Combinatorial and Probabilistic techniques in Property Testing

Artur Czumaj

DIMAP (Centre for Discrete Maths and it Applications)

& Department of Computer Science

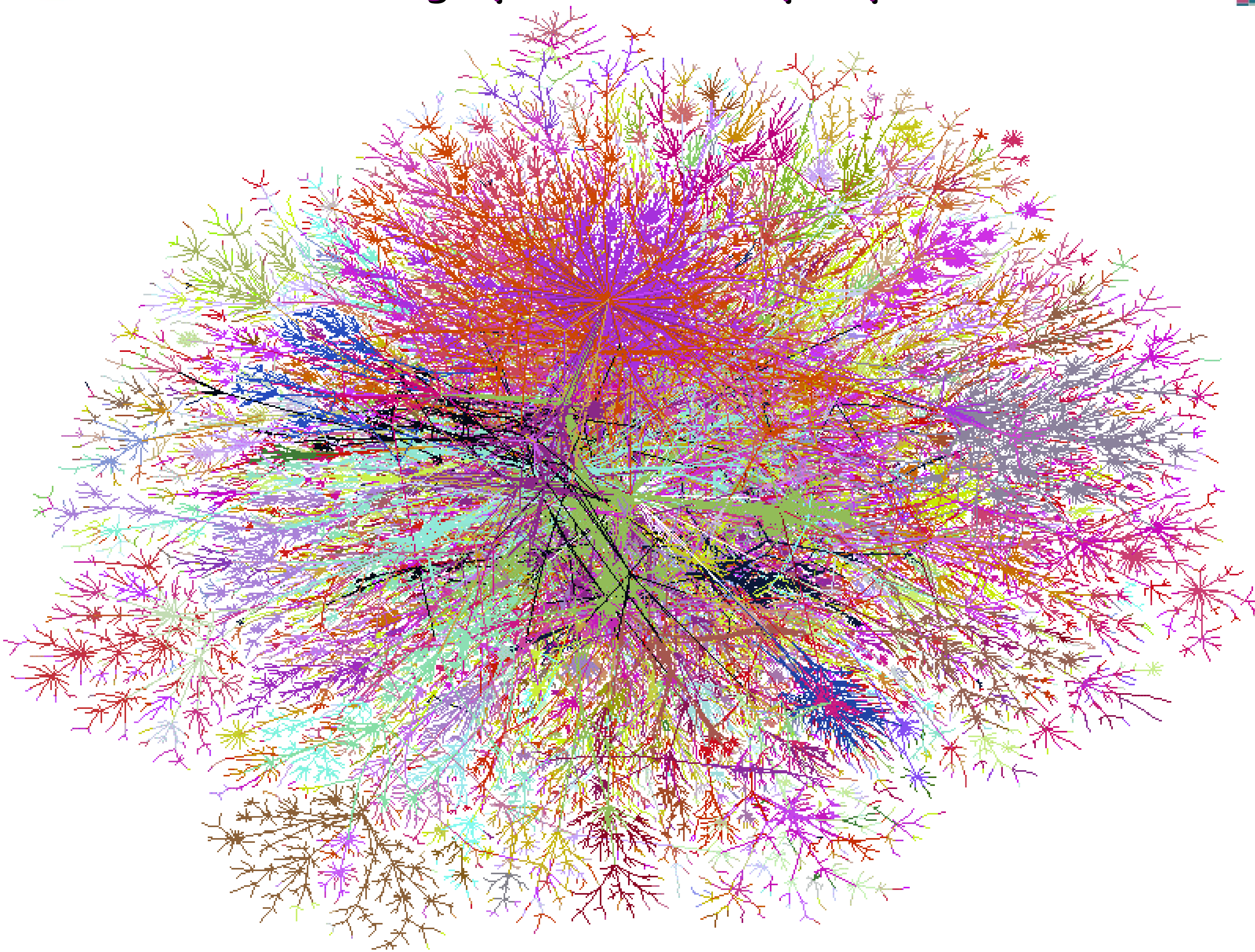
University of Warwick

THE UNIVERSITY OF WARWICK

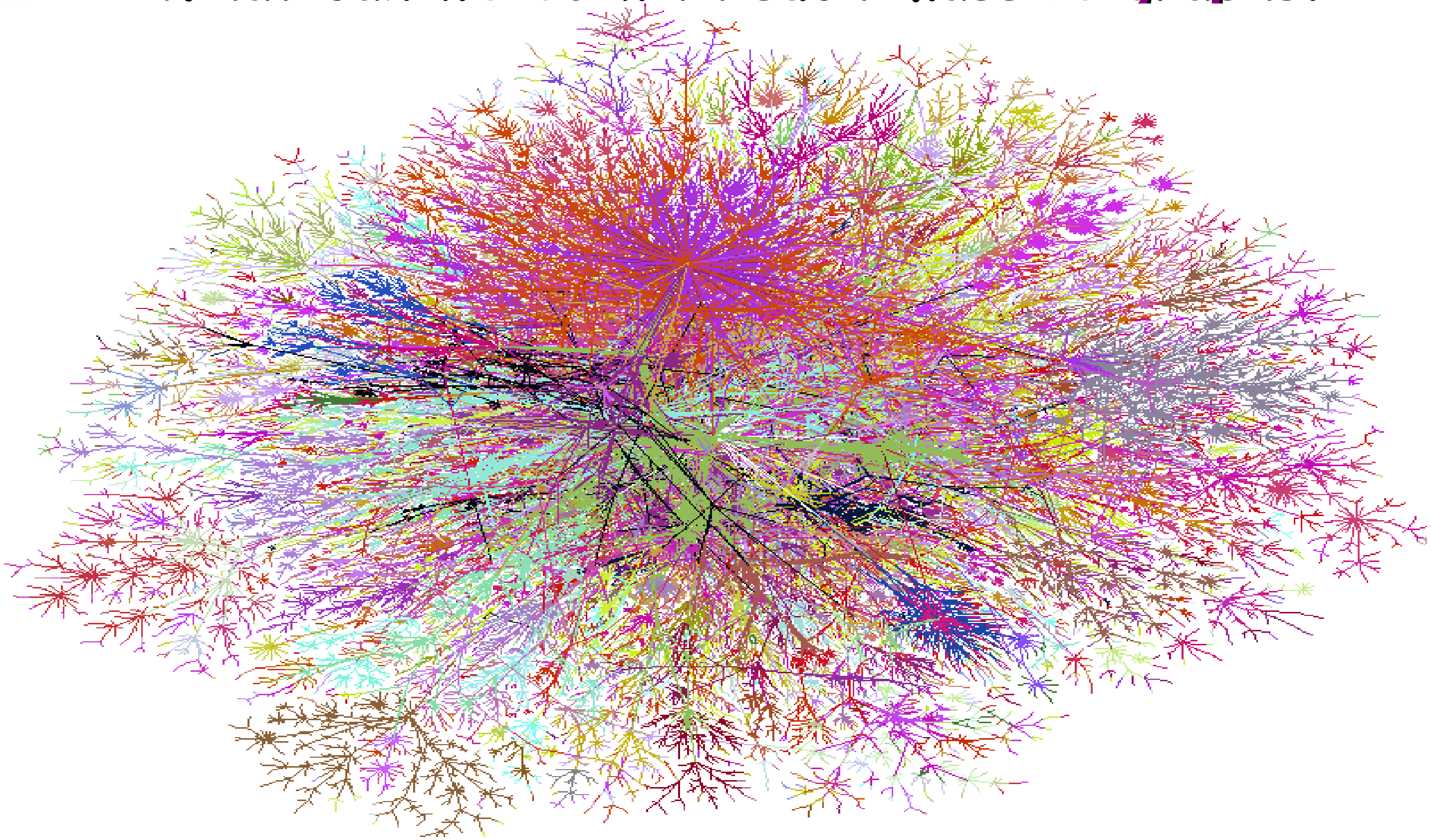
DIMAP

Massive graphs

Massive graphs – modern perspective

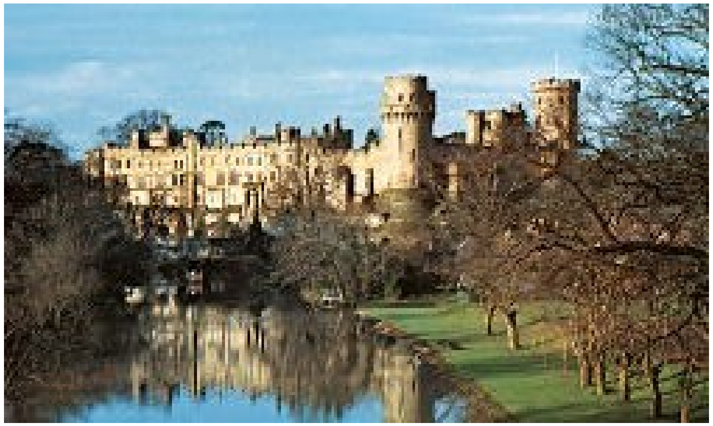


What can we do with such massive graphs?



