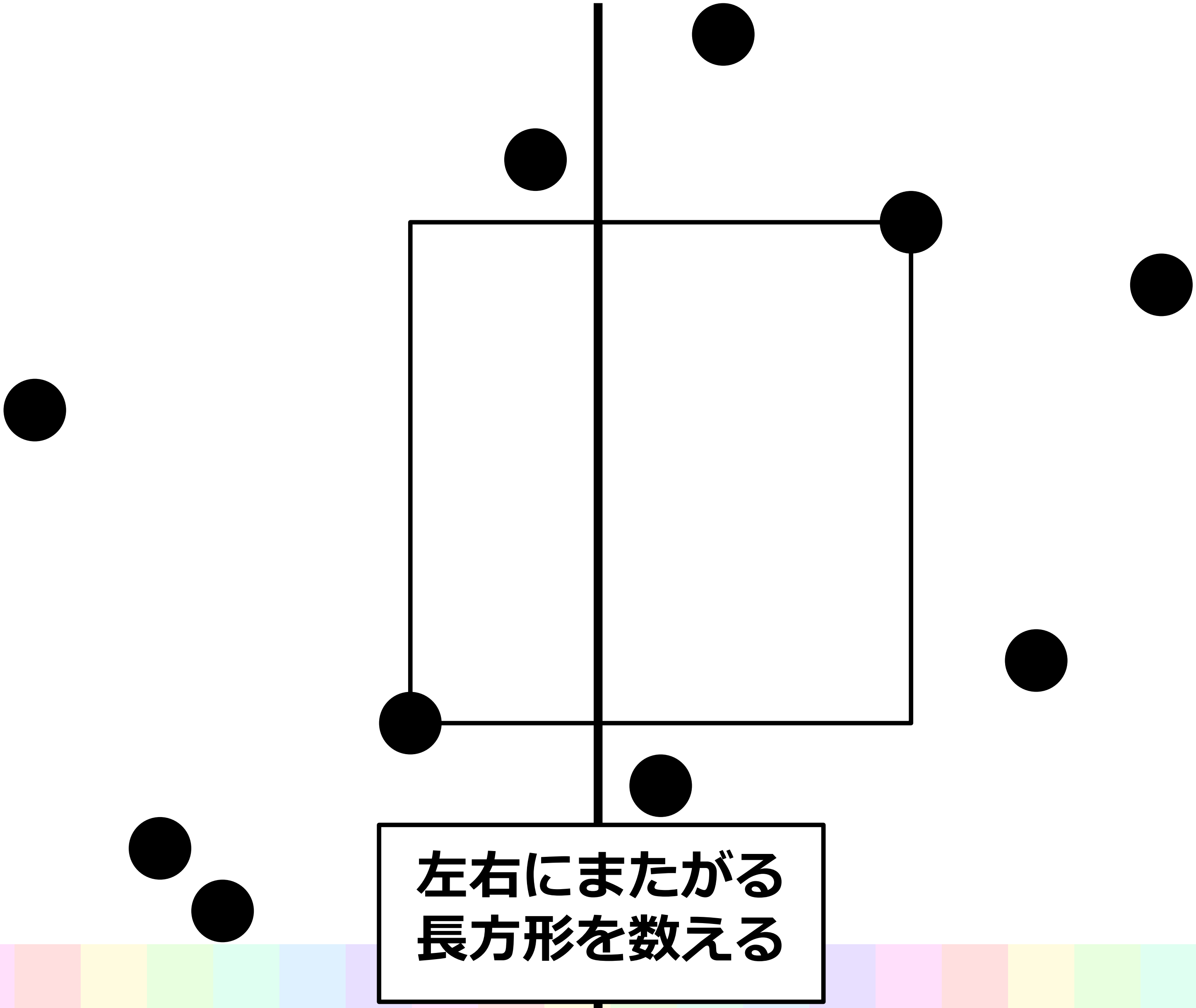
分割統治法



分割統治法



分割統治法



分割統治法

- 左右にまたがる長方形をどれくらい速く数えるか？
  - これが $O(N(\log N)^k)$ 時間 $(k \geqq 0)$ でできるなら，全体では $O\left(N(\log N)^{k+1}\right)$ 時間
    - サイズ N のとき $T(N)$ ステップかかるとし，左右にまたがるのを $c\,N\,(\log N)^k$ ステップでできるとすると，$T(2^m) = 2\,T(2^{m-1}) + c\,2^m\,m^k$，これを解いて $\frac{T(2^m)}{2^m} = c\left(1^k + 2^k + \cdots + m^k\right) = O(m^{k+1})$

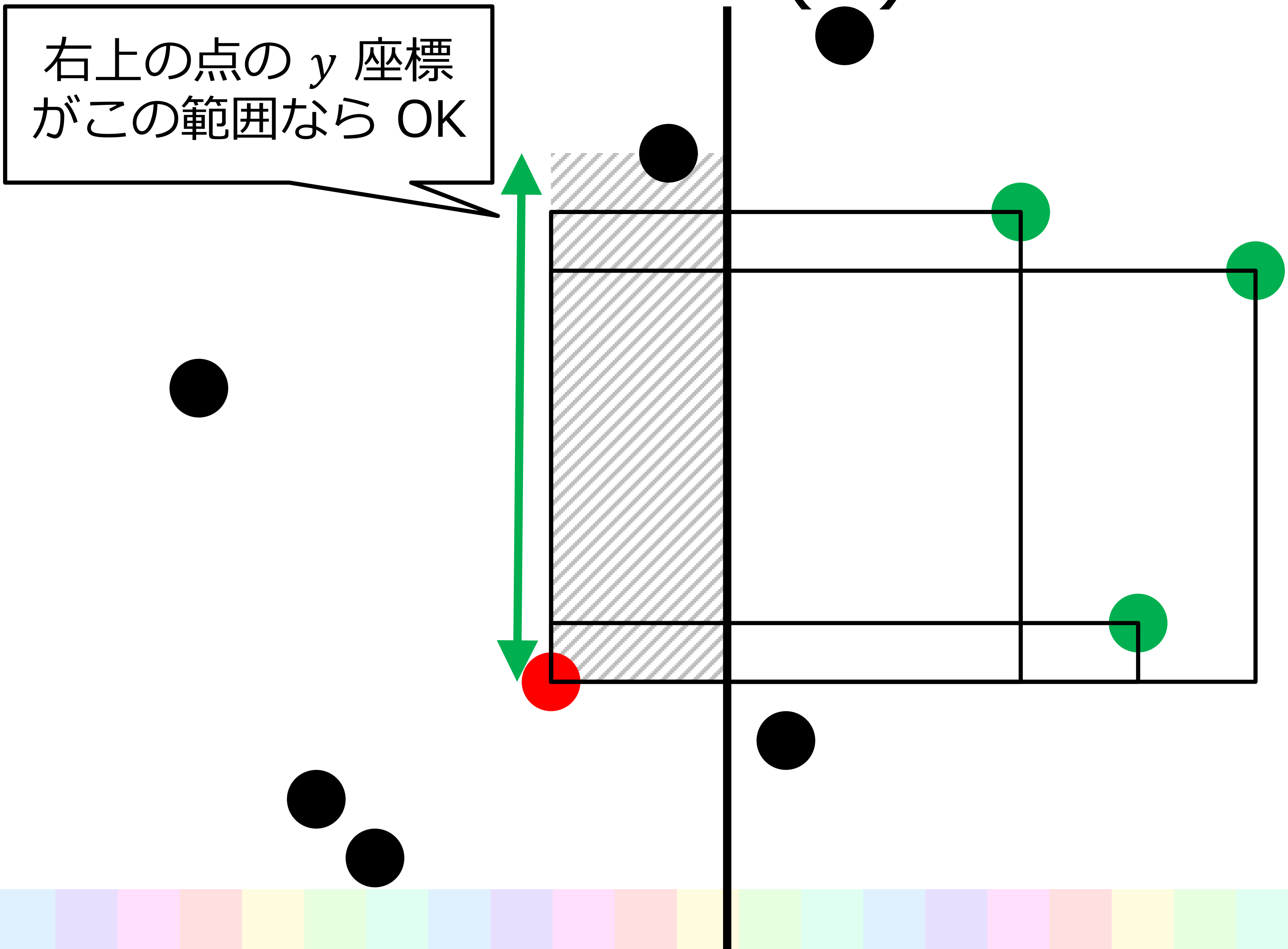
分割統治法

- 左右にまたがる長方形をどうやって数えるか？
- やっぱりデータ構造……？
  - はい！ → 解法 (2) へ
  - いいえ → 解法 (3) へ

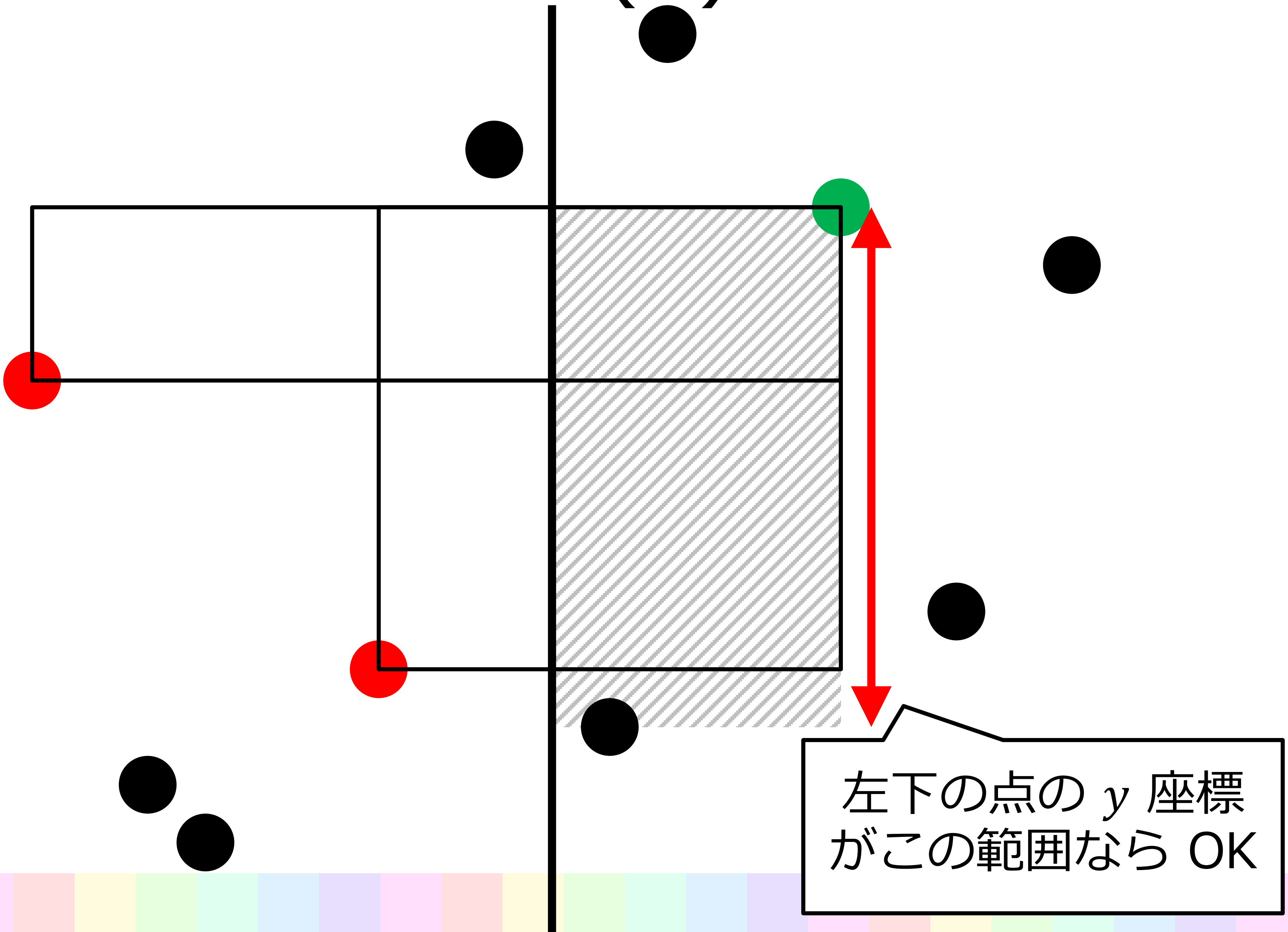
解法 (2)

- 「内部に他の点がない」
  - 中央線より左側に他の点がない
  - 中央線より右側に他の点がない

解法 (2)



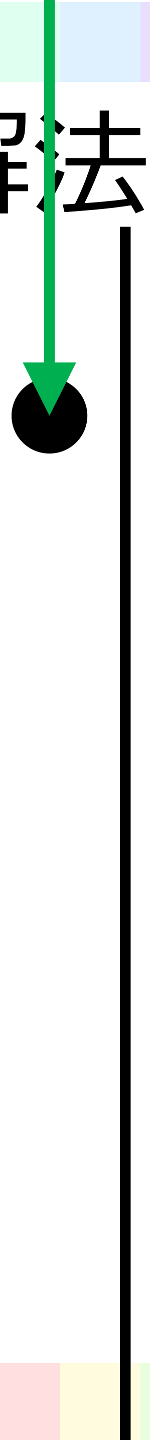
解法 (2)



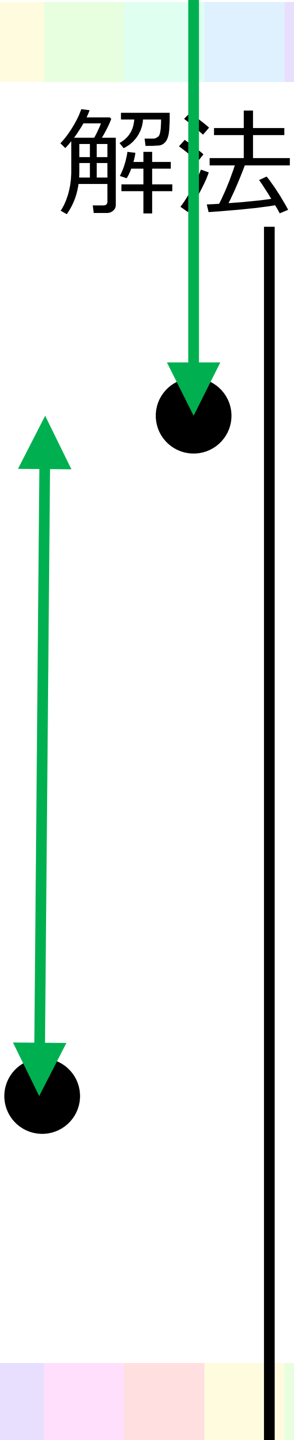
解法 (2)

中央線に近い方から
範囲を求める

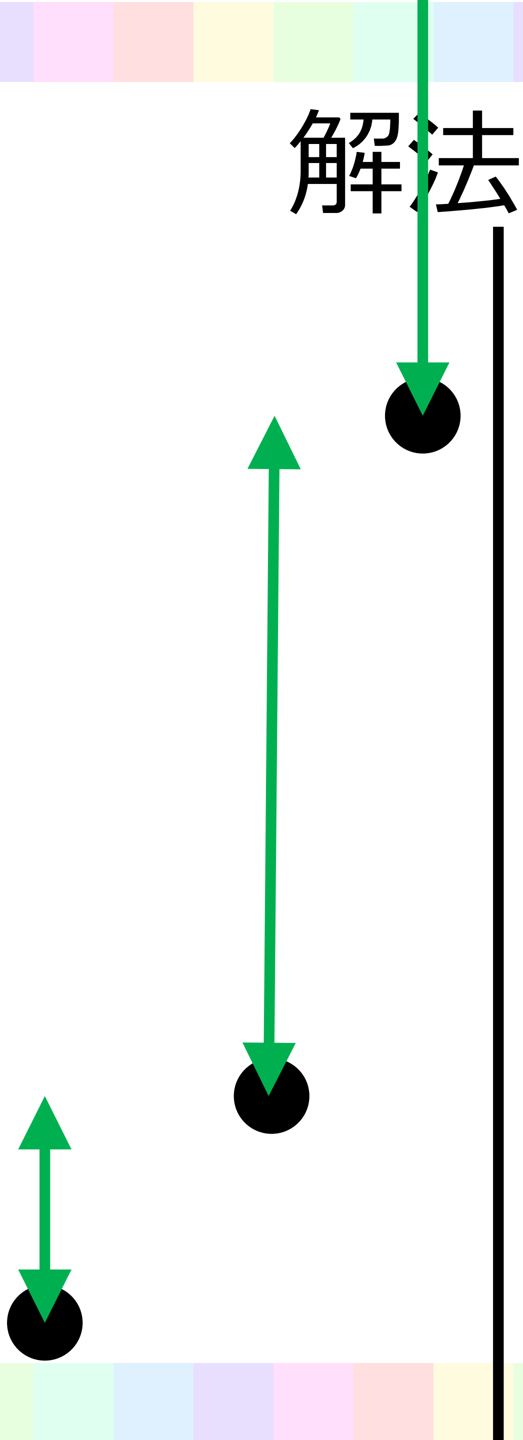
解法 (2)



