Sublinear-Time Algorithms
Lecture 1

Sofya Raskhodnikova
Penn State University

Thanks to Madhav Jha (Penn State) for help with creating these slides.

Tentative Plan

Lecture 1. Background. Testing properties of images and lists.

Lecture 2. Properties of graphs. Sublinear approximation.

Lecture 3. Properties of functions. Monotonicity and linearity testing.

Lecture 4. Techniques for proving hardness. Other models for sublinear computation.

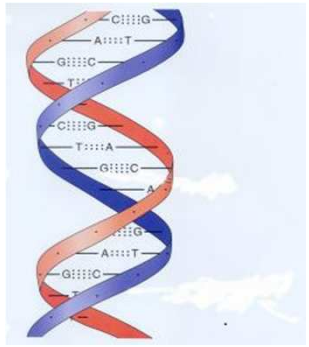
Motivation for Sublinear-Time Algorithms

Massive datasets

- world-wide web
- online social networks
- genome project
- sales logs
- census data
- high-resolution images
- scientific measurements

Long access time

- communication bottleneck (dial-up connection)
- implicit data (an experiment per data point)



Some material in this lecture is based on slides by Ronitt Rubinfeld:
http://stellar.mit.edu/S/course/6/fa10/6.896/courseMaterial/topics/topic3/lectureNotes/lecst11/lecst11.pdf

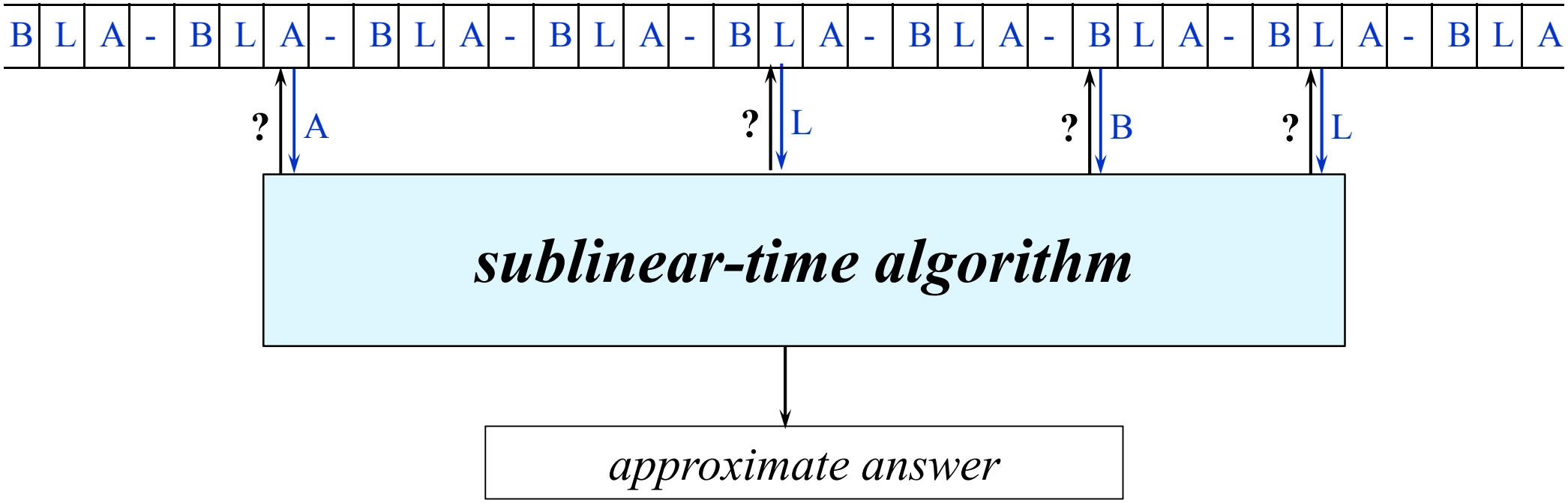
What Can We Hope For?

- What can an algorithm compute if it
  - reads only a **sublinear** portion of the data?
  - runs in **sublinear** time?

- Some problems have exact deterministic solutions

- For most interesting problems algorithms must be
  - approximate
  - randomized

A Sublinear-Time Algorithm



Quality of approximation

vs.

Resources

- number of queries
- running time

Types of Approximation

Classical approximation

- need to compute a value
  - output is close to the desired value
  - examples: average, 90th percentile
- need to compute the best structure
  - output is a structure with "cost" close to optimal
  - examples: furthest pair of points, minimum spanning tree

Property testing

- need to answer YES or NO
  - output is a correct answer for a given input,
    or at least some input close to it

Classical Approximation

A Simple Example

Approximate Diameter of a Point Set [Indyk]

Input: m points, described by a distance matrix D

- D_{ij} is the distance between points i and j
- D satisfies triangle inequality and symmetry

(Note: input size is $n = m^2$)

Let i, j be indices that maximize D_{ij}.