- Can we quickly test if this graph has some properties?
  - What is quickly if graph has billions of nodes?
  - Quickly ~ better than in $\Theta(n)$ time!

“in the castle” vs. “out of the castle”
Property Testing

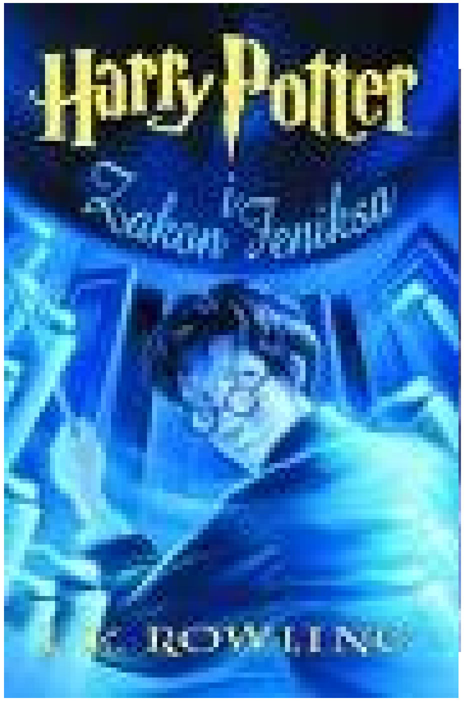


- Distinguish inputs that have specific property
  from those that are far from having the property

- Benefits:
  - May be the natural question to ask
  - May be just as good when data constantly changing
  - Gives fast sanity check to rule out very “bad” inputs (i.e., restaurant bills) or to decide when expensive processing is worth it

An example:

- Is this book written in English?
- We (usually) don't have to read entire book to make a good guess



Test: pick a few random pages
if they're in English ➔ book is in English

Another example:

- Given: list $x_1\, x_2 \ldots\, x_n$
- Question: is the list sorted?

- Clearly requires $\Omega(n)$ time

Another example:

- Given: list $x_1\, x_2 \ldots\, x_n$
- Question: is the list almost sorted?
  - i.e., can change at most ε fraction of list to make it sorted

- Can test in $O(1/\varepsilon \cdot \log n)$ time
  - [Ergun, Kannan, Kumar, Rubinfeld, Viswanathan]
  - best possible

Property testing

- Classical decision problem:
  - Given a property P and input instance I
  - Does I has property P?

Often it's computationally hard (NP-complete/undecidable)

- What we want to study [relaxation]:
  - Is I close to satisfy property P?

Can work fast even for NP-hard or undecidable properties

Property Testing definition

- Given input x
- If x has the property , **tester passes**
- If x is ε-far from any string that has the property , **tester fails**
- error probability < 1/3

Notion of **ε-far** depends on the problem;
Typically: one needs to change ε fraction of the input to obtain object satisfying the property

Typically we think about **ε** as on a small constant, say, **ε = 0.1**

Property Testing definition

- Given input x
- If x has the property , **tester passes**
- If x is ε-far from any string that has the property ,
tester fails
- error probability < 1/3

- This is 2-sided-error tester
- 1-sided error: errs only for x being ε-far

So, what is property testing

- Early motivation:
  - Program checking
  - Program verification
  - Learning theory
- Big boost (in theory)
  - Probabilistically Checkable Proofs
    - "Correctness of any proof in NP can be verified by testing only O(1) positions in the proof and using only O(log n) random bits"

Properties of functions

- Linearity test [Blum Luby Rubinfeld] [Bellare Coppersmith Hastad Kiwi Sudan] (various improvements by many others)
  $\forall x,y \quad f(x)+f(y)=f(x+y)$
- Low total degree polynomial tests [Rubinfeld Sudan] [Arora Safra] [Arora Lund Motwani Sudan Szegedy] [Arora Sudan] ...
- Functions definable by functional equations – trigonometric, elliptic functions
- Groups, Fields
- Finite precision [Gemmell Lipton Rubinfeld Sudan Wigderson] ....
- Low complexity functions [Parnas Ron Samorodnitsky] .....

- Useful in
  - Program checking
  - PCP constructions

Properties of distributions:

- some properties:
  - are two given distributions similar or very different?
  - approximate the entropy of a distribution
  - are two random variables independent?

  [Batu Fortnow Rubinfeld Smith White] [Batu Dasgupta Kumar Rubinfeld][Batu Fischer Fortnow Kumar Rubinfeld White]
- access to samples of distribution, not explicit probabilities

Study of combinatorial properties
[Goldreich Goldwasser Ron]

- Graph properties
- Hypergraph properties
- Monotonicity
- Set properties
- Geometric properties
- String properties
- Membership in low complexity languages
  (regular languages, constant width branching programs, context-free languages …)

Properties of graphs
[Goldwasser, Goldreich, Ron]

- Graph properties:
  - Colorability
  - Not containing a forbidden subgraph
  - Connectivity
  - Acyclicity
  - Rapid mixing
  - Max-Cut

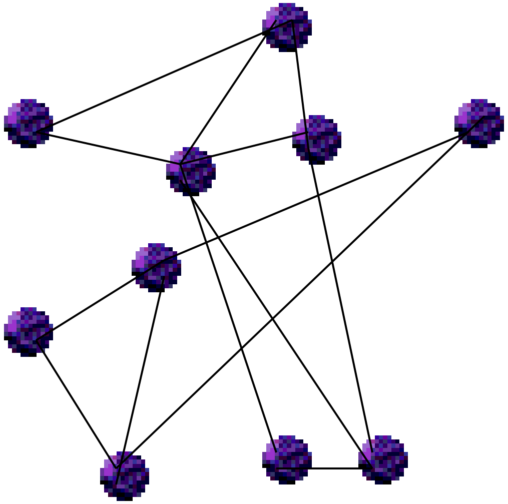
  ...

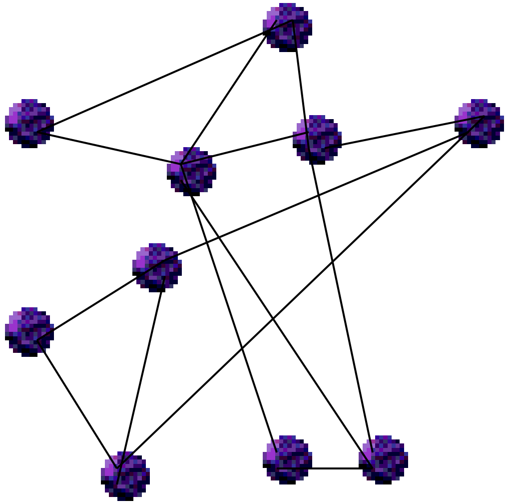
Some of these properties are NP-hard

Graph properties

- Measure of being far/close from a property
- Is graph connected or is *far* from being connected?

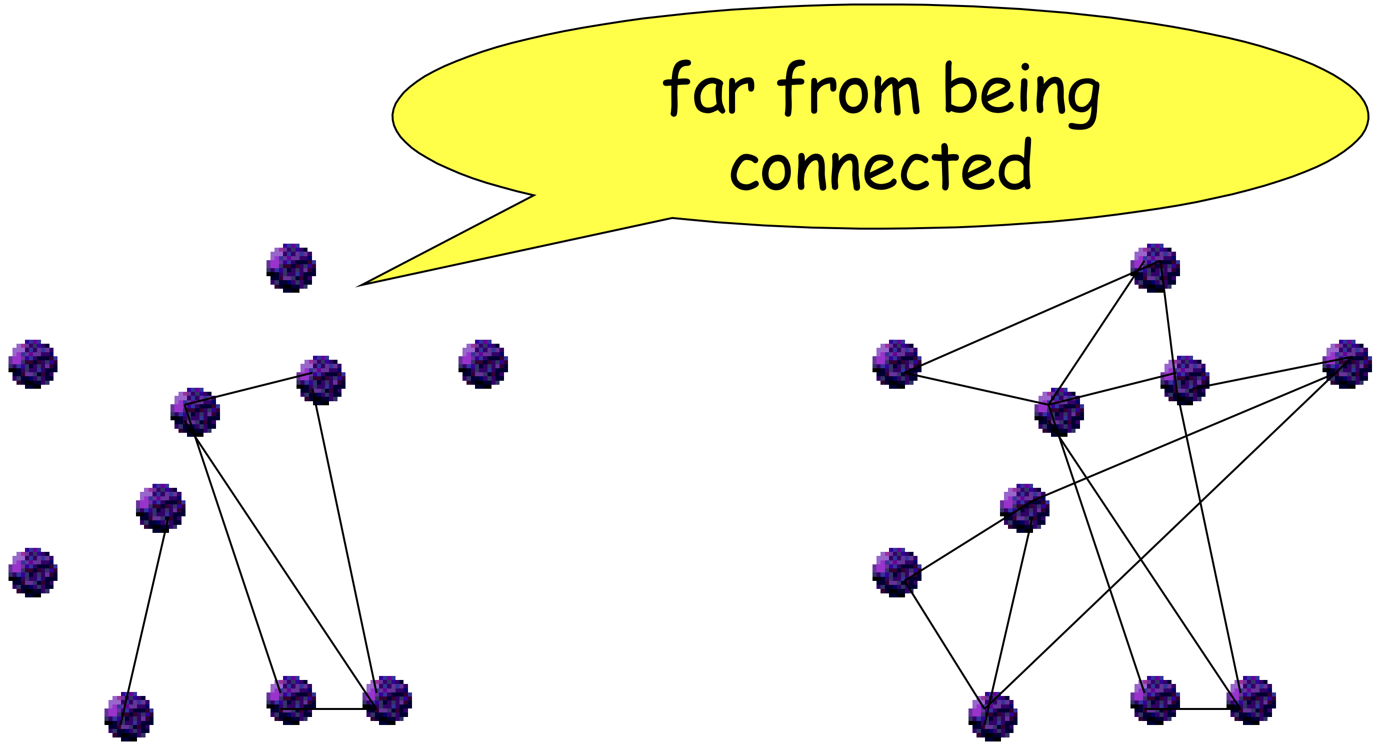
These two graphs are **close** to be connected





