Covering Many or Few Points with Unit Disks

Sergio Cabello
University of Ljubljana and IMFM

Mark de Berg
TU Eindhoven

Sariel Har-Peled
University of Illinois

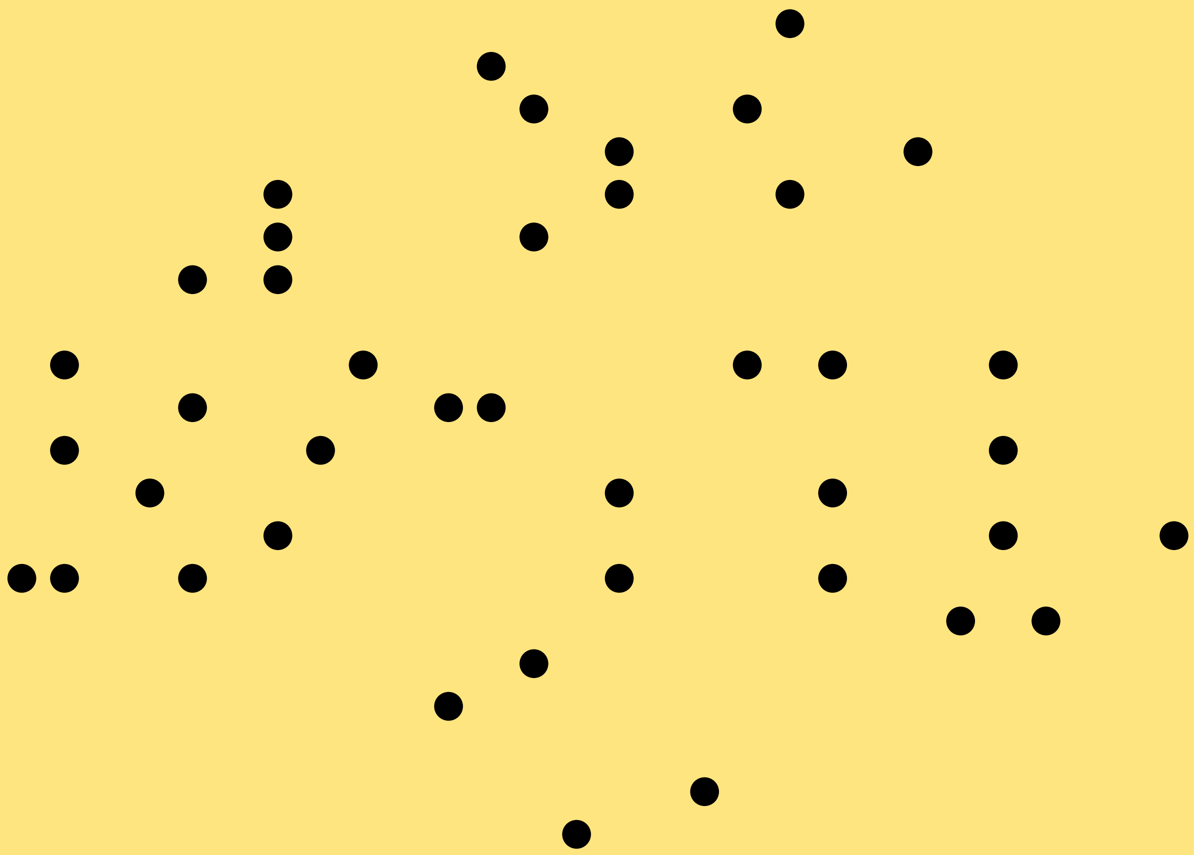
Overview

- MAX: covering many points with unit disks
- MIN: covering few points with a unit disk
- the problems and their context
- new results
- algorithm for MAX