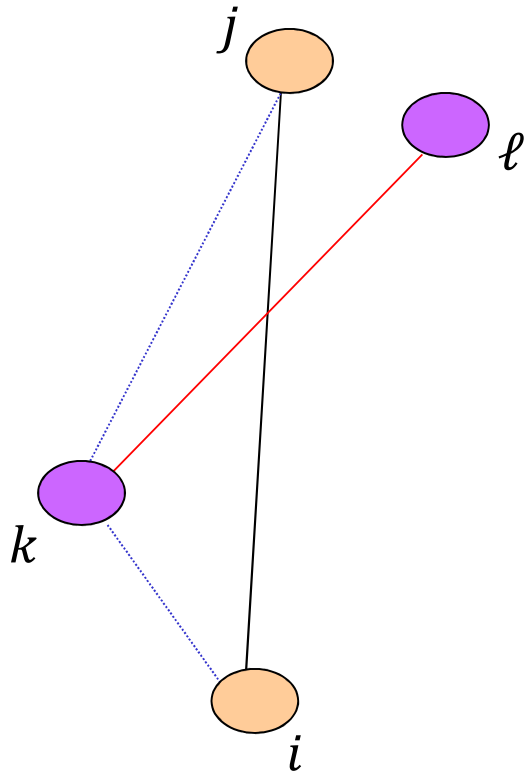
Maximum D_{ij} is the *diameter*.

- Output: (k, ℓ) such that $D_{k\ell} \geq D_{ij}/2$

Algorithm and Analysis

Algorithm (m, D)

1. Pick k arbitrarily
2. Pick ℓ to maximize $D_{k\ell}$
3. Output (k, ℓ)



- Approximation guarantee

$$D_{ij} \le D_{ik} + D_{kj} \text{ (triangle inequality)}$$
$$\le D_{k\ell} + D_{k\ell} \text{ (choice of } \ell \text{ + symmetry of } D)$$
$$\le 2D_{k\ell}$$

- Running time: $O(m) = O(m = \sqrt{n})$

A rare example of a deterministic sublinear-time algorithm

Property Testing

Property Testing: YES/NO Questions

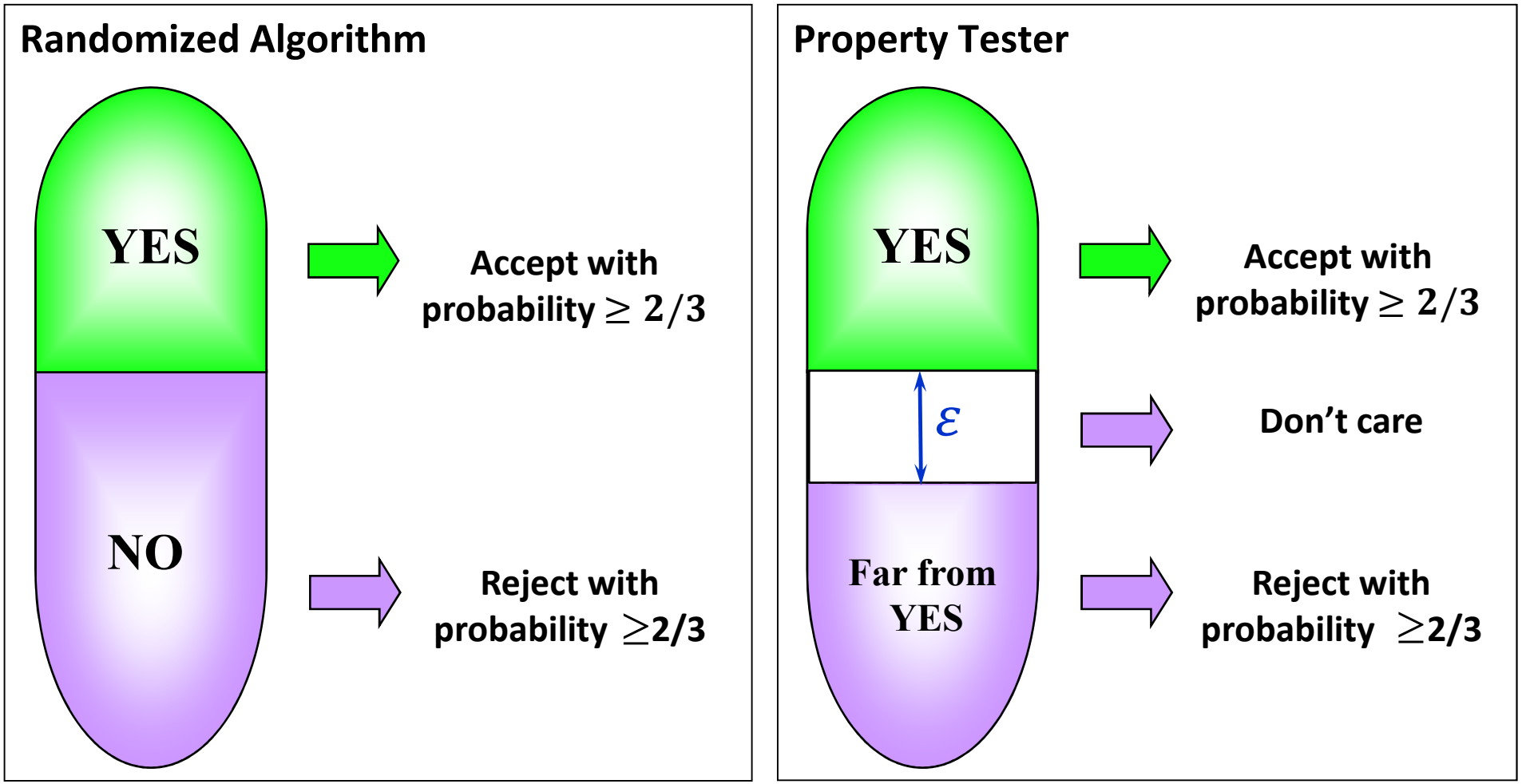
Does the input satisfy some property? (YES/NO)

"in the ballpark" vs. "out of the ballpark"

Does the input satisfy the property or is it far from satisfying it?