Graph properties

- Measure of being far/close from a property
- Is graph connected or is *far* from being connected?



1st definition

Graph G is ε-far from satisfying property P

If one needs to modify more than ε-fraction of entries in adjacency matrix to obtain a graph satisfying P

0	1	0	0	1
1	0	1	1	1
0	1	0	0	1
0	1	0	0	0
1	1	1	0	0

1st definition

Graph G is ε-far from satisfying property P

If one needs to modify more than ε-fraction of entries in adjacency matrix to obtain a graph satisfying P

$\varepsilon \cdot n^2$ edges have to be added/deleted

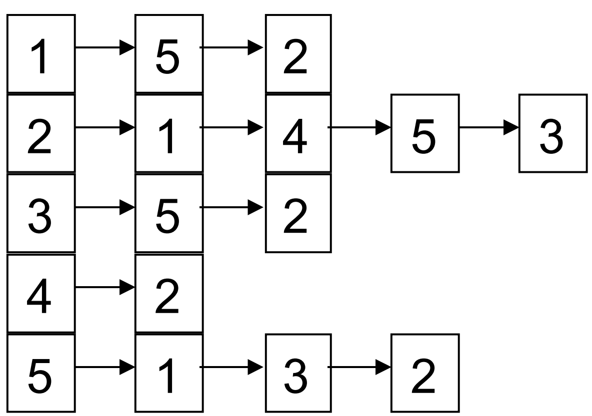
Suitable for dense graphs

Usually "trivial" for sparse graphs

2nd definition

Graph G is ε-far from satisfying property P

If one needs to modify more than ε-fraction of entries in **adjacency lists** to obtain a graph satisfying P



2^{nd} definition

Graph G is ε-far from satisfying property P

If one needs to modify more than ε-fraction of entries in adjacency lists to obtain a graph satisfying P

Suitable for sparse graphs

Main model: graphs of bounded degree

What is the complexity (running-time)?

- For simplicity:
  - Complexity = number of accesses to the input
    - In adjacency matrix model:
      - number of entries tested
      - Oracle: is (x,y) in E?
    - In adjacency list model:
      - number of edges tested
      - Oracle: give me the ith neighbor of vertex v

Plan

- We will discuss a few representative examples of property testing algorithms for both models of graphs
- Some proofs will be given
- Some won't (eg because they're too complex)
- You will have to do some proofs

Part I
Adjacency matrix model

Adjacency matrix model

- **Accept** every graph that satisfies property P
- **Reject** every graph that is ε–far from property P
  - **ε-far from P**: one has to modify at least εn^2 entries of the adjacency matrix to obtain a graph with property P
- **Arbitrary answer** if the graph doesn't satisfy P nor is ε–far from P
- Can err with probability $< 1/3$
  - Sometimes errs only for "rejects": **1-sided-error**

Adjacency matrix model

- There are very fast property testers
- They're very simple
  - Typical algorithm:

    - Select a random set of vertices U
    - Test the property on the subgraph induced by U

- The analysis is (often) very hard
- We understand this model very well
  - mostly because of very close relation to combinatorics

Adjacency matrix model

First example:

- Test if a graph G=(V,E) is planar
  - We've already said: testing is "trivial" for sparse graphs …

Adjacency matrix model

First example:

- Test if a graph $G=(V,E)$ is planar
  - We've already said: testing is "trivial" for sparse graphs …

- If G is dense then it's certainly not planar
- If G is sparse then it's not ε-far from planar
  - Every graph with less than $\varepsilon n^2/2$ edges is ε-close to planar: remove all its edges
- How to distinguish between sparse graphs and dense graphs?

Adjacency matrix model

Easy example:

- Test if a graph is planar

  Check if G is sparse [if it has less than $\varepsilon n^2/2$ edges]

  - If it's not then reject G
    - all planar graphs are sparse
  - If it sparse then accept G
    - every sparse graph can be "made planar": remove all edges

  - How to check if G has much less than $\varepsilon n^2/2$ edges?
    - Randomly sample $2/\varepsilon$ entries in the adjacency matrix
      - If all of them are 0 then accept
      - Otherwise reject
    - With probability 2/3 will give right answer

Adjacency matrix model

How to check if G has much less than $\varepsilon n^2/2$ edges

Randomly sample $2/\varepsilon$ entries in the adjacency matrix
- If all of them are 0 then accept
- Otherwise reject

Analysis:

- If G is planar then $|E|<3n$

$$Pr[accept] = (1-2|E|/n^2)^{2/\varepsilon} > (1-6/n)^{2/\varepsilon} > 2/3$$

- If $|E| > \varepsilon n^2/2$ then

$$Pr[reject] = 1-Pr[accept] = 1-(1-2|E|/n^2)^{2/\varepsilon} > 1-(1-\varepsilon)^{2/\varepsilon} > 1-e^{-2} > 2/3$$

2-sided-error $O(1/\varepsilon)$-tester for planarity

Adjacency matrix model

- All algorithms are of the following form:

Randomly sample set S of vertices
Consider subgraph of G induced by S
If the subgraph satisfies the property → accept
otherwise → reject

This property may be **different** from
the original one
(but it's usually the same)

Adjacency matrix model

- All algorithms are of the following form:

Randomly sample set S of vertices
Consider subgraph of G induced by S
If the subgraph satisfies the property ➔ accept
otherwise ➔ reject

Theorem: If there is a property testing algorithm for property P that performs $q(n,\varepsilon)$ queries to the adjacency matrix then there is a property testing algorithm for P of the form above that performs at most $O((q(n,\varepsilon))^2)$ queries to the adjacency matrix

Adjacency matrix model

Not so easy example:

- Test if a graph is bipartite

- One can show that the following algorithm will work:
  - Randomly sample $O(1/\varepsilon)$ nodes
  - Check if the graph induced by these nodes is bipartite
    - If it is then accept
    - Otherwise reject
  - With probability 2/3 will give right answer

Adjacency matrix model

Not so easy example:

- Test if a graph is bipartite

  - One can show that the following algorithm will work:
    - Randomly sample $O(1/\varepsilon)$ nodes
    - Check if the graph induced by these nodes is bipartite
      - If it is then accept
      - Otherwise reject
    - With probability 2/3 will give right answer

- Proof with query complexity $O(1/\varepsilon^2)$ is non-trivial
- Let's try to give a proof with $O(1/\varepsilon^{O(1)})$ nodes

Adjacency matrix model

- We sample $O(1/\varepsilon^{O(1)})$ random nodes of G
- Call H the graph induced by these nodes

- To prove
  - If G is bipartite then H is bipartite
    - obvious
  - If G is ε-far from bipartite then with prob. > 2/3 graph H is not bipartite
    - challenging

Adjacency matrix model

- We sample $O(1/\varepsilon^{O(1)})$ random nodes of G
- Call H the graph induced by these nodes
- If G is ε–far from bipartite then with prob. > 2/3 graph H is not bipartite

- **Idea: choose a smaller subset of selected nodes U**
- **Show that any bipartition determined by U (U = U1+U2) with constant probability enforce the bipartition on the remaining part of H + some edges will clash**

Adjacency matrix model

- We sample $O(1/\varepsilon^{O(1)})$ random nodes of G
- Call H the graph induced by these nodes
- If G is ε–far from bipartite then with prob. > 2/3 graph H is not bipartite