MAX: covering many points with unit disks

$m \in \mathbb{N}$ a constant

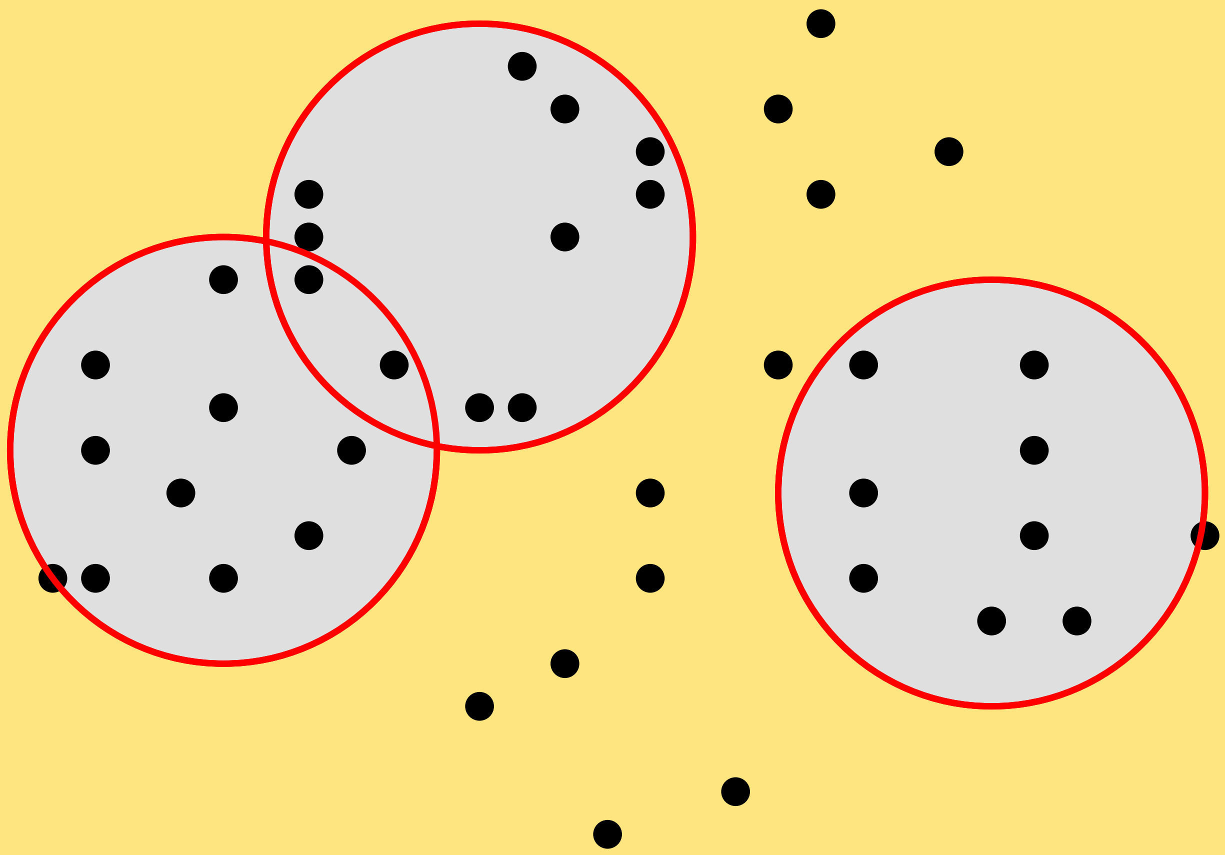
P: n points in $\mathbb{R}^2$



MAX: covering many points with unit disks

$m \in \mathbb{N}$ a constant

P: n points in $\mathbb{R}^2$

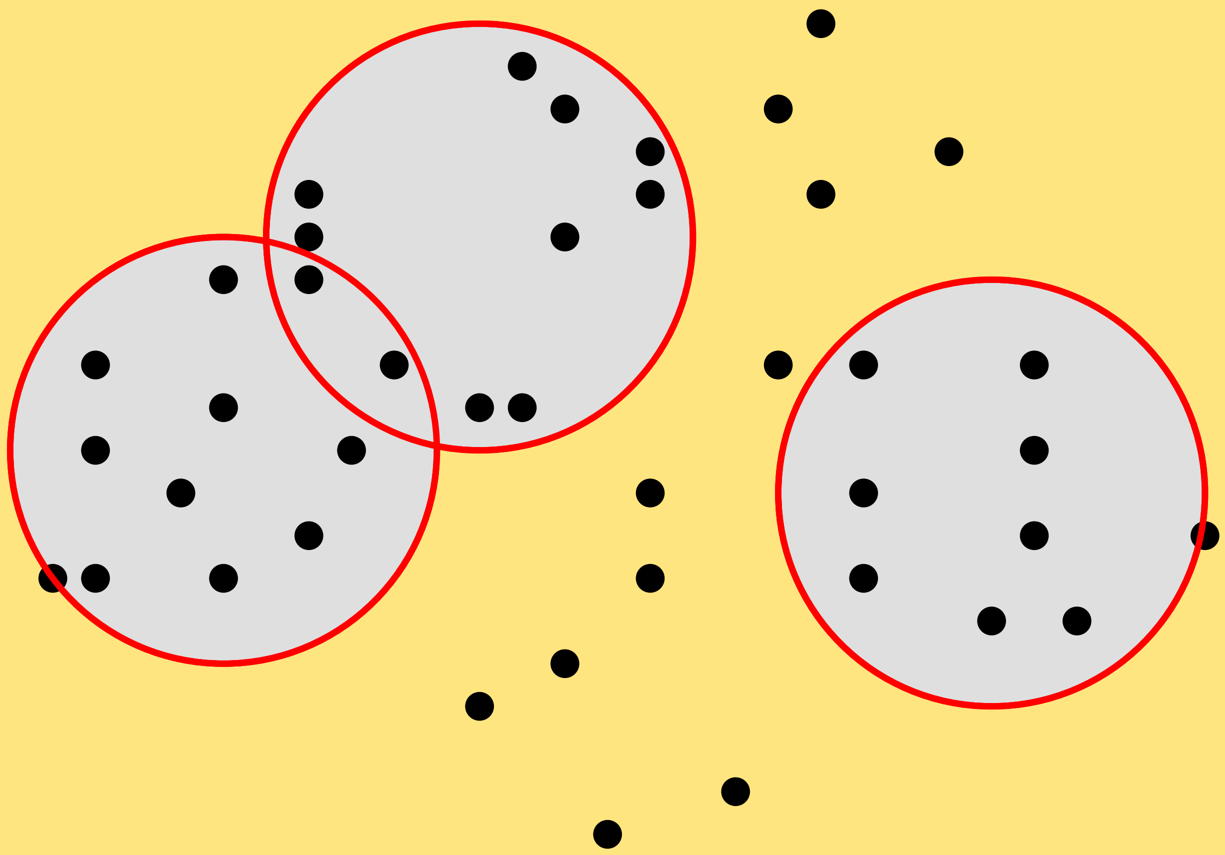


Place m unit disks,
max number covered points

MAX: covering many points with unit disks

$m \in \mathbb{N}$ a constant

P: n points in $\mathbb{R}^2$



Place m unit disks,
max number covered points

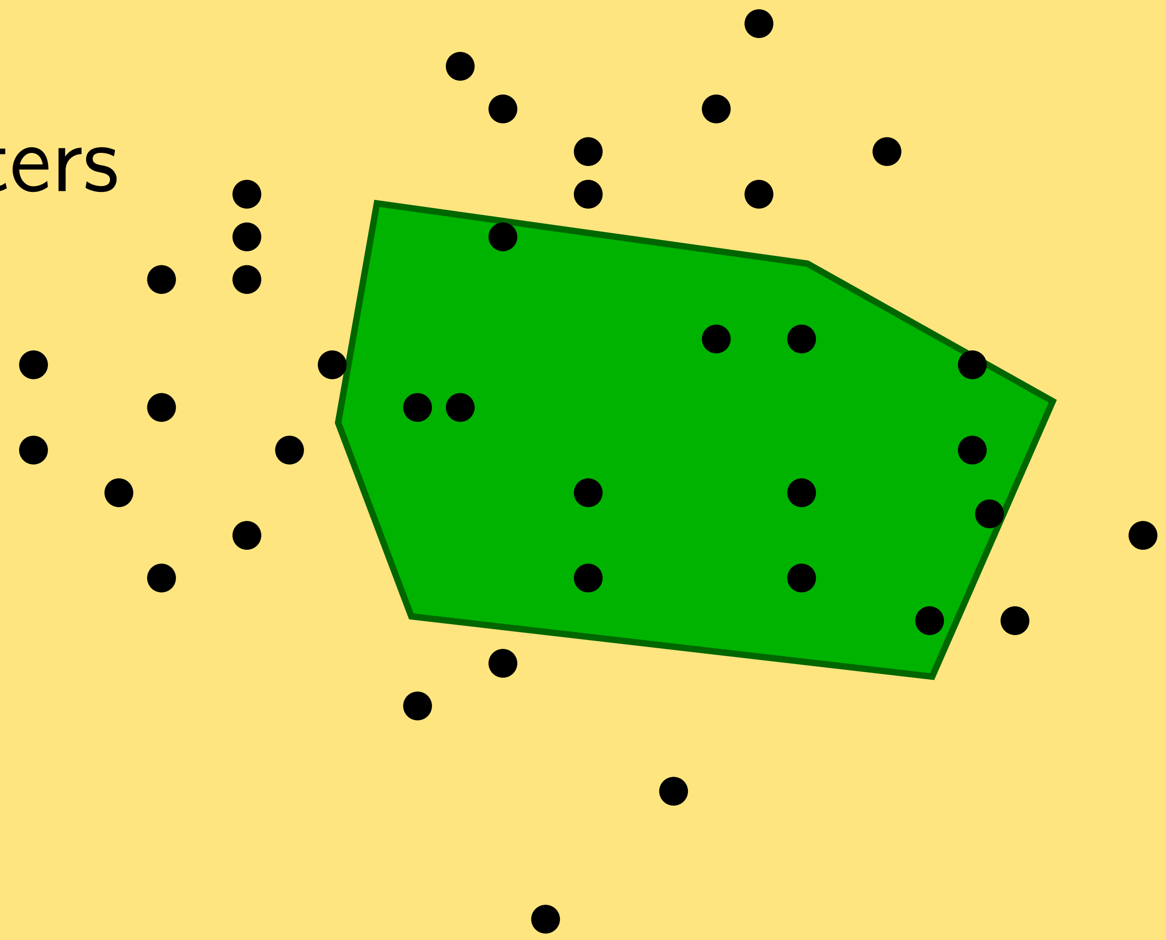
Weighted points $\Rightarrow$ maximize sum of weights

Disks may overlap, no multiplicity when counting
(Non-overlapping disks: collides with packing)