- sometimes it is the right question (probabilistically checkable proofs (PCPs))
- as good when the data is constantly changing (WWW)
- fast sanity check to rule out inappropriate inputs (airport security questioning)

Property Tester Definition [Rubinfeld Sudan, Goldreich Goldwasser Ron]



ε-far = differs in many places ($\geq \varepsilon$ fraction of places)

Randomized Sublinear Algorithms

Toy Examples

Property Testing: a Toy Example

Input: a string $w \in \{0,1\}^n$

0	0	0	1	…	0	1	0	0

Question: Is $w = 00 \ldots 0$?

Requires reading entire input.

Approximate version: Is $w = 00 \ldots 0$ or

does it have $\geq \varepsilon n$ 1's ("errors")?

Test (n, w)

1. Sample $s = 2/\varepsilon$ positions uniformly and independently at random
2. If 1 is found, **reject**; otherwise, **accept**

Analysis: If $w = 00 \ldots 0$, it is always accepted.

Used: $1 - x \leq e^{-x}$

If w is ε-far, Pr[error] = Pr[no 1's in the sample] $\leq (1-\varepsilon)^s \leq e^{-\varepsilon s} = e^{-2} < \frac{1}{3}$

Witness Lemma

If a test catches a witness with probability $\geq p$,

then $s = \frac{2}{p}$ iterations of the test catch a witness with probability $\geq 2/3$.

Randomized Approximation: a Toy Example

Input: a string $w \in \{0,1\}^n$

0	0	0	1	…	0	1	0	0

Goal: Estimate the fraction of 1's in w (like in polls)

It suffices to sample $s = 1 / \varepsilon^2$ positions and output the average to get the fraction of 1's $\pm\varepsilon$ (i.e., additive error ε) with probability $\geq$ 2/3

Hoeffding Bound

Let $Y_1, \ldots, Y_s$ be independently distributed random variables in [0,1] and let $Y = \sum_{i=1}^{s} Y_i$ (sample sum). Then $\Pr[|Y - E[Y]| \geq \delta] \leq 2e^{-2\delta^2/s}$.

Y_i = value of sample i. Then $E[Y] = \sum_{i=1}^{s} E[Y_i] = s \cdot$ (fraction of 1's in w)

$$\Pr[|(\text{sample average}) - (\text{fraction of 1's in } w)| \geq \varepsilon] = \Pr[|Y - E[Y]| \geq \varepsilon s]$$
$$\leq 2e^{-2(\varepsilon s)^2/s} \leq 2e^{-2\varepsilon^2 s} = 2e^{-2} < 1/3$$

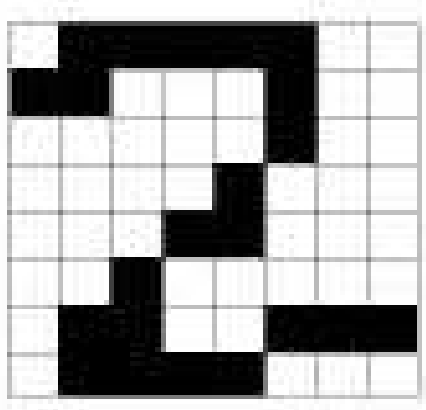
Apply Hoeffding Bound with $\delta = \varepsilon s$

substitute $s = 1 / \varepsilon^2$

Property Testing

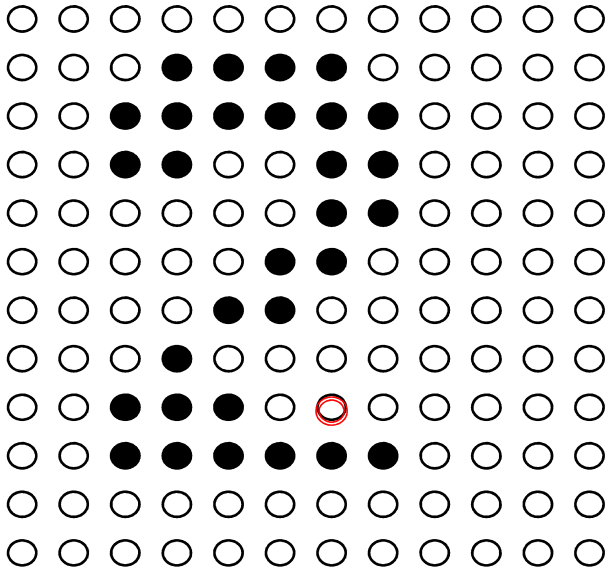
Simple Examples

Testing Properties of Images



Pixel Model

Input: $n \times n$ matrix of pixels
(0/1 values for black-and-white pictures)



Query: point (i_1, i_2)

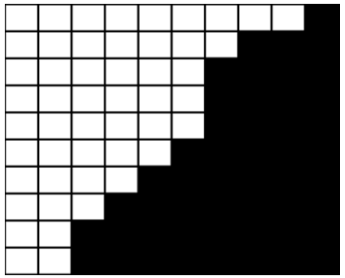
Answer: color of (i_1, i_2)

Testing if an Image is a Half-plane [R03]

