Approximate Testing of Visual Properties

Sofya Raskhodnikova

MIT + the Hebrew University of Jerusalem*

* Partly supported by the Lady Davis Fellowship

Motivation for sublinear algorithms

Massive data sets

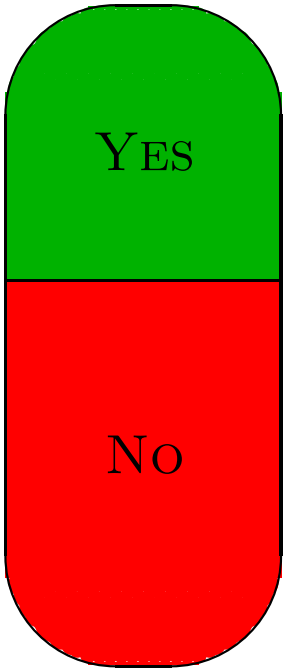
- scientific measurements
- human DNA sequence, completed by genome project
- world-wide web
- census data
- sales logs
- high-resolution images

Long access time

Property tester definition

[Rubinfeld Sudan 96, Goldreich Goldwasser Ron 98]

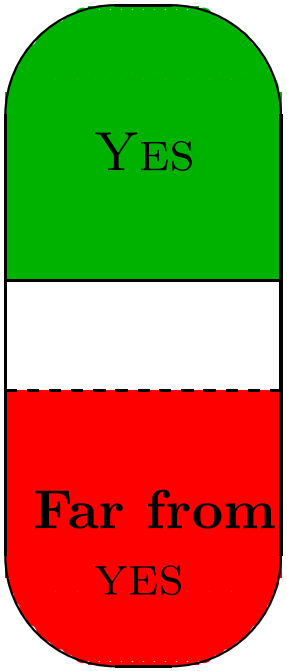
Probabilistic algorithm



Accept with high probability

Reject with high probability

Property tester

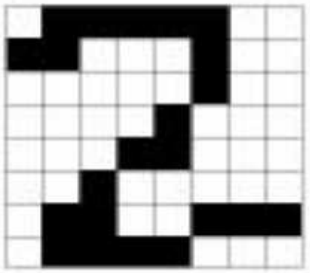


Accept with high probability

Do not care

Reject with high probability

Testing visual properties



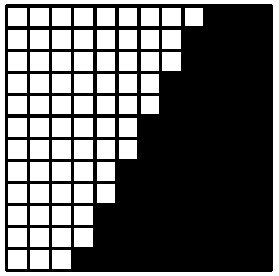
Our results

Being a half-plane

A half-plane or

ε-far from a half-plane?

$O(1/\varepsilon)$ queries

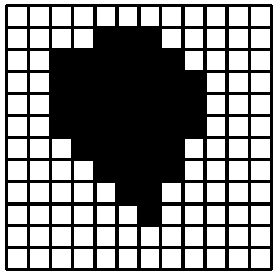


Convexity

Convex or

ε-far from a convex image?

$O(1/\varepsilon^2)$ queries

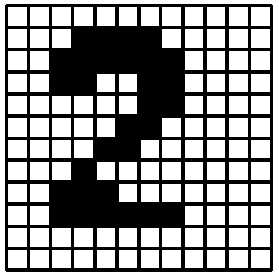


Connectedness

Connected or

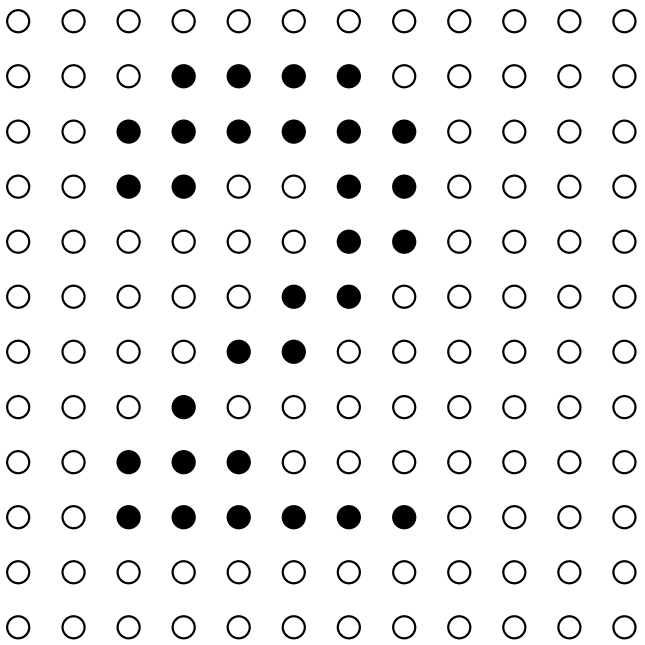
ε-far from a connected image?

$O(1/\varepsilon^4)$ queries



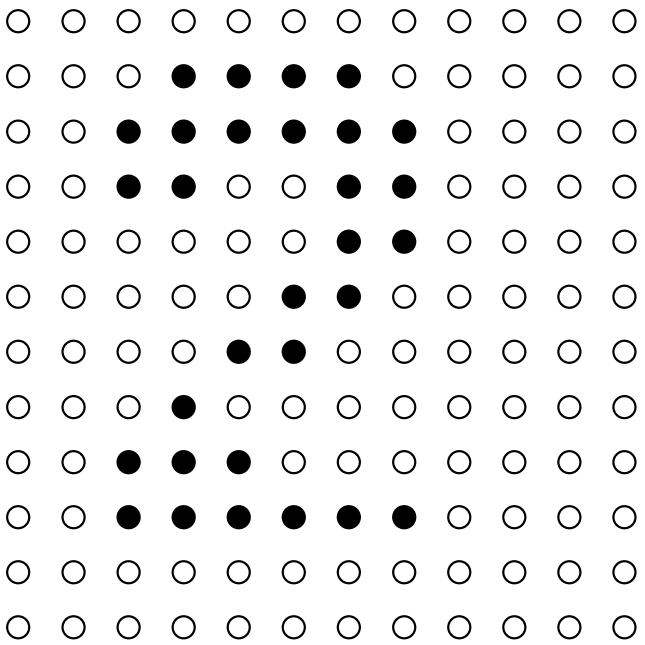
Pixel model

$n \times n$ matrix of pixels



Pixel model

$n \times n$ matrix of pixels

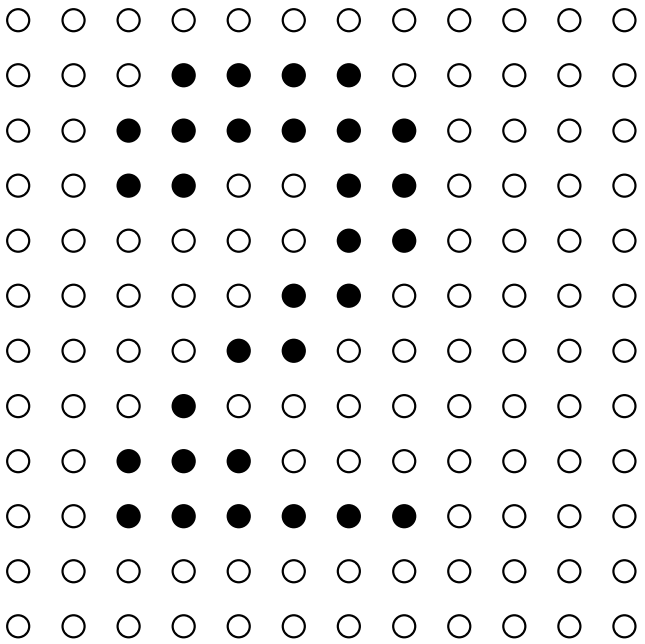


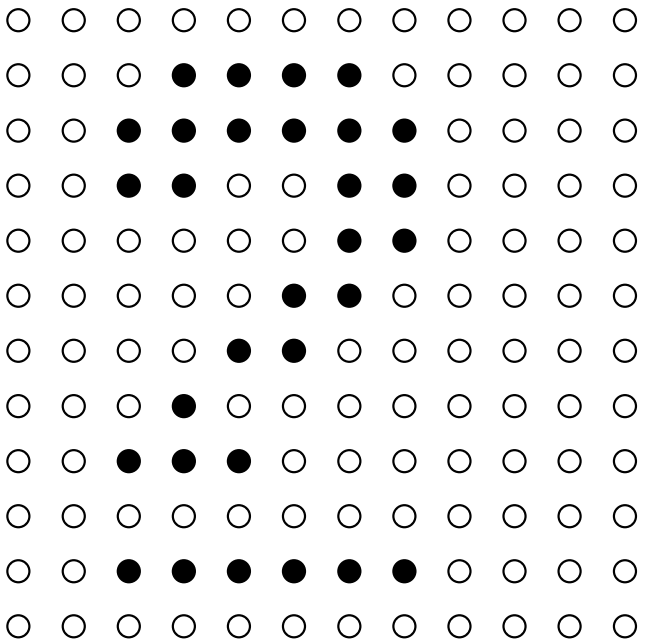
Query: point (i_1, i_2)

Answer: color of (i_1, i_2)

Pixel model

$n \times n$ matrix of pixels



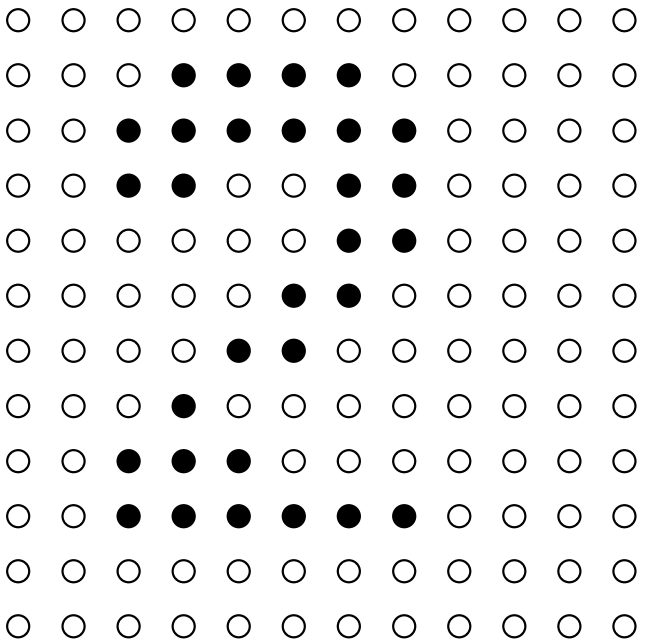


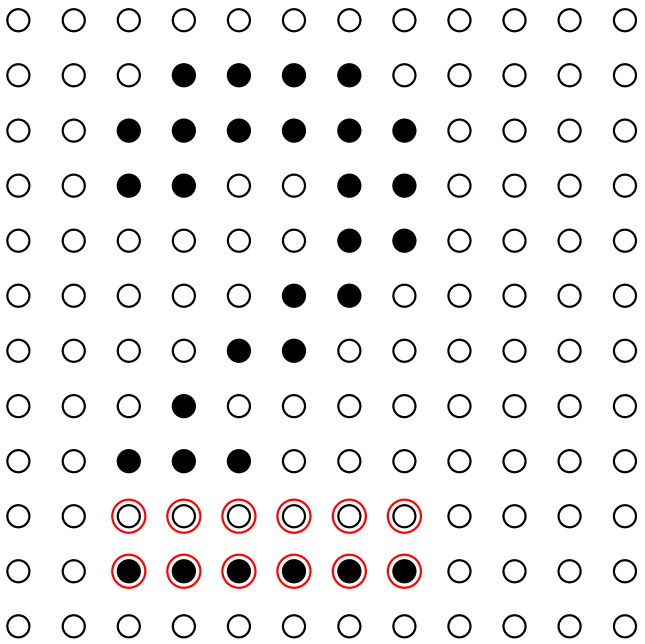
Query: point (i_1, i_2)

Answer: color of (i_1, i_2)

Pixel model

$n \times n$ matrix of pixels





Query: point (i_1, i_2)

Answer: color of (i_1, i_2)

Previous work: property testing

- Computational geometry [Czumaj Sohler Ziegler 00]

  Query: point i Answer: (i_1, i_2)

Previous work: property testing

- Computational geometry [Czumaj Sohler Ziegler 00]

  Query: point i Answer: (i_1, i_2)

- Properties of d-dimensional matrices [Fischer Newman 01]

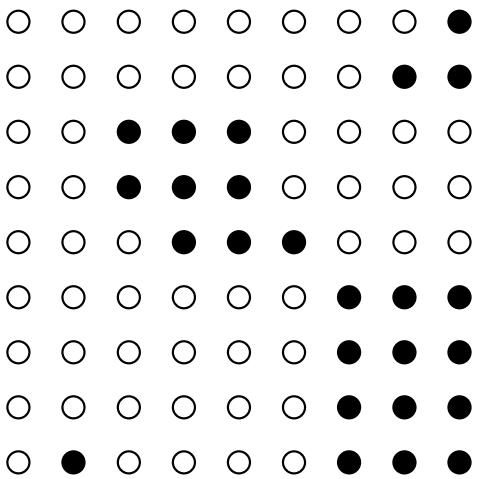
Previous work: property testing

- Computational geometry [Czumaj Sohler Ziegler 00]

  Query: point i Answer: (i_1, i_2)

- Properties of d-dimensional matrices [Fischer Newman 01]

- Properties of bounded degree graphs [Goldreich Ron 97]



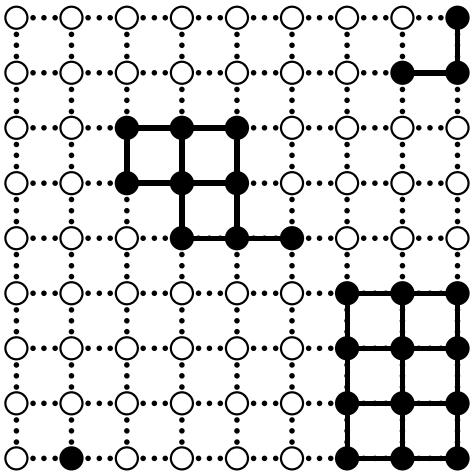
Previous work: property testing

- Computational geometry [Czumaj Sohler Ziegler 00]

  Query: point i Answer: (i_1, i_2)

- Properties of d-dimensional matrices [Fischer Newman 01]

- Properties of bounded degree graphs [Goldreich Ron 97]