- $U \subseteq V(H)$ - of size $t=O(\varepsilon^{-1} \log(1/\varepsilon))$
- $v \in V$ is heavy: $\deg(v) \geq \varepsilon n/3$
- U is good for V if all but at most $\varepsilon n/3$ of the heavy nodes in V have a neighbor in U

- Claim: With prob $\geq$ 5/6 randomly chosen set U is good

Adjacency matrix model

- $U \subseteq V(H)$ - of size $t=O(\varepsilon^{-1} \log(1/\varepsilon))$
- $v \in V$ is heavy: $deg(v) \geq \varepsilon n/3$
- U is good for V if all but at most $\varepsilon n/3$ of the heavy nodes in V have a neighbor in U

- Claim: With prob $\geq 5/6$ randomly chosen set U is good

Proof:

Pr[heavy node v has no neighbor in U] $\leq (1-\varepsilon/3)^t < \varepsilon/18$

E[# heavy nodes with no neighbor in U] $< \varepsilon n/18$

The claim follows now from Markov's inequality

Adjacency matrix model

- $U \subseteq V(H)$ - of size $t=O(\varepsilon^{-1} \log (1/\varepsilon))$
- $v \in V$ is heavy: $\deg(v) \geq \varepsilon n/3$
- U is good for V if all but at most $\varepsilon n/3$ of the heavy nodes in V have a neighbor in U
- Claim: With prob $\geq 5/6$ randomly chosen set U is good
- Edge disturbs a partition (U_1,U_2) of U if both endpoints are in the same $N(U_i)$ for some $i \in \{1,2\}$
- Claim: For any good set U and any bipartition of U, at least $\varepsilon n^2/3$ edges disturb the partition

Adjacency matrix model

- $v \in V$ is heavy: $\deg(v) \geq \varepsilon n/3$
- U is good for V if all but at most $\varepsilon n/3$ of the heavy nodes in V have a neighbor in U
- Claim: With prob $\geq 5/6$ randomly chosen set U is good
- Edge disturbs a partition (U_1,U_2) of U if both endpoints are in the same $N(U_i)$ for some $i \in \{1,2\}$
- Claim: For any good set U and any bipartition of U, at least $\varepsilon n^2/3$ edges disturb the partition

Proof:
Each partition of V has at least εn^2 violating edges
We upper bound the number of these edges with an endpoint not in N(U):
- # edges incident to heavy nodes with no neighbor in U $\leq \varepsilon n^2/3$
- # edges incident to non-heavy nodes $\leq \varepsilon n^2/3$

→There are at least $\varepsilon n^2/3$ violating edges connecting vertices in N(U)
These edges disturb the partition

Adjacency matrix model

- $U \subseteq V(H)$ - of size $t=O(\varepsilon^{-1} \log(1/\varepsilon))$
- $v \in V$ is heavy: $deg(v) \geq \varepsilon n/3$
- U is good for V if all but at most $\varepsilon n/3$ of the heavy nodes in V have a neighbor in U
- Claim: With prob $\geq 5/6$ randomly chosen set U is good
- Edge disturbs a partition (U_1,U_2) of U if both endpoints are in the same $N(U_i)$ for some $i \in \{1,2\}$
- Claim: For any good set U and any bipartition of U, at least $\varepsilon n^2/3$ edges disturb the partition
- H is bipartite only if either
  1) U is not good or
  2) U is good & $\exists$ a bipartition of U with no disturbing edge in H

Last step of the proof

H is bipartite only if either

1) U is not good or
2) U is good & $\exists$ a bipartition of U with no disturbing edge in H

Pr[U is not good] $\leq 1/6$

Pr[event (2)] $\leq$ #[bipartitions of U] $\times$ Pr[given bipartition is "bad"]

$S = V(H) - U$

$|S| = \Omega(|U|/\varepsilon)$, where $|U| = O(\varepsilon^{-1} \log (1/\varepsilon))$

$$\Pr[\text{event } (2)] \leq 2^{|U|} (1-\varepsilon/3)^{|S|/2} < 1/6$$

Pr[H is bipartite] $\leq$ Pr[U is not good]+Pr[event (2)] $\leq 1/3$

Adjacency matrix model

Not so easy example:

- Test if a graph is bipartite

- One can show that the following algorithm will work:
  - Randomly sample $O(1/\varepsilon)$ nodes
  - Check if the graph induced by these nodes is bipartite
    - If it is then accept
    - Otherwise reject
  - With probability 2/3 will give right answer

- Proof with query complexity $O(1/\varepsilon^2)$ is non-trivial
- We proved it with $O(\log(1/\varepsilon)/\varepsilon^2)$ nodes → $O(\log^2(1/\varepsilon)/\varepsilon^4)$ query complexity

Adjacency matrix model

Open question:
is it possible to test bipartitness with query complexity $o(\varepsilon^{-2})$?

Not so easy example:

- Test if a graph is bipartite

- One can show that the following algorithm will work:
  - Randomly sample $O(1/\varepsilon)$ nodes
  - Check if the graph induced by these nodes is bipartite
    - If it is then accept
    - Otherwise reject
  - With probability 2/3 will give right answer

- Proof with query complexity $O(1/\varepsilon^2)$ is non-trivial
- We proved it with $O(\log(1/\varepsilon)/\varepsilon^2)$ nodes ➔ $O(\log^2(1/\varepsilon)/\varepsilon^4)$ query complexity

Adjacency matrix model

Very easy example:

- Test if a graph contains a triangle (cycle of length 3)

> **Return YES (always)**

Highly nontrivial example:

- Test if a graph is triangle-free

> - Can be done in $f(\varepsilon) = O(1)$ time
> - Proof: deep combinatorics

Testing triangle-freeness

- Test if a graph is triangle-free
- Can be done in $f(\varepsilon) = O(1)$ time using Szemeredi regularity lemma

Szemeredi regularity lemma

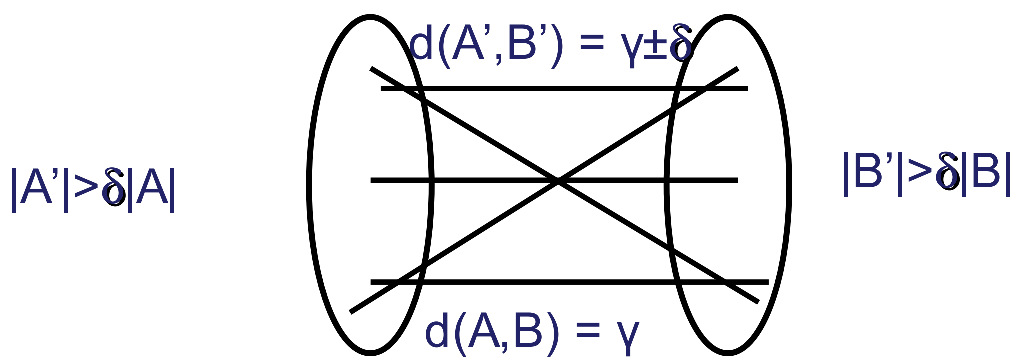
Regular pairs

- For two vertex sets A and B, let

  d(A,B) = edge-density connecting A and B

$$d(A,B) = \frac{\#\text{ edges connecting } A \text{ and } B}{|A||B|}$$

(A,B) is **δ-regular** if for every $A' \subseteq A$, $B' \subseteq B$, with $|A'| > \delta|A|$ and $|B'| > \delta|B|$, we have $|d(A,B) - d(A',B')| < \delta$



Szemeredi regularity lemma

Szemeredi Regularity Lemma:

For any δ, any graph G can be partitioned into k, $1/\delta \le k \le T(\delta)$, subsets $V_1,\ldots,V_k$ of equal size, such that all but at most δk^2 of the pairs (V_i, V_j) are δ-regular

Triangle-freeness & Szemeredi lemma

Szemeredi Lemma can be used to show that

- if G is ε-far from triangle-free
- then G has $\Theta(n^3)$ triangles [at least $n^3/f(\varepsilon)$]

Once this is proven, we have the following property testing algorithm (with 1-sided error):

Repeat $O(f(\varepsilon))$ times:
 choose 3 nodes i.u.r.
 if they form a triangle then reject
accept

Using Szemeredi lemma

Szemeredi Regularity Lemma:
For any δ, any graph G can be partitioned into k, $1/\delta \le k \le T(\delta)$, subsets $V_1,\dots,V_k$ of equal size, such that all but at most δk^2 of the pairs (V_i, V_j) are δ-regular

We have to prove that if G is ε-far from triangle-free then G has $\Theta(n^3)$ triangles

Find a partition of V into $V_1,\dots,V_k$ with $k<f(\varepsilon)$ and $k \gg 1/\varepsilon$, such that all but at most δk^2 of the pairs are δ-regular for some constant $\delta=\delta(\varepsilon) \ll \varepsilon$

Edge $e=(x,y)$ with $x \in V_i$ and $y \in V_j$, is **useful** if

- $V_i \neq V_j$,
- (V_i,V_j) is δ-regular, and
- the density between V_i and V_j is at least $\varepsilon/15$