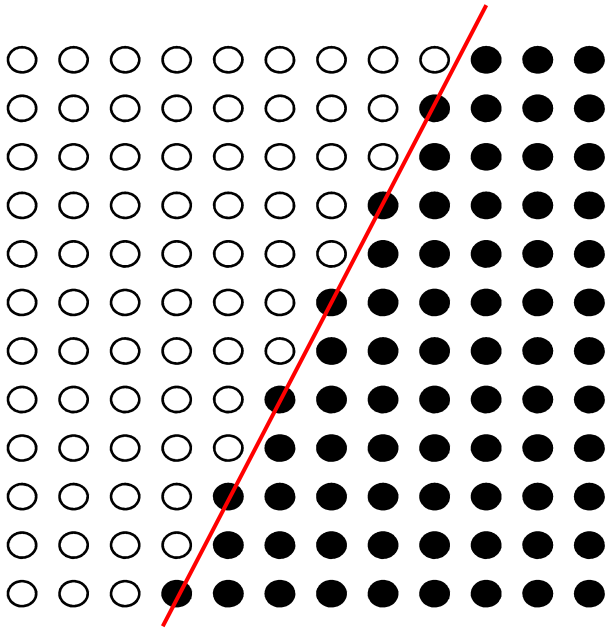
A half-plane or

ε-far from a half-plane?