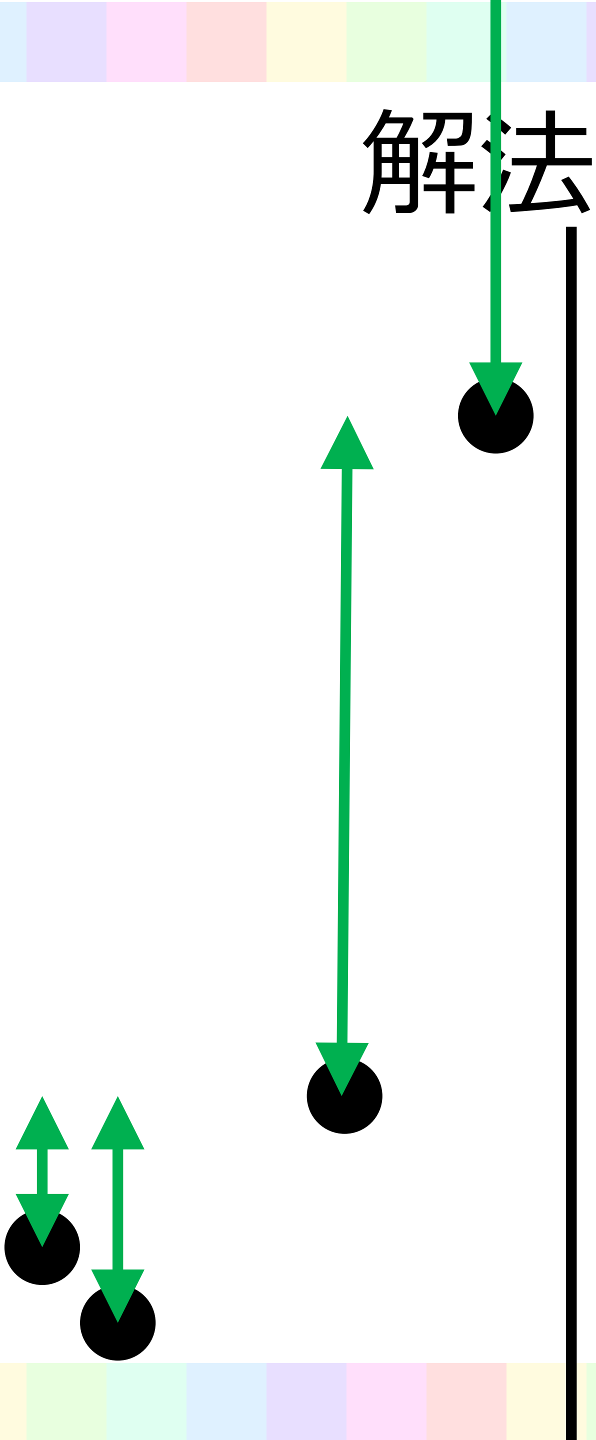
解法 (2)



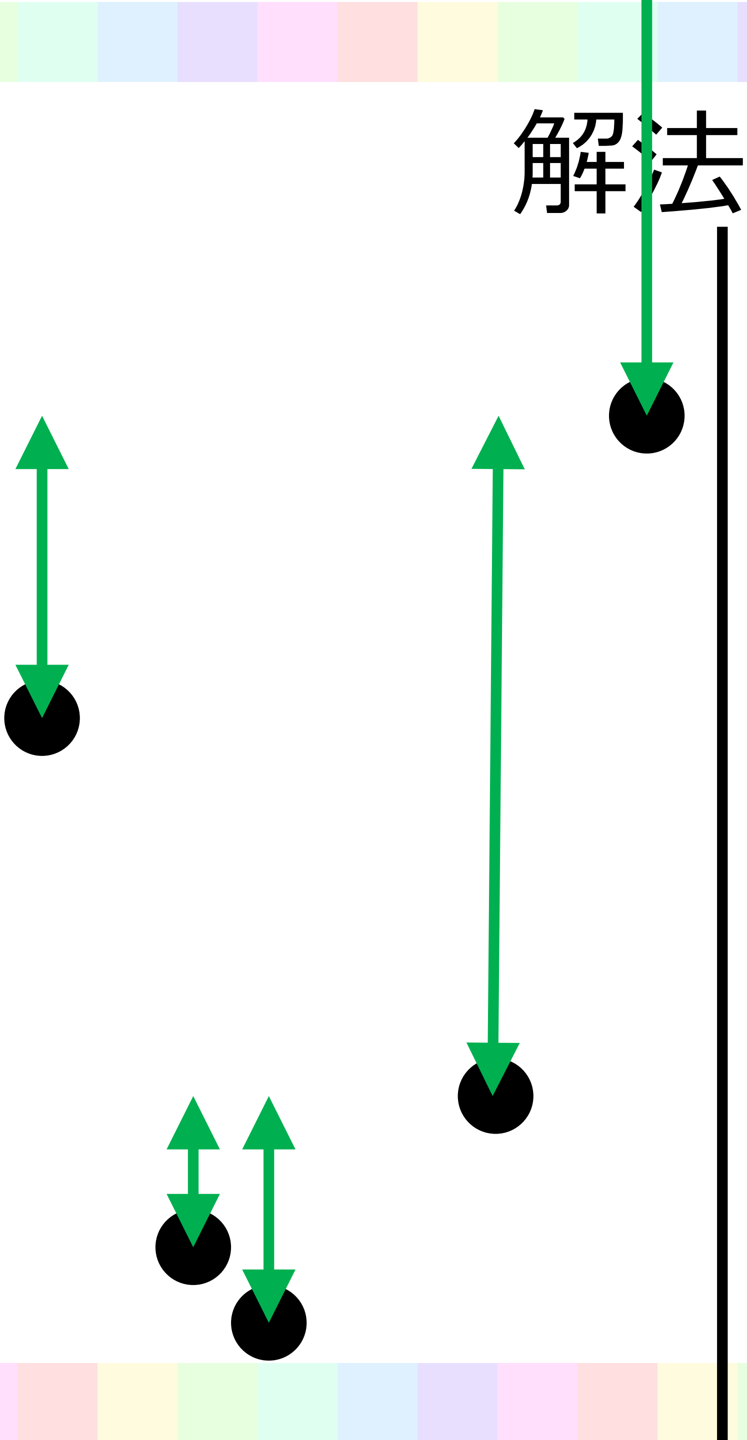
解法 (2)



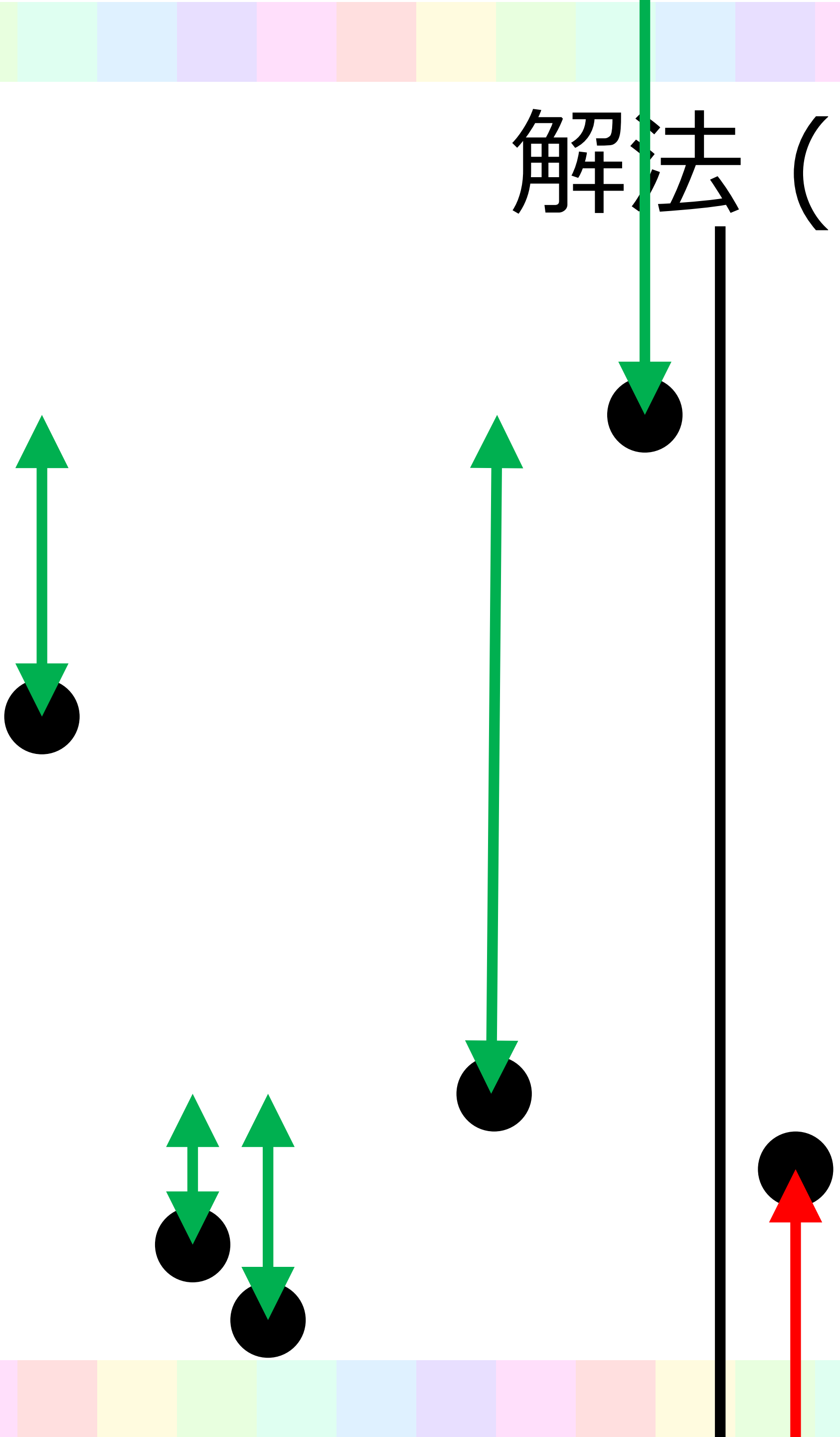
解法 (2)



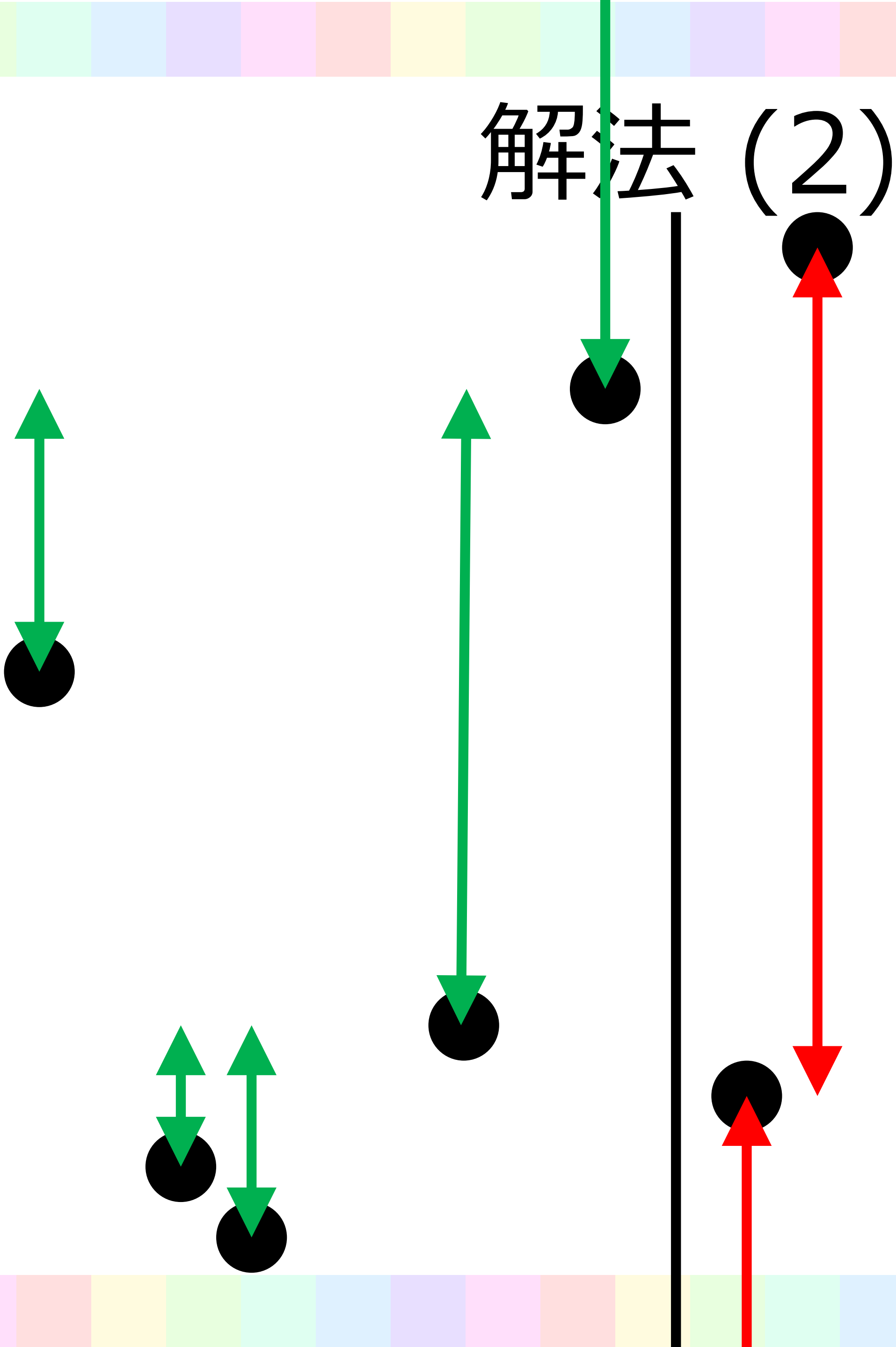
解法 (2)



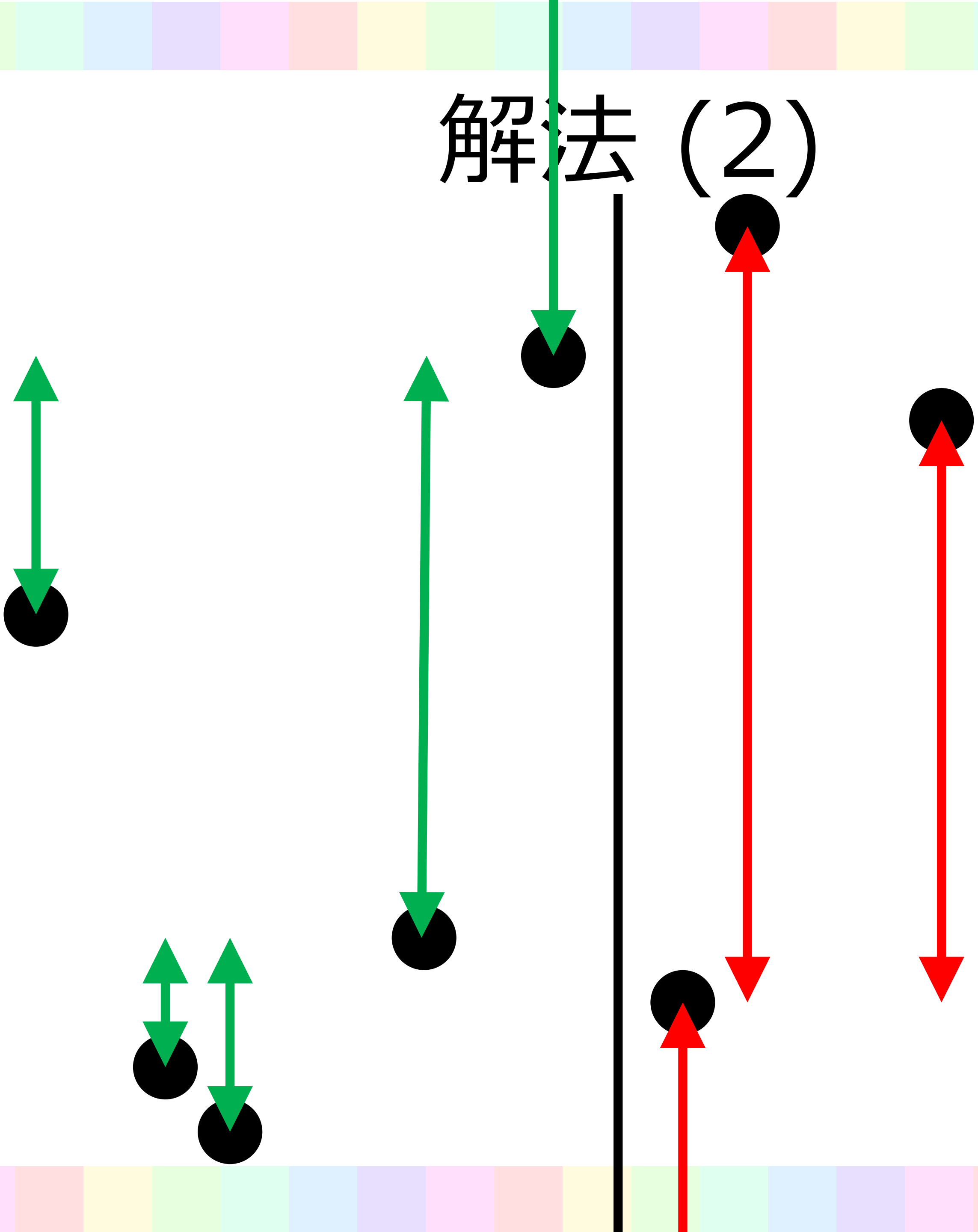
解法 (2)