Using Szemeredi lemma

Szemeredi Regularity Lemma:
For any δ, any graph G can be partitioned into k, $1/\delta \le k \le T(\delta)$, subsets $V_1,\dots,V_k$ of equal size, such that all but at most δk^2 of the pairs (V_i, V_j) are δ-regular

Find a partition of V into $V_1,\dots,V_k$ with $k<f(\varepsilon)$ and $k \gg 1/\varepsilon$, such that all but at most of the pairs are δ-regular for some constant $\delta=\delta(\varepsilon)$

Edge $e=(x,y)$ with $x \in V_i$ and $y \in V_j$, is **useful** if

- $V_i \neq V_j$,
- (V_i,V_j) is δ-regular, and
- the density between V_i and V_j is at least $\varepsilon/15$

Lemma: There are less than εn^2 non-useful edges

Using Szemeredi lemma

- Let G be ε–far from triangle free
- Remove all non-useful edges to define graph G′
- Since G has less than εn^2 non-useful edges, G′ must has at least one triangle ➔
  - There are three useful edges (x,y), (y,z), (z,x) with $x \in V_i$, $y \in V_j$, $z \in V_s$, such that
    - all sets V_i, V_j, V_s are distinct,
    - all sets V_i, V_j, V_s are pairwise δ-regular, and
    - the density between each pair V_i, V_j, V_s is at least ε/15.

➔ There are $\Theta(n^3)$ triangles between V_i, V_j, V_s

Complexity of some properties

- Is G 2-colorable (bipartite) or is ε-far
  - We can test just **$O(1/\varepsilon)$** vertices!

Testing in time independent of the size of G

- G contain no clique of size 17?
  - We can test just **$f(1/\varepsilon)$** vertices!
- Does G contain no subgraph isomorphic to a given graph with 122 vertices?
  - We can test just **$f(1/\varepsilon)$** vertices!

First order graph properties

- Any first-order graph property of type $\forall\exists$ is testable in $O(1)=f(\varepsilon)$ time.
- There are first order properties of type $\exists\forall$ that require superconstant time.

Complexity of some properties

- We've first seen a long series of exciting results showing that many graph properties can be tested in $f(1/\varepsilon)$ time [independent of the input size] and then …

General result

- Every hereditary property can be tested in **constant-time**!

  [Alon & Shapira, 2003-2005]

- Property is **hereditary** if
  - Invariant under vertex removal
    - bipartitness
    - being perfect
    - being chordal
    - having no induced subgraph H
    - …

Main Lemma

Main Lemma:

If G is ε-far from satisfying a hereditary property P, then whp a random subgraph of size $W_P(\varepsilon)$ does not satisfy P

Proof: by a strengthened version of Szemeredi regularity lemma

Can be extended to **hypergraphs**

- via a strengthened version of Szemeredi regularity lemma for hypergraphs

Is hereditary needed?

- There is an NP property, which is closed under edge removal, that cannot be tested with $o(n^2)$ edge queries, even with 2-sided error.

THE UNIVERSITY OF WARWICK

DIMAP

Adjacency matrix model

- There are very fast property testers
- They're very simple
  - Typical algorithm:

> •Select a random set of vertices U
> •Test the property on the subgraph induced by U

- The analysis is (often) very hard
- We understand this model very well
  - mostly because of very close relation to combinatorics
  - Typical running time: (via Szemeredi regularity lemma)

$$2^{2^{\cdot^{\cdot^{\cdot^{2^{1/\varepsilon}}}}}}$$

three or more towers of height $O(1/\varepsilon)$

Adjacency matrix model

- There are very fast property testers
- They're very simple
  - Typical algorithm:

> •Select a random set of vertices U
> •Test the property on the subgraph induced by U

- The analysis is (often) very hard
- We understand this model very well
  - mostly because of very close relation to combinatorics
  - Typical running time: (via Szemeredi regularity lemma)

$$\text{Tower}(c) = \left.2^{2^{2^{\cdot^{\cdot^{2}}}}}\right\} c \text{ times}$$

Adjacency matrix model

- There are very fast property testers
- They're very simple
  - Typical algorithm:

    - Select a random set of vertices U
    - Test the property on the subgraph induced by U

- The analysis is (often) very hard
- We understand this model very well
  - mostly because of very close relation to combinatorics
  - Typical running time: (via Szemeredi regularity lemma) Tower(Tower(Tower(1/ε)))

For ε=0.5 we have Tower(Tower(Tower(1/ε))) =Tower(65536)

Adjacency matrix model

- There are very fast property testers
- They're very simple
  - Typical algorithm:

> •Select a random set of vertices U
> •Test the property on the subgraph induced by U

- The analysis is (often) very hard
- We understand this model very well
  - mostly because of very close relation to combinatorics

- Still: sometimes the runtime is better

  $O(1/\varepsilon)$, $O(1/\varepsilon^2)$, $O(1/2^{\varepsilon})$

Adjacency matrix model

- When can we get $1/\varepsilon^{O(1)}$ query complexity?
- Alon: For any non-bipartite H, testing the property of being H-free with 1-sided error requires $(1/\varepsilon)^{\Omega(\log 1/\varepsilon)}$ queries
- For any $f(\varepsilon)$ there is a monotone graph property, which cannot be tested with $o(f(\varepsilon))$ queries and 1-sided error
  - Monotone properties are hereditary ➔ have $O(1)$ 1-sided error testers

Part II
Adjacency lists model

Testing graph properties in adjacency lists model

- We consider bounded-degree model
  - graph has maximum degree **d** [constant]
- Relatively little is known
- Connection to **combinatorics**!
- Connection to **random walks**!

CS notation:

$f(n) = \tilde{O}(g(n))$ iff $\exists$ $k>0$ $f(n) = O(g(n) \log^k (g(n)))$

Bounded-degree adjacency list model

- **Accept** every graph of max-degree d that satisfies property P
- **Reject** every graph of max-degree d that is ε–far from property P
  - ε-far from P: one has to modify at least $\varepsilon dn/2$ edges to obtain a graph with property P
- **Arbitrary answer** if the graph doesn't satisfy P nor is ε–far from P
- Can err with probability < 1/3
  - Sometimes errs only for "rejects": **1-sided-error**

Bounded-degree adjacency list model

Testing connectivity

What does it mean that a graph G with maximum degree at most d is ε-far from connected?

➔ G has at least $\varepsilon dn/8$ connected components

- not enough…we need many small connected components

Bounded-degree adjacency list model

What does it mean that a graph G with maximum degree at most d is ε-far from connected?

G has at least $\varepsilon dn/8$ connected components

G has $\geq \varepsilon dn/16$ connected components of size $\leq 16/\varepsilon d$

Bounded-degree adjacency list model

What does it mean that a graph G with maximum degree at most d is ε-far from connected?

G has at least $\varepsilon dn/8$ connected components

G has $\geq \varepsilon dn/16$ connected components of size $\leq 16/\varepsilon d$

Proof:
Let G have t connected components of size $\leq 16/\varepsilon d$
and s connected components of size $> 16/\varepsilon d$

Then, $t+s \geq \varepsilon dn/8$

Since $s\,(16/\varepsilon d) \leq n$ we have
$s \leq \varepsilon dn/16$
Hence, $t \geq \varepsilon dn/16$

Bounded-degree adjacency list model

What does it mean that a graph G with maximum degree at most d is ε-far from connected?

G has $\geq$ $\varepsilon dn/16$ connected components of size $\leq 16/\varepsilon d$

Repeat $O(\varepsilon^{-1} d)$ times:
- choose a random vertex v
- run BFS from v until either $16/\varepsilon d+1$ vertices have been visited or the entire connected component has been visited
- if v is contained in a connected component of size $\leq 16/\varepsilon d$ then **reject**

accept

Bounded-degree adjacency list model

Testing connectivity:

Can be done in $O(\varepsilon^{-2} d)$ time

Repeat $O(\varepsilon^{-1} d)$ times:
    choose a random vertex v
    run BFS from v until either $16/\varepsilon d+1$ vertices have been
        visited or the entire connected component has been visited
    if v is contained in a connected component of size $\leq 16/\varepsilon d$
        then reject
accept

Can be improved to $O(\varepsilon^{-1} \operatorname{polylog}(\varepsilon^{-1} d))$ time

Bounded-degree adjacency list model

- Testing bipartitness (2-colorability)
  - Can be done in $\tilde{O}(n^{1/2} / \varepsilon^{O(1)})$ time (Goldreich & Ron)

Algorithm:

- Select $O(1/\varepsilon)$ starting vertices
- For each vertex run poly$(\log n/\varepsilon)$ $n^{1/2}$ random walks of length poly$(\log n/\varepsilon)$
- If any of the starting vertices lies on an odd-length cycle then reject
- Otherwise accept

Bounded-degree adjacency list model

- Testing bipartitness (2-colorability)
  - Can be done in $\tilde{O}(n^{1/2} / \varepsilon^{O(1)})$ time (Goldreich & Ron)
  - Cannot be done faster (Goldreich & Ron)

Bounded-degree adjacency list model

- Testing bipartitness (2-colorability)
  - Can be done in **$\tilde{O}(n^{1/2} / \varepsilon^{O(1)})$** time (Goldreich & Ron)
  - Cannot be done faster (Goldreich & Ron)

Consider two classes of graphs (wlog N – even):

- G_1^N: Hamiltonian cycle + a perfect matching on N nodes
- G_2^N: Hamiltonian cycle + a perfect matching on N nodes,
  but every matching connects two nodes
  at odd distance on the Hamiltonian cycle

G_2^N is bipartite, and whp G_1^N is not; whp G_1^N is 0.01-far from bipartite

Then: an algorithm that performs $o(n^{1/2})$ queries is unable
to distinguish between a graph chosen at random from G_1^N
and a graph chosen at random from G_2^N:

In both cases, the algorithm is unlikely to encounter a cycle

Bounded-degree adjacency list model

- Testing 3-colorability

… requires checking (almost) all vertices and edges!