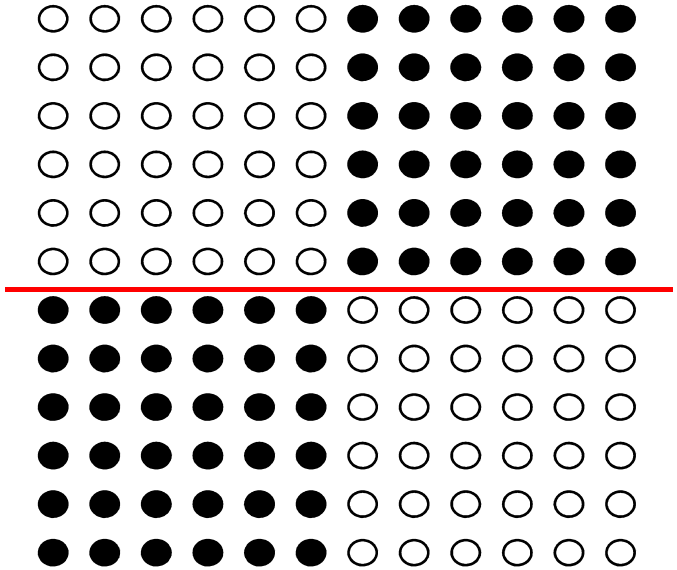
$O(1/\varepsilon)$ time



Half-plane Instances

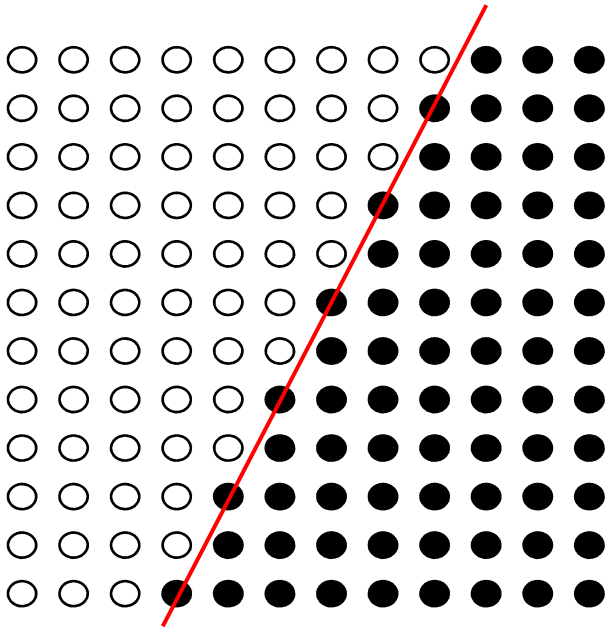


A half-plane

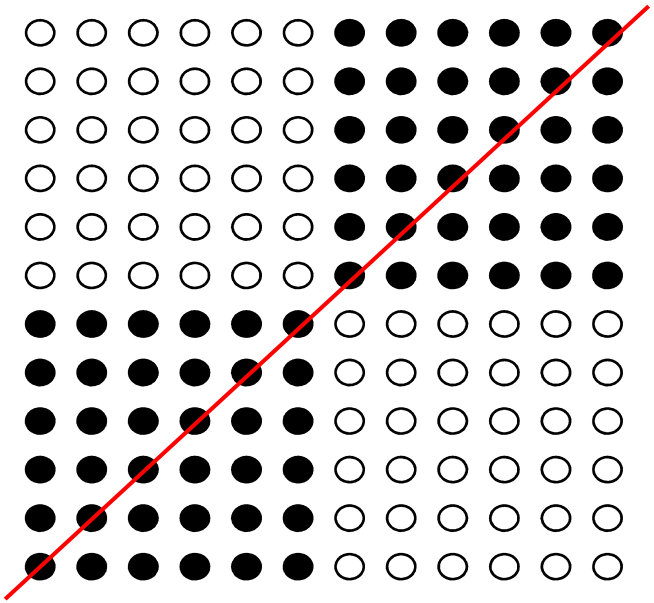


$\frac{1}{4}$-far from a half-plane

Half-plane Instances

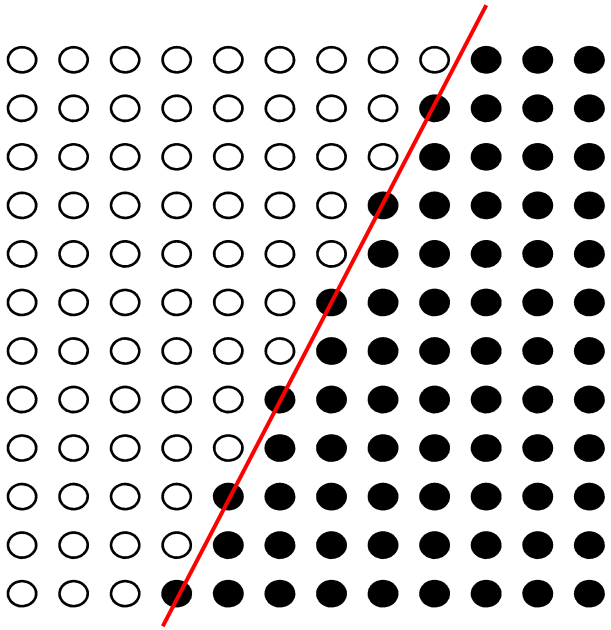


A half-plane

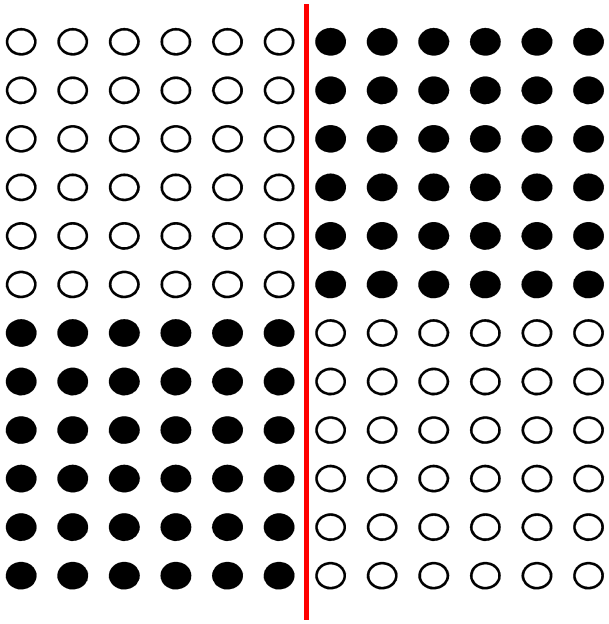


$\frac{1}{4}$-far from a half-plane

Half-plane Instances

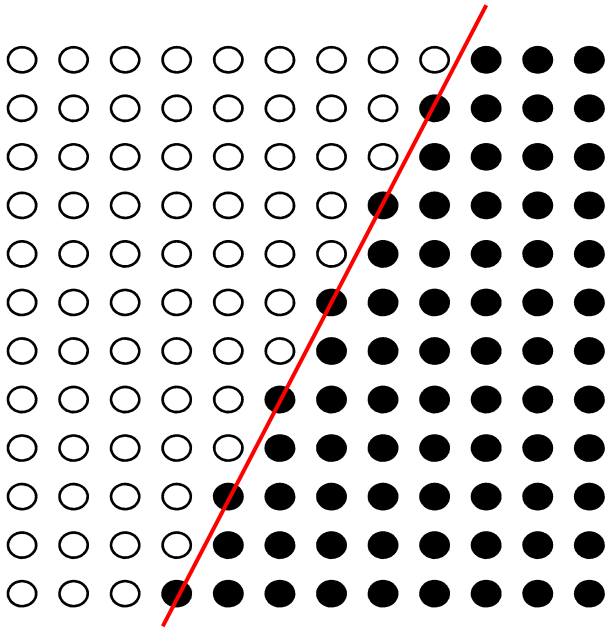


A half-plane

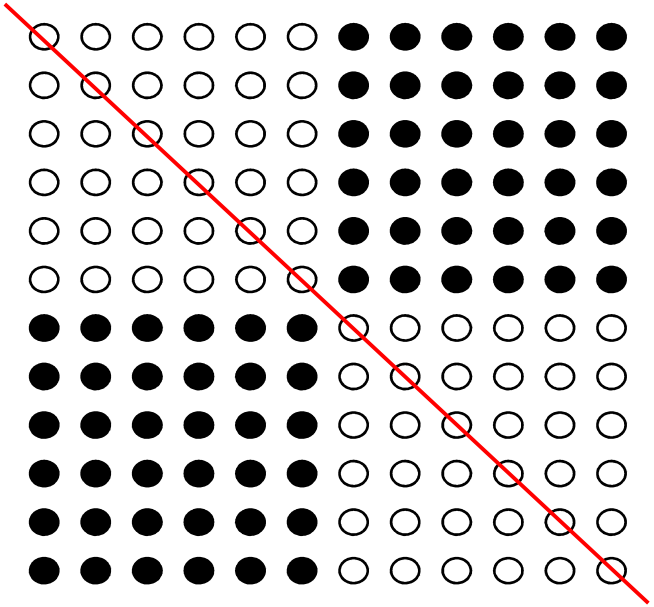


$\frac{1}{4}$-far from a half-plane

Half-plane Instances