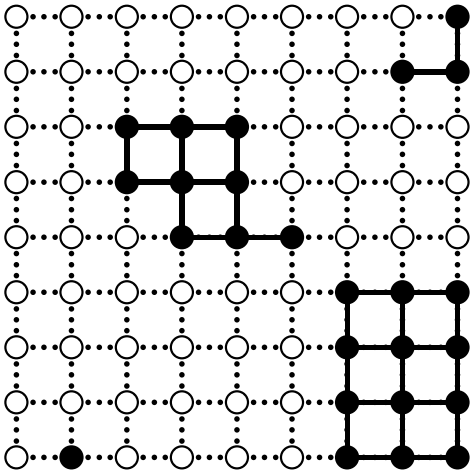
Previous work: property testing

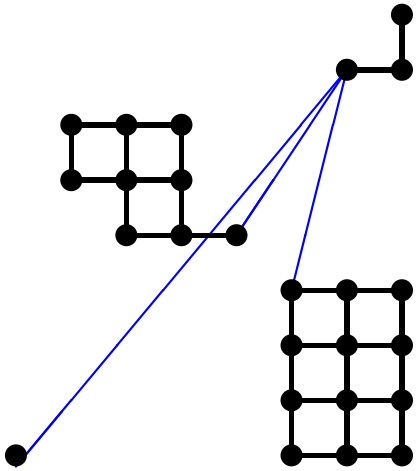
- Computational geometry [Czumaj Sohler Ziegler 00]

  Query: point i Answer: (i_1, i_2)

- Properties of d-dimensional matrices [Fischer Newman 01]

- Properties of bounded degree graphs [Goldreich Ron 97]





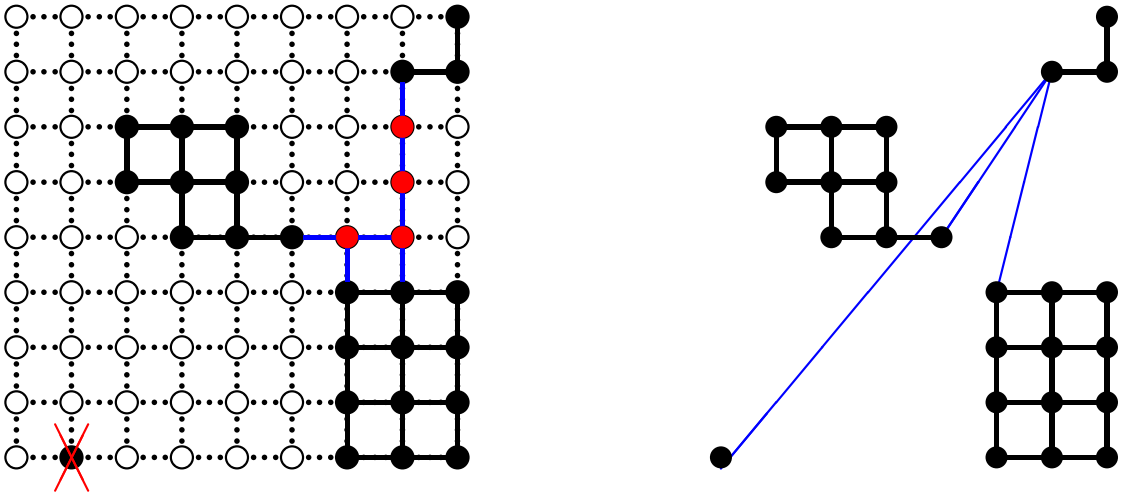
Previous work: property testing

- Computational geometry [Czumaj Sohler Ziegler 00]

  Query: point i Answer: (i_1, i_2)

- Properties of d-dimensional matrices [Fischer Newman 01]

- Properties of bounded degree graphs [Goldreich Ron 97]



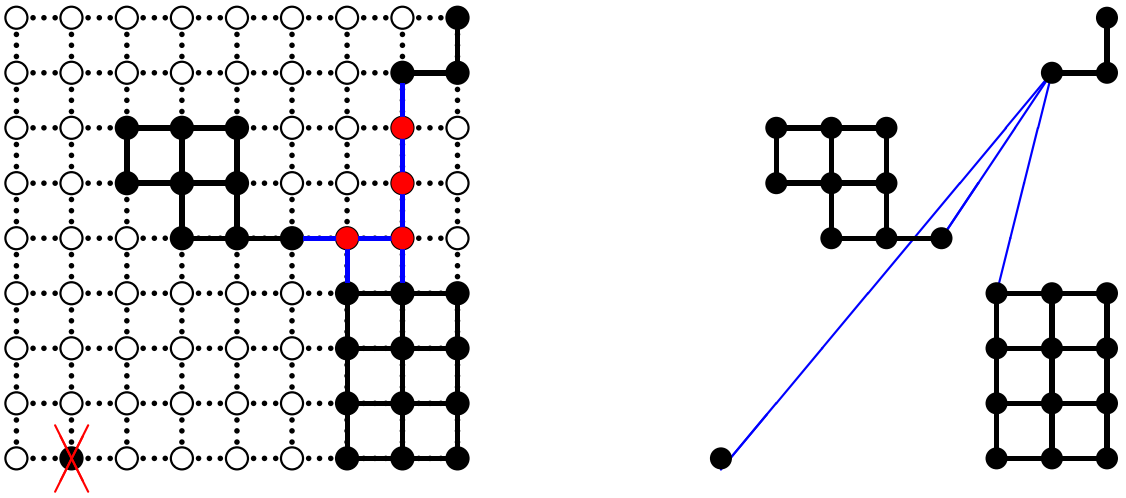
Previous work: property testing

- Computational geometry [Czumaj Sohler Ziegler 00]

  Query: point i Answer: (i_1, i_2)

- Properties of d-dimensional matrices [Fischer Newman 01]

- Properties of bounded degree graphs [Goldreich Ron 97]



- Approximating the number of connected components in a graph in sublinear time [Chazelle Rubinfeld Trevisan 01]

Previous work: connection to learning

- PAC learning [Valiant 84]

  Given: *target* with the property $\mathcal{P}$.

  Output: *hypothesis* ε-close to target w.h.p.

  Proper learning: hypothesis must have property $\mathcal{P}$.

- Proper learning $\Rightarrow$ testing [Goldreich Goldwasser Ron 98]

  Proper PAC learning algorithm for $\mathcal{P}$ with $q(\varepsilon)$ queries

  $\Downarrow$

  (2-sided error) ε-test for $\mathcal{P}$ with $q(\varepsilon/2) + O(1/\varepsilon)$ queries

Previous work: learning

- Learning half-planes
  - exactly [Maas Turan 89]: $\Theta(\log n)$.
  - Proper PAC [Blumer Ehrenfeucht Haussler Warmuth 89]: $O(1/\varepsilon \log(1/\varepsilon))$.
- Complexity of PAC learning $\mathcal{P} = \Omega(\text{VC-dim of } \mathcal{P})$ [Ehrenfeucht Haussler Kearns Valiant 89].
  - VC-dim of convexity is $\Theta(n)$.
  - VC-dim of connectedness is $\Theta(n^2)$.

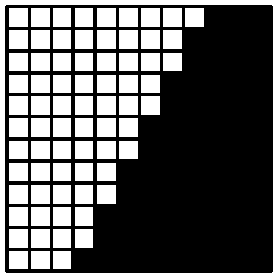
Testing if an image is a half-plane

A half-plane or

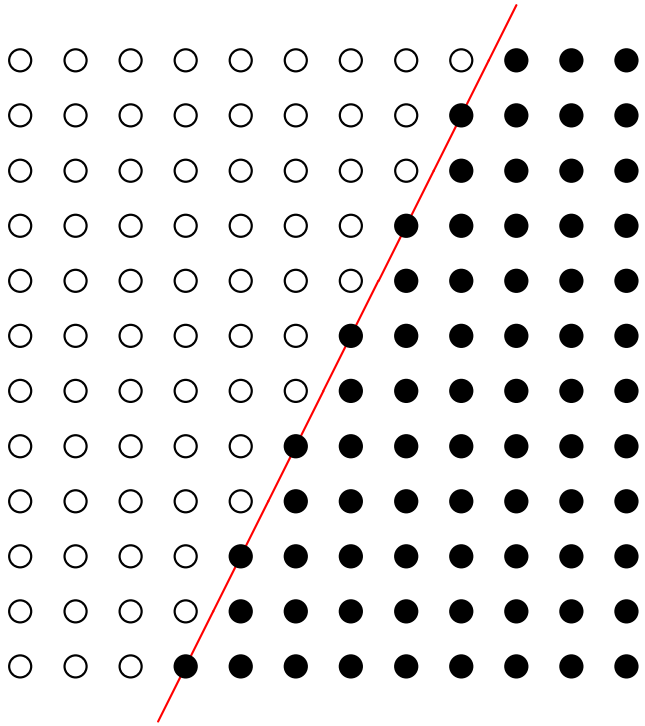
ε-far from a half-plane?