MIN: covering few points with a unit disk

X: constraint region for the centers

P: n points in $\mathbb{R}^2$

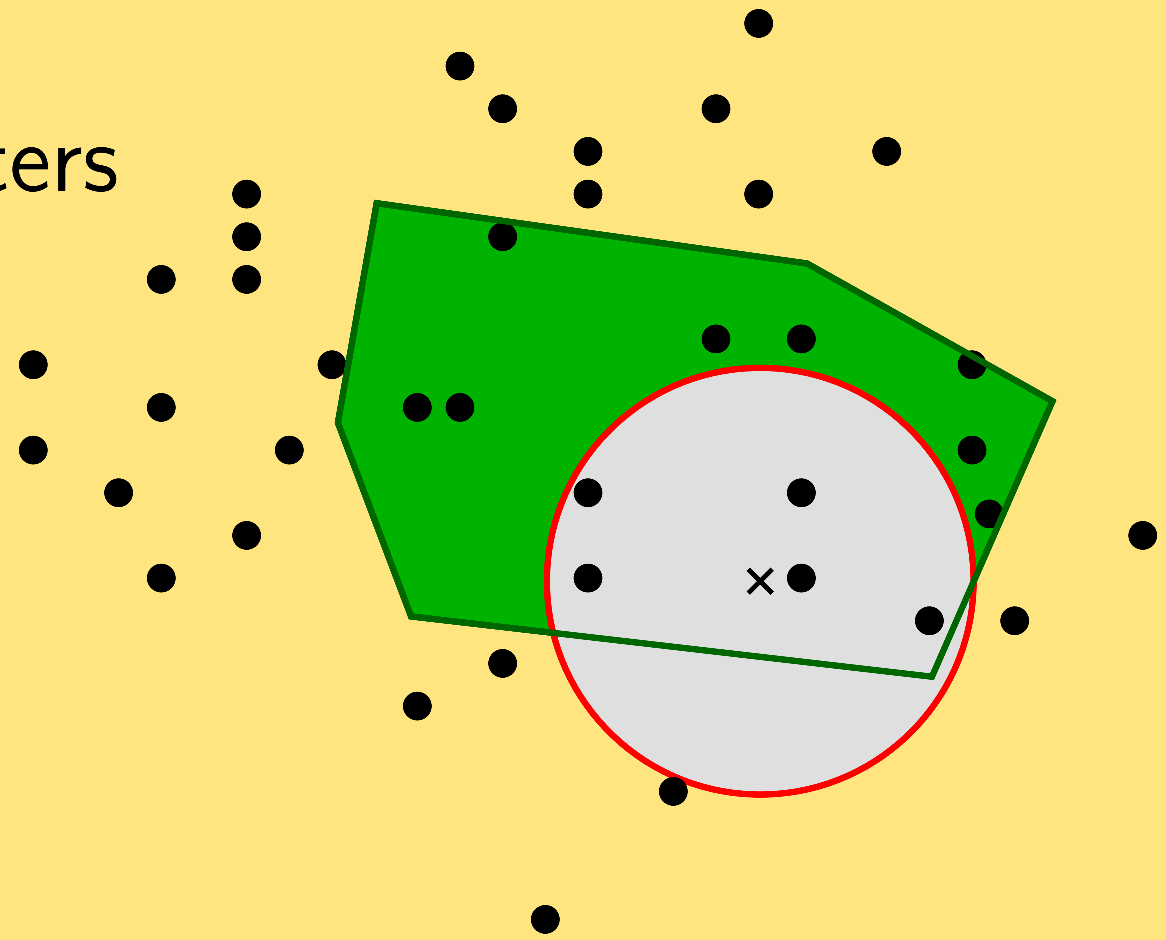


MIN: covering few points with a unit disk

X: constraint region for the centers

P: n points in $\mathbb{R}^2$

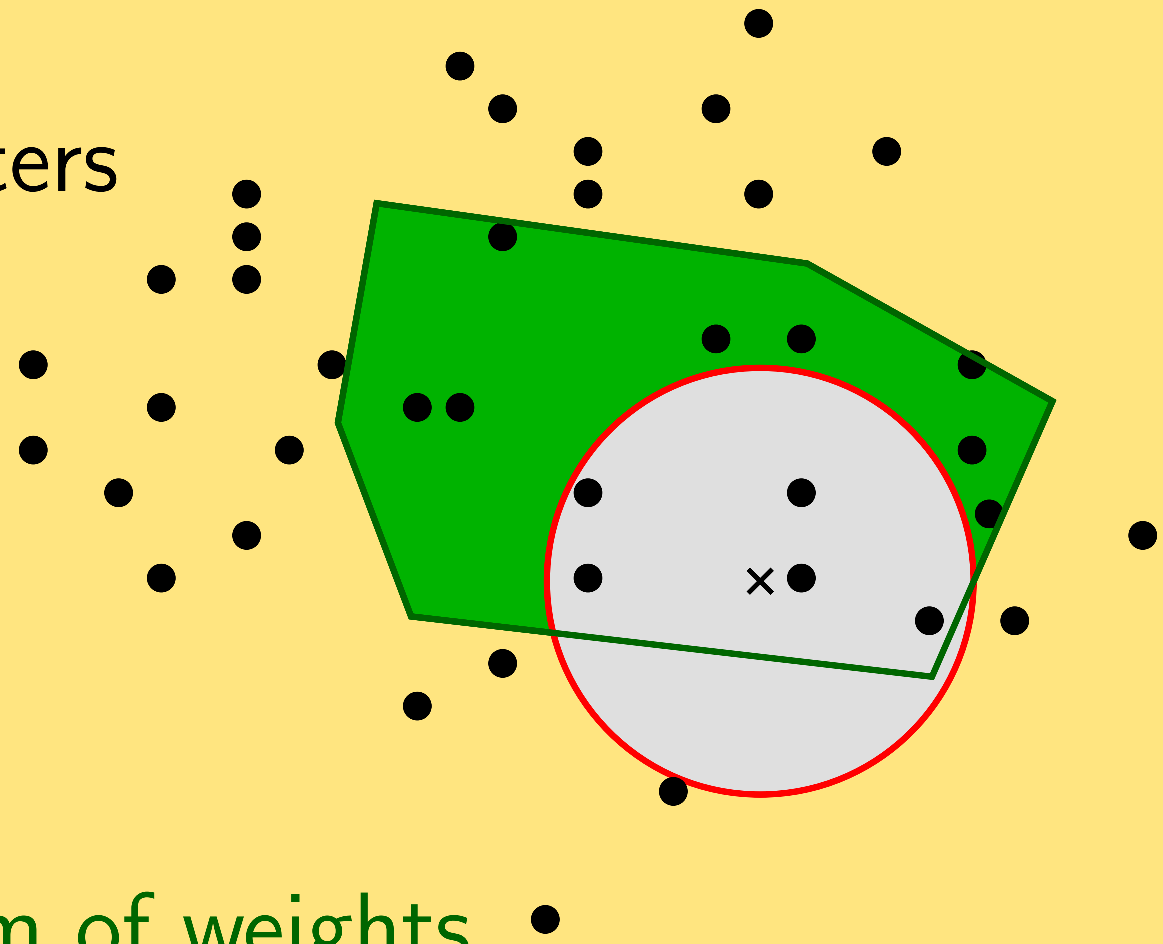
Place a unit disk, centered at X
min number covered points



MIN: covering few points with a unit disk

X: constraint region for the centers

P: n points in $\mathbb{R}^2$



Place a unit disk, centered at X
min number covered points

Weighted points $\Rightarrow$ minimize sum of weights

(X constant complexity)

$X = \mathbb{R}^2$ or placing $m > 1$ disks $\Rightarrow$ problems in the definition

The problems and their context

- MAX(P, m): place m unit disks maximizing the weight of the covered points; m is a constant.
- MIN(P, X): place a unit disk with center in X and minimizing the weight of the covered points.