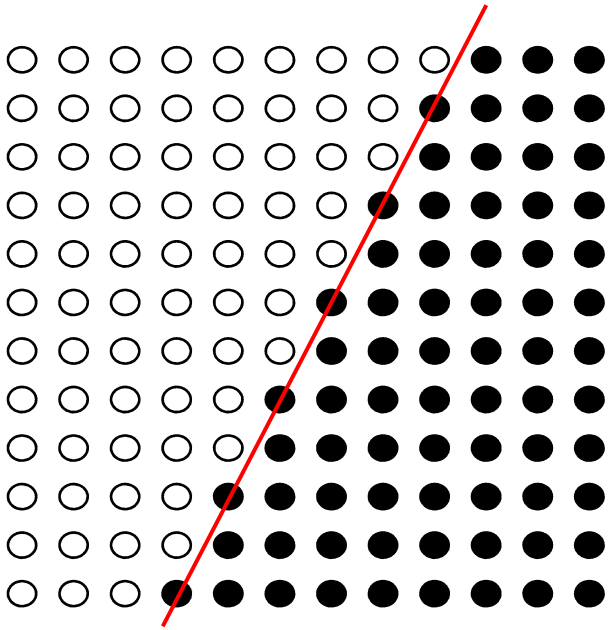


A half-plane

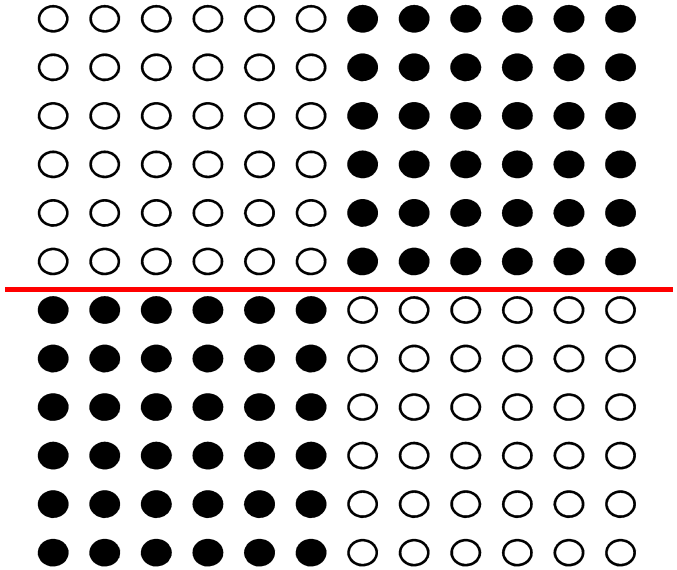


$\frac{1}{4}$-far from a half-plane

Half-plane Instances

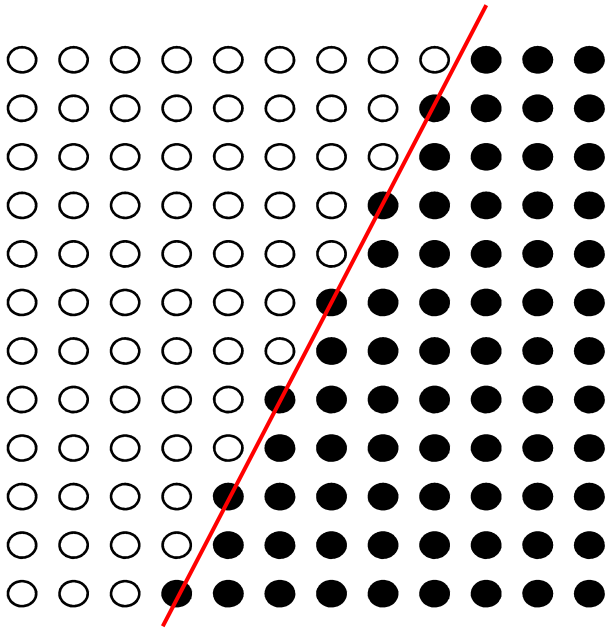


A half-plane

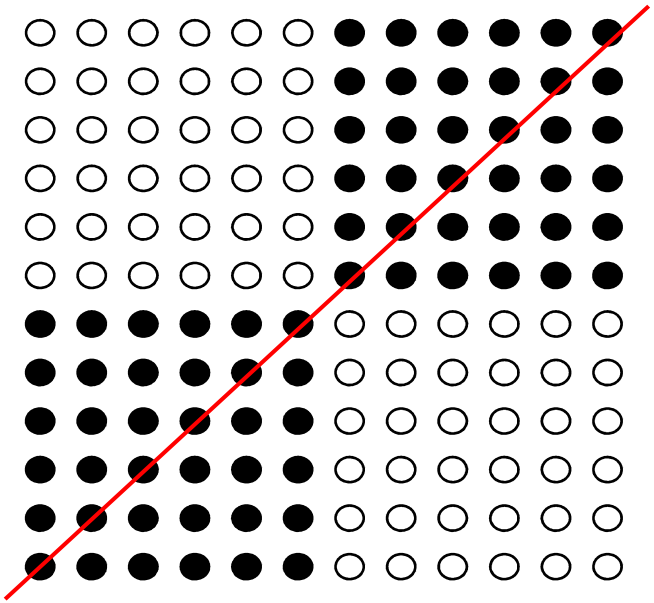


$\frac{1}{4}$-far from a half-plane

Half-plane Instances

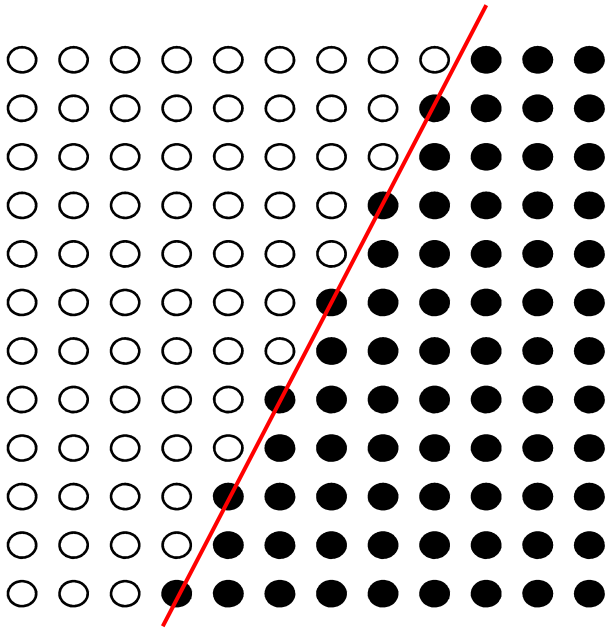


A half-plane

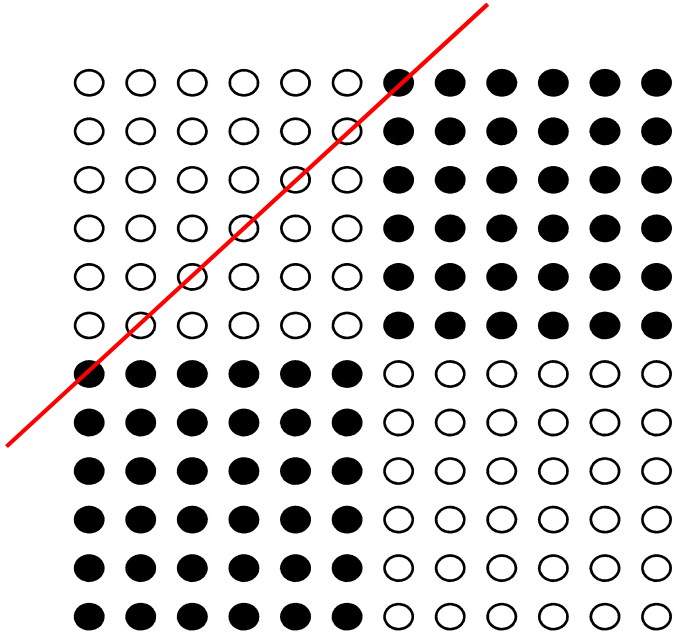


$\frac{1}{4}$-far from a half-plane

Half-plane Instances



A half-plane



$\frac{1}{4}$-far from a half-plane

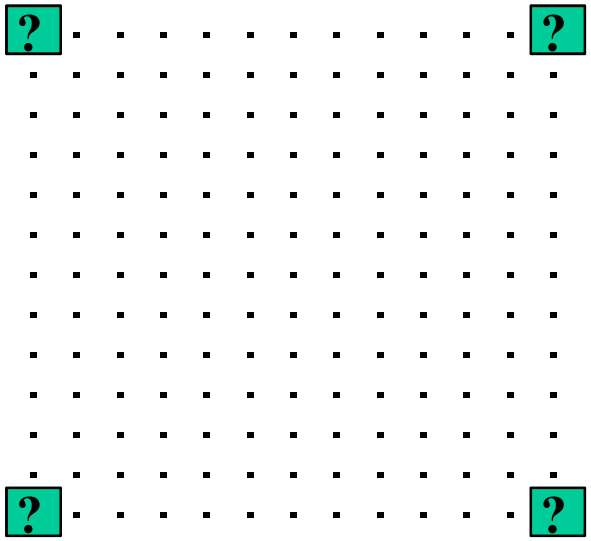