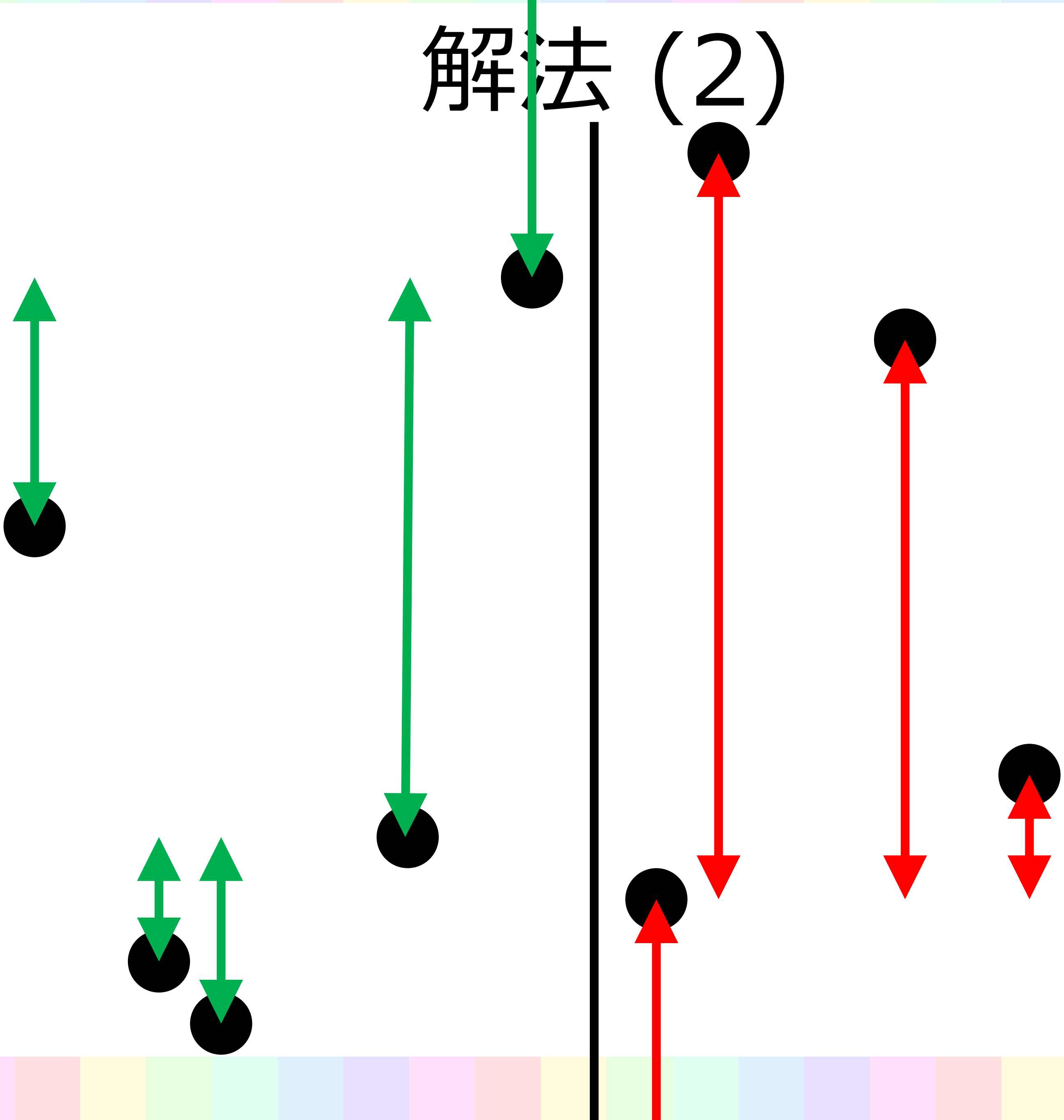
解法 (2)



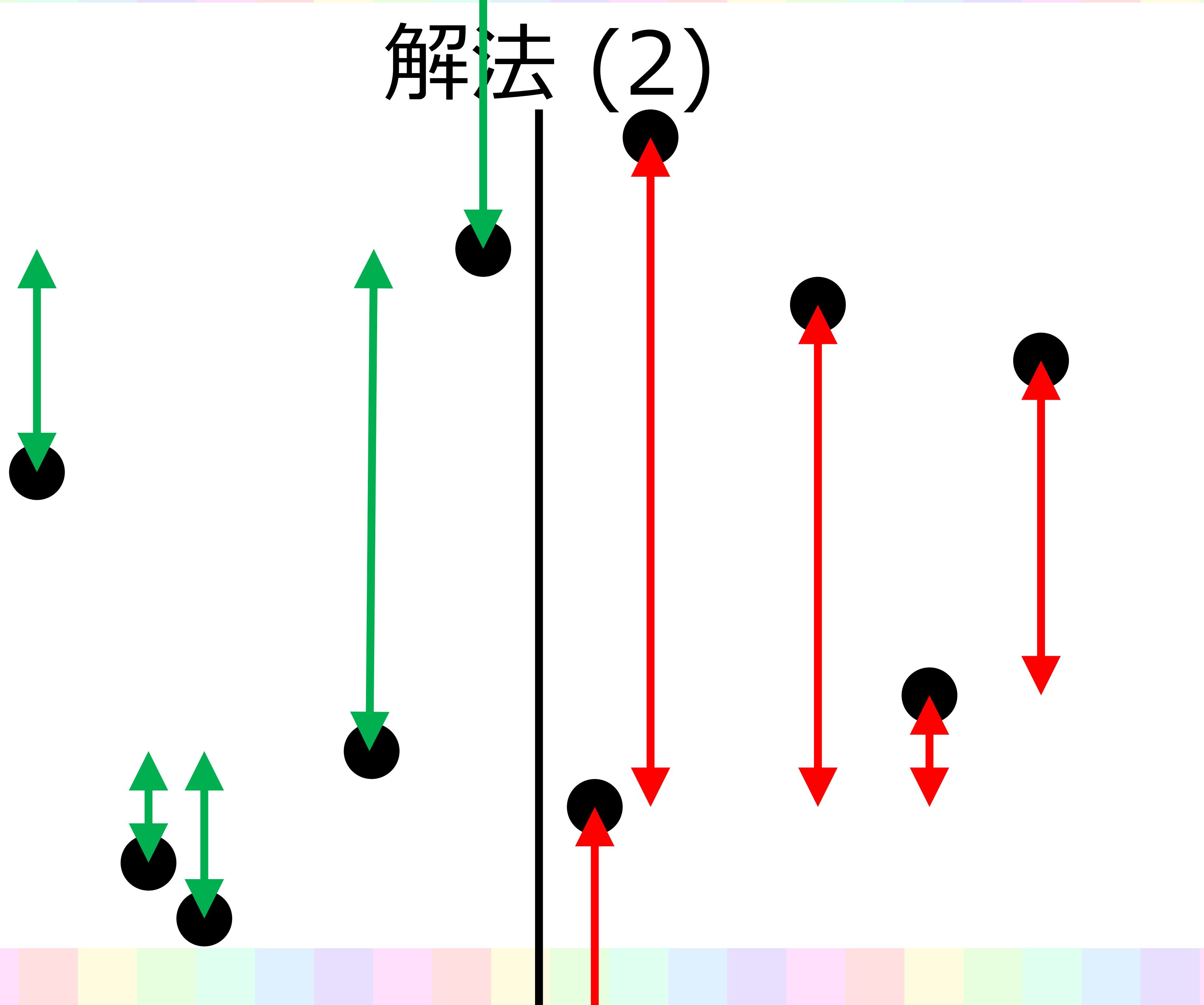
解法 (2)



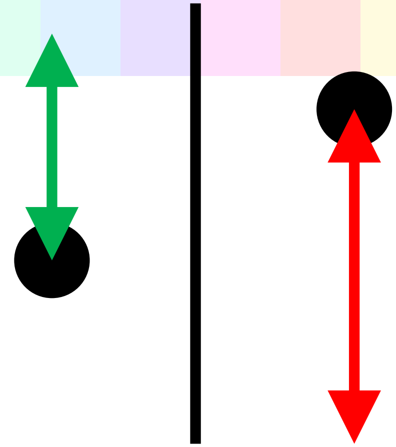
解法 (2)



解法 (2)



解法 (2)



- こういう区間の組を数える：
- 左側 $[a_i, b_i]$，右側 $[c_j, d_j]$ とする
  - OK な条件は $c_j \leqq a_i \leqq d_j \leqq b_i$
- 下端 (a_i や c_j) の値でソート

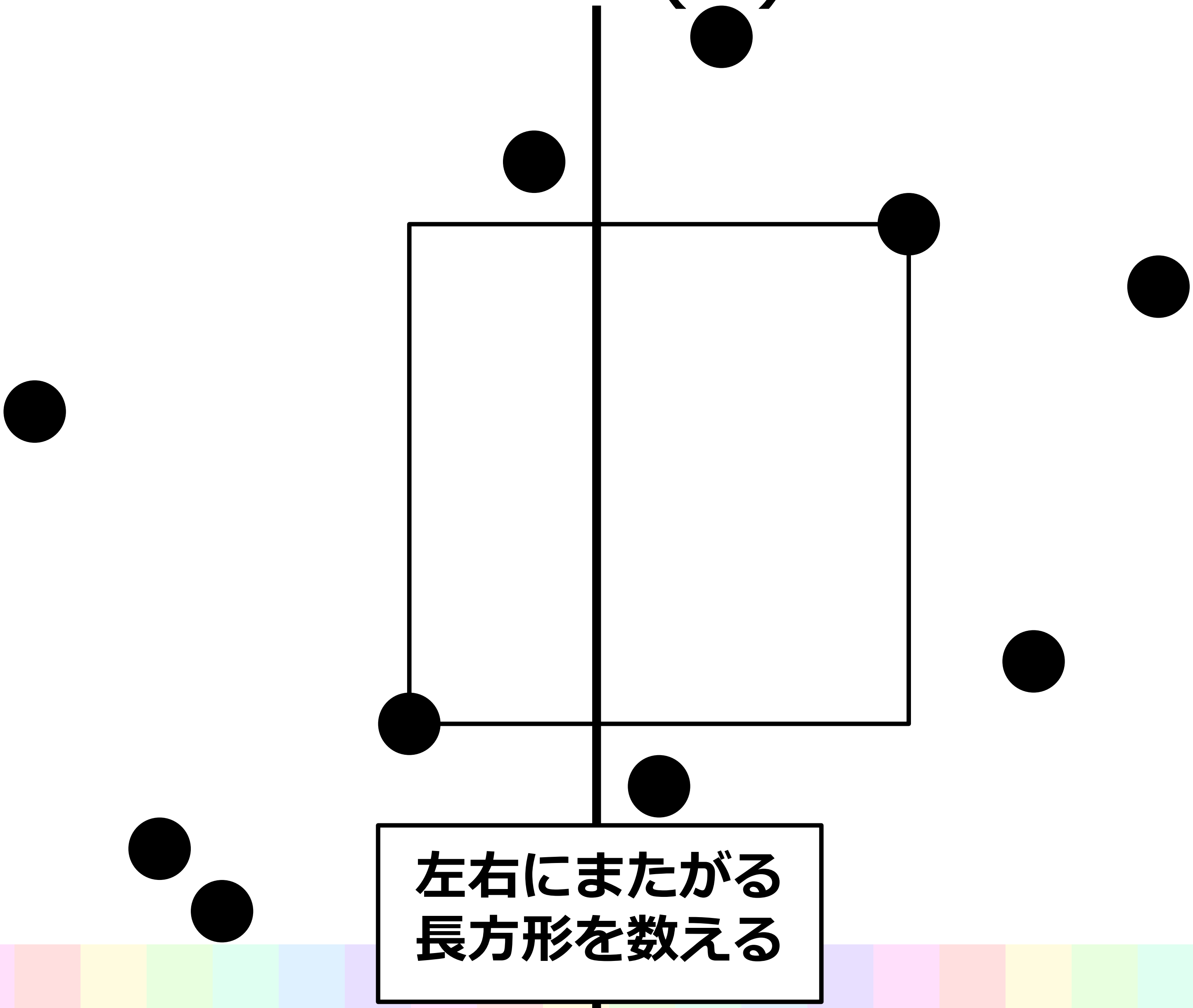
1. 区間 $[c_j, d_j]$ が来たら，d_j を追加する
2. 区間 $[a_i, b_i]$ が来たら，追加された d_j のうち $a_i \leqq d_j \leqq b_i$ なるものを数える

解法 (2)

- 区間を求めるのは std::set とその lower_bound などを利用
- y 座標を座標圧縮しておけば Segment Tree や BIT で数えられる
- $O(N \log N)$ 時間

- 全体で $O(N(\log N)^2)$ 時間
  - 実装：比較的かんたん

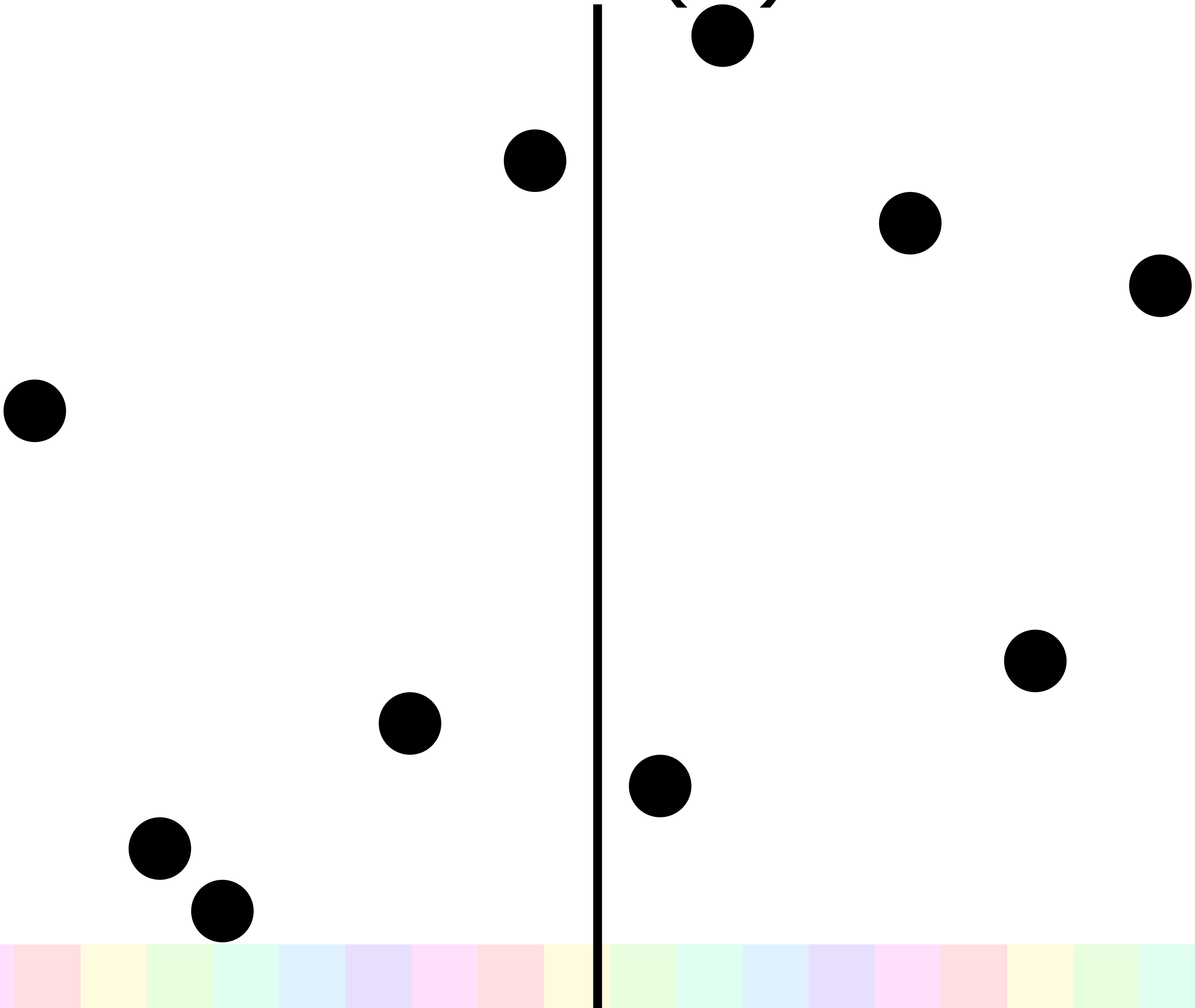
解法 (3)