Motivation: Location of attractive or obnoxious facilities with fixed range of impact.

The problems and their context

Known results for MAX(P, m) and MIN(P, X):

- solvable in polynomial time. [folklore]
- $O(n^2)$ time for MAX$(P, 1)$ and MIN(P, X).
  [Drezner '81, Drezner & Wesolowsky '94,
  Chazelle & Lee '86]
- 3SUM-hard $\Rightarrow$ no subquadratic algorithm known.
- randomized $(1 + \varepsilon)$-approximation for unweighted MAX$(P, 1)$ and MIN(P, X) in $O(n\varepsilon^{-2} \log n)$ time.
  [Aronov & Har-Peled '05]

The problems and their context

Variations on MAX(P, m) and MIN(P, X):

- MIN(P, X) but placing a unit square: $O(n \log n)$ time.
  [Katz & Kedem & Segal, '02]

- MAX(P,1) but placing convex object of constant complexity: randomized near-linear time.
  [Agarwal et al. '02]

- MAX$(P, 2)$ but with disjoint disks: $O(n^{8/3} \log^2 n)$ time.
  [Cabello et al '06]

New results

$(1 \pm \varepsilon)$-approximation algorithms for:

- MAX(P, m) in $O(n(\log n + \varepsilon^{-O(m)}))$ time.

- MIN(P, X) in $O(n\,(\log^3 n + \varepsilon^{-4}\log^2 n))$ expected time.

New results

$(1 \pm \varepsilon)$-approximation algorithms for:

- MAX(P, m) in $O(n(\log n + \varepsilon^{-O(m)}))$ time.

  First near-linear deterministic result for any m.

- MIN(P, X) in $O(n\,(\log^3 n + \varepsilon^{-4} \log^2 n))$ expected time.

  "Adapt" [Aronov & Har-Peled '05] for weighted point sets $\Rightarrow$ Extra logs and ε's.

Overview

- ~~MAX: covering many points with unit disks~~
- ~~MIN: covering few points with a unit disk~~
- ~~the problems and their context~~
- ~~new results~~
- algorithm for MAX

Algorithm for MAX(P, m)

Ingredients:

- bounded VC-dimension $\Rightarrow (1/r)$-approximations
- shifted grids
- dynamic programming

Algorithm for MAX$(P, 1)$

Ingredients:

- bounded VC-dimension $\Rightarrow (1/r)$-approximations
- shifted grids
- ~~dynamic programming~~

Algorithm for MAX$(P, 1)$

A biased course on discrepancy.

Algorithm for MAX$(P, 1)$

P a weighted n-point set.
r a parameter.

Point set A is a $(1/r)$-approximation for P if

$$|w(D \cap P) - w(D \cap A)| \leq \frac{1}{r} \cdot w(P)$$

for any unit disk D.

Algorithm for MAX$(P, 1)$

P a weighted n-point set.
r a parameter.

Point set A is a $(1/r)$-approximation for P if

$$|w(D \cap P) - w(D \cap A)| \leq \frac{1}{r} \cdot w(P)$$

for any unit disk D.

Thm: There is a $(1/r)$-approximation A for P with $O(r^2 \log r)$ points. It takes $O(nr^{O(1)})$ time to construct it.

Algorithm for MAX$(P, 1)$

Aim: $(1+\varepsilon)$-approximation algorithm.

Algorithm for MAX$(P, 1)$

Aim: $(1+\varepsilon)$-approximation algorithm.

Warning!

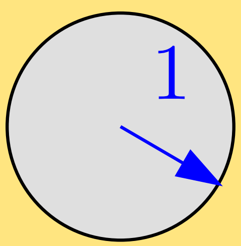
finding an ε-approximation A and
an optimal solution for A is not good.

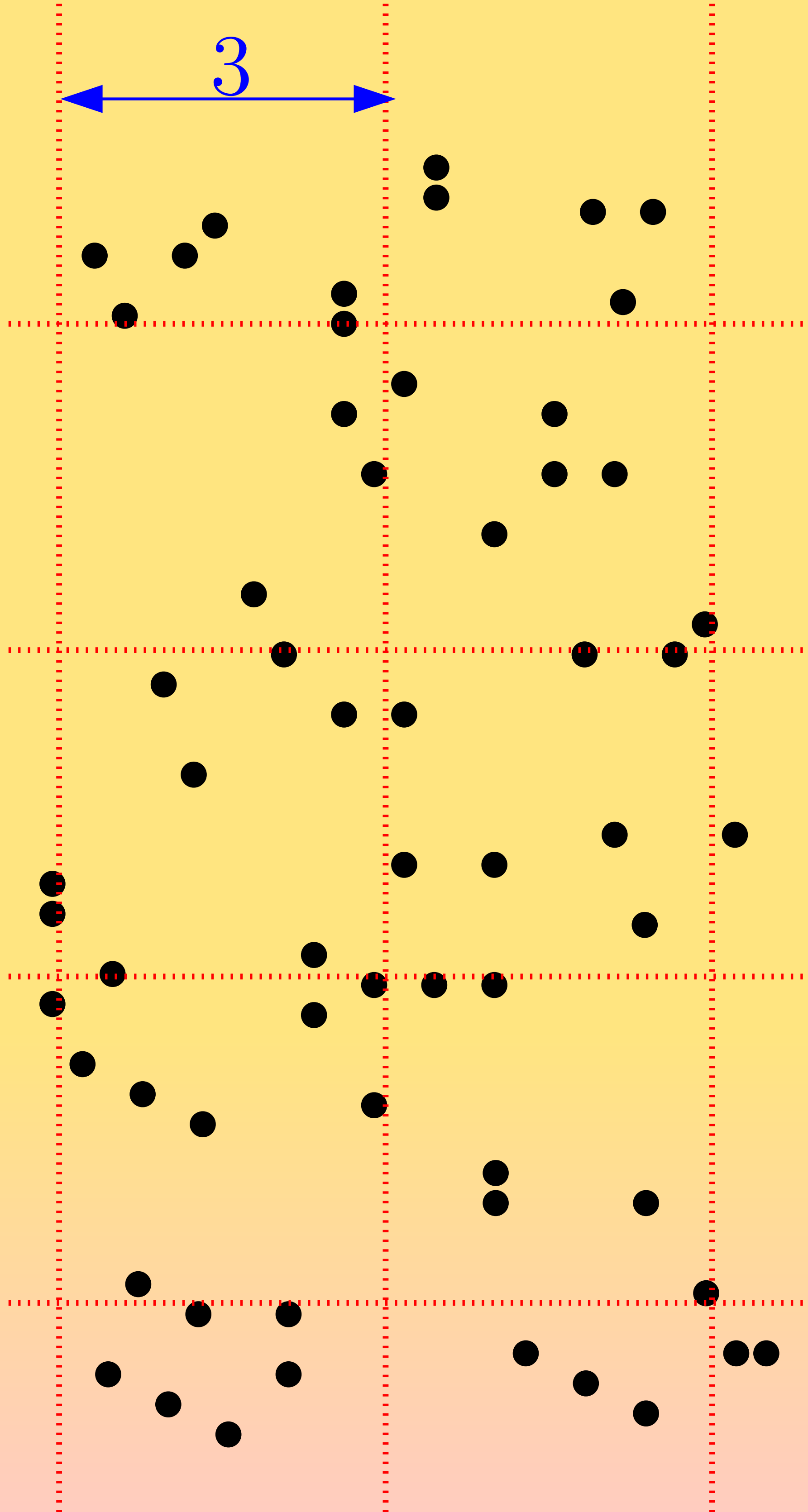
So, why did I explain it...?