1-sided error

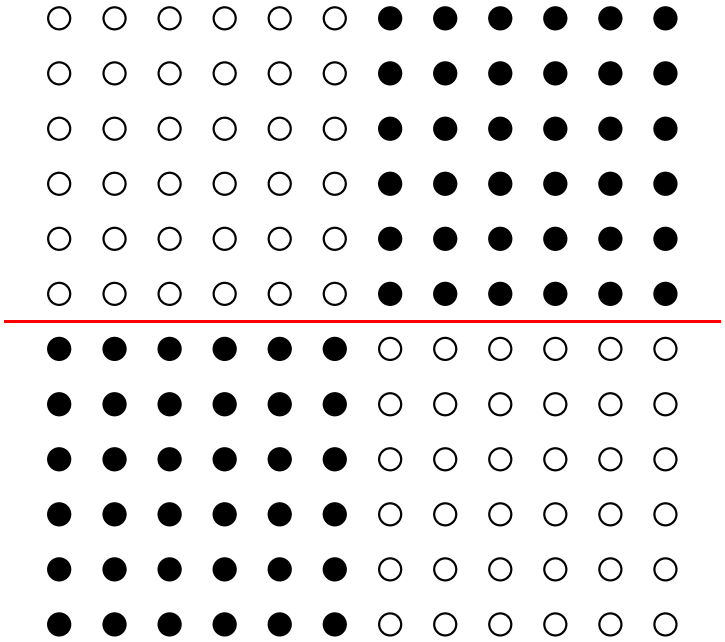
$O(1/\varepsilon)$ queries



Half-plane instances

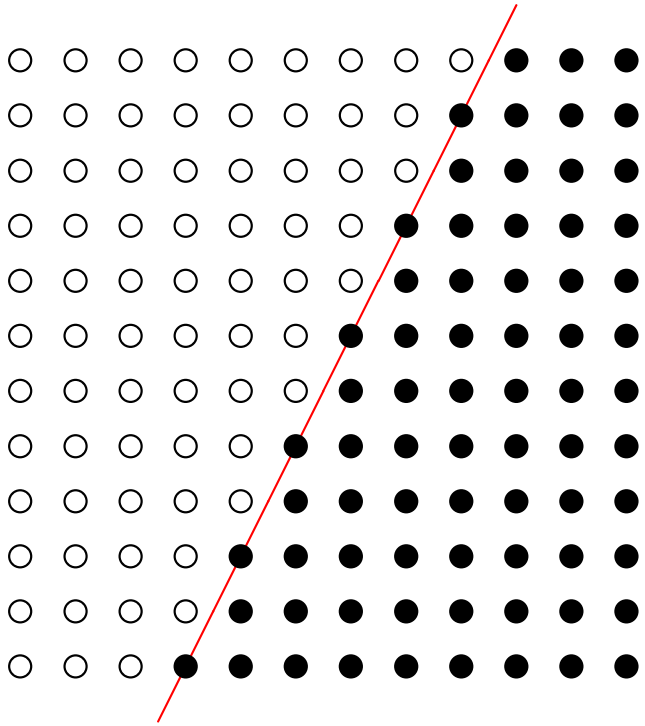


A HALF-PLANE

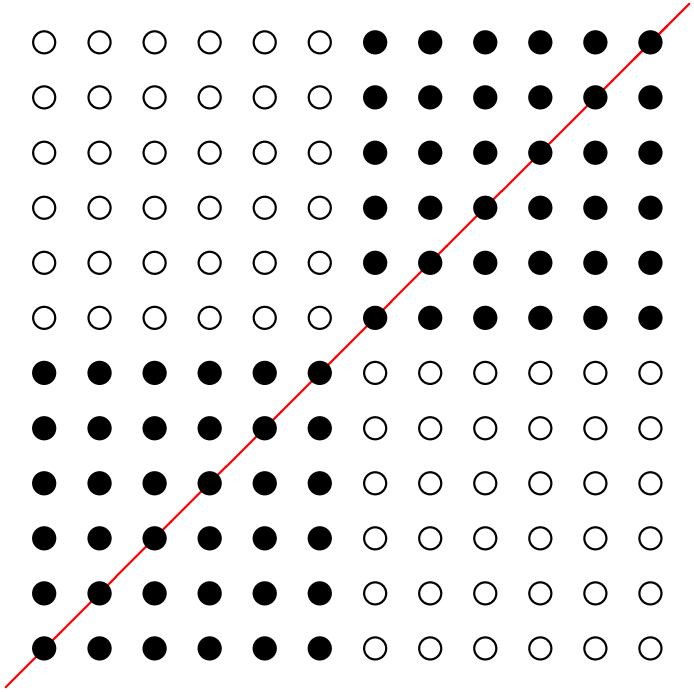


$\frac{1}{4}$-FAR FROM A HALF-PLANE

Half-plane instances

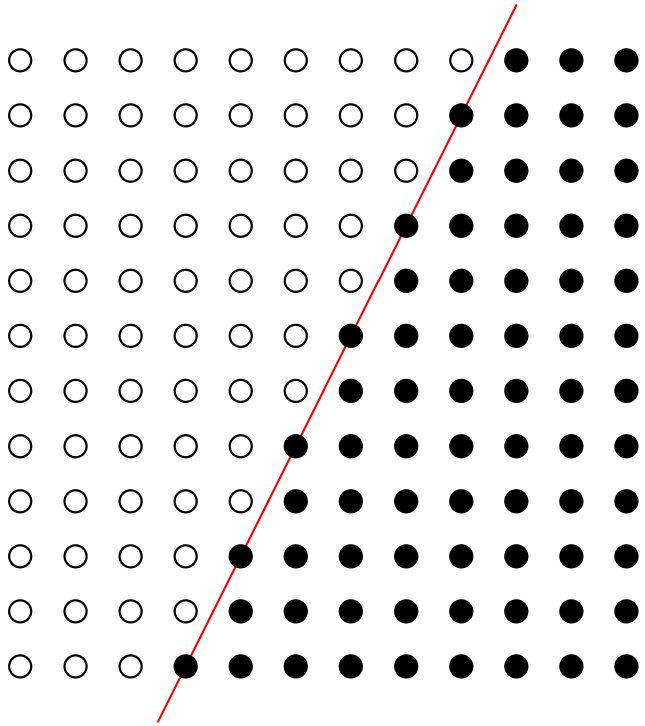


A HALF-PLANE

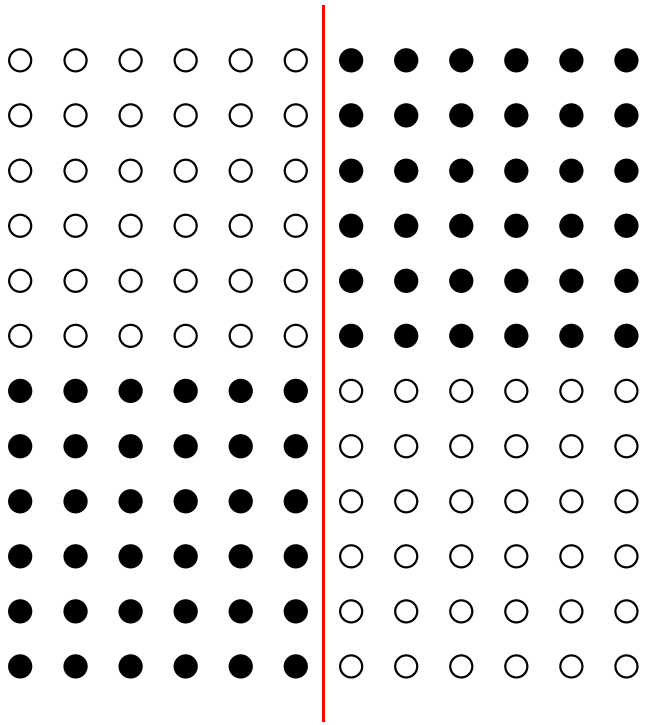


$\frac{1}{4}$-FAR FROM A HALF-PLANE

Half-plane instances

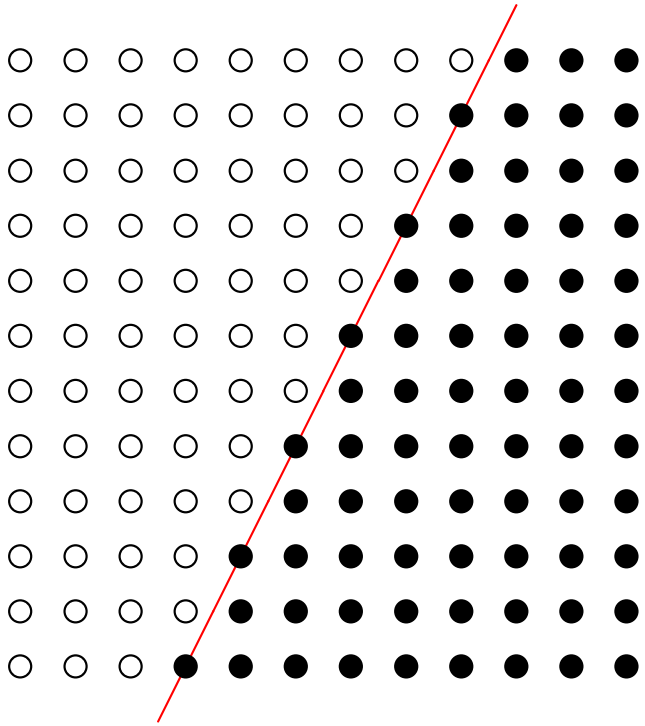


A HALF-PLANE

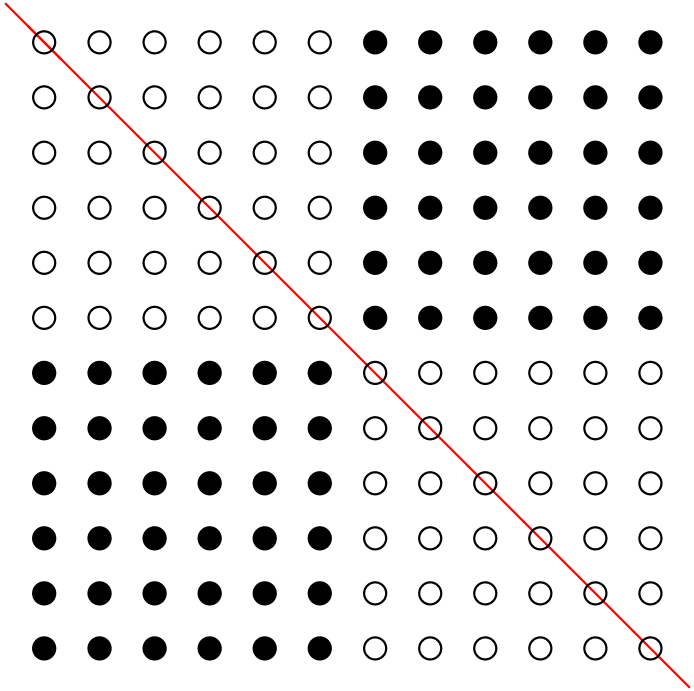


$\frac{1}{4}$-FAR FROM A HALF-PLANE

Half-plane instances

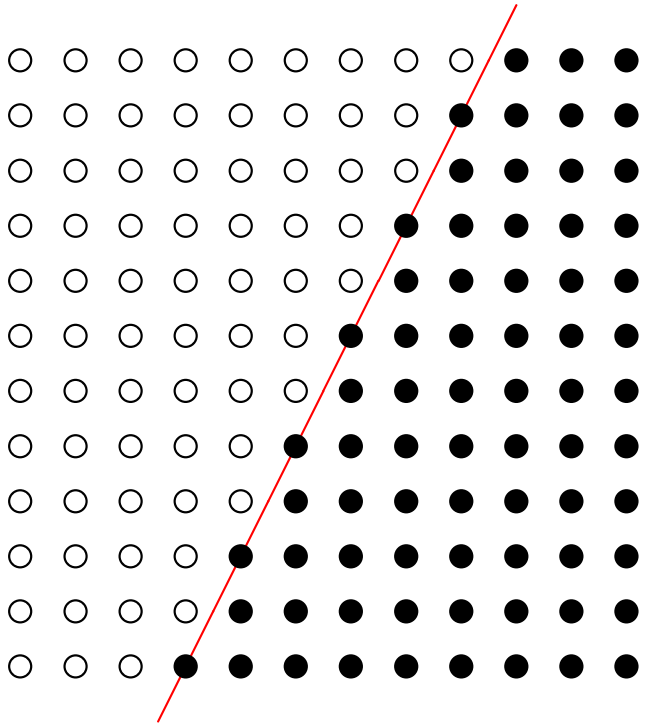


A HALF-PLANE

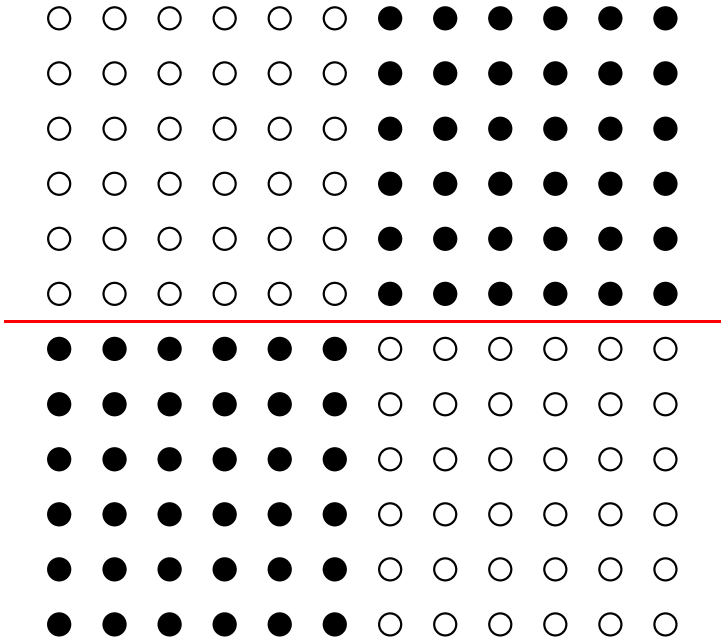


$\frac{1}{4}$-FAR FROM A HALF-PLANE

Half-plane instances

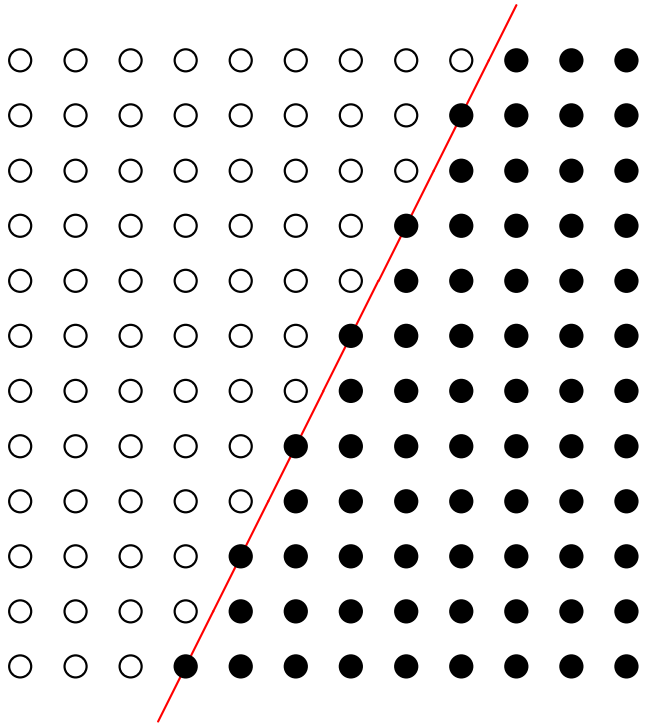


A HALF-PLANE

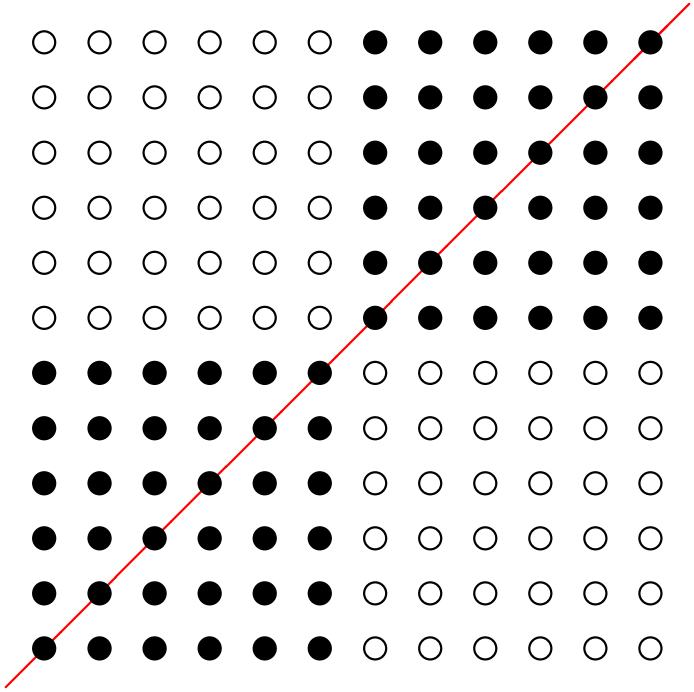


$\frac{1}{4}$-FAR FROM A HALF-PLANE

Half-plane instances

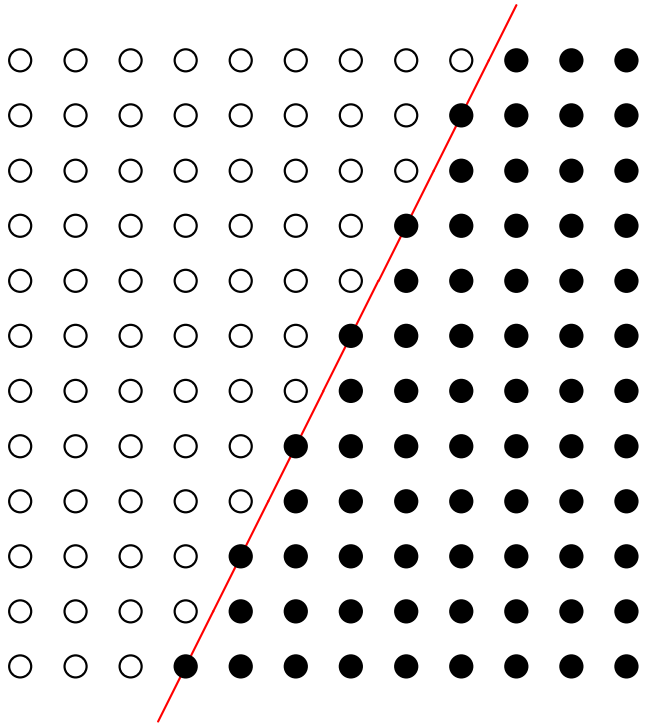


A HALF-PLANE

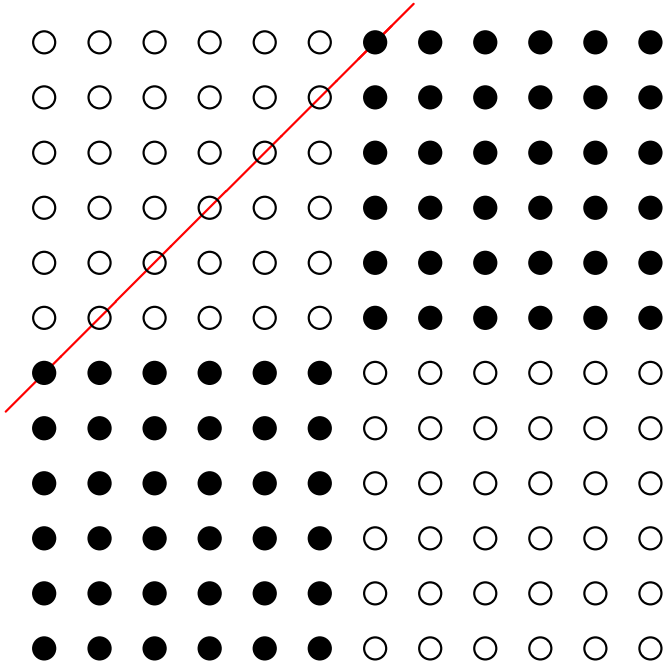


$\frac{1}{4}$-FAR FROM A HALF-PLANE

Half-plane instances

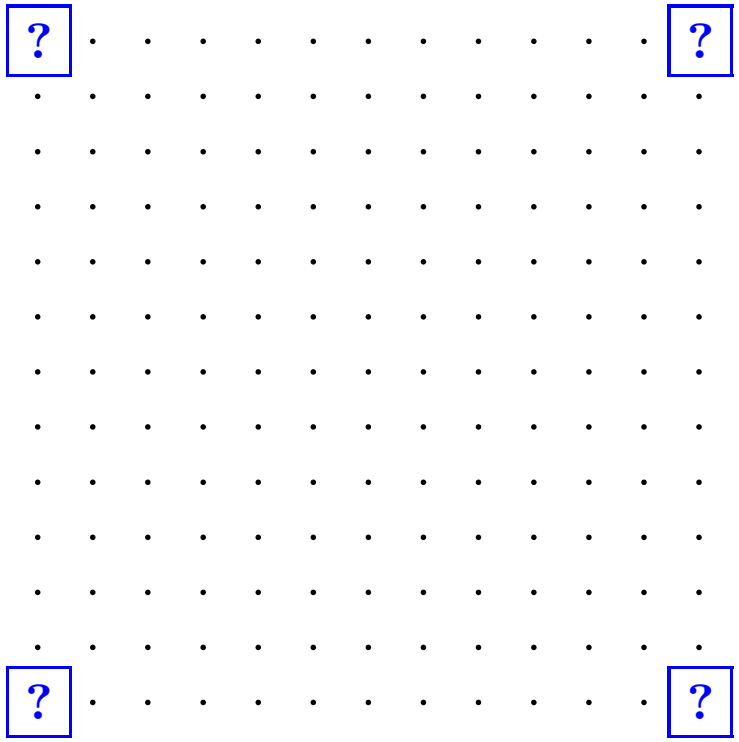


A HALF-PLANE



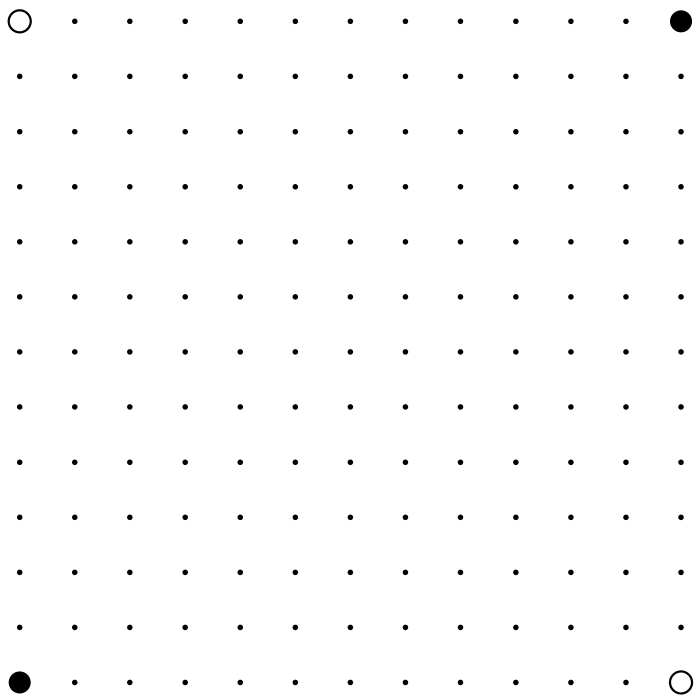
$\frac{1}{4}$-FAR FROM A HALF-PLANE

Half-plane test



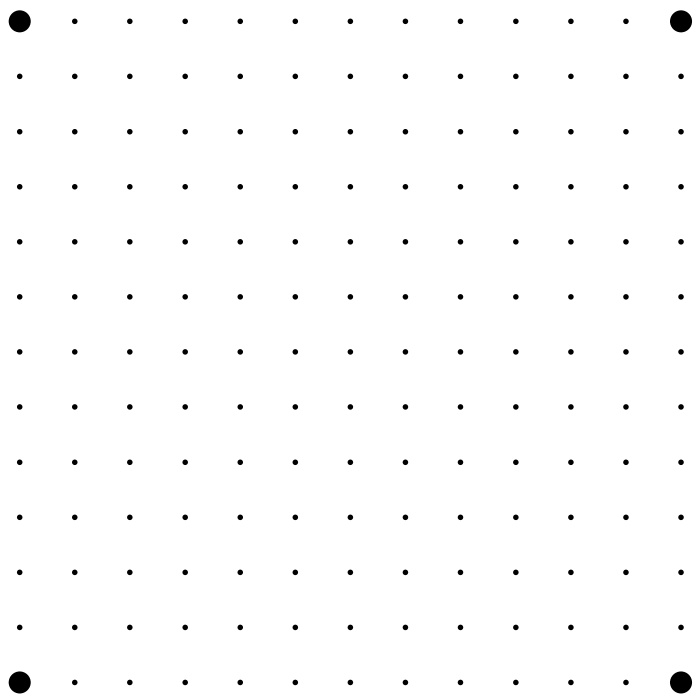
1. *Query the corners.*