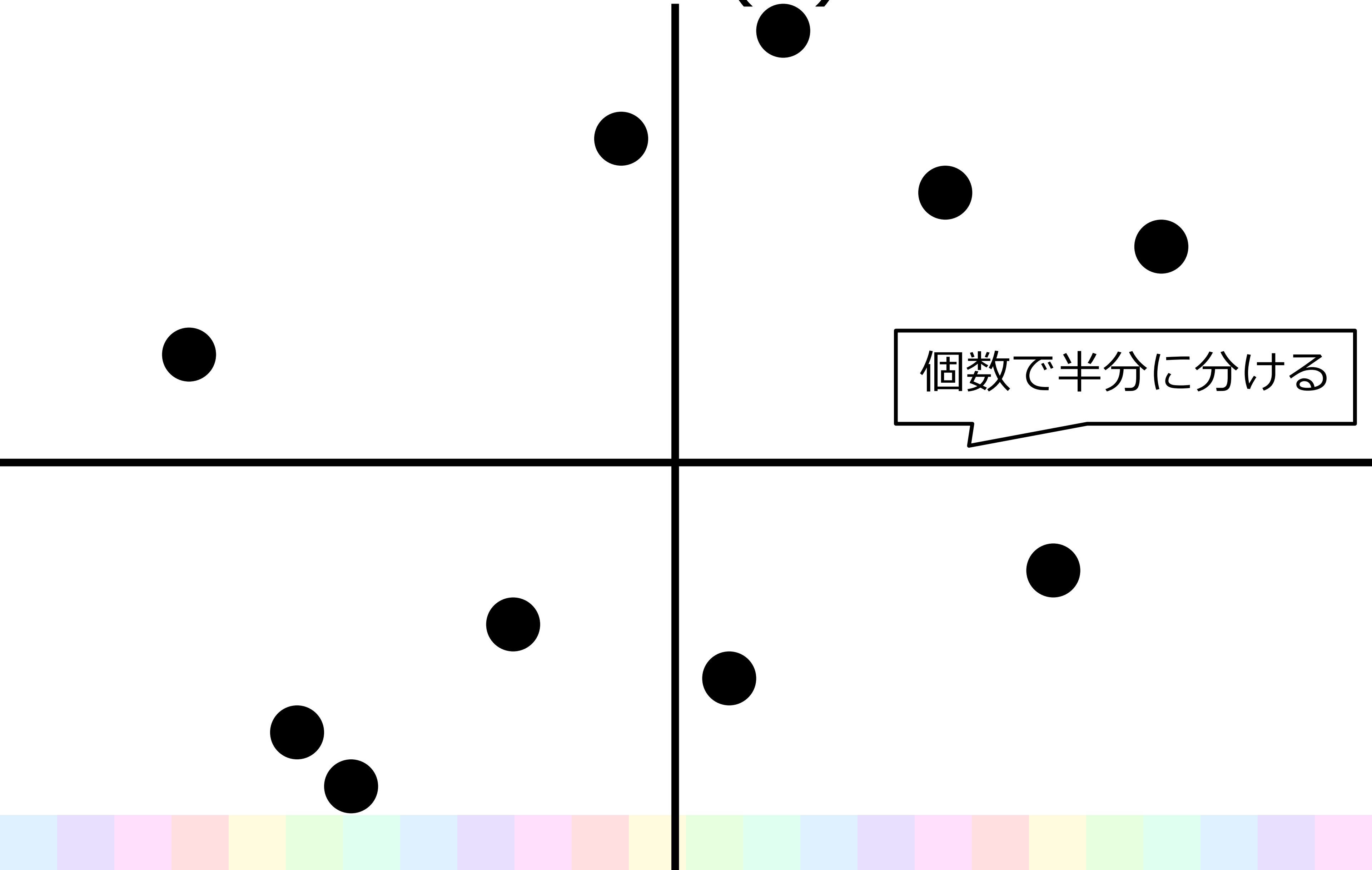
分 割 統 治 法

–Divide and Conquer–

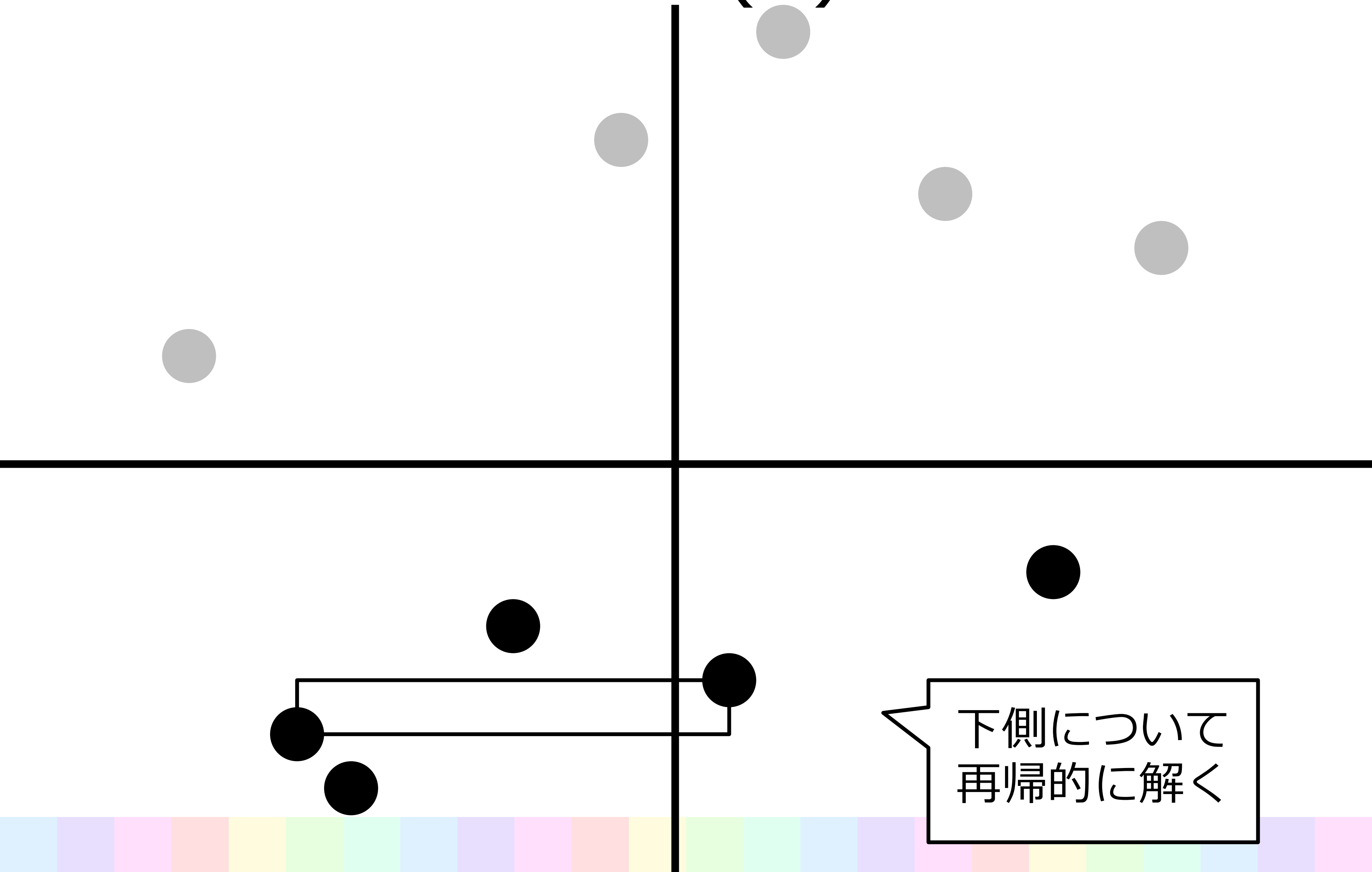
解法 (3)



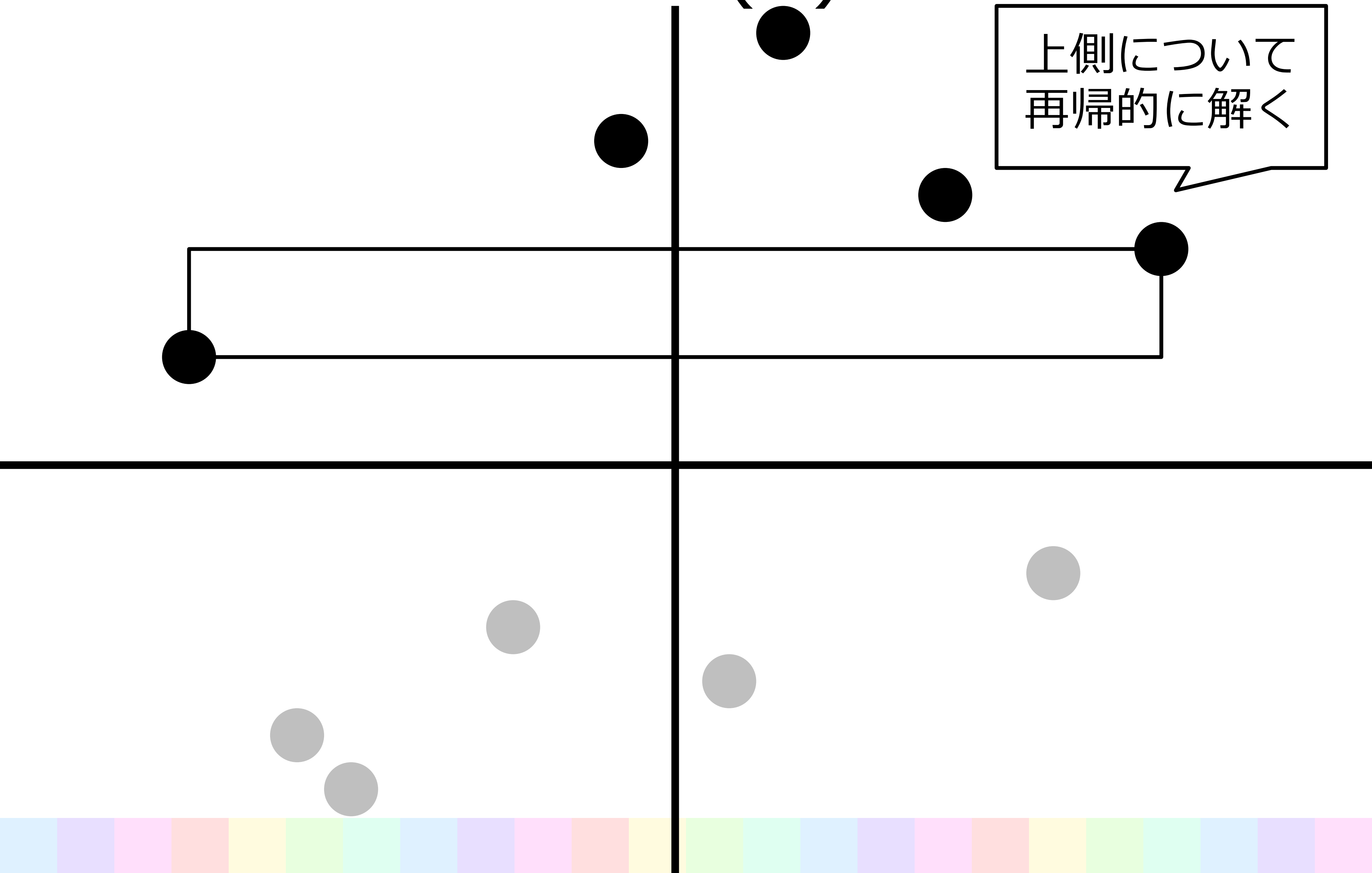
解法 (3)



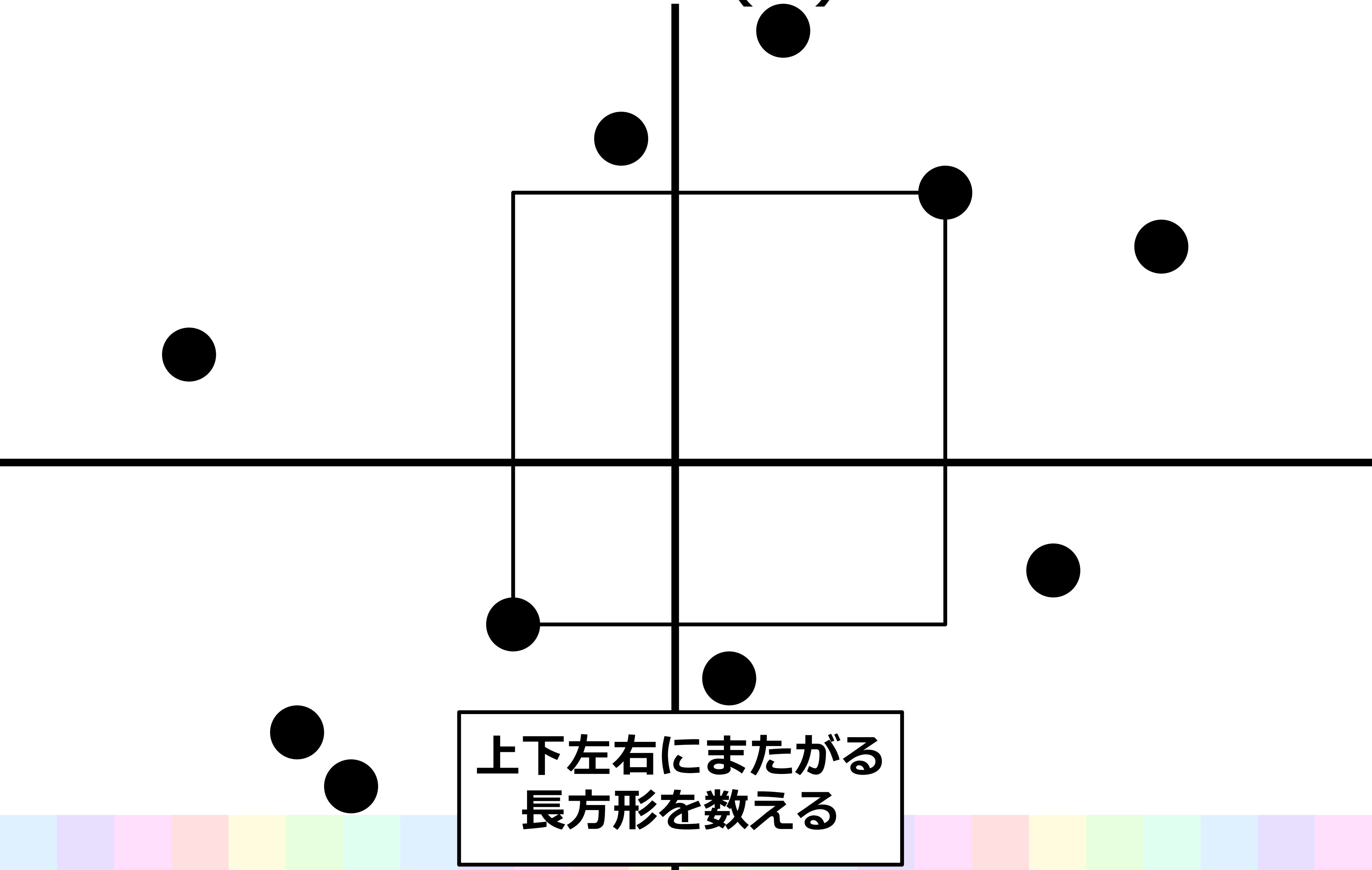
解法 (3)



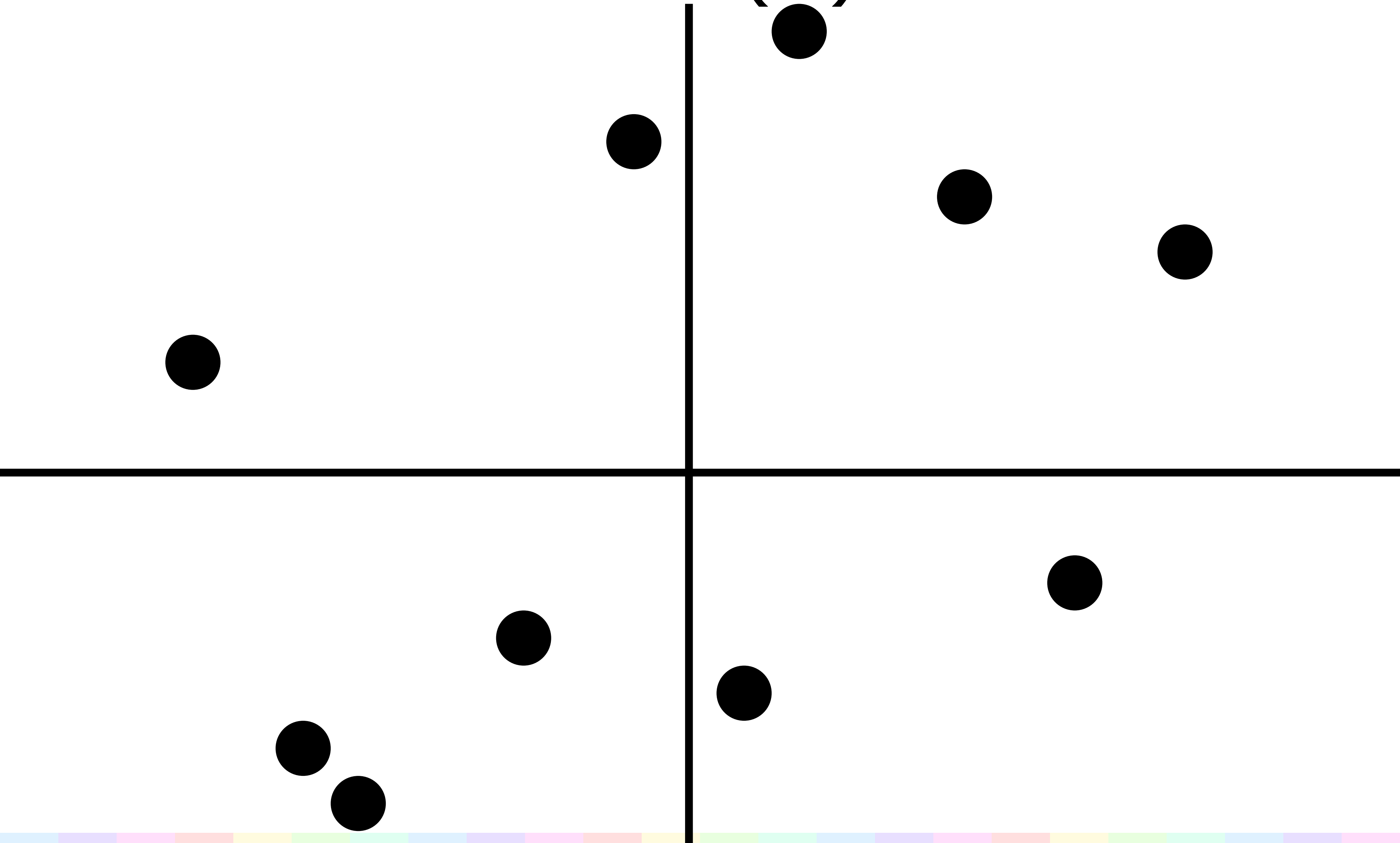
解法 (3)