Strategy

"Testing by implicit learning" paradigm

- Learn the outline of the image by querying a few pixels.
- Test if the image conforms to the outline by random sampling, and reject if something is wrong.

Half-plane Test

Claim. The number of sides with different corners is 0, 2, or 4.

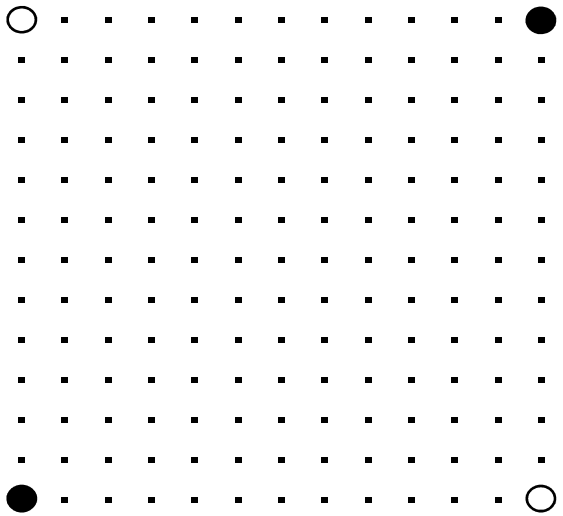


Algorithm

1. Query the corners.

Half-plane Test: 4 Bi-colored Sides

Claim. The number of sides with different corners is 0, 2, or 4.



Analysis

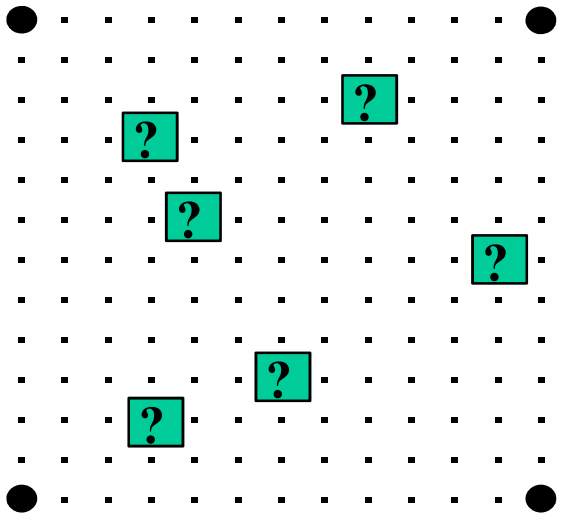
- If it is 4, the image cannot be a half-plane.