2. (The number of sides with different corners is 0,2 or 4.)

Half-plane test: 4 bi-colored sides



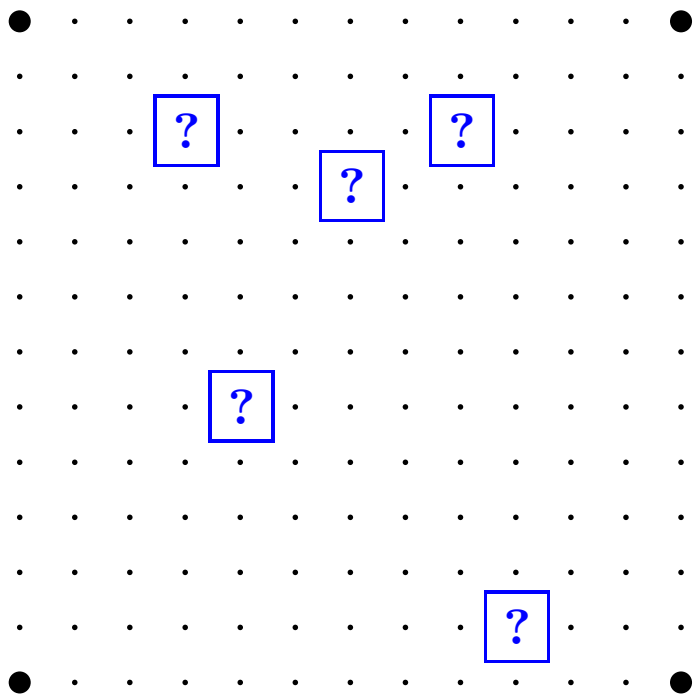
1. *Query the corners.*
2. *If the number of sides with different corners is 4, reject.*
   (The image cannot be a half-plane.)

Half-plane test: 0 bi-colored sides



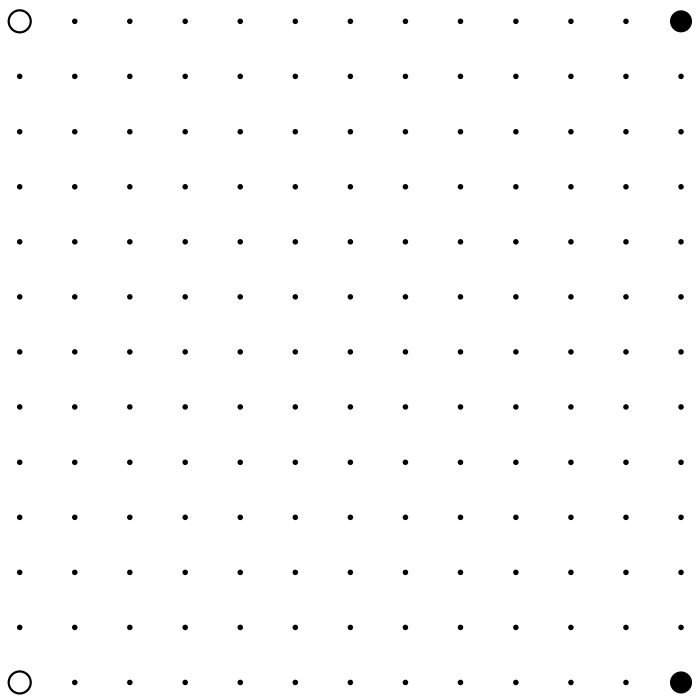
1. *Query the corners.*
2. (If all corners are colored c,
   the image is a half-plane iff it is unicolored.)

Half-plane test: 0 bi-colored sides



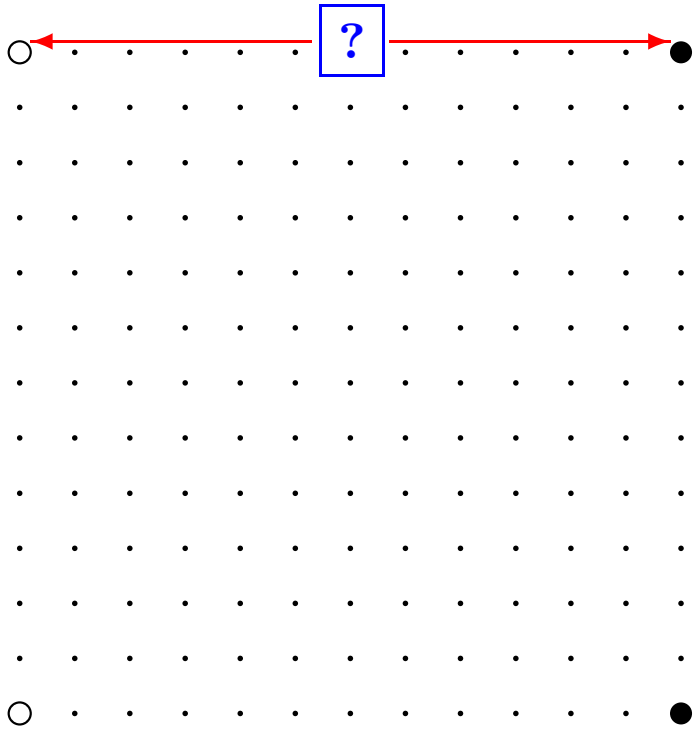
1. *Query the corners.*
2. (If all corners are colored c,
   the image is a half-plane iff it is unicolored.)
3. *Query* $\frac{\ln 3}{\varepsilon}$ *random pixels.*
   Accept if all of them are colored c*, otherwise reject.*

Half-plane test: 2 bi-colored sides



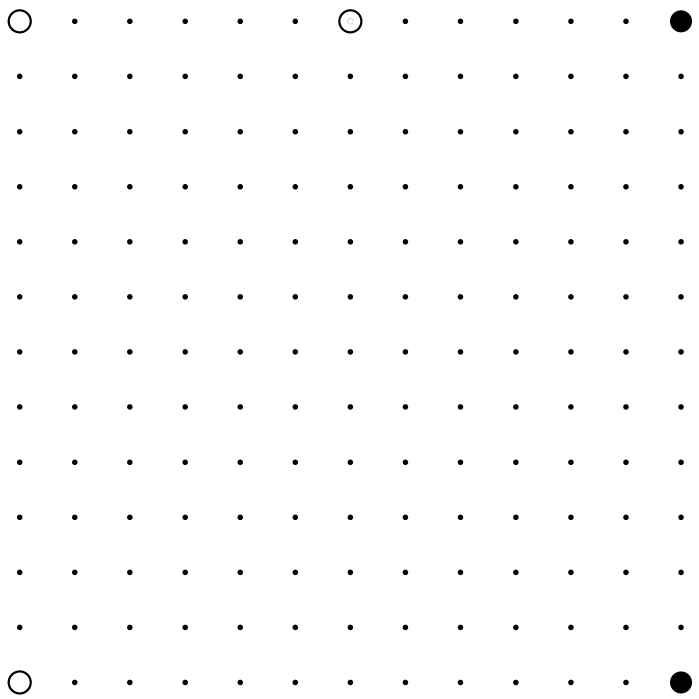
1. *Query the corners.*
2. *If the number of sides with different corners is 2, on both find 2 different pixels within distance* $< \frac{\varepsilon n}{2}$ *by binary search.*

Half-plane test: 2 bi-colored sides



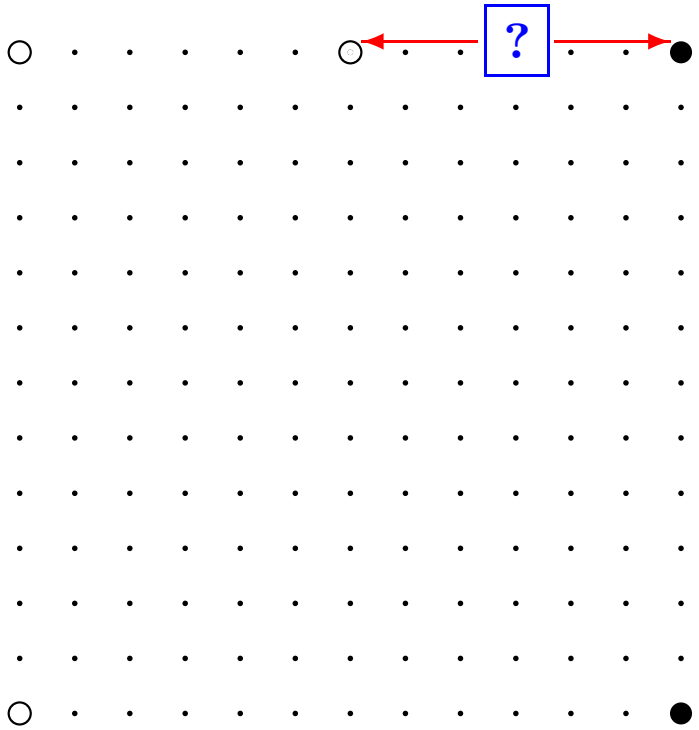
1. *Query the corners.*
2. *If the number of sides with different corners is 2, on both find 2 different pixels within distance* $< \frac{\varepsilon n}{2}$ *by binary search.*