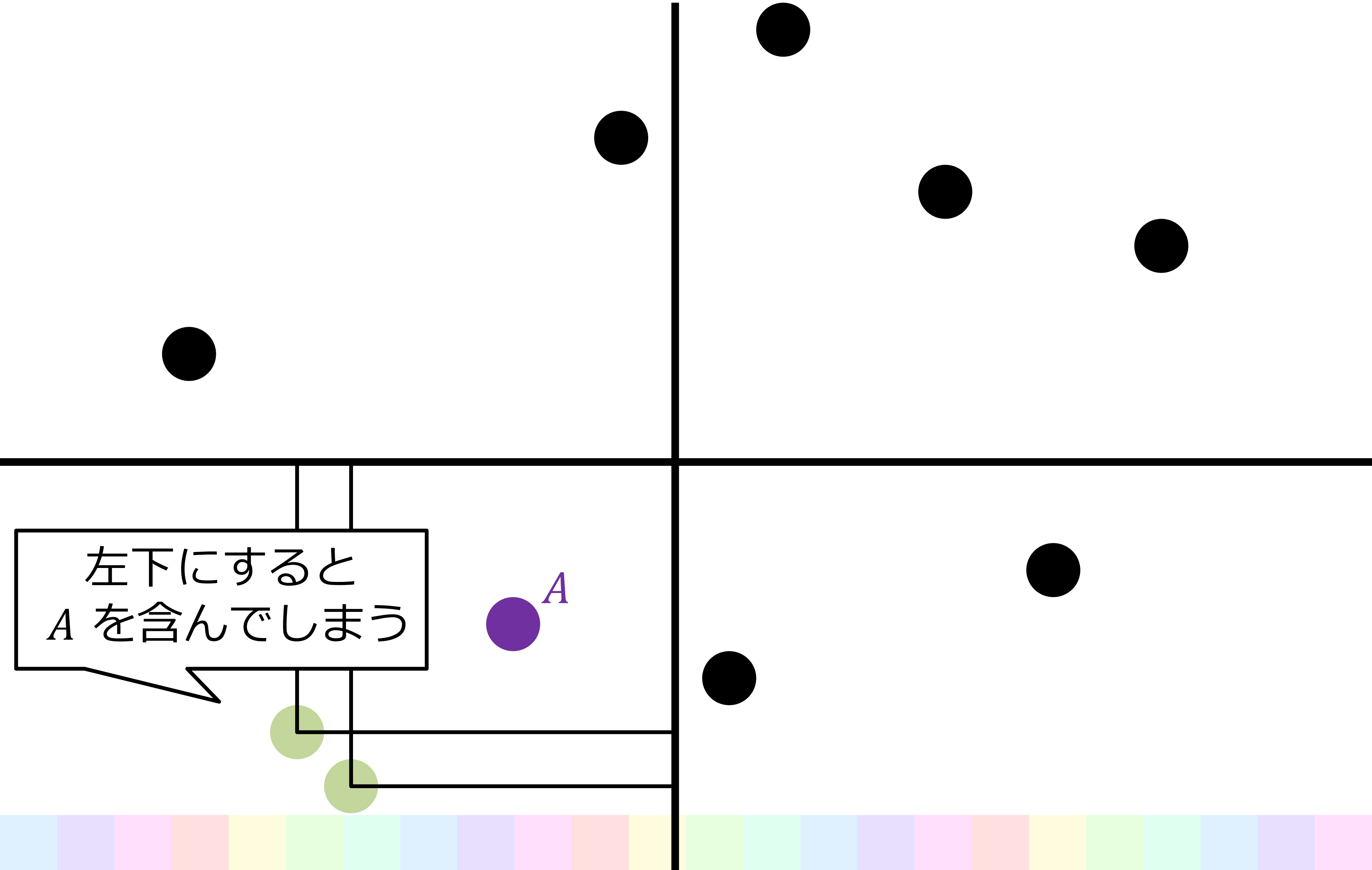
解法 (3)



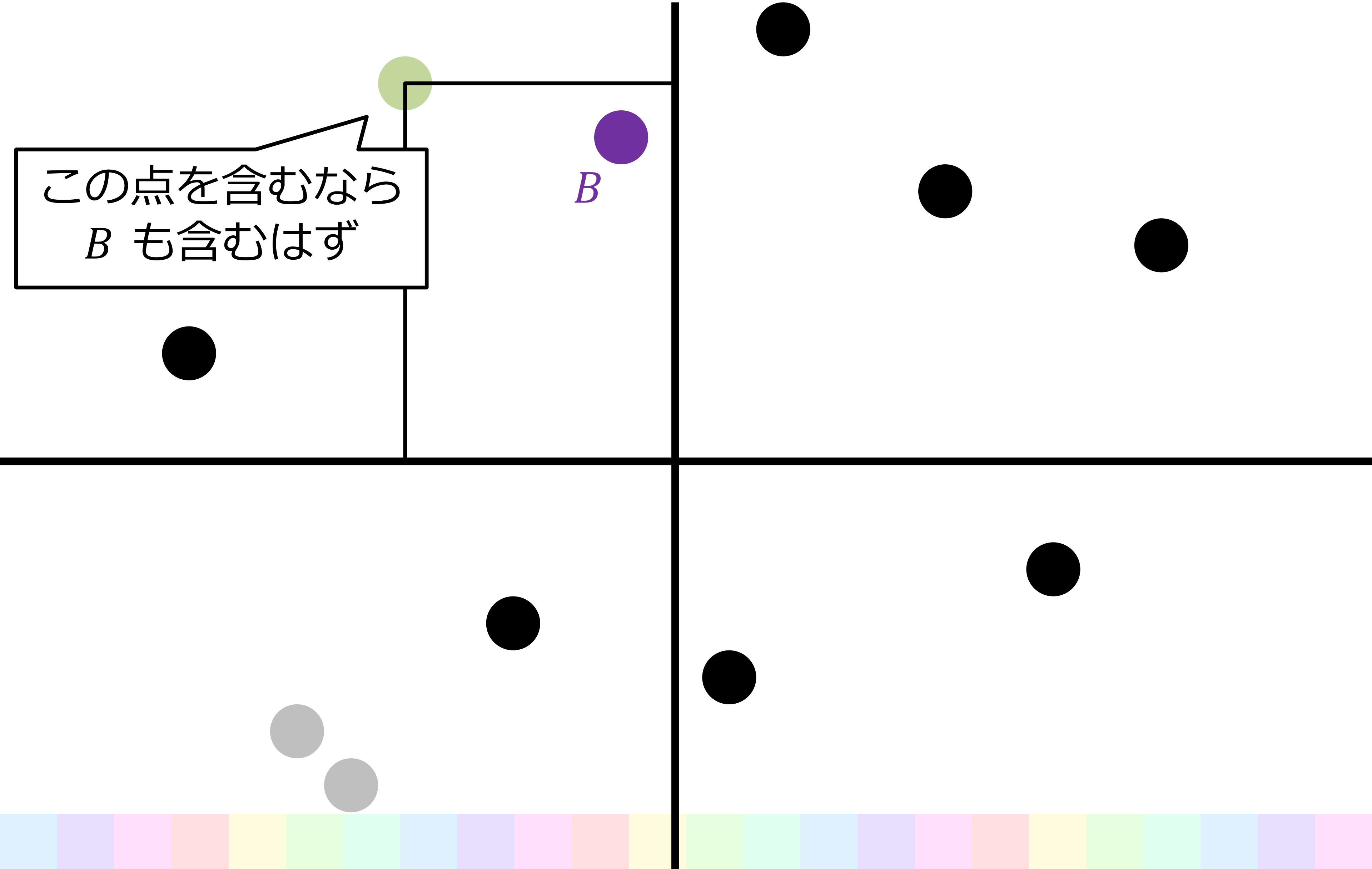
解法 (3)



不要な点を削除



不要な点を削除

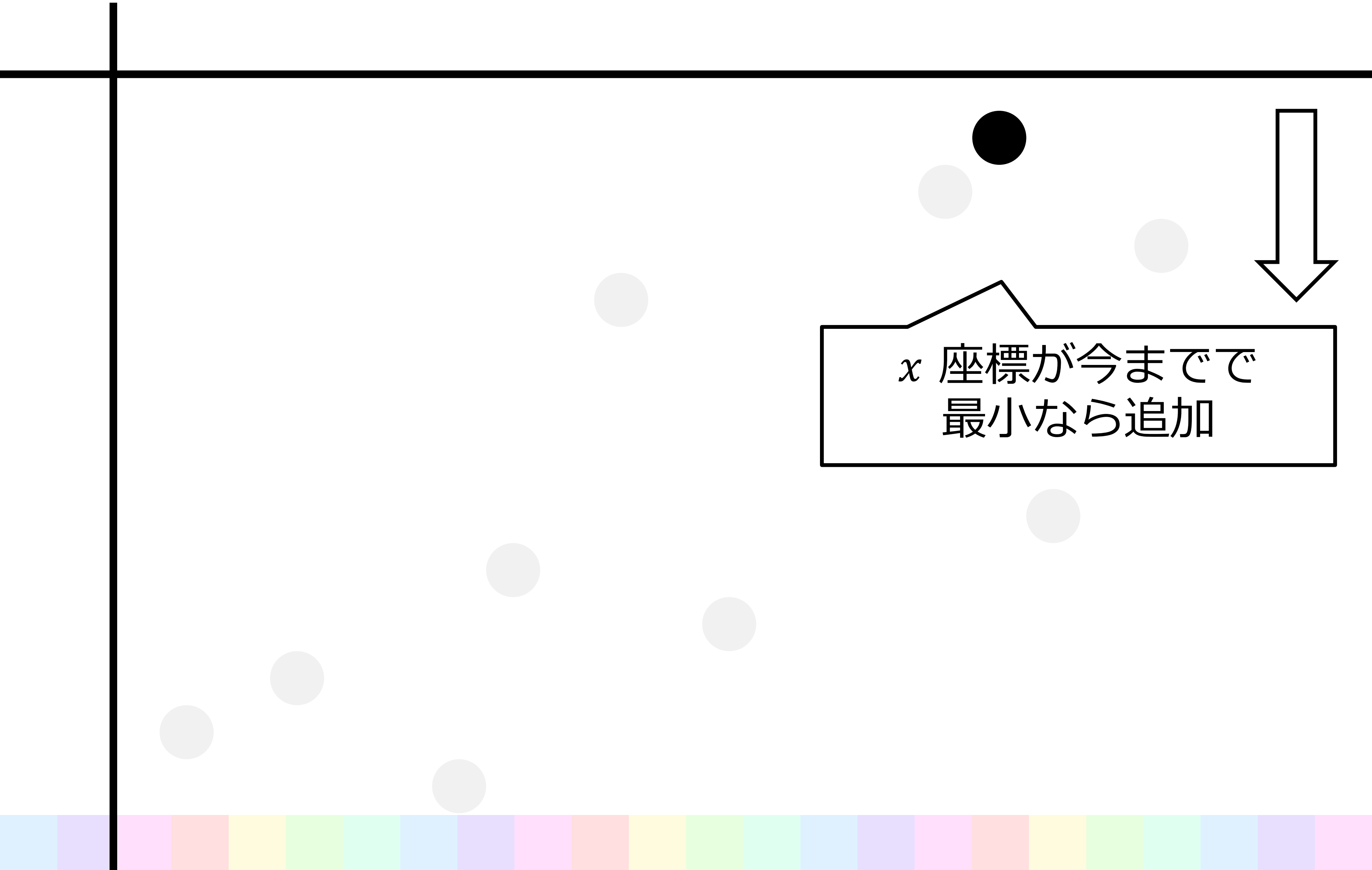


不要な点を削除

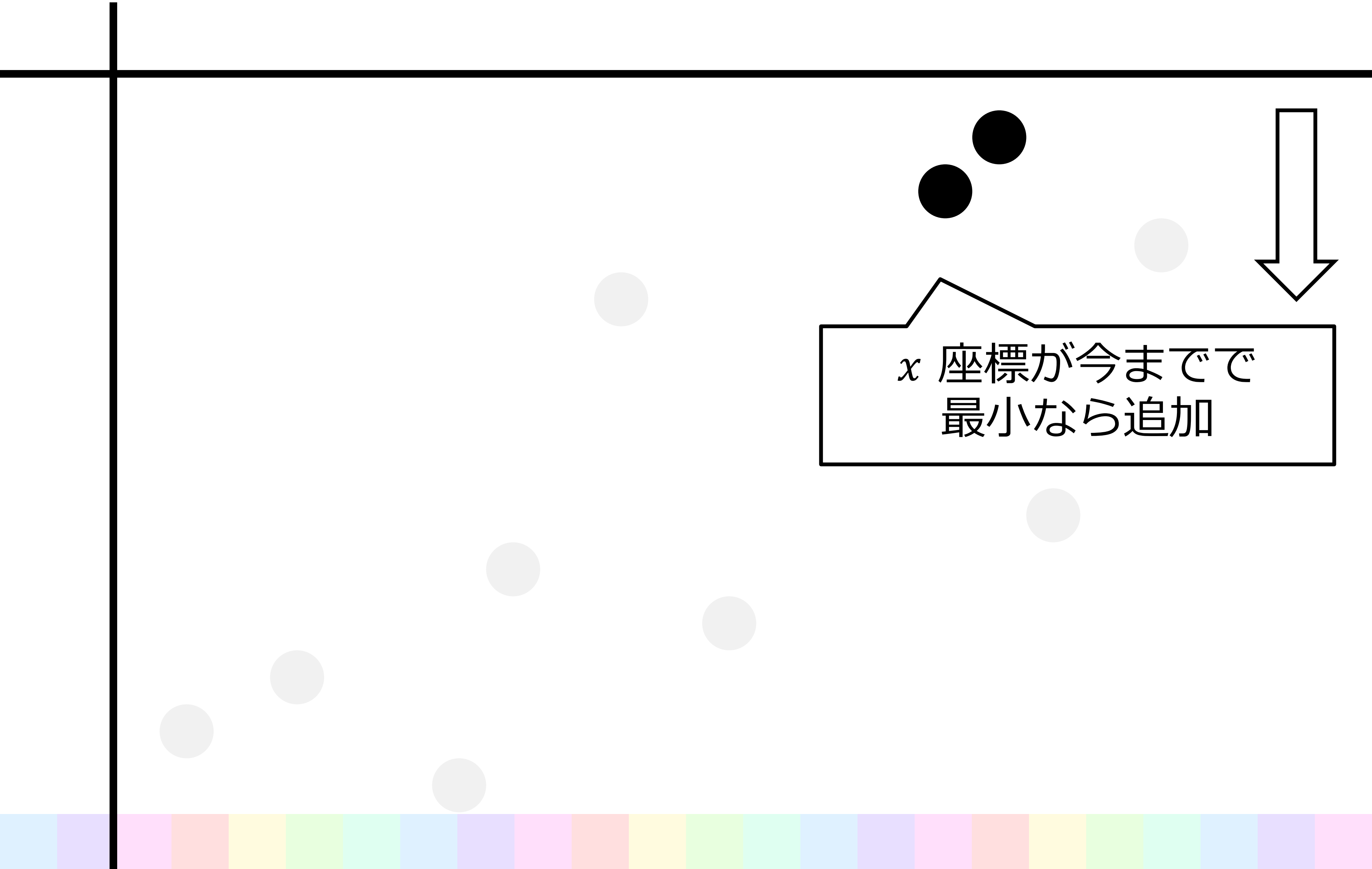
- 4 領域とも，「中央のほうに並んでいる点たち」だけ残せばよい
  - 正確には，それと中央線の交点とで長方形領域を作ったとき，他の点が含まれないような点たち
  - 残った点は，y 座標の小さい順に並べると，x 座標は小さい順 or 大きい順になる