Algorithm for MAX$(P, 1)$

Aim: $(1+\varepsilon)$-approximation algorithm.

Grid of spacing 3.



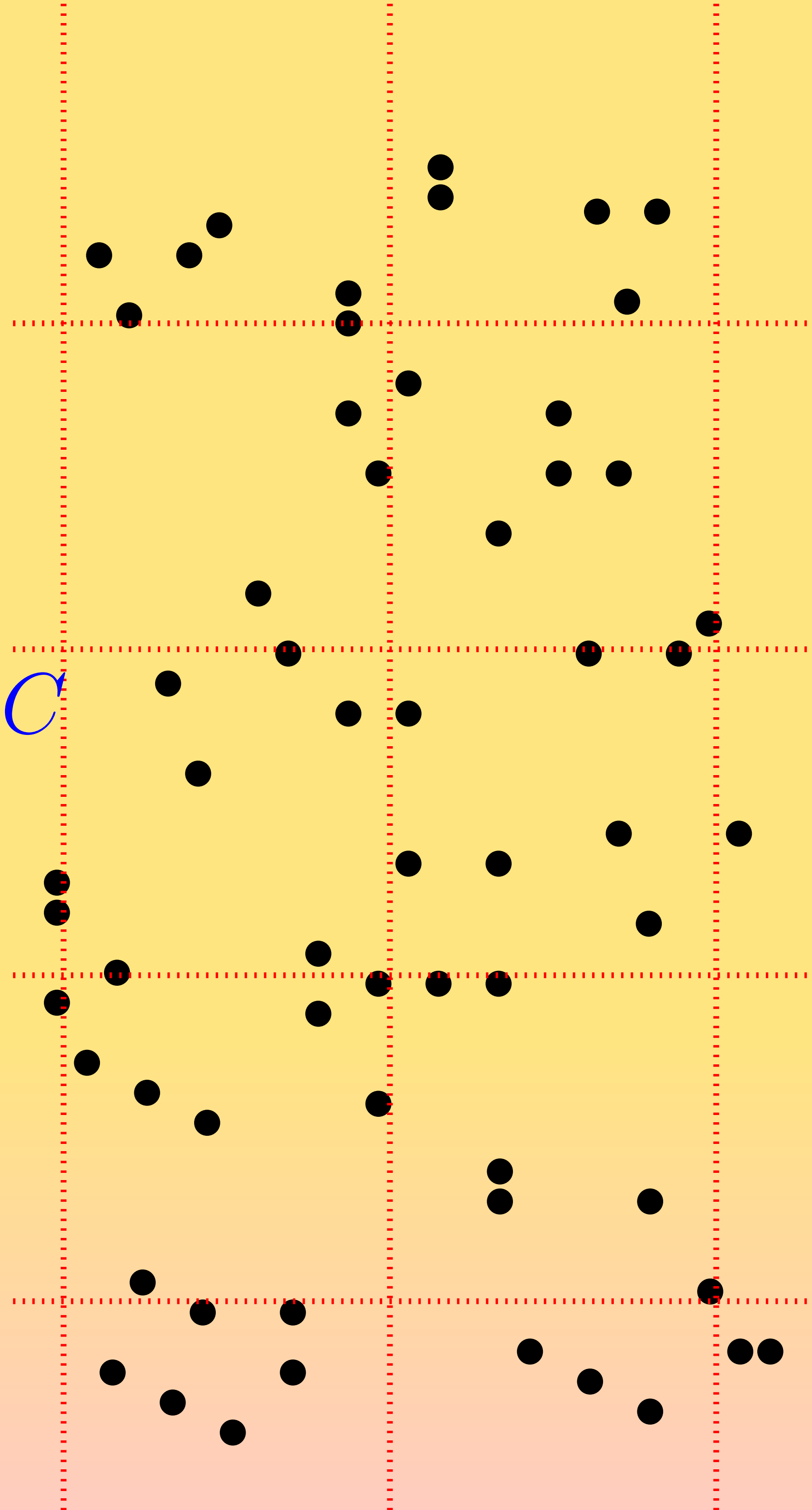


Algorithm for MAX$(P, 1)$

Aim: $(1+\varepsilon)$-approximation algorithm.

Grid of spacing 3.

set $r = 100/\varepsilon$
set $A = \emptyset$
for each cell C
    find $(1/r)$-approximation A_C for $P \cap C$
    add A_C to A