Half-plane test: 2 bi-colored sides



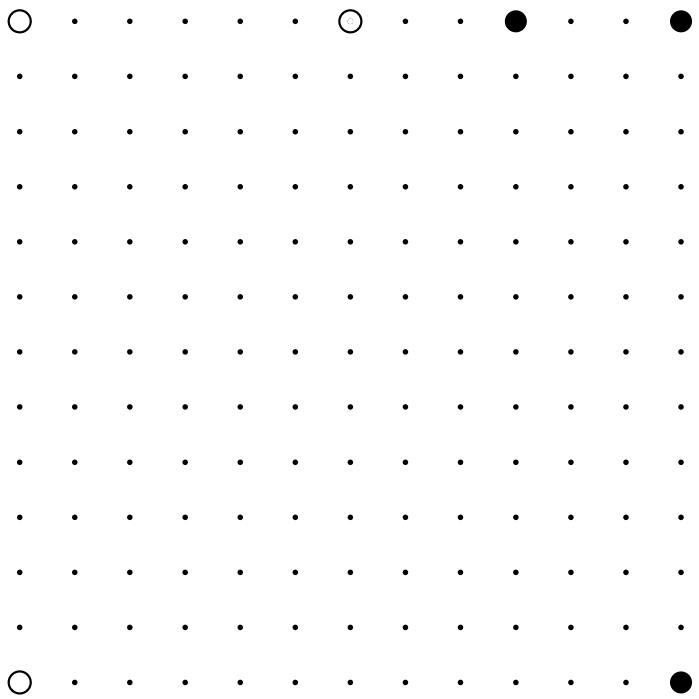
1. *Query the corners.*
2. *If the number of sides with different corners is 2, on both find 2 different pixels within distance* $< \frac{\varepsilon n}{2}$ *by binary search.*

Half-plane test: 2 bi-colored sides



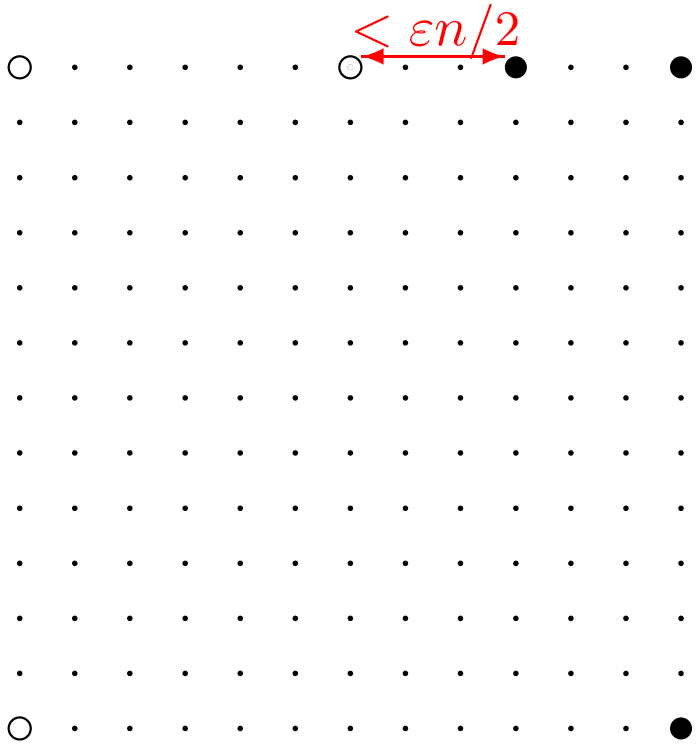
1. *Query the corners.*
2. *If the number of sides with different corners is 2, on both find 2 different pixels within distance* $< \frac{\varepsilon n}{2}$ *by binary search.*

Half-plane test: 2 bi-colored sides



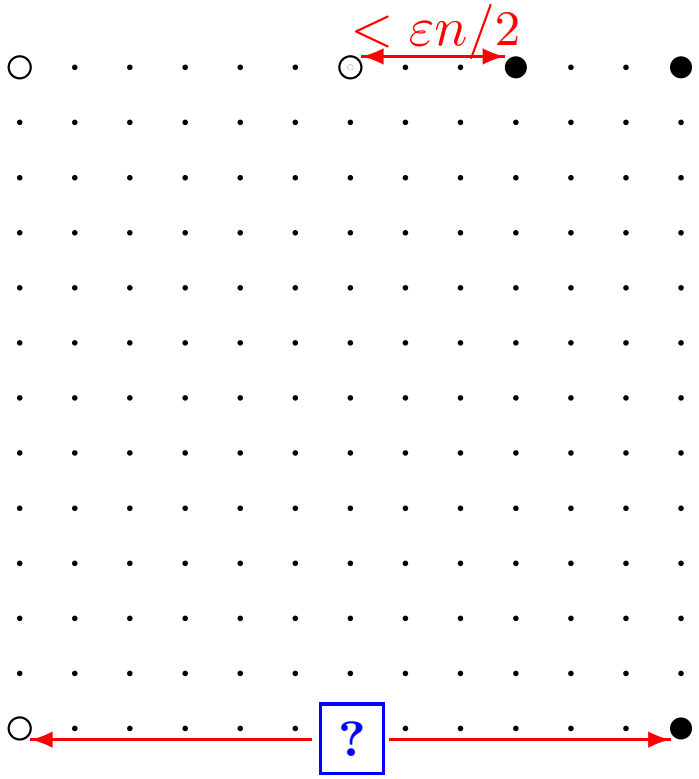
1. *Query the corners.*
2. *If the number of sides with different corners is 2, on both find 2 different pixels within distance* $< \frac{\varepsilon n}{2}$ *by binary search.*

Half-plane test: 2 bi-colored sides



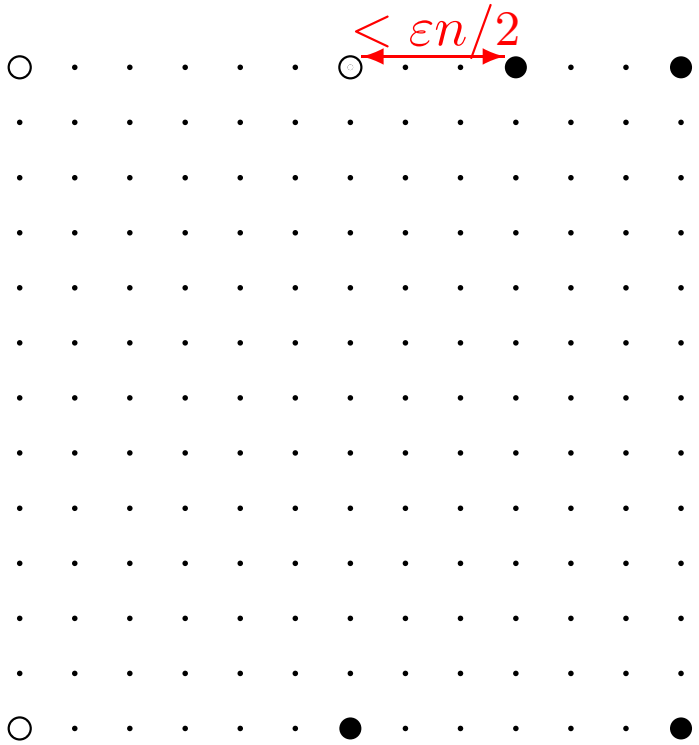
1. *Query the corners.*
2. *If the number of sides with different corners is 2, on both find 2 different pixels within distance* $< \frac{\varepsilon n}{2}$ *by binary search.*

Half-plane test: 2 bi-colored sides



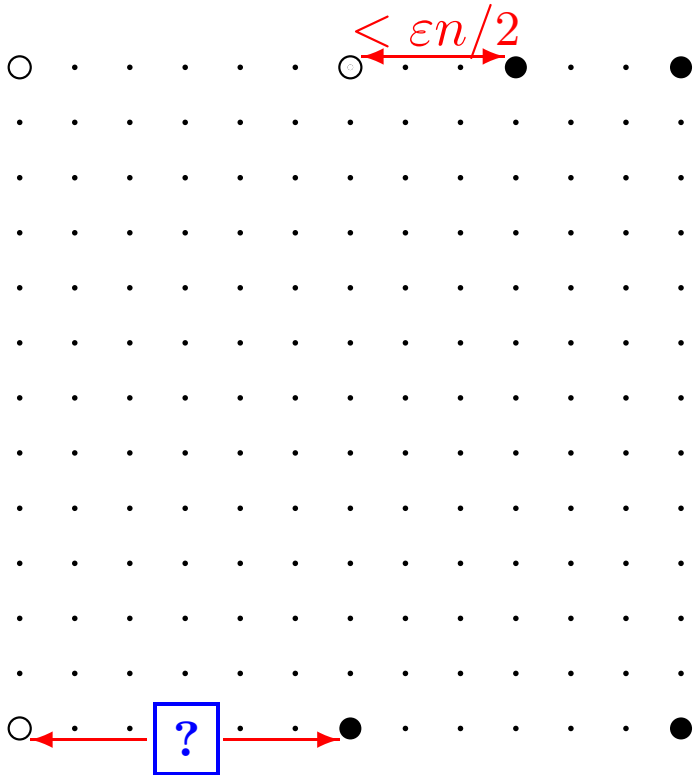
1. *Query the corners.*
2. *If the number of sides with different corners is 2, on both find 2 different pixels within distance* $< \frac{\varepsilon n}{2}$ *by binary search.*

Half-plane test: 2 bi-colored sides



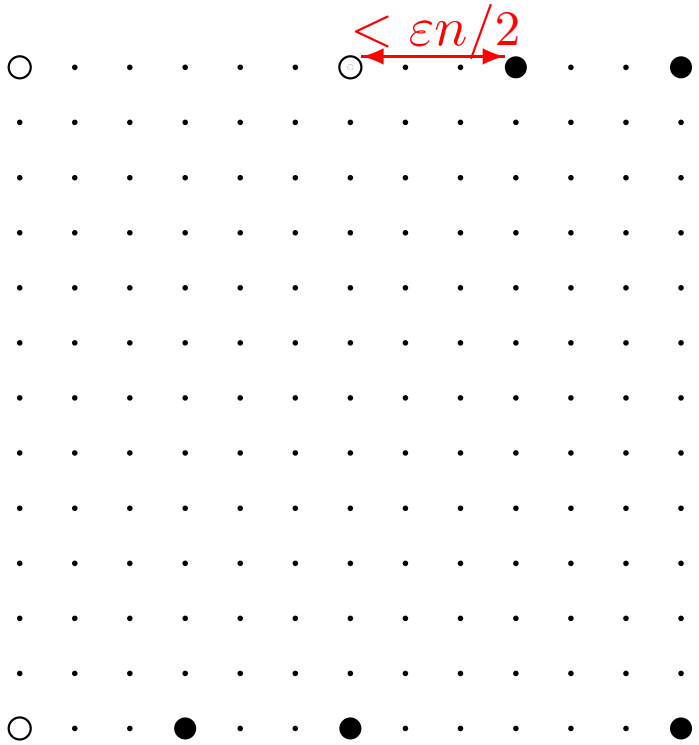
1. *Query the corners.*
2. *If the number of sides with different corners is 2, on both find 2 different pixels within distance* $< \frac{\varepsilon n}{2}$ *by binary search.*

Half-plane test: 2 bi-colored sides



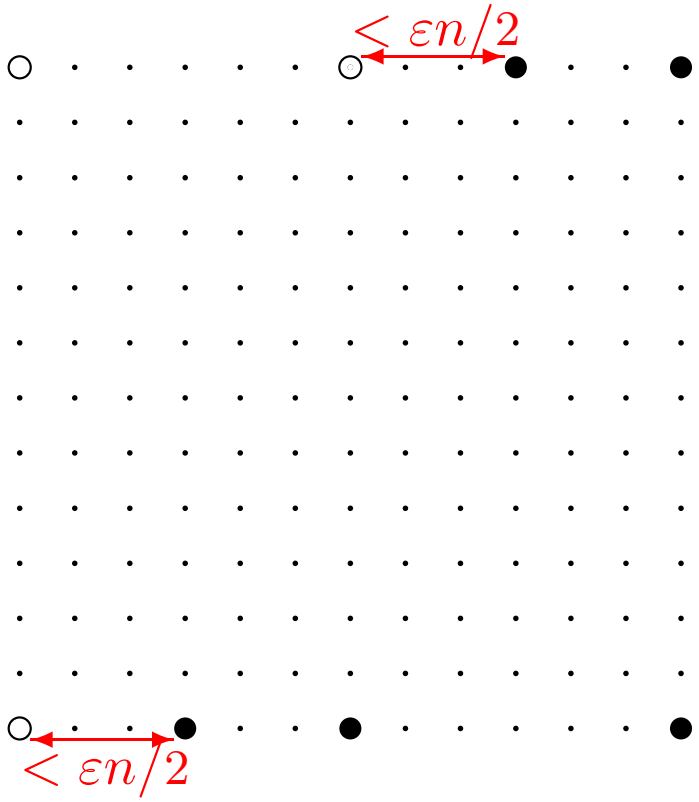
1. *Query the corners.*
2. *If the number of sides with different corners is 2, on both find 2 different pixels within distance* $< \frac{\varepsilon n}{2}$ *by binary search.*

Half-plane test: 2 bi-colored sides



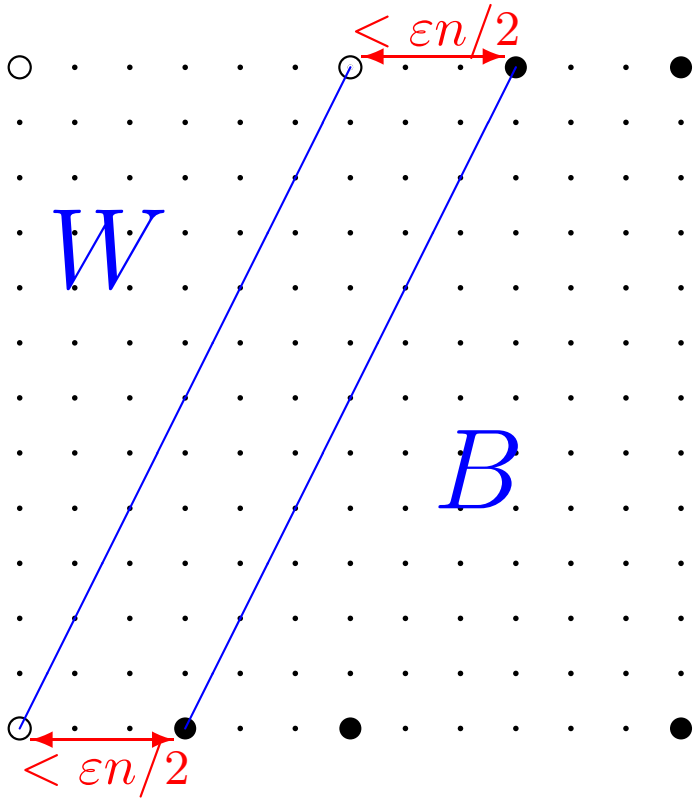
1. *Query the corners.*
2. *If the number of sides with different corners is 2, on both find 2 different pixels within distance* $< \frac{\varepsilon n}{2}$ *by binary search.*