不要な点を削除

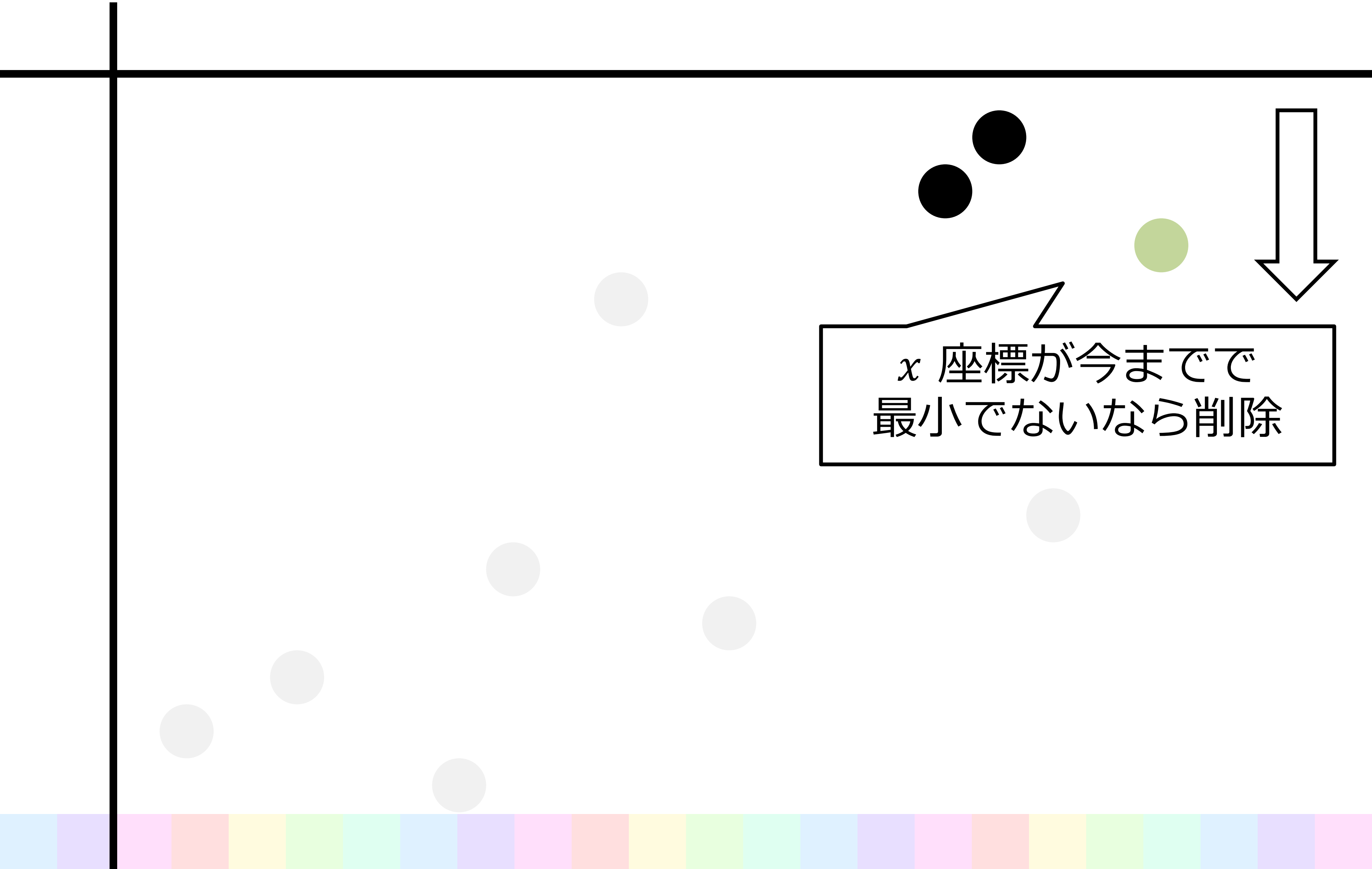
不要な点を削除



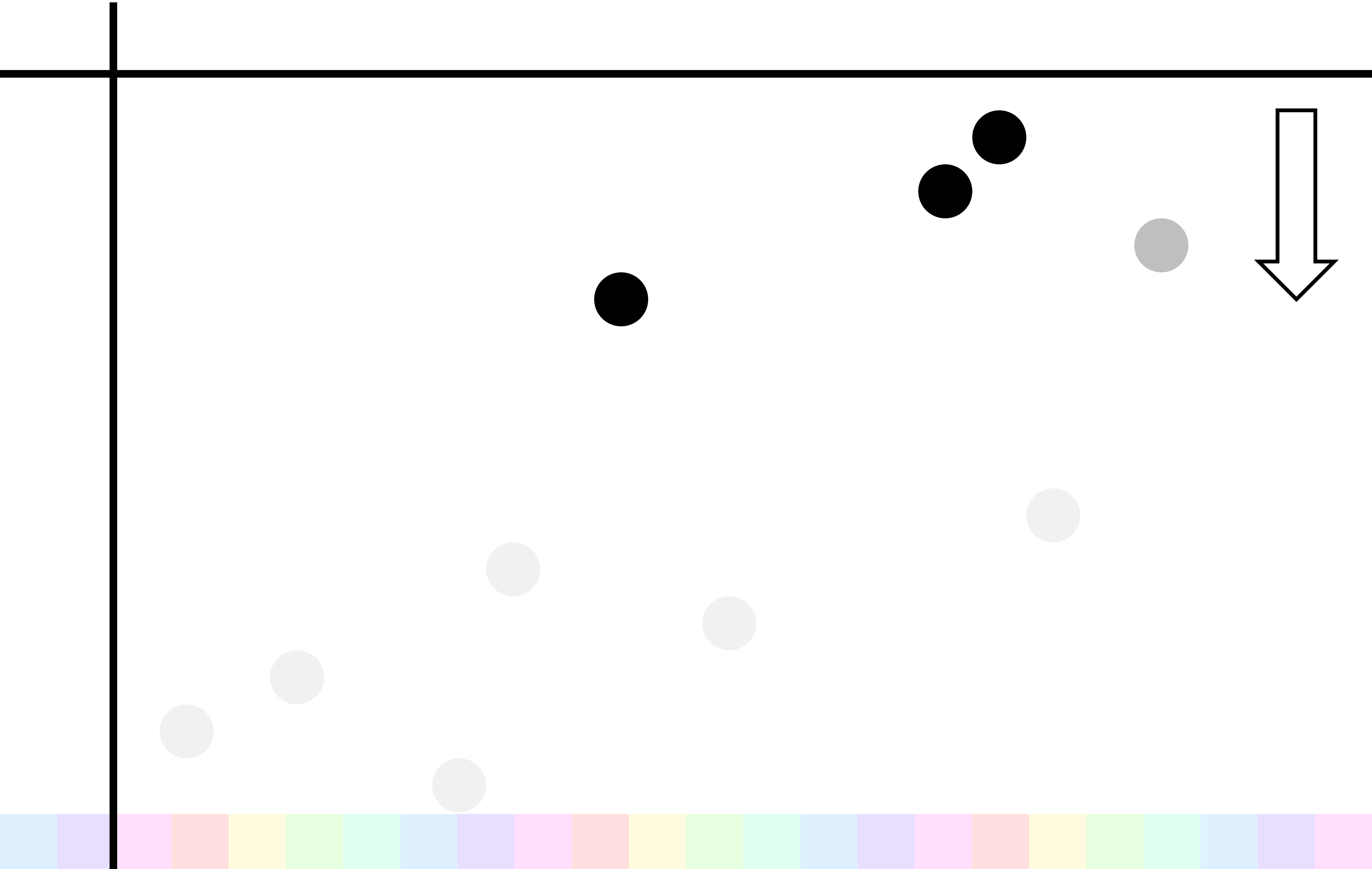
不要な点を削除



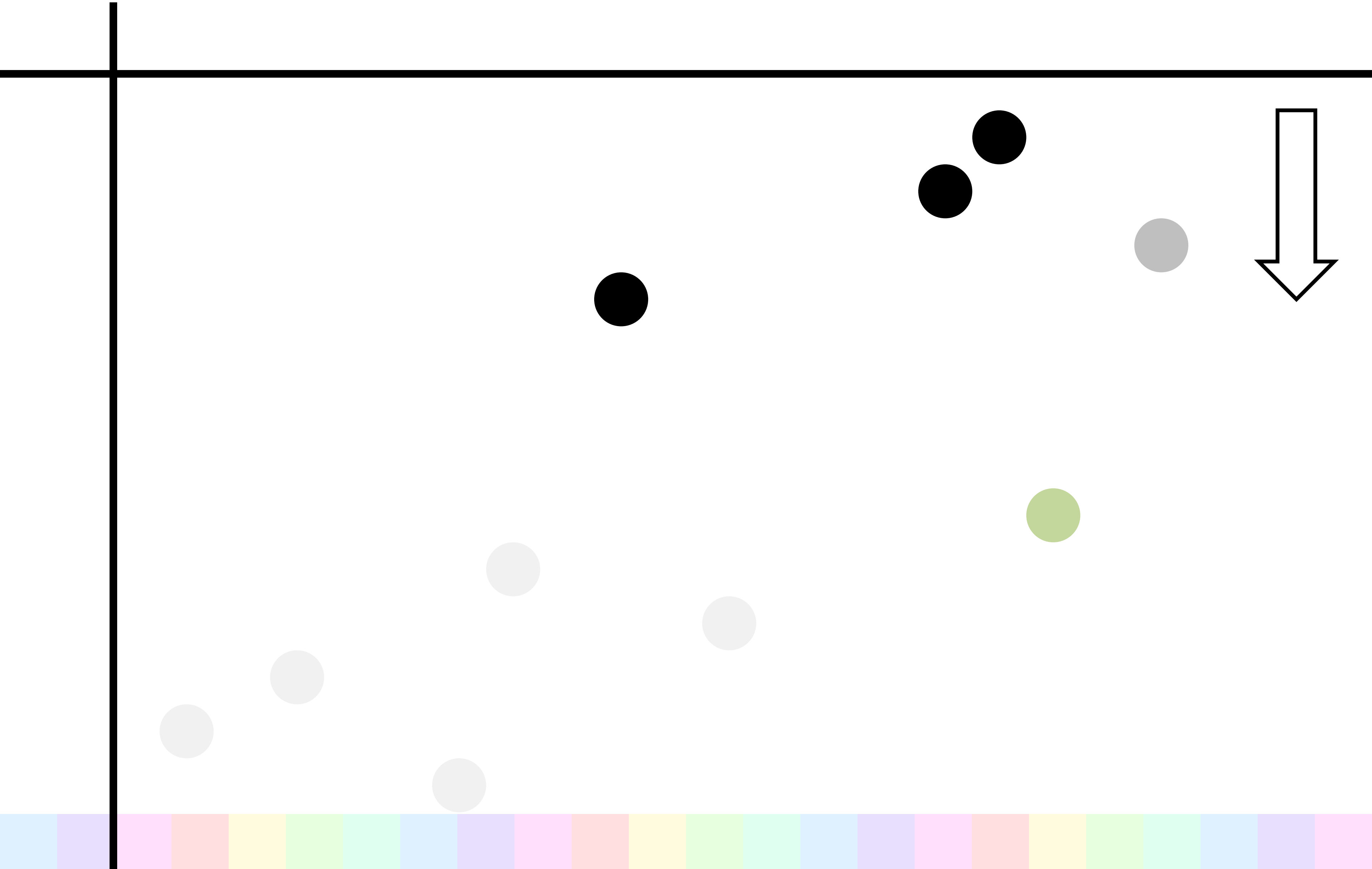
不要な点を削除



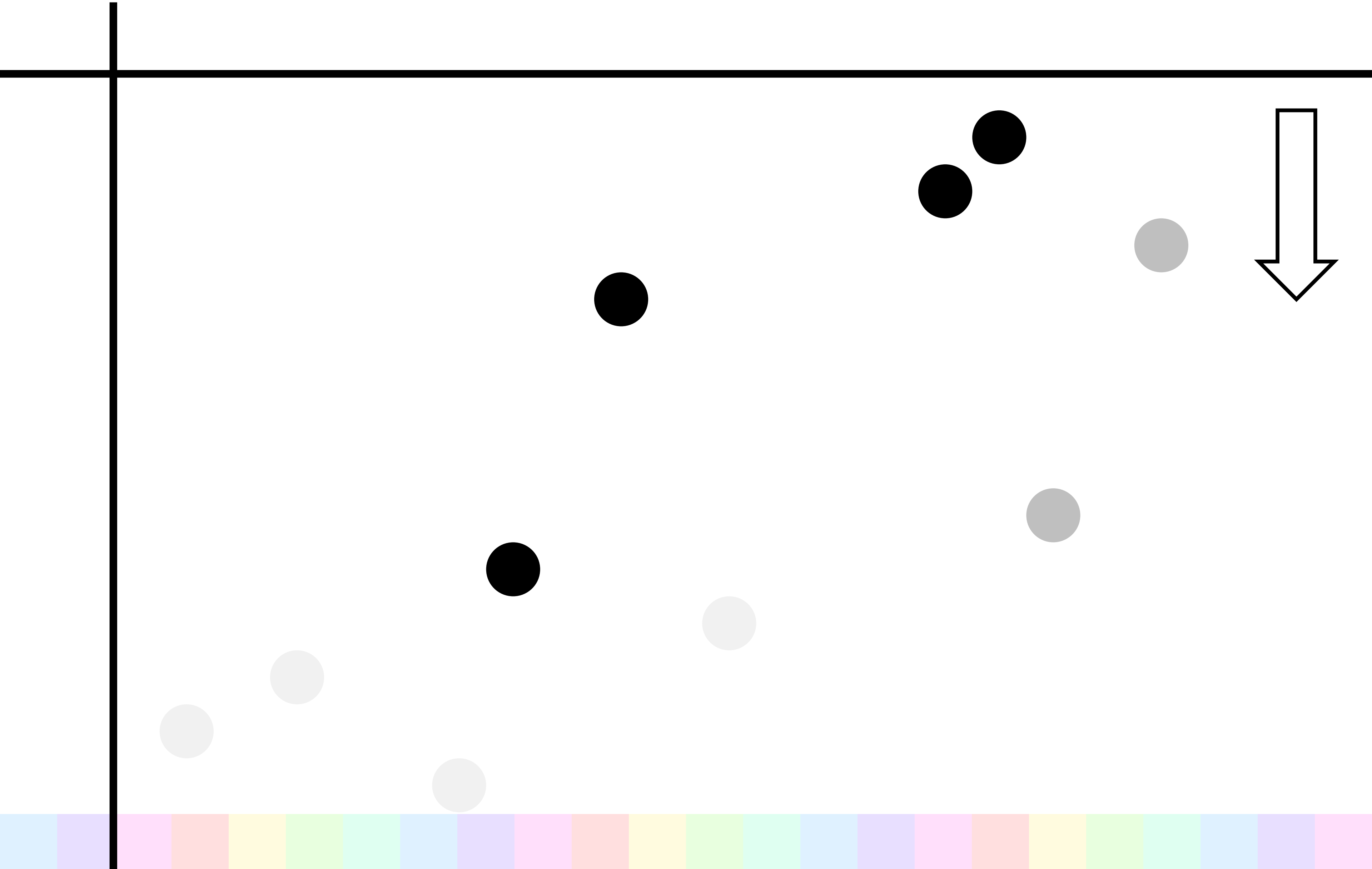
不要な点を削除



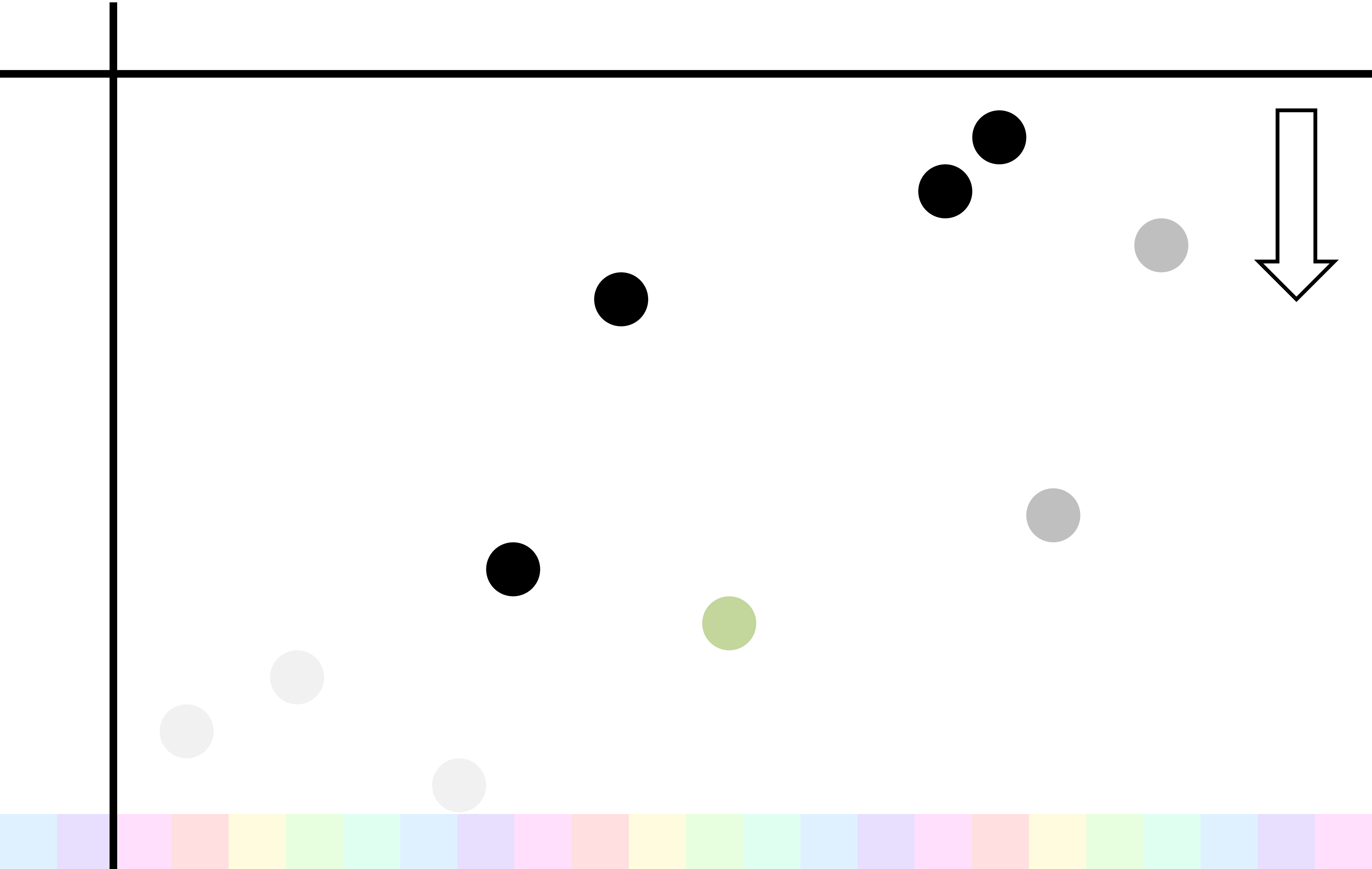
不要な点を削除



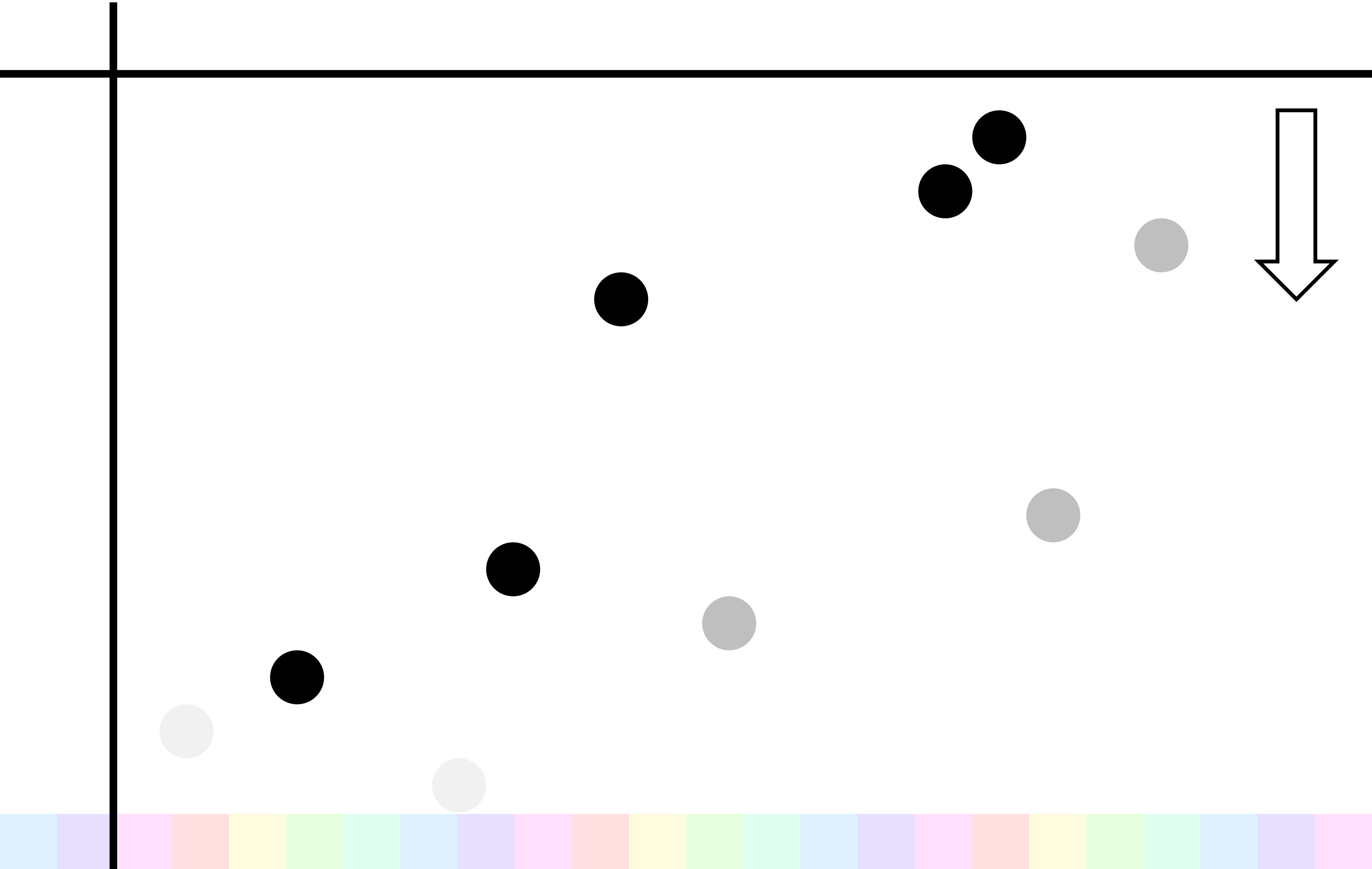
不要な点を削除



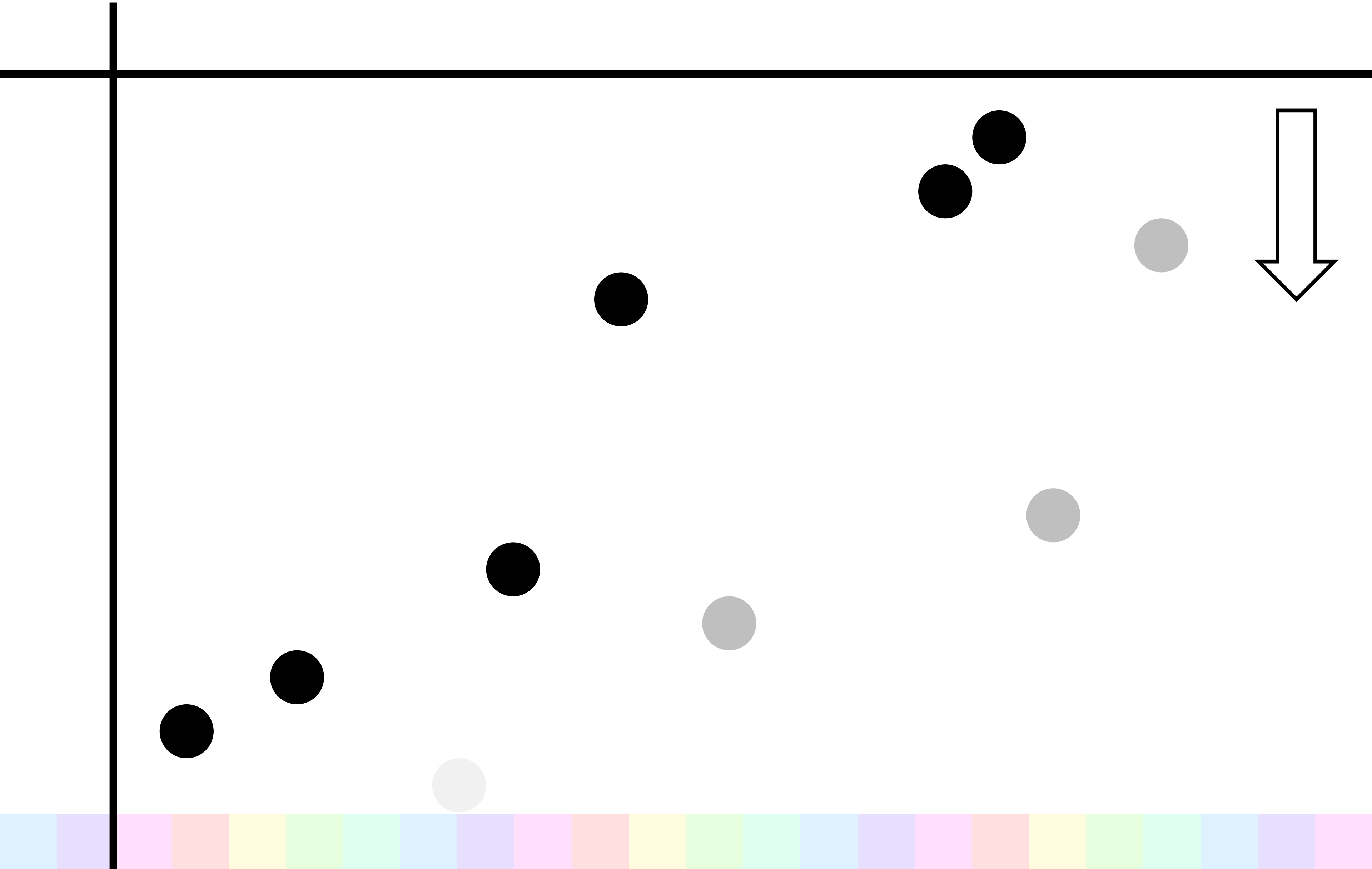
不要な点を削除



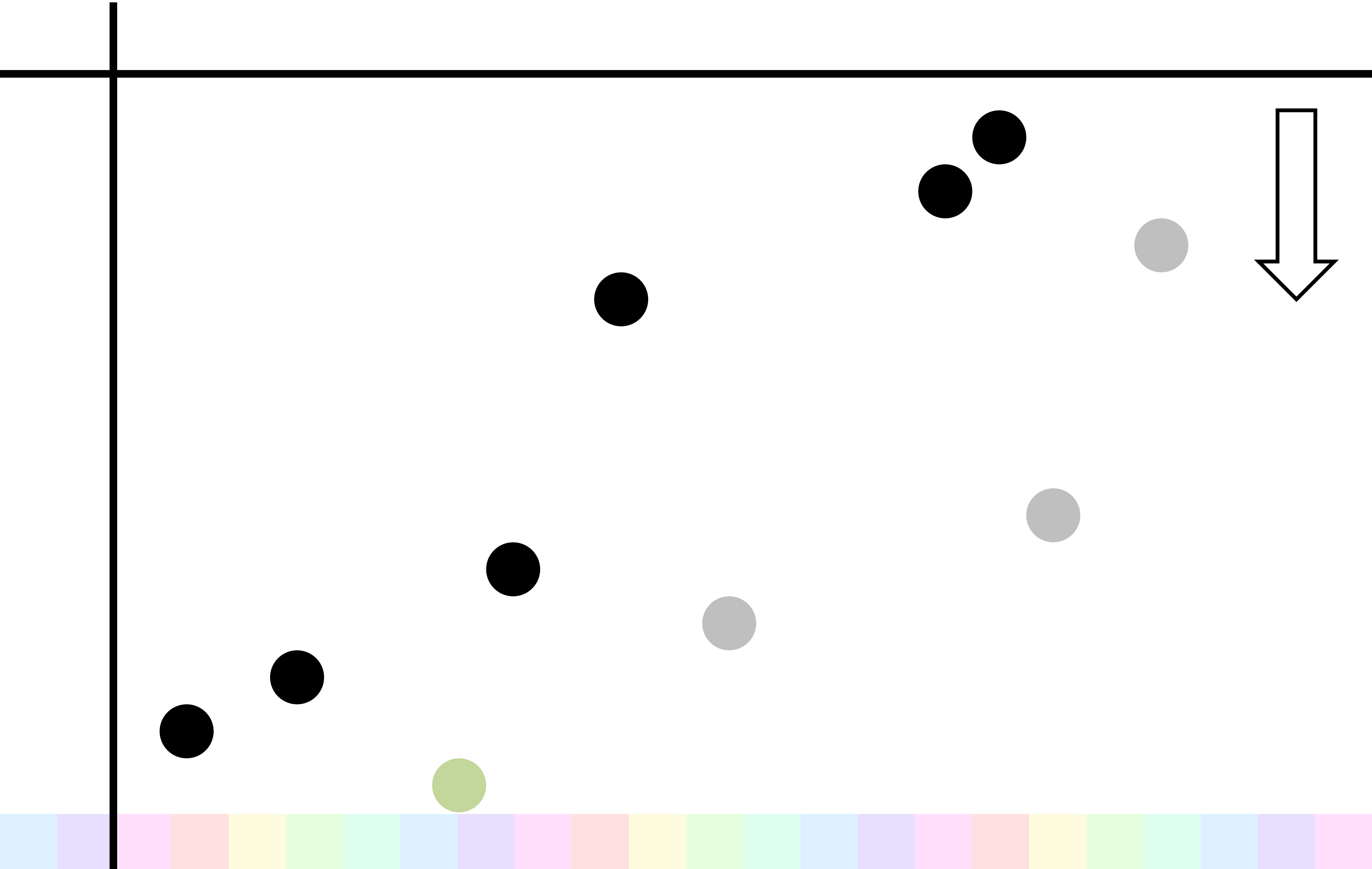
不要な点を削除



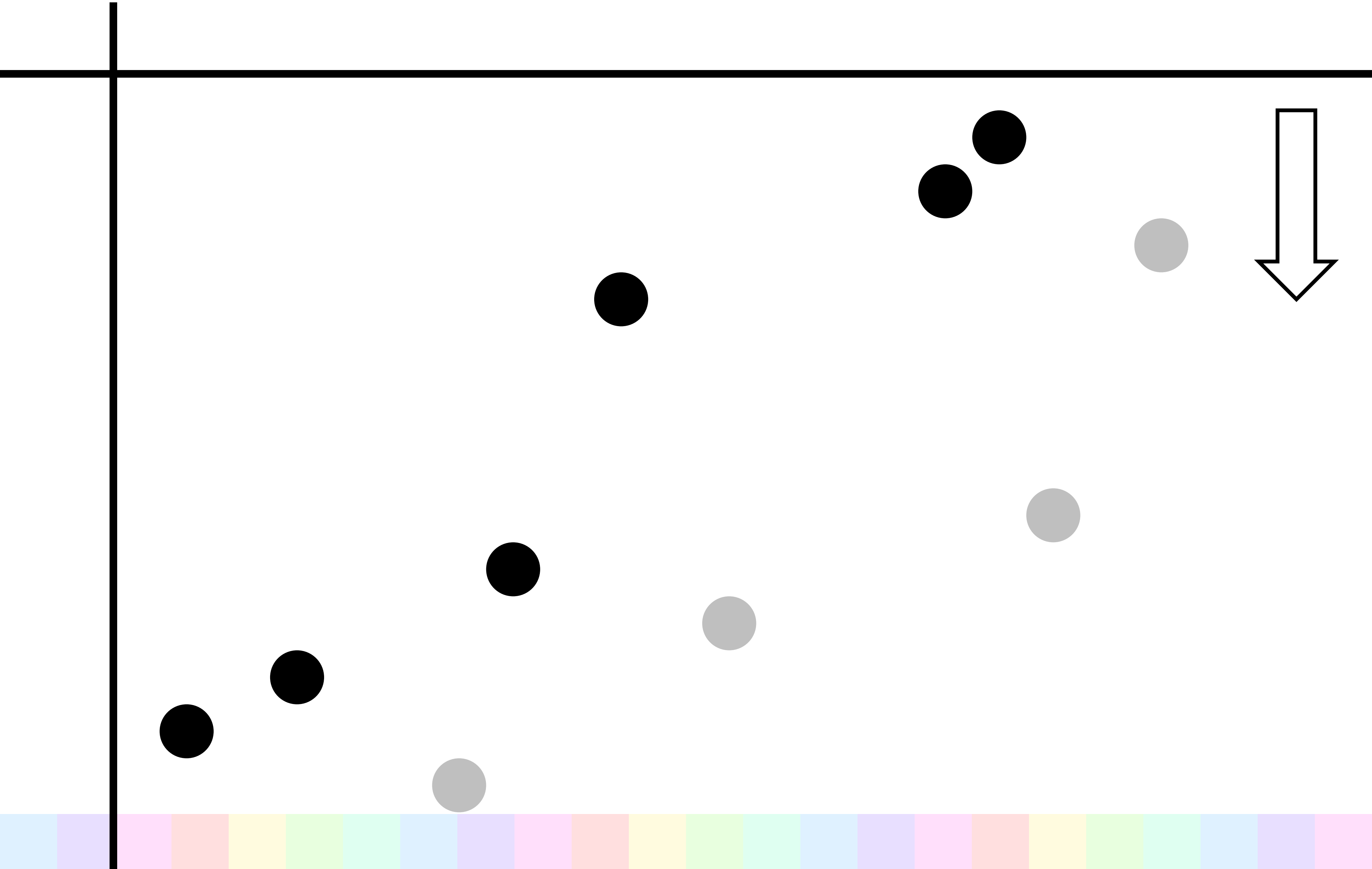