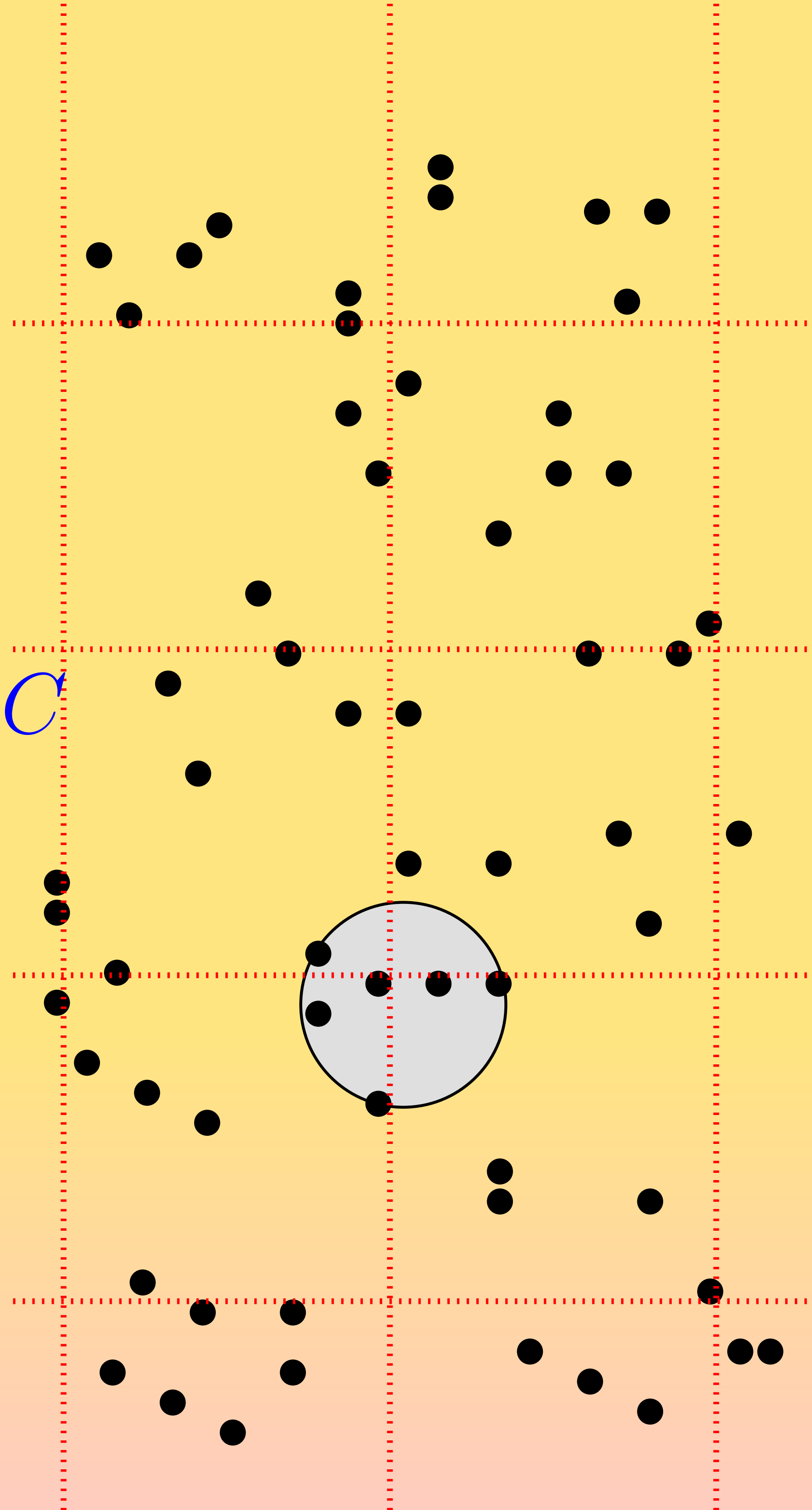


Algorithm for MAX$(P, 1)$

Aim: $(1 + \varepsilon)$-approximation algorithm.

Grid of spacing 3.

set $r = 100/\varepsilon$
set $A = \emptyset$
for each cell C
    find $(1/r)$-approximation A_C for $P \cap C$
    add A_C to A



Lem: Optimal solution for A is a $(1 + \varepsilon)$-approximation.

Proof: ...

Algorithm for MAX(P, 1)

Aim: $(1+\varepsilon)$-approximation algorithm.

Lem: Optimal solution for A is a $(1+\varepsilon)$-approximation.

set $r = 100/\varepsilon$
set $A = \emptyset$
for each cell C
    find $(1/r)$-approximation A_C for $P \cap C$
    add A_C to A

Algorithm for MAX($P, 1$)

Aim: $(1+\varepsilon)$-approximation algorithm.

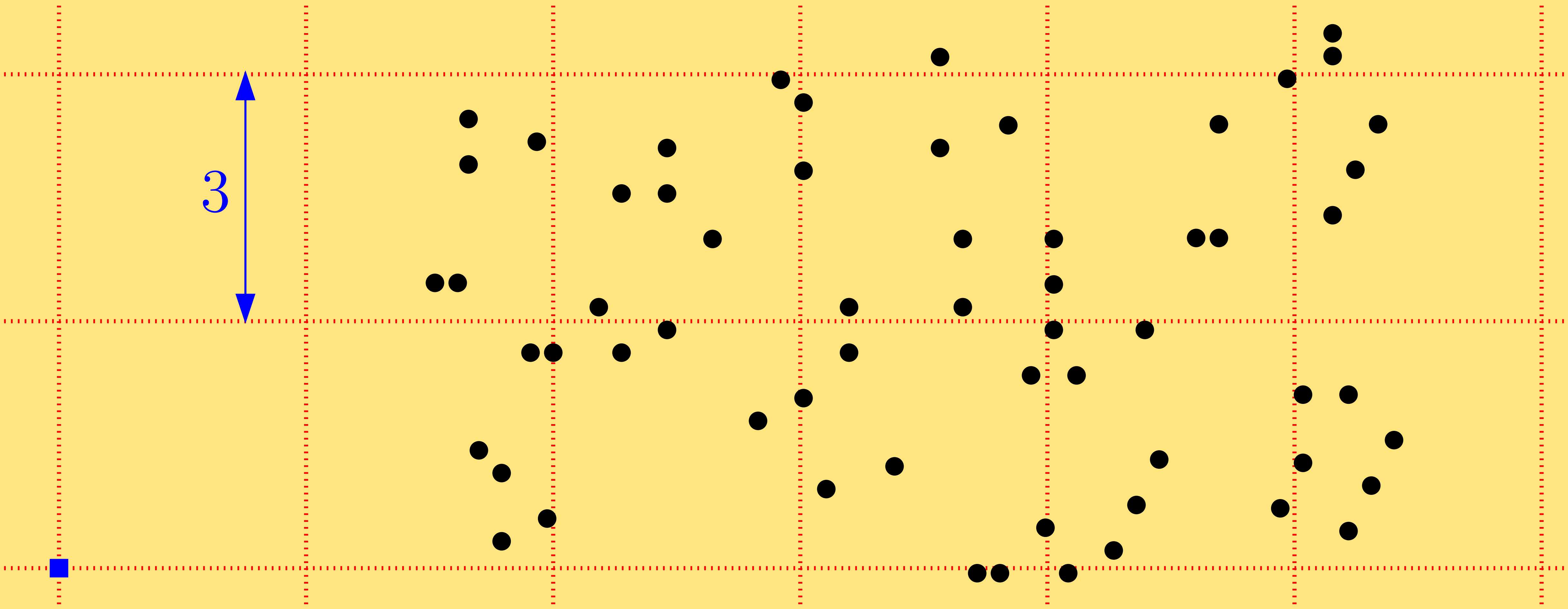
Lem: Optimal solution for A is a $(1+\varepsilon)$-approximation.

set $r = 100/\varepsilon$
set $A = \emptyset$
for each cell C
    find $(1/r)$-approximation A_C for $P \cap C$
    add A_C to A

Did we gain anything?

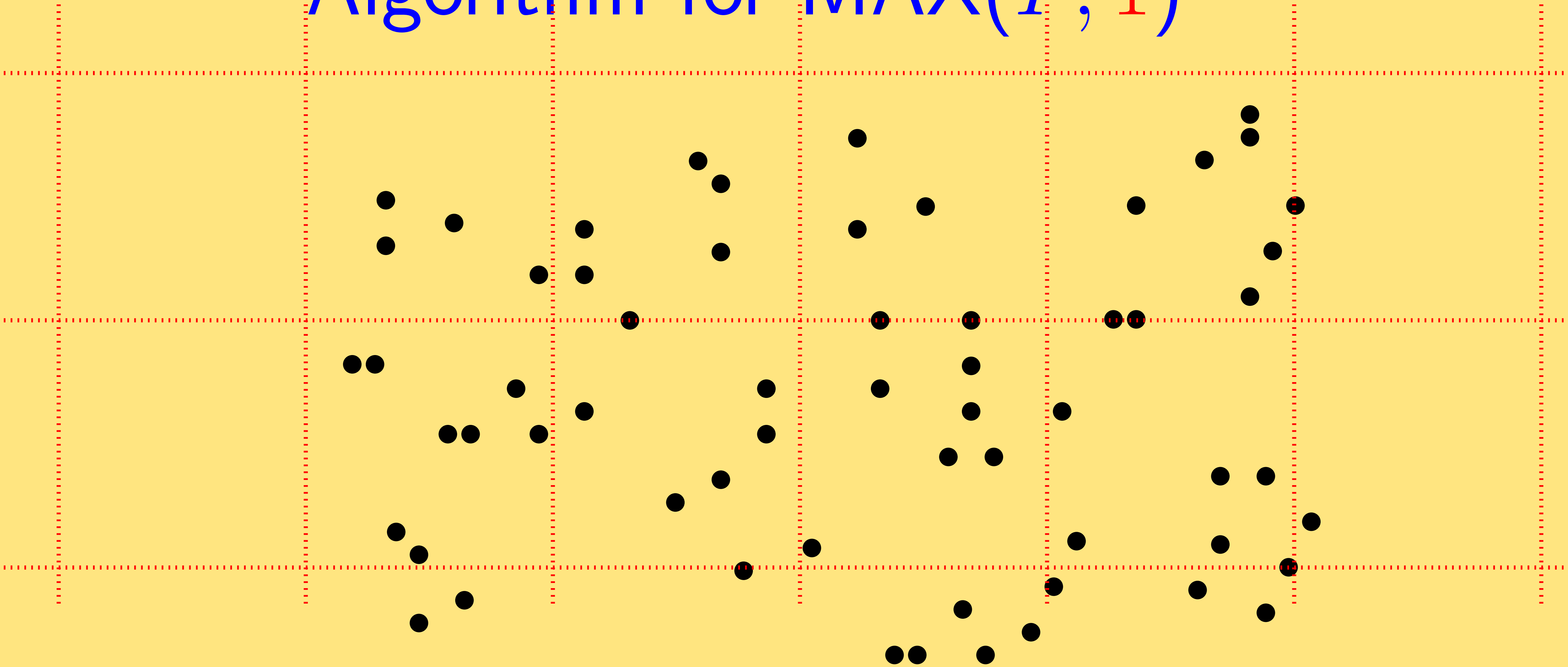
Each grid cell has $O(r^2 \log r) = O(\varepsilon^{-O(1)})$ points.