Algorithm

1. Query the corners.
2. If the number of sides with different corners is 4, **reject**.

Half-plane Test: 0 Bi-colored Sides

Claim. The number of sides with different corners is 0, 2, or 4.



Analysis

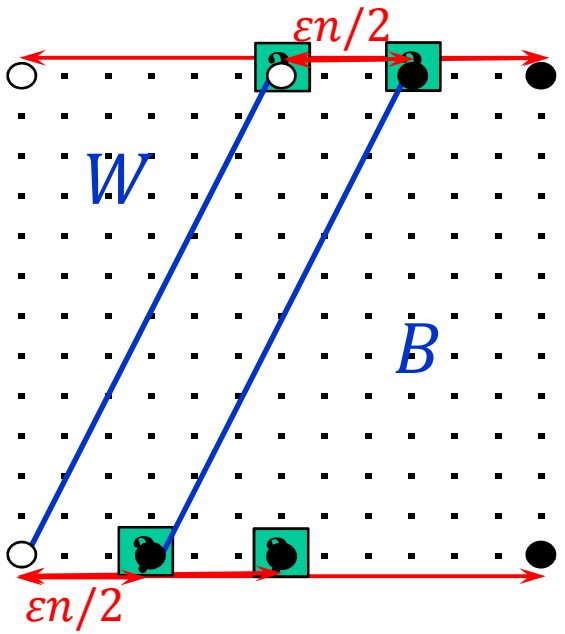
- If all corners have the same color, the image is a half-plane if and only if it is unicolored.

Algorithm

1. Query the corners.
2. If all corners have the same color c, test if all pixels have color c (as in Toy Example 1).

Half-plane Test: 2 Bi-colored Sides

Claim. The number of sides with different corners is 0, 2, or 4.



Analysis

- The area outside of $W \cup B$ has $\leq \varepsilon n^2/2$ pixels.
- If the image is a half-plane, W contains only white pixels and B contains only black pixels.
- If the image is ε-far from half-planes, it has $\geq \varepsilon n^2/2$ wrong pixels in $W \cup B$.
- By Witness Lemma, $4/\varepsilon$ samples suffice to catch a wrong pixel.

Algorithm

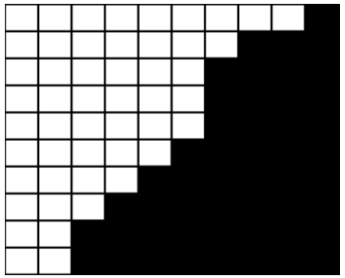
1. Query the corners.
2. If # of sides with different corners is 2, on both sides find 2 different pixels within distance $\varepsilon n/2$ by binary search.
3. Query $4/\varepsilon$ pixels from $W \cup B$
4. **Accept** iff all W pixels are white and all B pixels are black.

Testing if an Image is a Half-plane [R03]

A half-plane or

ε-far from a half-plane?

$O(1/\varepsilon)$ time ✓



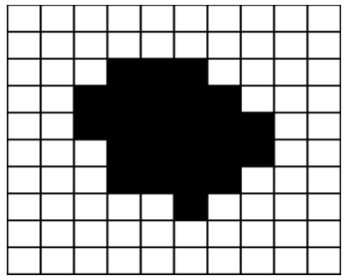
Other Results on Properties of Images

- Pixel Model

  Convexity [R03]

  Convex or ε-far from a half-plane?

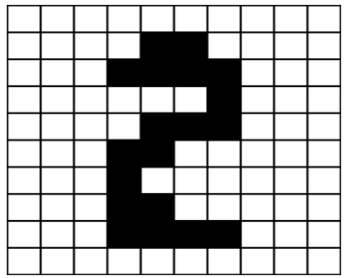
  $O(1/\varepsilon^2)$ time



  Connectedness [R03]

  Connected or ε-far from connected?

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

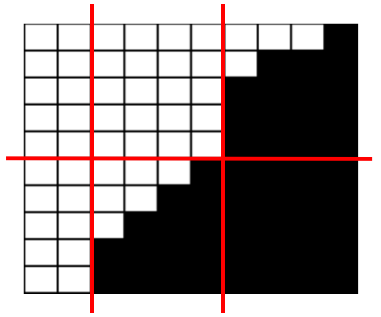


  Partitioning [Kleiner Keren Newman 10]

  Can be partitioned according to a template

  or or ε-far?

  time independent of image size



- Properties of sparse images [Ron Tsur 10]

Testing if a List is Sorted

Input: a list of n numbers $x_1, x_2, \ldots, x_n$

- Question: Is the list sorted?

  Requires reading entire list: $\Omega(n)$ time

- Approximate version: Is the list sorted or ϵ-far from sorted?

  (An ϵ fraction of x_i's have to be changed to make it sorted.)

  [Ergün Kannan Kumar Rubinfeld Viswanathan 98, Fischer 01]: $O((\log n)/\epsilon)$ time

  $\Omega(\log n)$ queries

- Attempts:

  1. Test: Pick a random i and reject if $x_i > x_{i+1}$.

     Fails on: 1 1 1 1 1 1 1 0 0 0 0 0 0 0 ← 1/2-far from sorted

  2. Test: Pick random $i < j$ and reject if $x_i > x_j$.

     Fails on: 1 0 2 1 3 2 4 3 5 4 6 5 7 6 ← 1/2-far from sorted