不要な点を削除



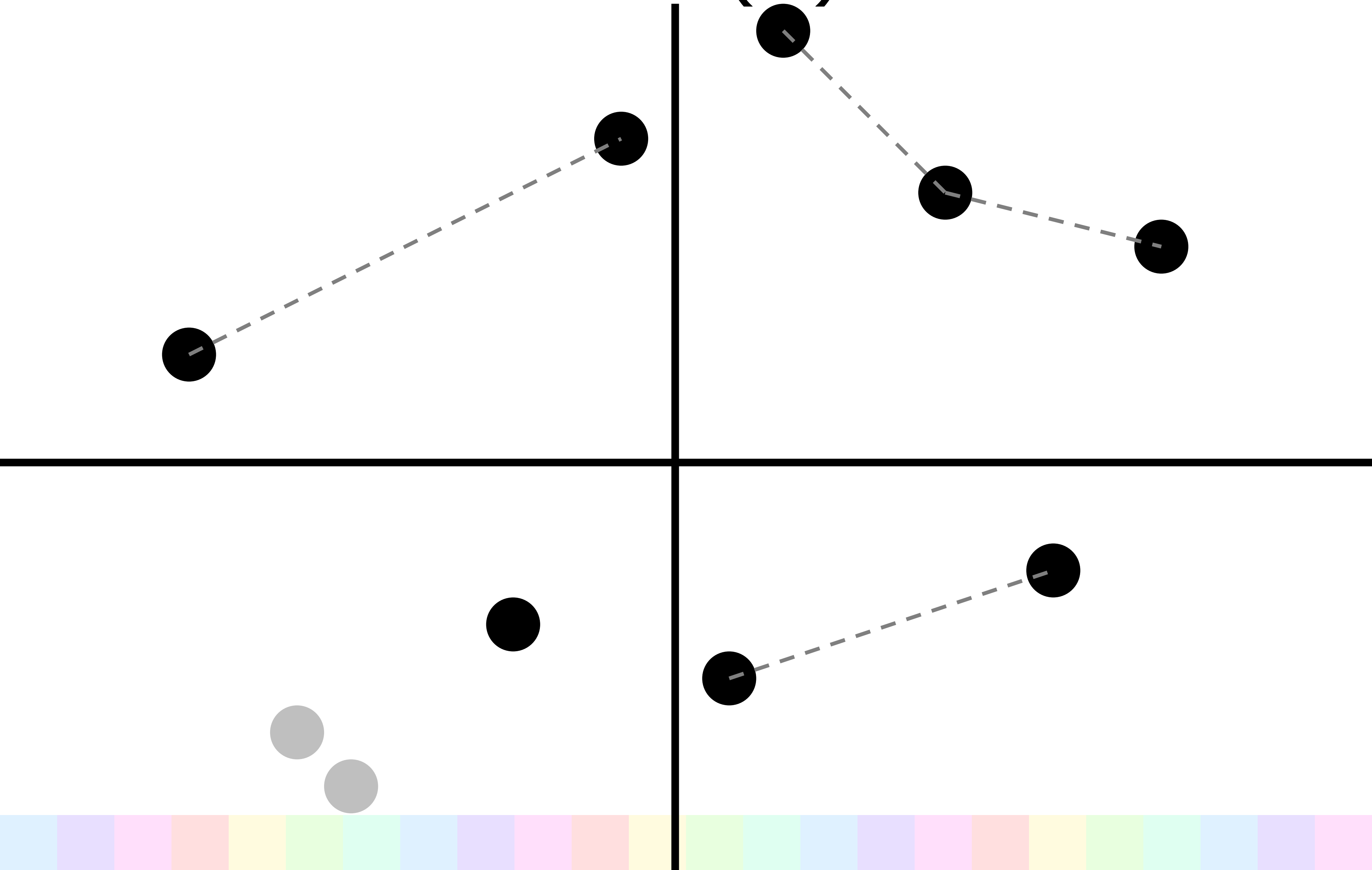
不要な点を削除



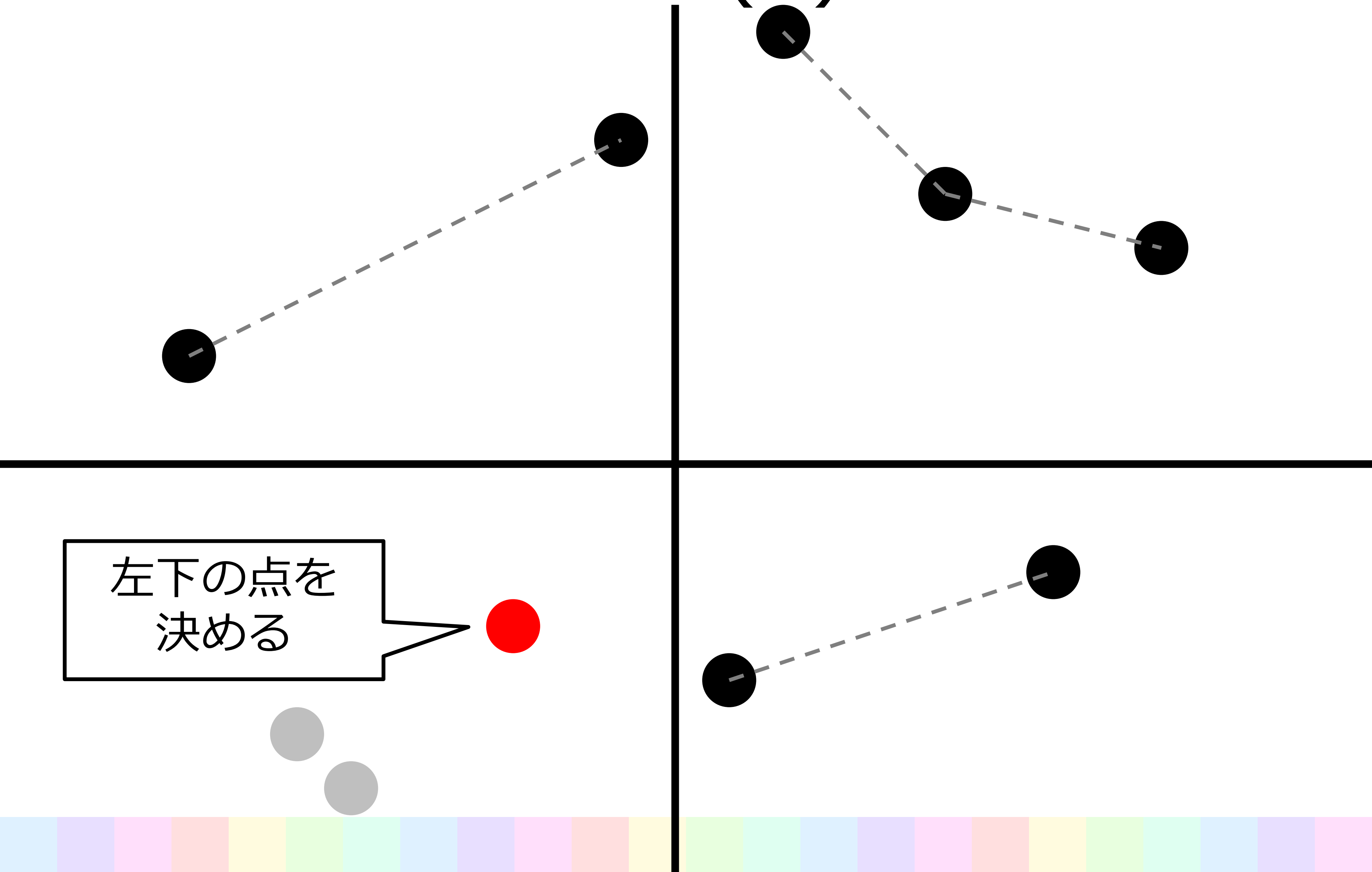
不要な点を削除



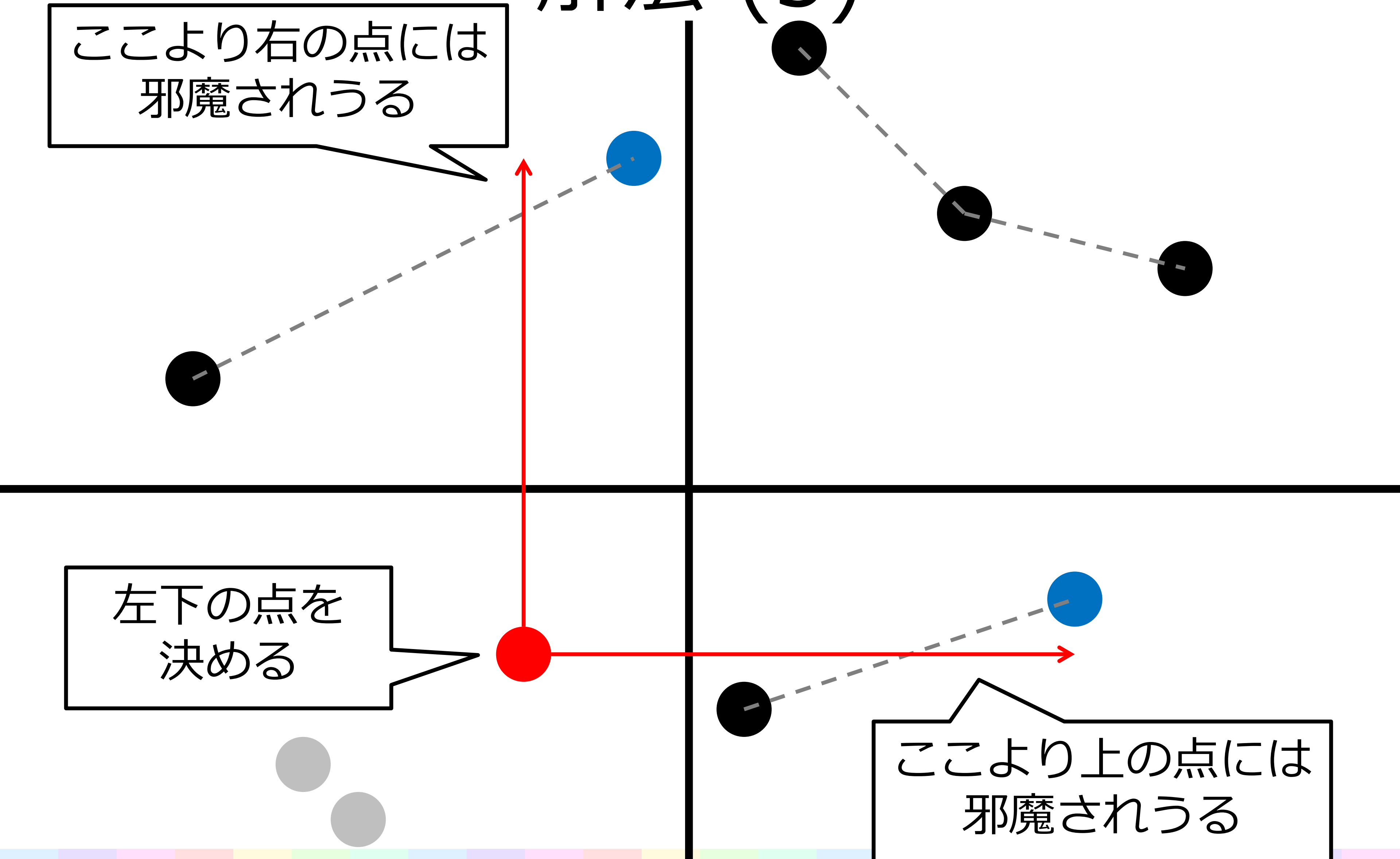
解法 (3)



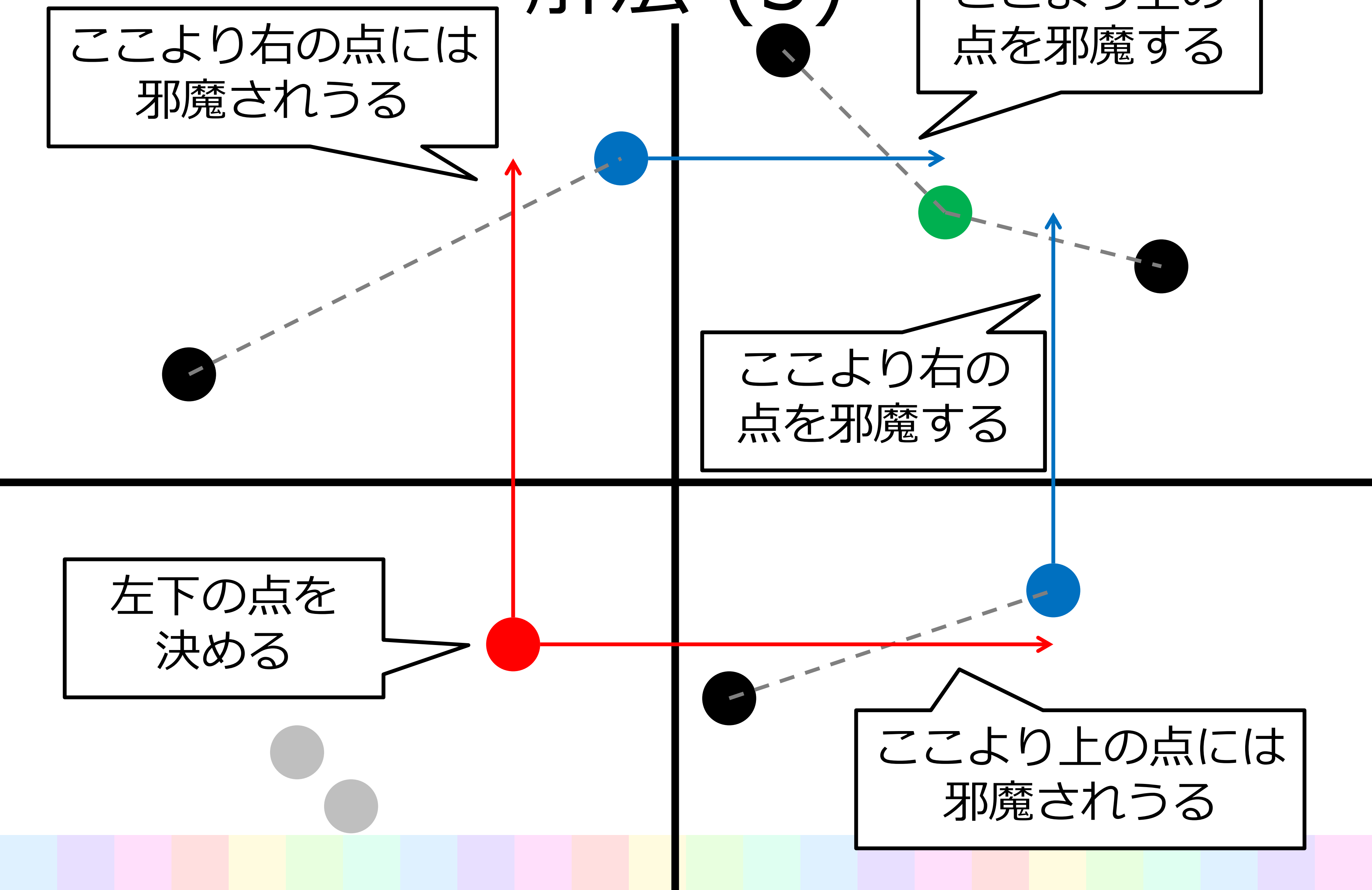
解法 (3)



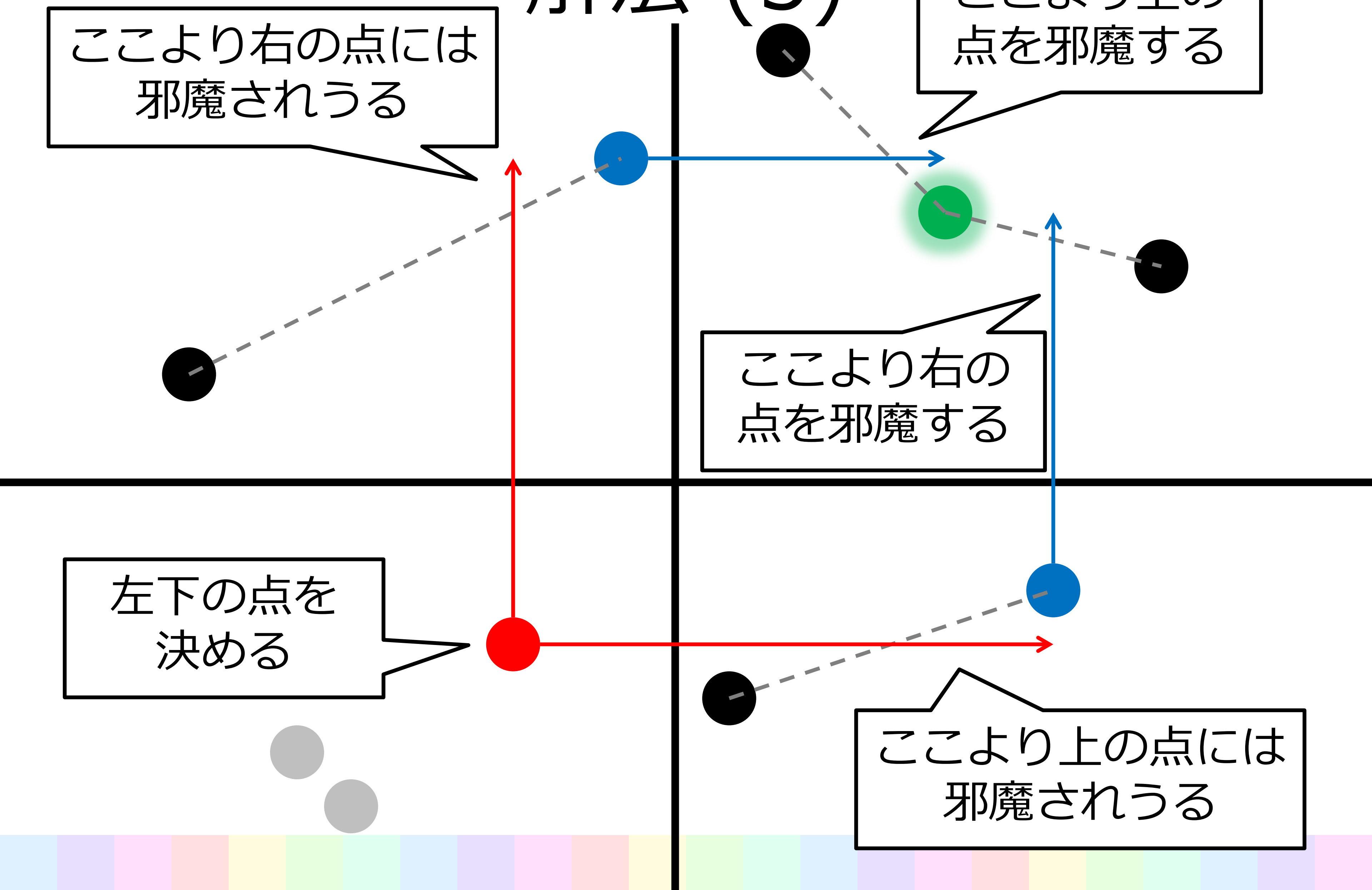
解法 (3)



解法 (3)


ここより右の点には
邪魔されうる
ここより上の
点を邪魔する
ここより右の
点を邪魔する
左下の点を
決める
ここより上の点には
邪魔されうる

解法 (3)


ここより右の点には
邪魔されうる
ここより上の
点を邪魔する
ここより右の
点を邪魔する
左下の点を
決める
ここより上の点には
邪魔されうる

解法 (3)

- 左下の点を決める
- 4 回二分探索する
- 右上としてとれる点の範囲がわかる

- $O(N \log N)$ 時間
- 全体で $O(N(\log N)^3)$ 時間
  - 実装：不等号や x, y で混乱しがちな単純作業

解法 (3)

- 尺取法が使える
  - 左下の点を動かしたとき，二分探索する先が単調に変化するから
- ソートも分割統治のついでにできる

- $O(N)$ 時間
- 全体で $O(N(\log N)^2)$ 時間
  - 速度・実装ともあまり変わりません

分割統治法に関して

- 分割統治してみたら簡単な問題に変わらないか？は重要なアイデア
  - しかも直接的には気づきにくい
- 典型的な例
  - マージソート
  - 最近点対
    - 点集合から距離がもっとも短い点の対を見つける
    - 応用： Google Code Jam 2009 World Finals B
    - 応用： ACM-ICPC 2013 Asia Aizu F

得点分布