Half-plane test: 2 bi-colored sides



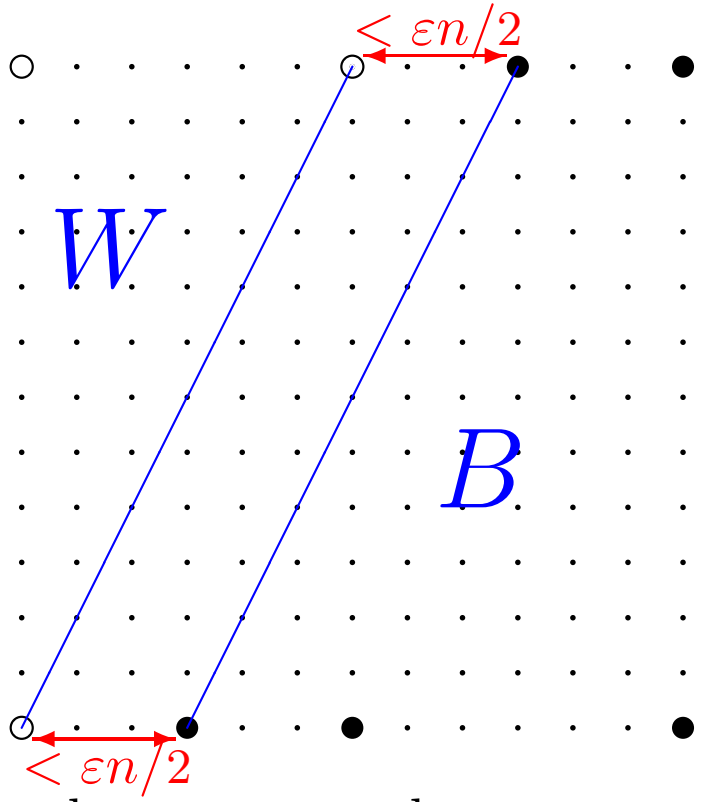
1. *Query the corners.*
2. *If the number of sides with different corners is 2, on both find 2 different pixels within distance* $< \frac{\varepsilon n}{2}$ *by binary search.*

Half-plane test: 2 bi-colored sides



1. *Query the corners.*
2. *If the number of sides with different corners is 2, on both find 2 different pixels within distance* $< \frac{\varepsilon n}{2}$ *by binary search.*
3. *Query* $\frac{2 \ln 3}{\varepsilon}$ *pixels from* $W \cup B$*.*
   Accept if all W *pixels are white, all* B *pixels are black.*

Half-plane test: correctness



- All half-planes are always accepted.
- ε-far images are rejected with probability $\geq \frac{2}{3}$.

  Contrapositive: If an image is rejected with probability $< \frac{2}{3}$,

  $W \cup B$ has $< \frac{\varepsilon n^2}{2}$ wrong pixels.

  Changing $< \varepsilon n^2$ pixels turns the image into a half-plane.

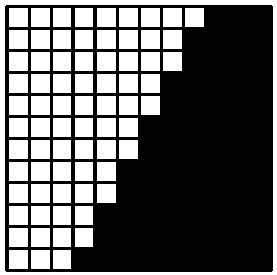
Testing convexity

A half-plane or

ε-far from a half-plane?

1-sided error

$O(1/\varepsilon)$ queries

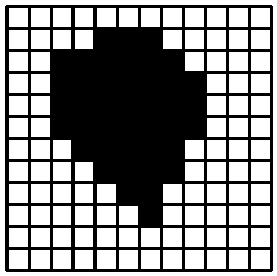


Convex or

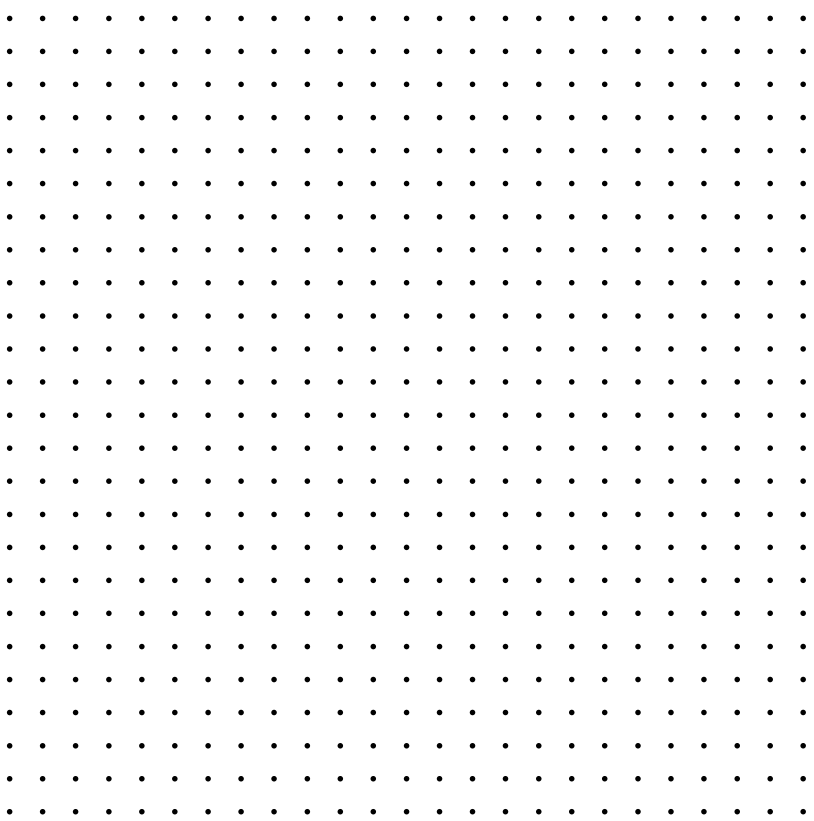
ε-far from a convex image?

2-sided error

$O(1/\varepsilon^2)$ queries

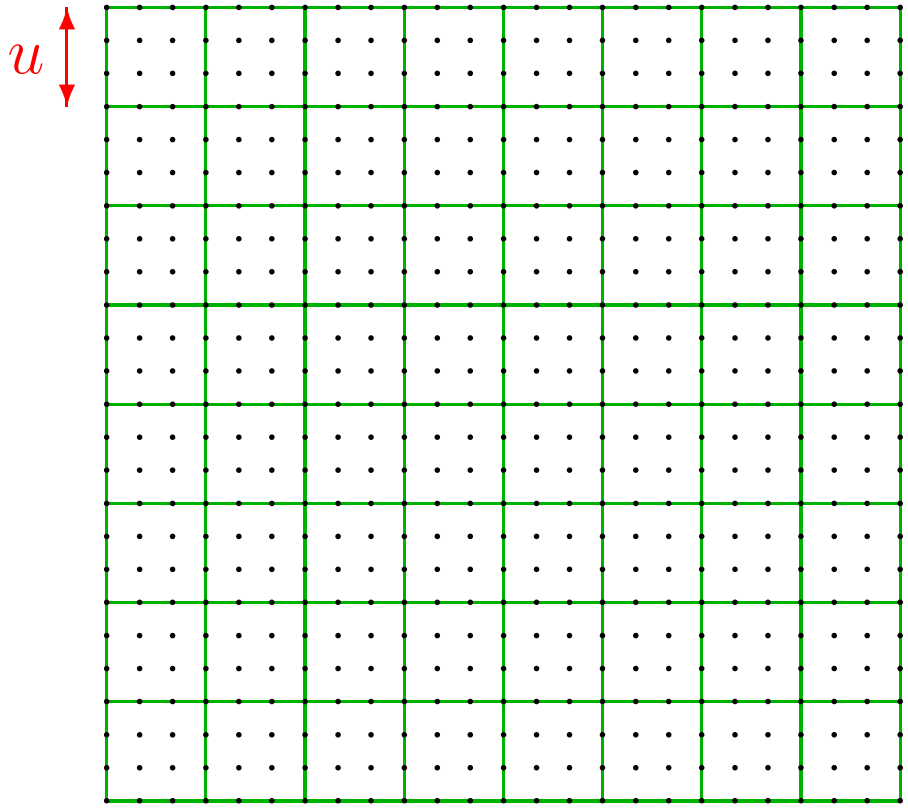


Convexity test



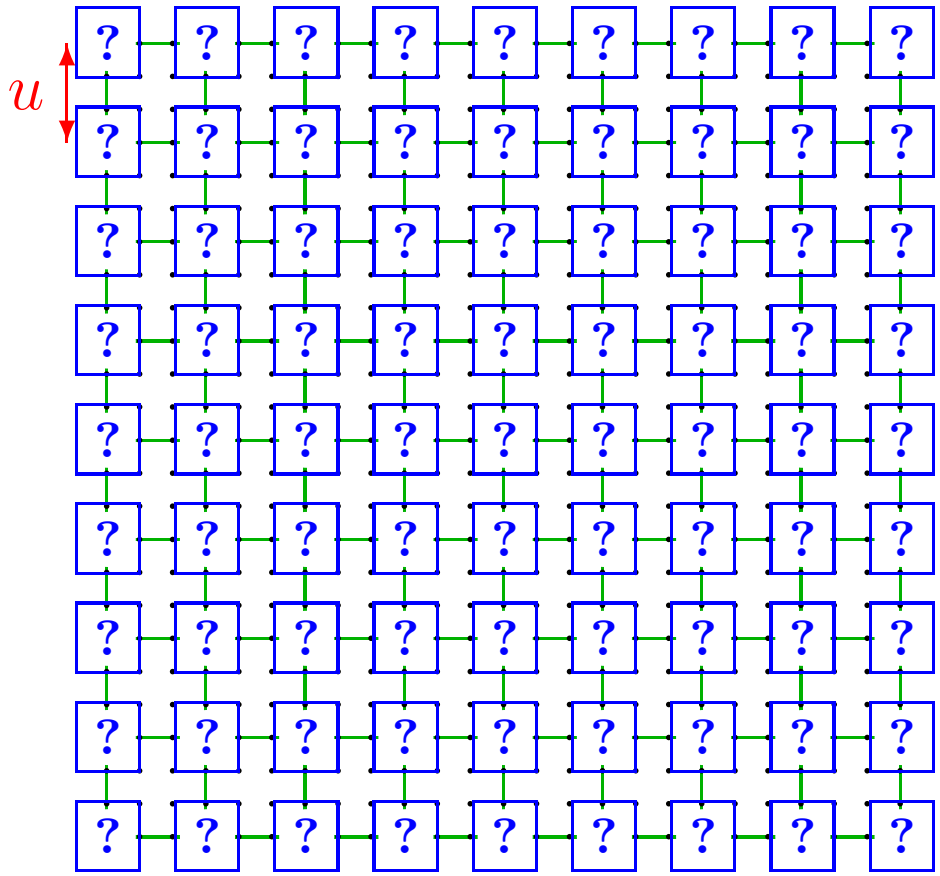
1. *Query all pixels with both coordinates divisible by* $u = \Theta(\varepsilon n)$.

Convexity test



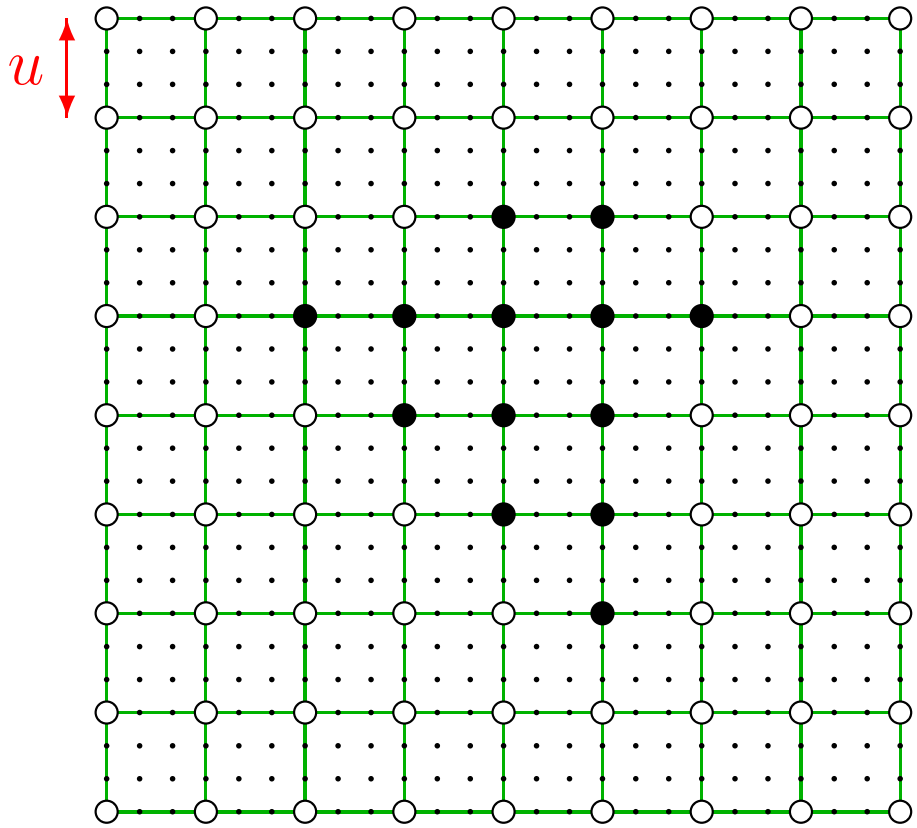
1. Query all pixels with both coordinates divisible by $u = \Theta(\varepsilon n)$.