For general bounded degree graphs,
testing most of natural properties require
superconstant-time (typically, $\Omega(n^{1/2})$)
or even linear-time

Bounded-degree adjacency list model

Which properties can be tested in constant time in the adjacency list model?

Constant time testing

- Even if we cannot test (in constant-time) many properties for general graphs, we can test them for large classes of graphs

THE UNIVERSITY OF WARWICK

DIMAP

Non-expanding families of graphs

- $G=(V,E)$ is a **λ-expander** if
  - $N(S) \geq \lambda |S|$ for all $S \subset V$ with $|S| \geq |V|/2$
- **Graph G is C-strongly non-expanding** if
  - every induced subgraph of G with at least C vertices is not a $(1/\log^2 n)$-expanders

Key property:
non-expanding families of graphs have good separators

Testing in non-expanding families of graphs

- In the bounded degree graph model any hereditary property is testable in constant-time if the input graph belongs to a C-strongly non-expanding family of graphs (for some constant C)

Example:
Testing in bounded degree planar graphs

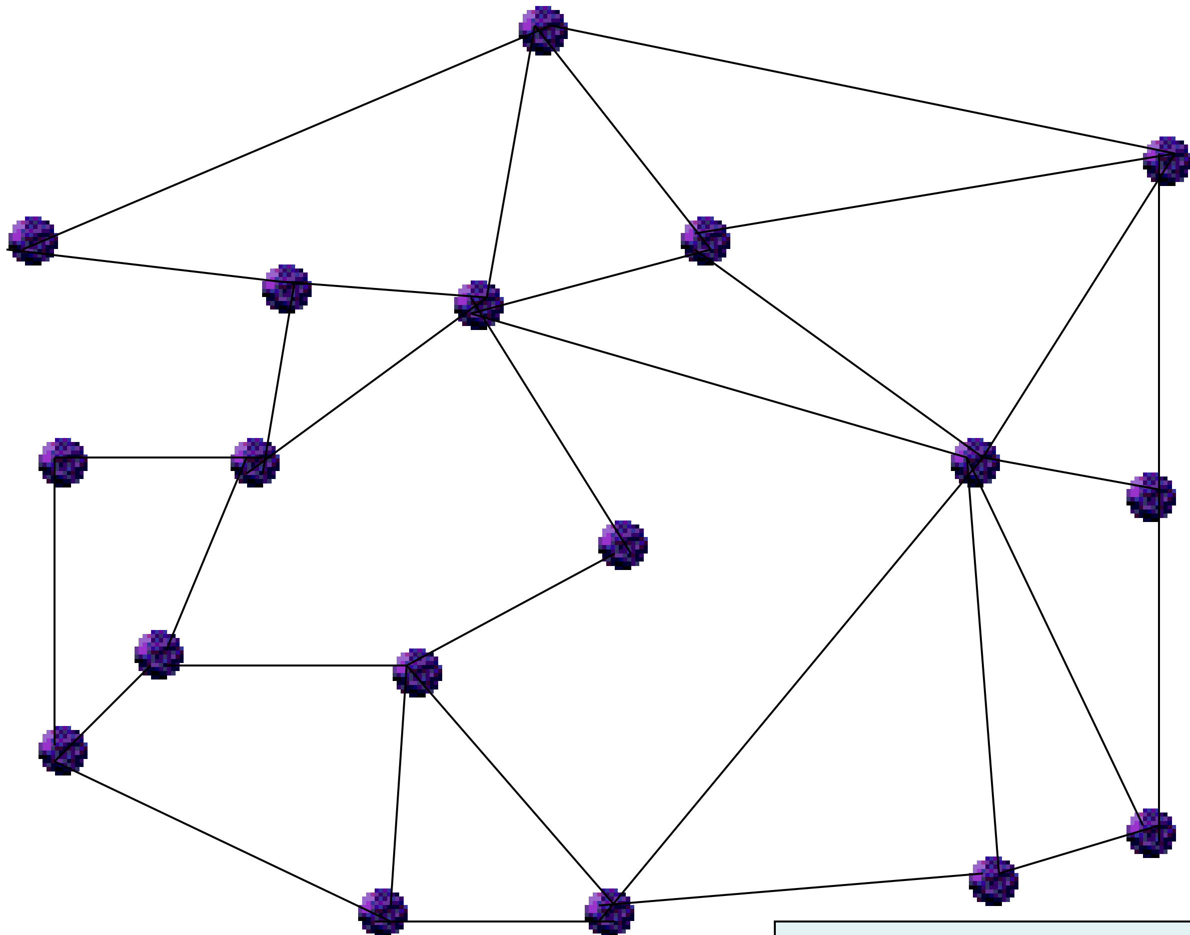
- Testing any hereditary property in planar graphs of constant degree can be done in constant time
  - bipartitness
  - being perfect
  - being chordal
  - having no induced subgraph H
  - …

- We'll sketch a proof that testing hereditary properties in planar graphs of bounded degree can be done in constant time

[assuming ε is a constant]

- For a given hereditary property P
  We have to design an algorithm that for any planar graph G of maximum degree d
  - will accept G if G satisfies P
  - [with prob $\geq$ 2/3] will reject G if G is ε-far from P

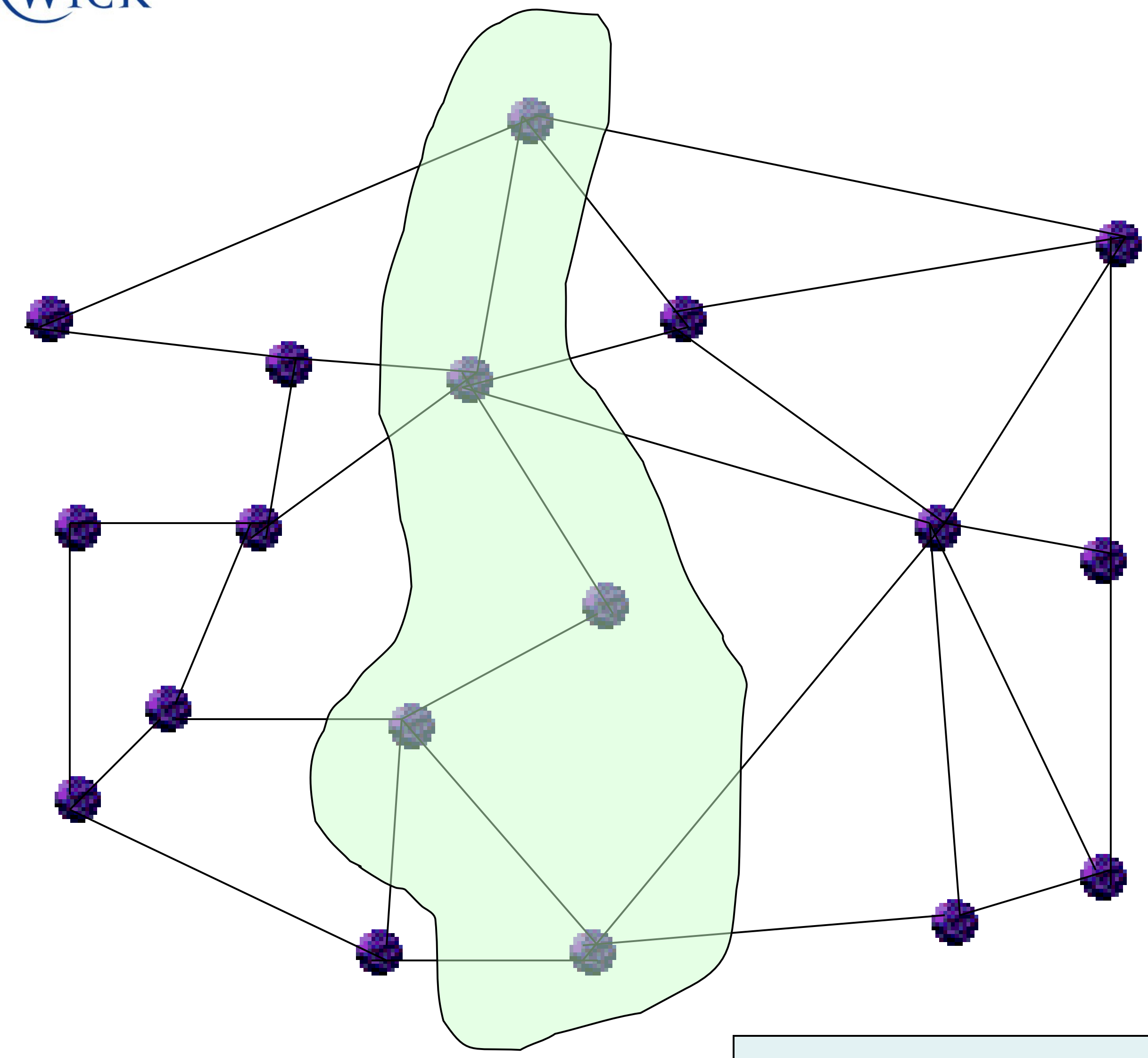
- Algorithm will accept unless it finds a "proof" that G doesn't satisfy P