Algorithm for MAX($P, 1$)



Each grid cell has $O(\varepsilon^{-O(1)})$ points.

Algorithm for MAX(P, 1)

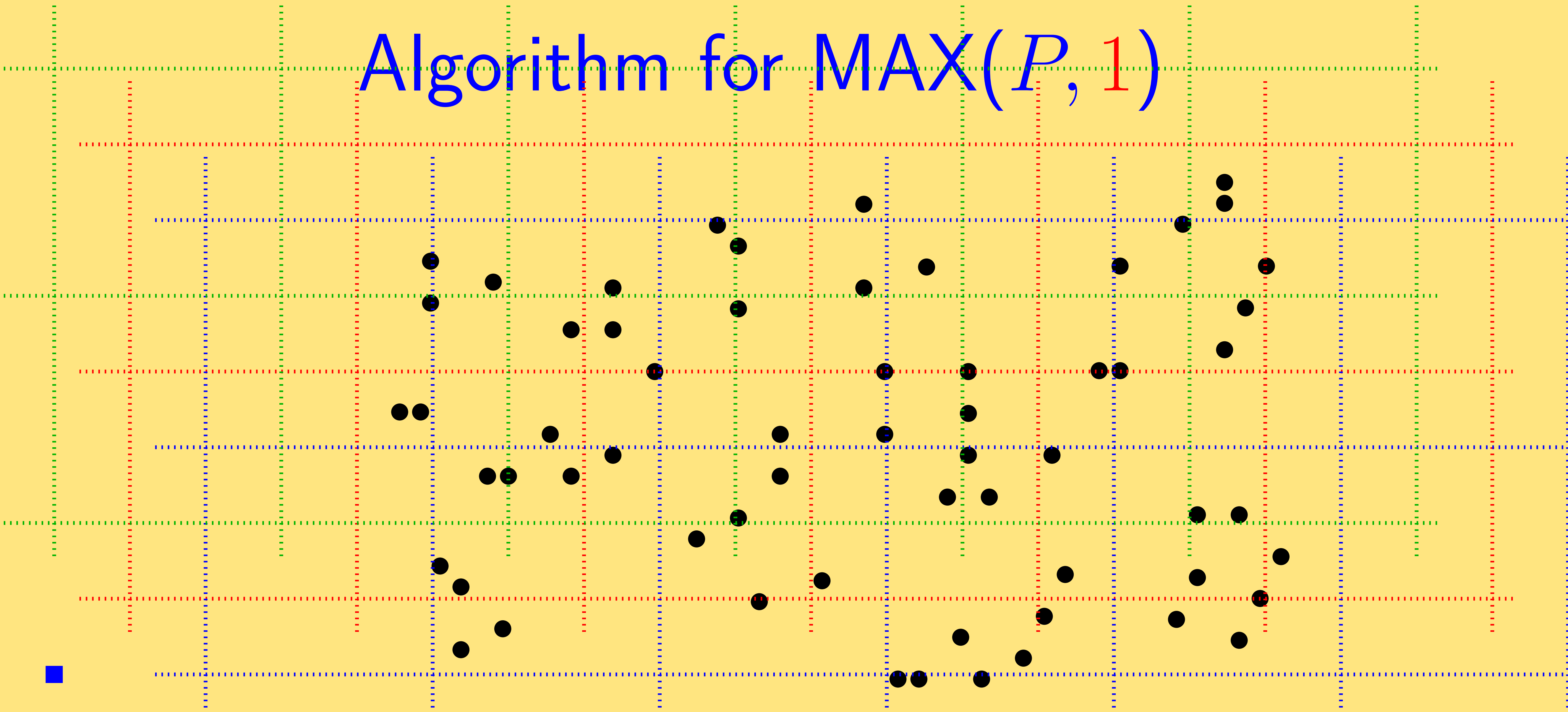


-

Each grid cell has $O(\varepsilon^{-O(1)})$ points.

Take its 3^2 integer shifts. Each cell $O(\varepsilon^{-O(1)})$ points.

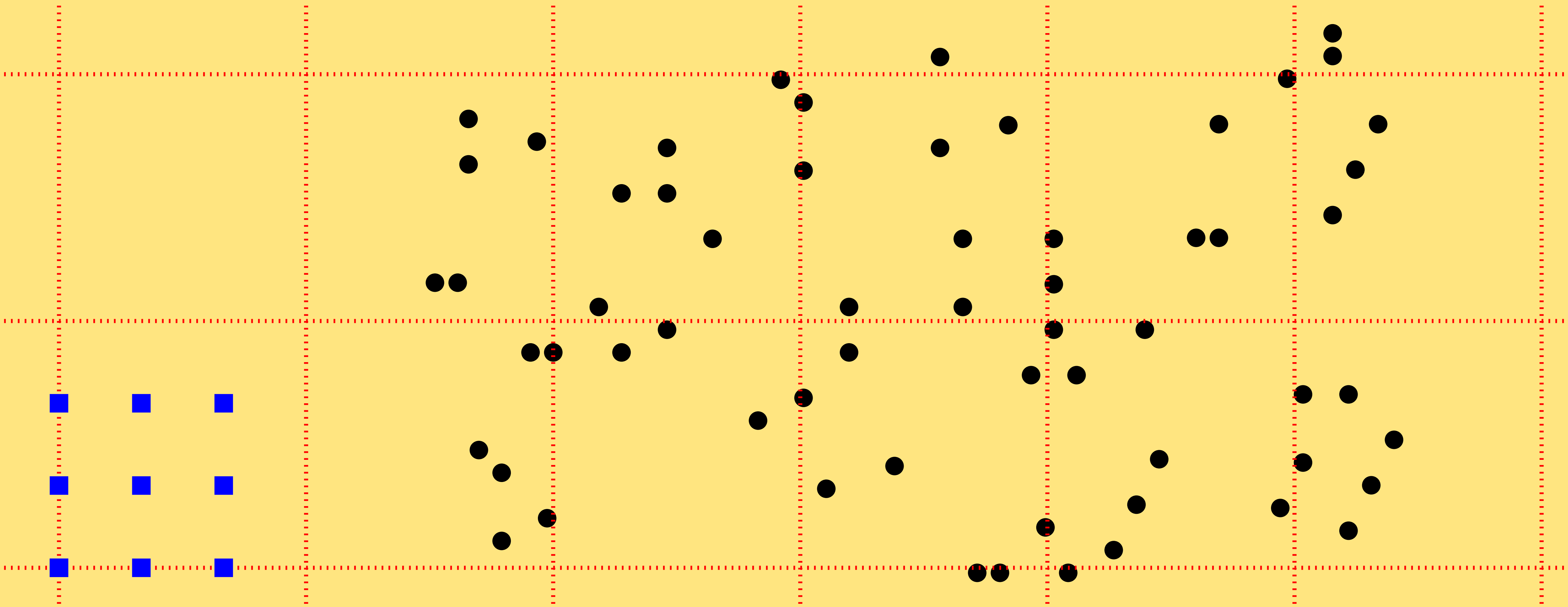
Algorithm for MAX($P, 1$)



Each grid cell has $O(\varepsilon^{-O(1)})$ points.