Convexity test



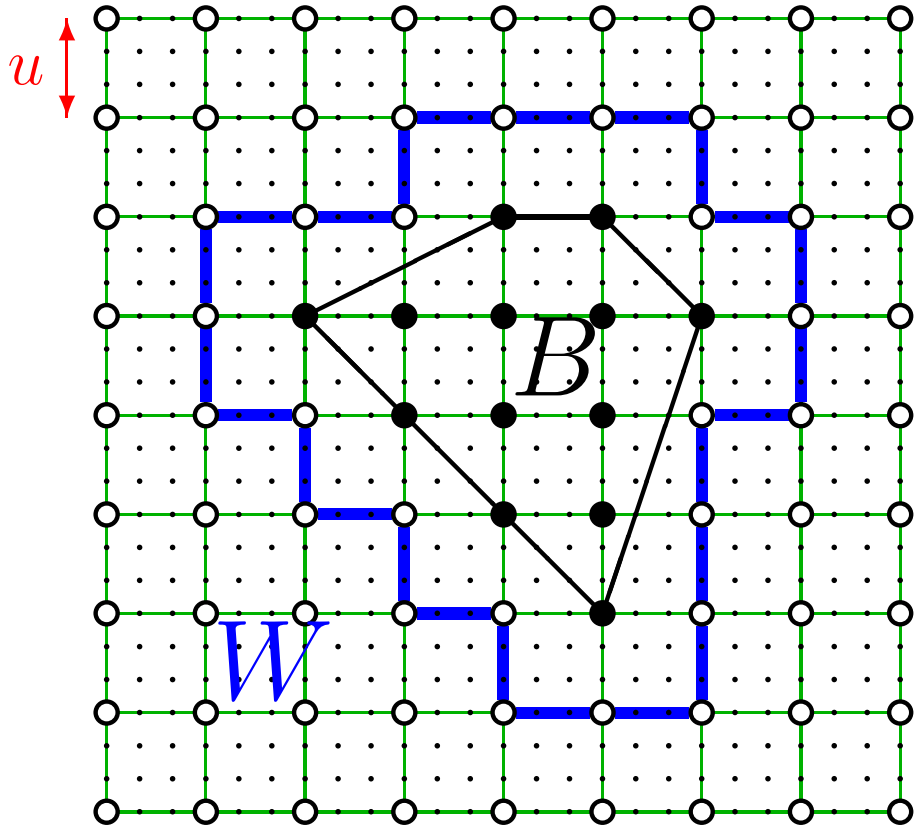
1. Query all pixels with both coordinates divisible by $u = \Theta(\varepsilon n)$.

Convexity test



1. Query all pixels with both coordinates divisible by $u = \Theta(\varepsilon n)$.

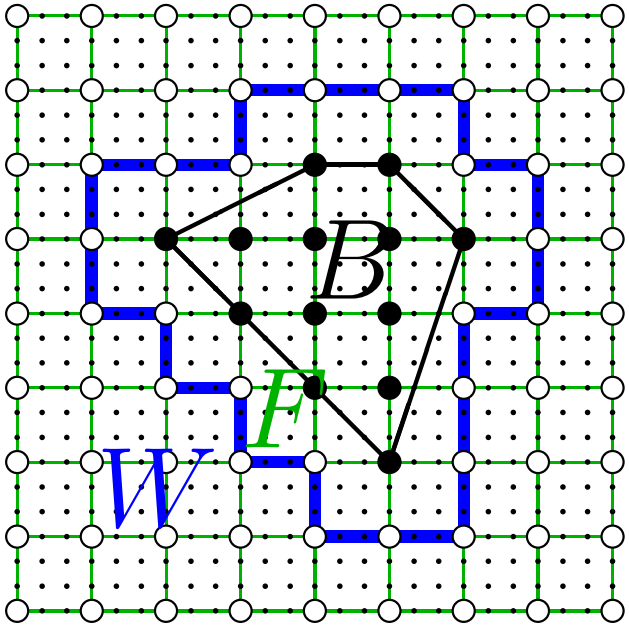
Convexity test



1. *Query all pixels with both coordinates divisible by* $u = \Theta(\varepsilon n)$.

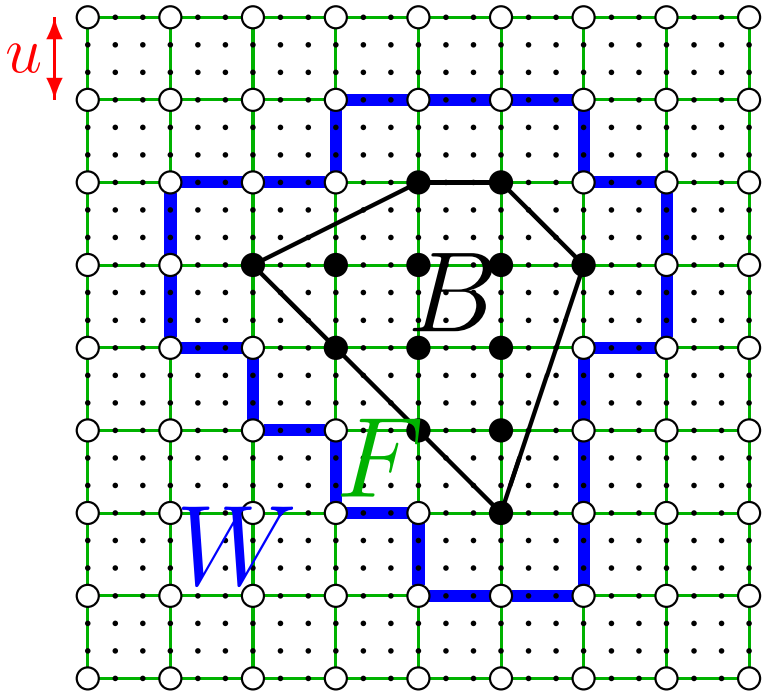
2. *Query* $O(\frac{1}{\varepsilon})$ *pixels from* B *and* W.
   Accept if all B *pixels are black, and all* W *pixels are white.*

Convexity test correctness



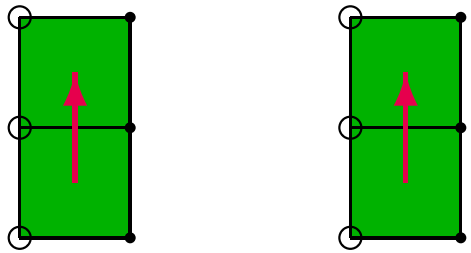
- If an image is convex,
  - B has no white pixels.
  - W has a small fraction of black pixels [Lemma 2].
- If an image is rejected with probability $< \frac{2}{3}$,
  - $B \cup W$ has a small fraction of wrong pixels.
  - F is a small fraction of the image [Lemma 1].
  - The image is close to convex.

Convexity test correctness: Lemma 1



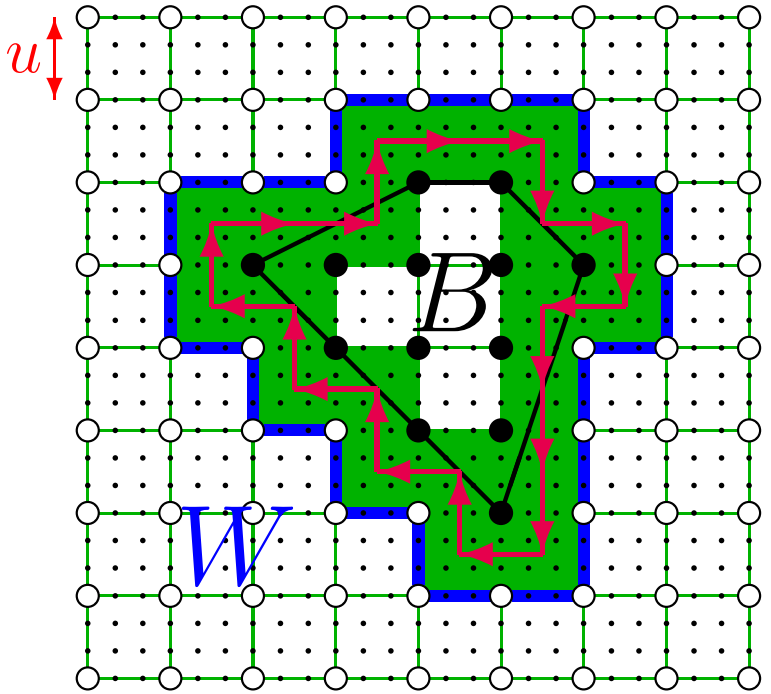
Lemma 1 *F contains $\leq 4un$ pixels.*

- Count the squares covering F.
- Walk on centers of the squares.
- **Claim**: The walk can go up (down) at most once per row.



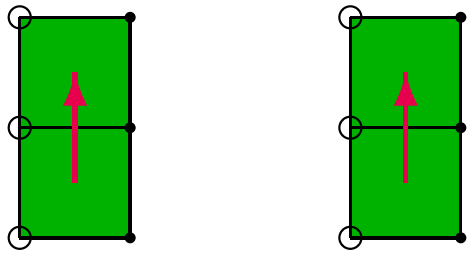
- There are $\frac{n}{u}$ rows (columns) of squares.

Convexity test correctness: Lemma 1



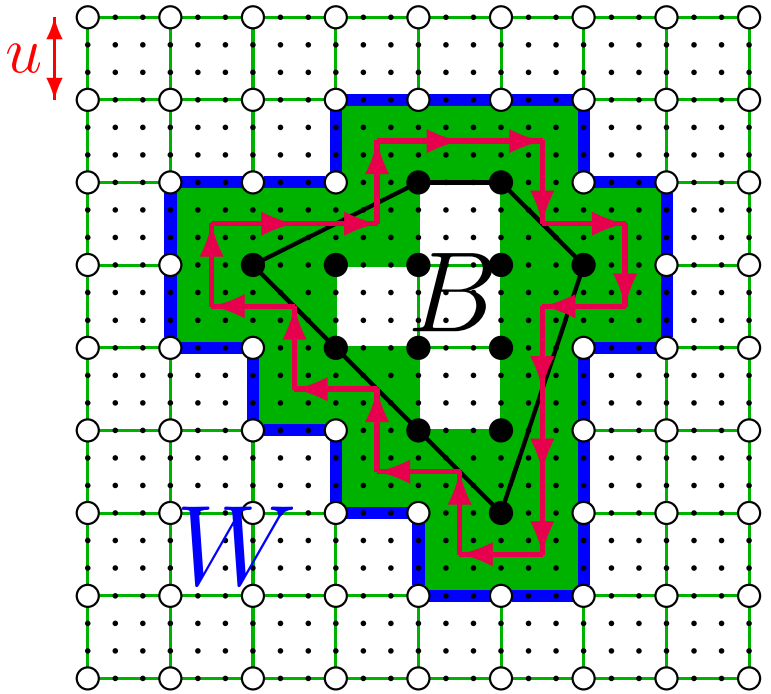
Lemma 1 *F contains $\leq 4un$ pixels.*

- Count the squares covering F.
- Walk on centers of the squares.
- **Claim**: The walk can go up (down) at most once per row.



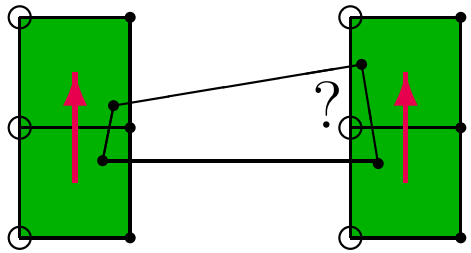
- There are $\frac{n}{u}$ rows (columns) of squares.

Convexity test correctness: Lemma 1



Lemma 1 *F contains $\leq 4un$ pixels.*

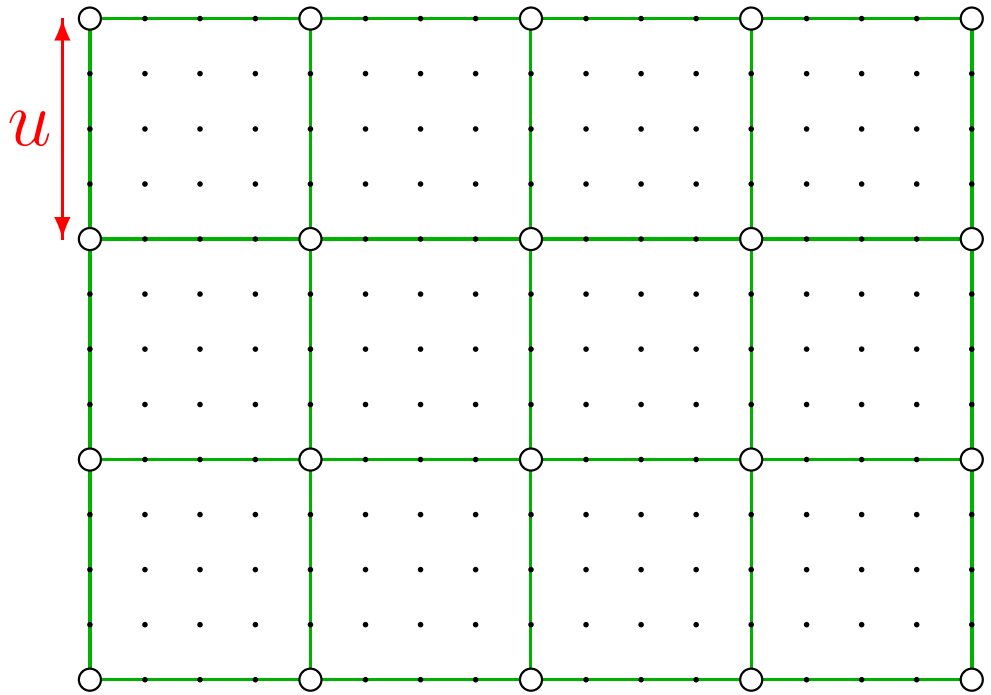
- Count the squares covering F.
- Walk on centers of the squares.
- **Claim**: The walk can go up (down) at most once per row.



- There are $\frac{n}{u}$ rows (columns) of squares.