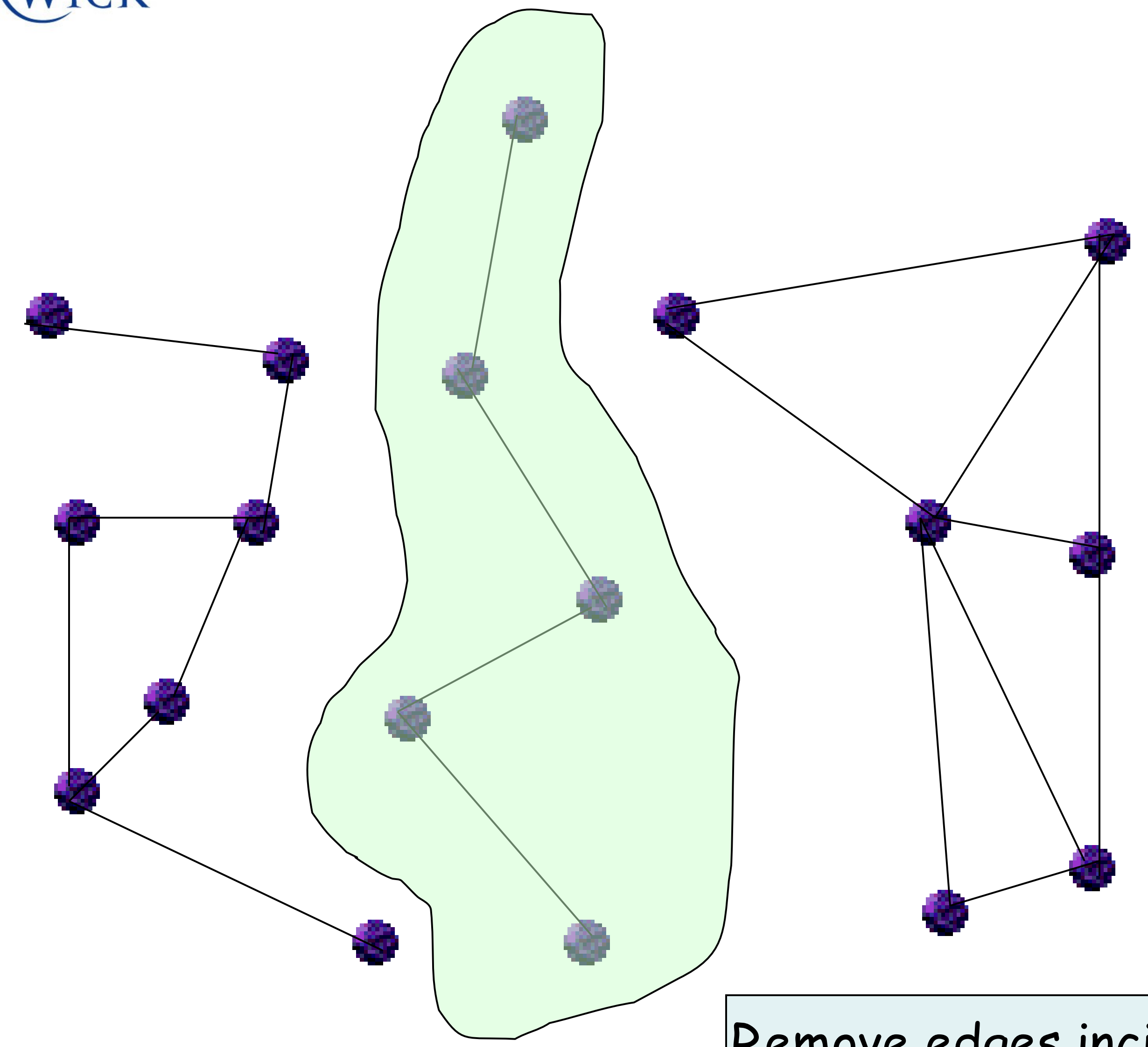
Consider testing a simple hereditary property – say, being bipartite

We only have to consider graphs that are ε-far from satisfying the property

How to **REJECT** them?