Take its 3^2 integer shifts. Each cell $O(\varepsilon^{-O(1)})$ points.

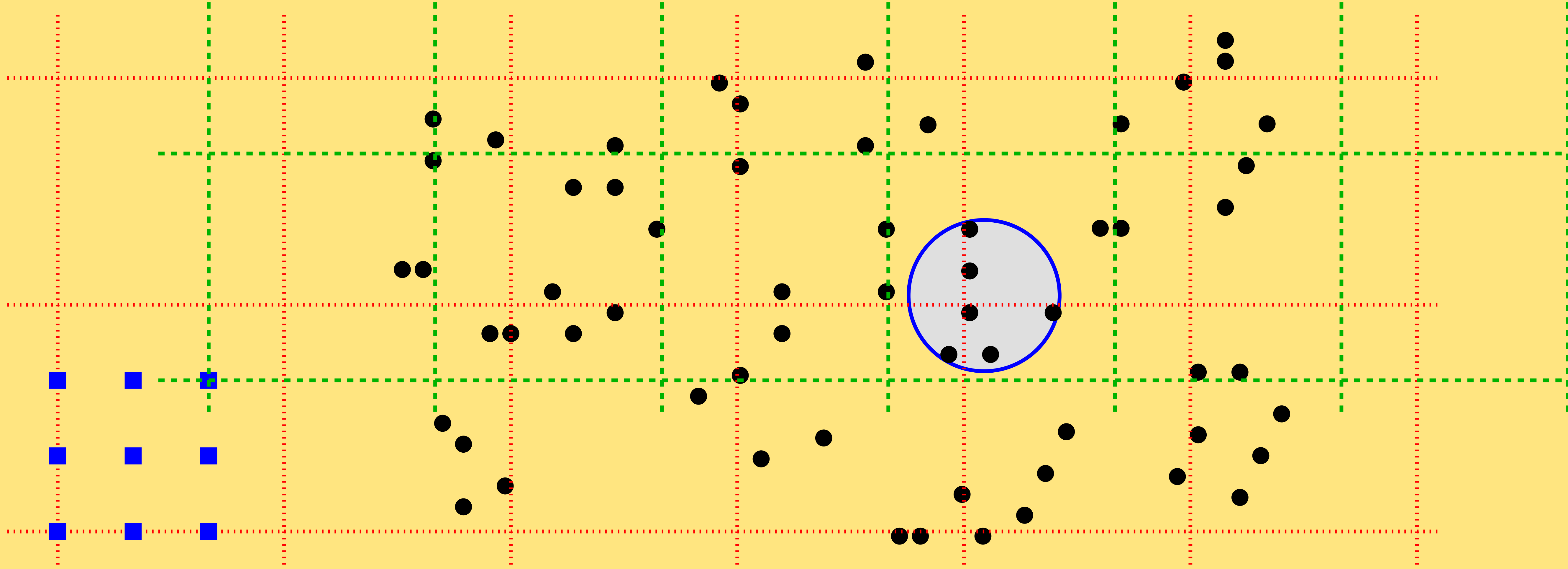
Algorithm for MAX(P, 1)



Each grid cell has $O(\varepsilon^{-O(1)})$ points.

Take its 3^2 integer shifts. Each cell $O(\varepsilon^{-O(1)})$ points.

Algorithm for MAX($P, 1$)



Each grid cell has $O(\varepsilon^{-O(1)})$ points.

Take its 3^2 integer shifts. Each cell $O(\varepsilon^{-O(1)})$ points.

Lem: One of them does not intersect the optimal solution.

Algorithm for MAX$(P, 1)$

replace P by A
for each of the 9 shifted grids G'
    for each cell C in G'
        find best disk inside C
report the best disk you found.

Algorithm for MAX($P, 1$)

replace P by A
for each of the 9 shifted grids G'
    for each cell C in G'
        find best disk inside C
report the best disk you found.

C has $O(\varepsilon^{-O(1)})$ points

takes $O(\varepsilon^{-O(1)})$ time

Algorithm for MAX($P, 1$)

replace P by A
for each of the 9 shifted grids G'
    for each cell C in G'
        find best disk inside C
report the best disk you found.

C has $O(\varepsilon^{-O(1)})$ points

takes $O(\varepsilon^{-O(1)})$ time

Thm: MAX($P, 1$) can be $(1+\varepsilon)$-approximated in
$O(n \log n + n\varepsilon^{-2} \log(1/\varepsilon))$ time.

Algorithm for MAX(P, m)

replace P by A
for each of the 9 shifted grids G'
for each cell C in G'
find best disk inside C
report the best disk you found.

replace P by A
for each of the $O(m^2)$ shifted grids G'
find best m disks avoiding G'
report the best group you found.

Algorithm for MAX(P, m)

replace P by A
for each of the 9 shifted grids G'
for each cell C in G'
find best disk inside C
report the best disk you found.

replace P by A
for each of the $O(m^2)$ shifted grids G'
find best m disks avoiding G'
report the best group you found.

Dynamic programming
accross cells of G'

Grid of size $3m$.

Algorithm for MAX(P, m)

replace P by A
for each of the 9 shifted grids G'
for each cell C in G'
find best disk inside C
report the best disk you found.

replace P by A
for each of the $O(m^2)$ shifted grids G'
find best m disks avoiding G'
report the best group you found.

Dynamic programming accross cells of G'

Grid of size $3m$.

Thm: For $m > 1$, MAX(P, m) can be $(1+\varepsilon)$-approximated in $O(n \log n + n\varepsilon^{-4m+4} \log^{2m-1}(1/\varepsilon))$ time.

Summary

$(1+\varepsilon)$-approximation algorithm for

- MAX(P, m) in $O(n \log n + n\varepsilon^{-O(m)})$; deterministic.
- MIN(P, X) in $O(n \log^3 n + n\varepsilon^{-4} \log^2 n)$ time; randomized MC and LV.

What remains?

- subcubic exact for MAX$(P, 2)$ with disks or squares?
- is it true that nobody studied MAX(P, m) before?

Summary

$(1+\varepsilon)$-approximation algorithm for

- MAX(P, m) in $O(n \log n + n\varepsilon^{-O(m)})$; deterministic.
- MIN(P, X) in $O(n \log^3 n + n\varepsilon^{-4} \log^2 n)$ time; randomized MC and LV.

What remains

- subcubic exact for MAX$(P, 2)$ with disks or squares?
- is it true that nobody studied MAX(P, m) before?

THANKS!