Convexity test: "invisible" objects

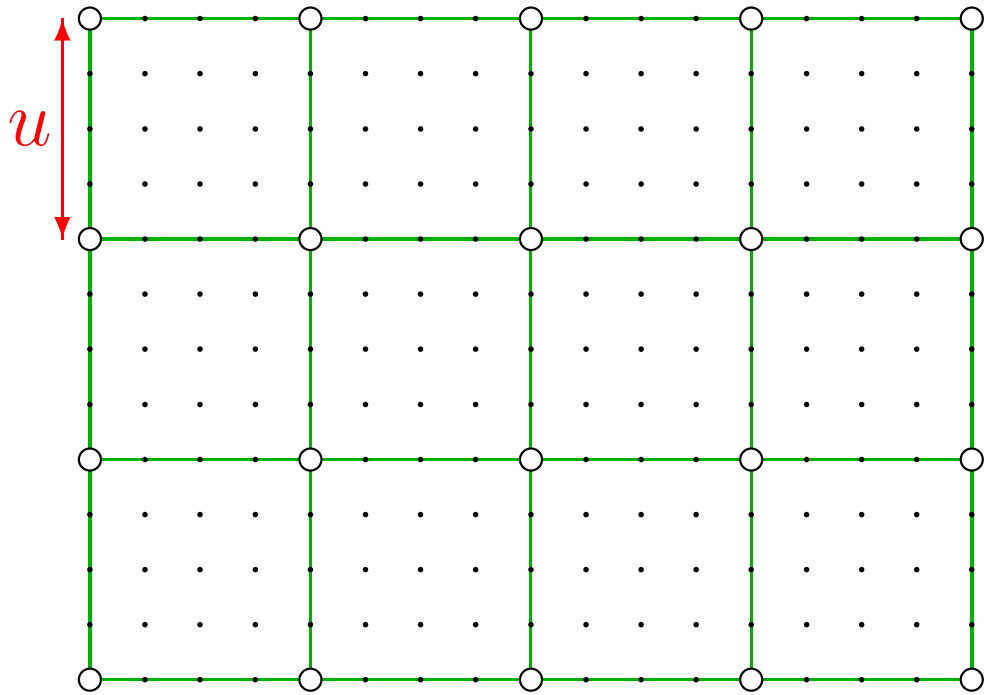
Claim *An "invisible" convex object contains* $< 2un$ *pixels.*



Convexity test: "invisible" objects

Claim *An "invisible" convex object contains* $< 2un$ *pixels.*

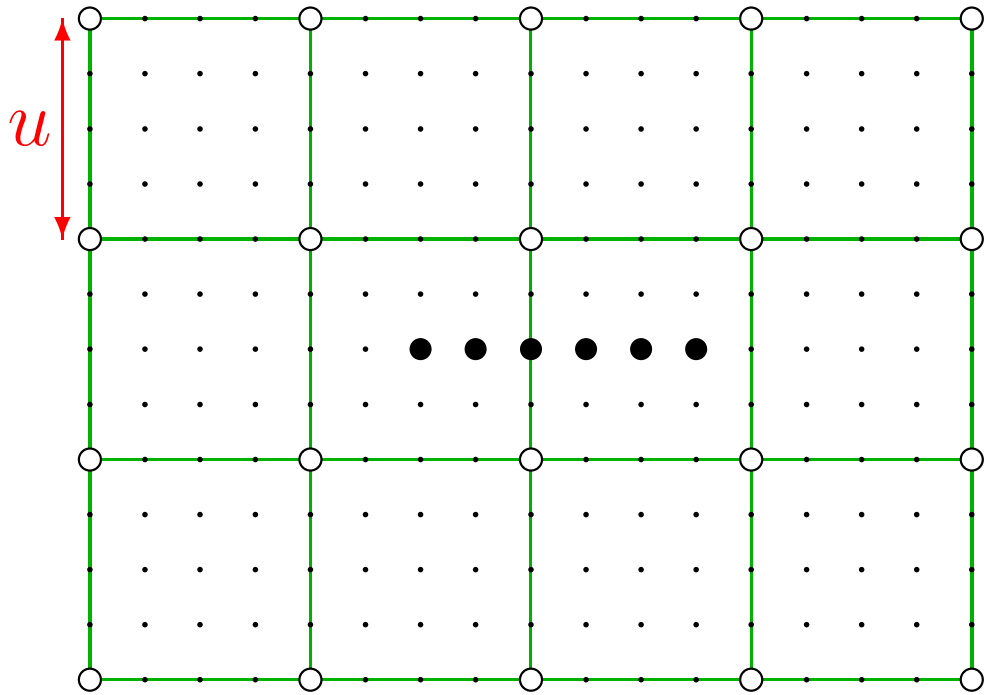
- Suppose some row r has $\geq u$ black pixels (o.w. done).



Convexity test: "invisible" objects

Claim *An "invisible" convex object contains* $< 2un$ *pixels.*

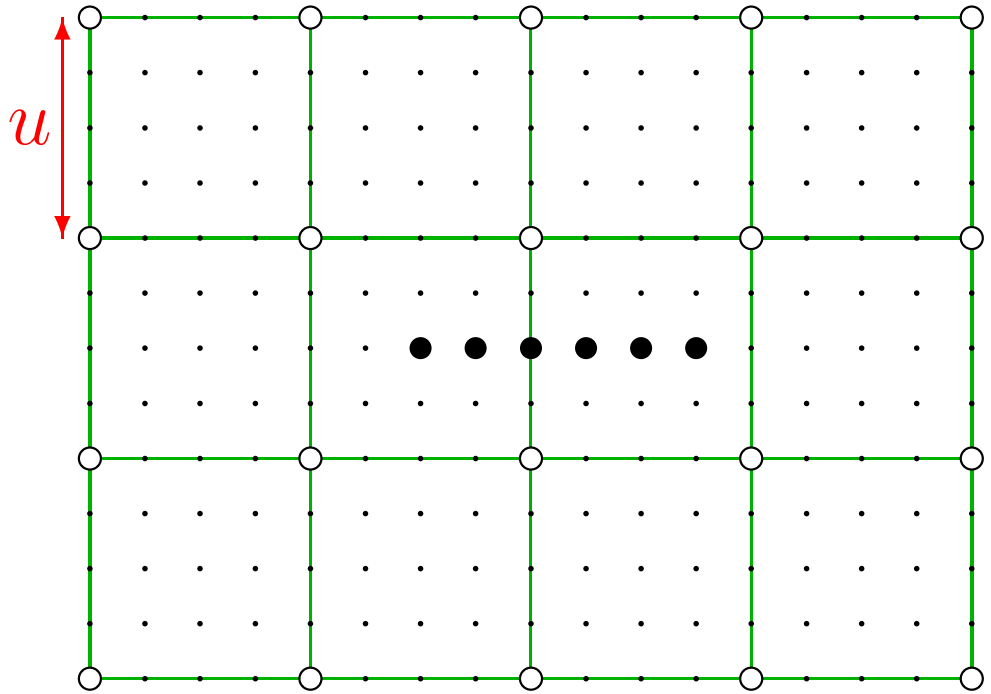
- Suppose some row r has $\geq u$ black pixels (o.w. done).



Convexity test: "invisible" objects

Claim *An "invisible" convex object contains* $< 2un$ *pixels.*

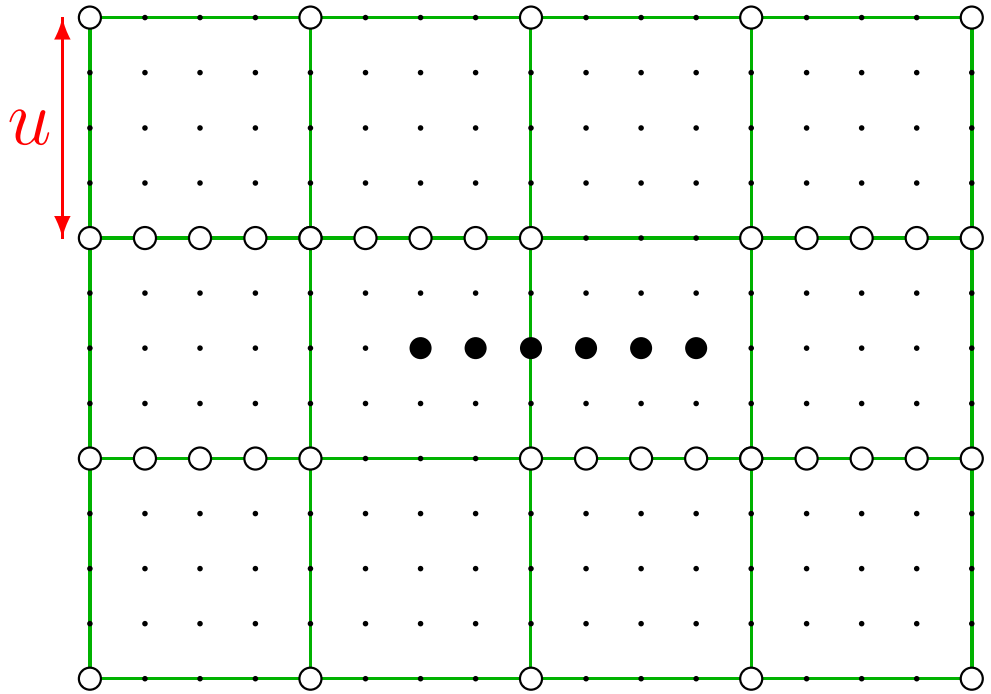
- Suppose some row r has $\geq u$ black pixels (o.w. done).
- In a fixed row, black pixels are consecutive.
- Green row above (below) r contains $< u$ black pixels.



Convexity test: "invisible" objects

Claim *An "invisible" convex object contains* $< 2un$ *pixels.*

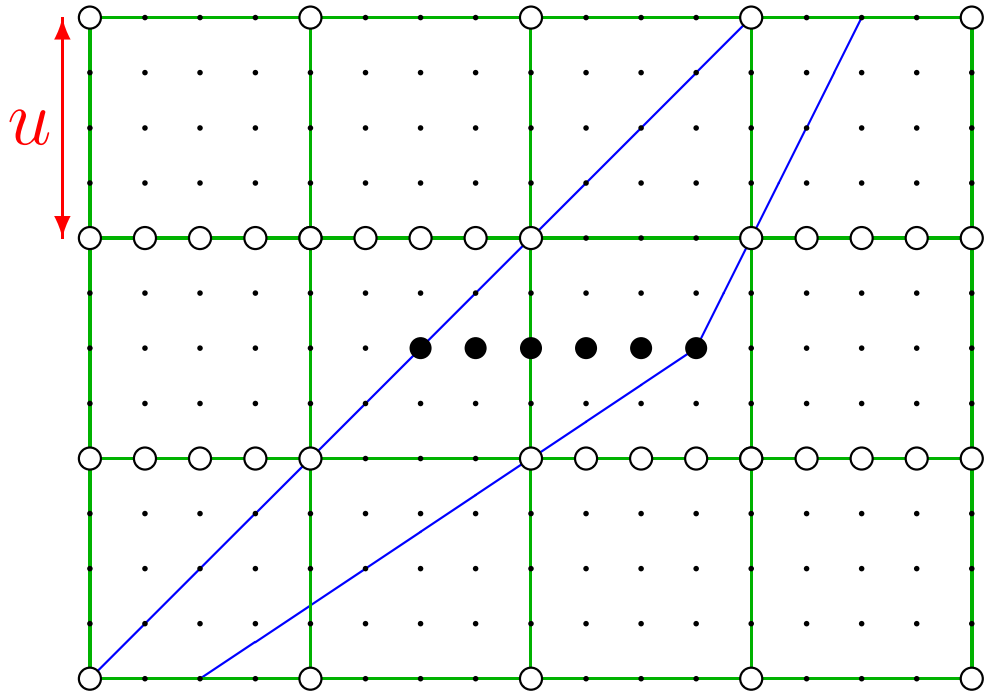
- Suppose some row r has $\geq u$ black pixels (o.w. done).
- In a fixed row, black pixels are consecutive.
- Green row above (below) r contains $< u$ black pixels.



Convexity test: "invisible" objects

Claim *An "invisible" convex object contains* $< 2un$ *pixels.*

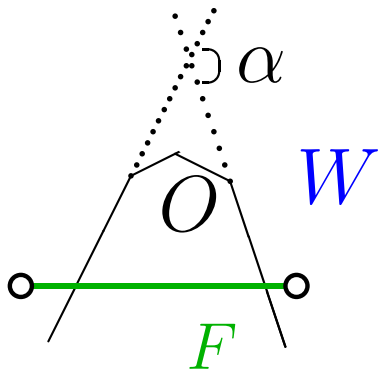
- Suppose some row r has $\geq u$ black pixels (o.w. done).
- In a fixed row, black pixels are consecutive.
- Green row above (below) r contains $< u$ black pixels.



Convexity test correctness: Lemma 2

Lemma 2 *In a convex image, W has $\leq 8un$ black pixels.*

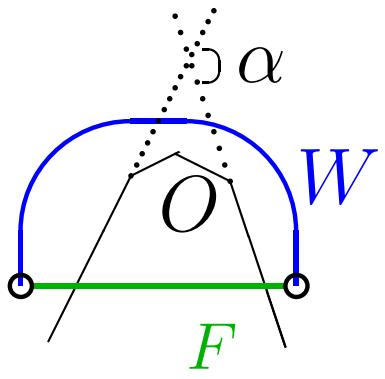
- A chunk of the object protruding from F into W is *large* if $\alpha > 90°$.
- ≤ 3 chunks are large, each with $< 2un$ pixels.



Convexity test correctness: Lemma 2

Lemma 2 *In a convex image, W has $\leq 8un$ black pixels.*

- A chunk of the object protruding from F into W is *large* if $\alpha > 90°$.
- ≤ 3 chunks are large, each with $< 2un$ pixels.
- Small chunks fall next to F, contain $< 2un$ pixels total.



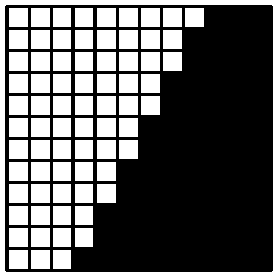
Done

A half-plane or

ε-far from a half-plane?

1-sided error

$O(1/\varepsilon)$ queries

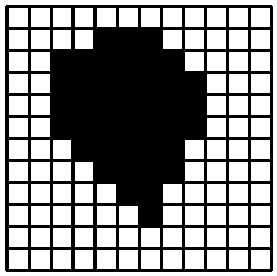


Convex or

ε-far from a convex image?

2-sided error

$O(1/\varepsilon^2)$ queries



Open questions

- Analyzing simpler tests:
  sample and accept iff no violation is found.
- Which visual properties are testable (not testable)?
- Matrices over larger alphabets.
- d-dimensional matrices.
- Something practical? E.g., distinguishing digits?