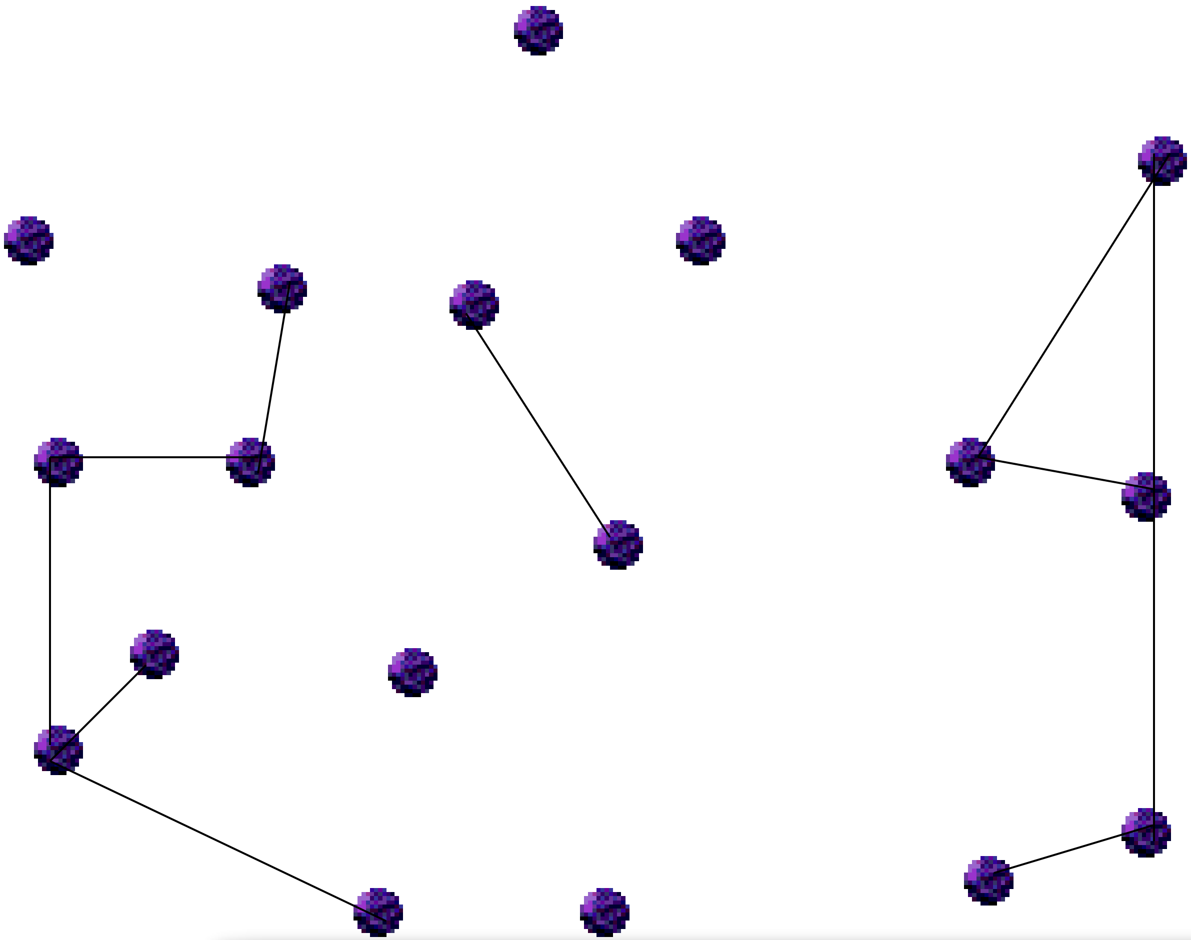


Separator of size $O(n^{1/2})$



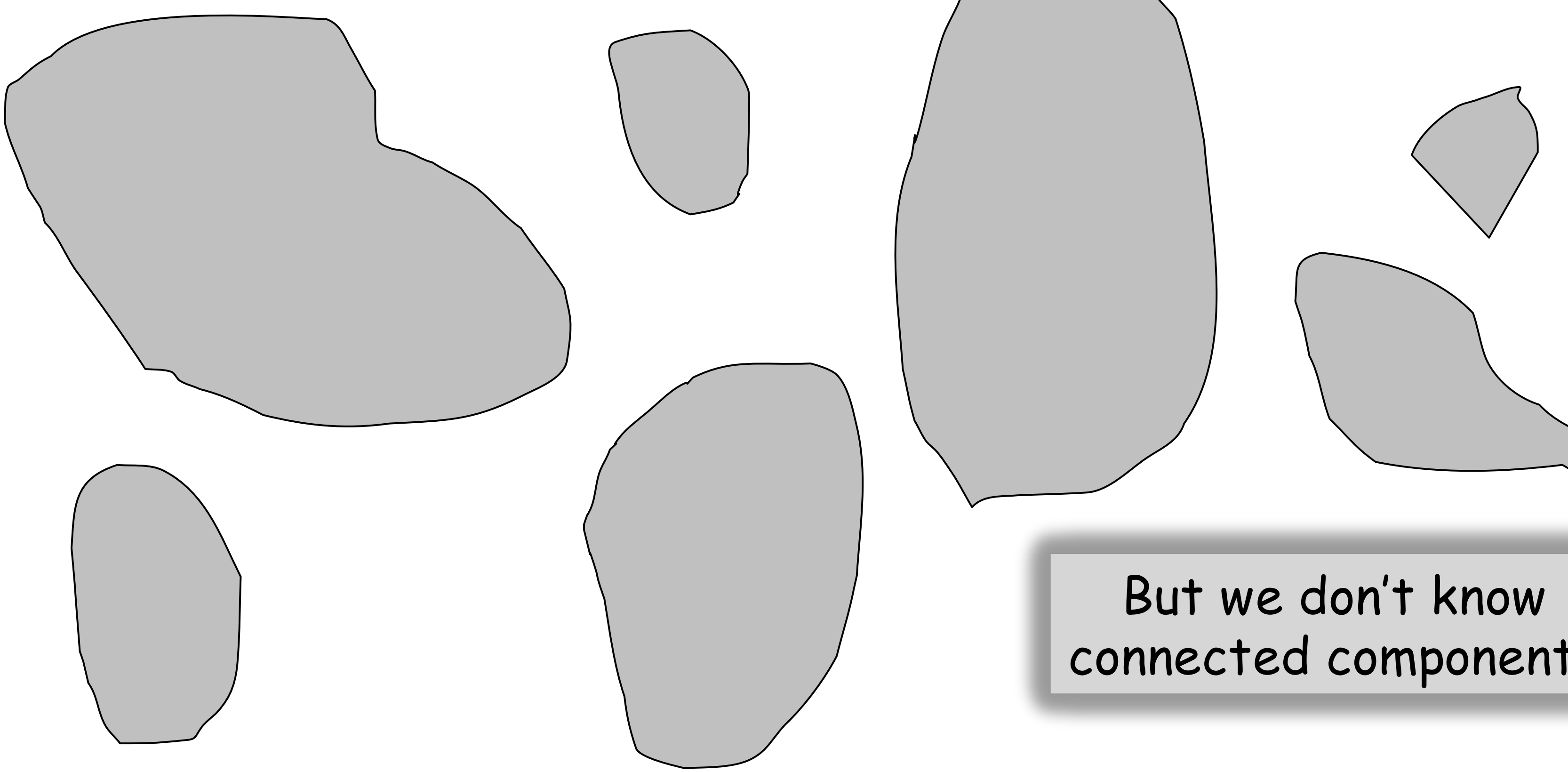
Remove edges incident to separator
And repeat recursively

Until only small connected components left



We've removed only a few edges (‹‹ εn)
If the input graph is ε–far then
The obtained graph is $\varepsilon/2$-far!

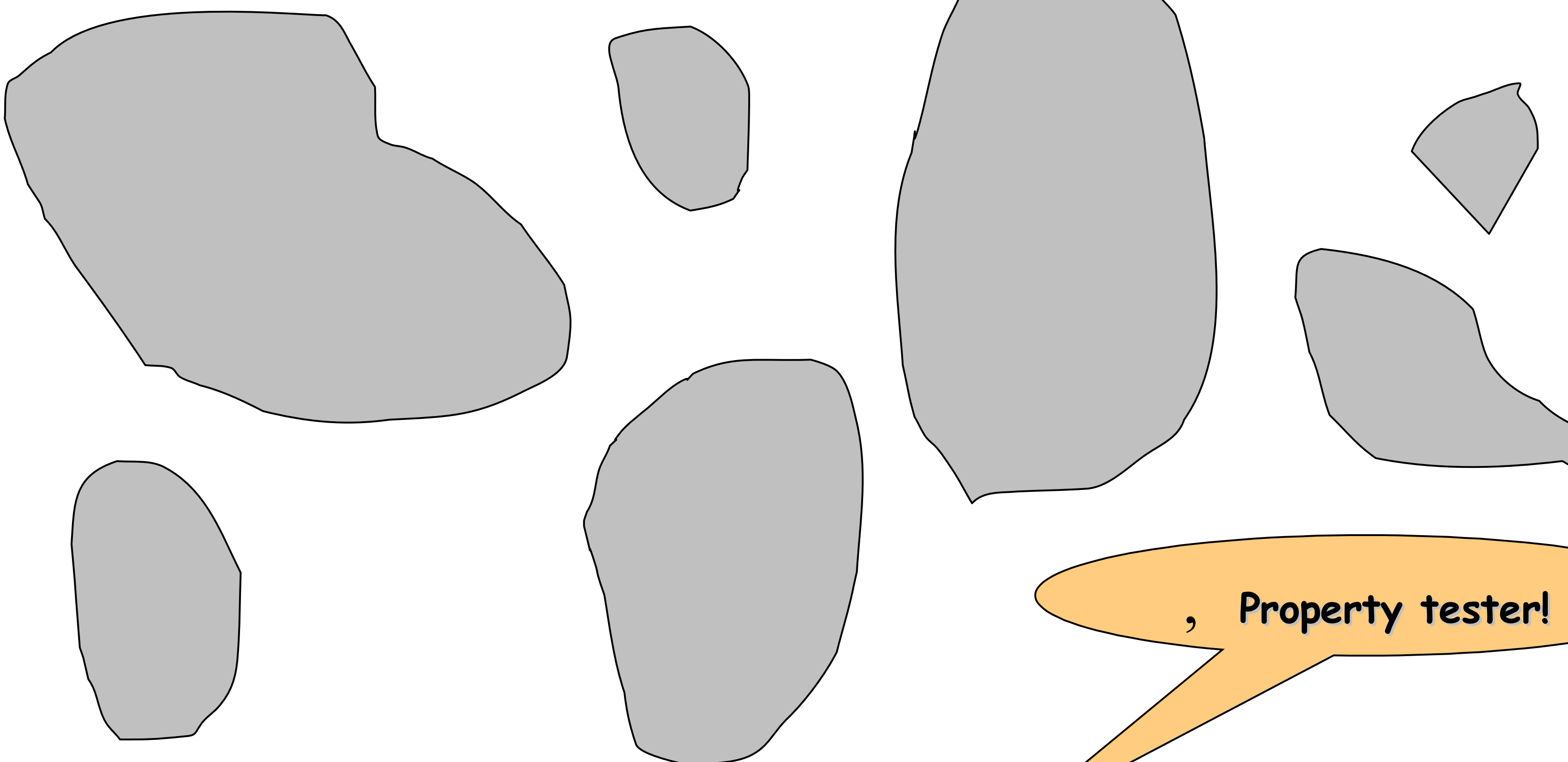
Obtained graph is ε/2-far from P ,
many [constant fraction of] components "prove"
that the graph doesn't satisfy P



But we don't know
connected components

**If we knew connected components
we could check if the obtained graph satisfies
the property by sampling ~O(d/ε) random vertices
and checking their connected components**

Obtained graph is $\varepsilon/2$-far from P ,
many [constant fraction of] components "prove" that the graph doesn't satisfy P



, **Property tester!**

Pick random sample of ~O(d/ε) vertices
For each vertex explore its neighborhood (of constant size)
If the input graph is ε-far:
the induced subgraph shouldn't satisfy the property!

What's the complexity/runtime? $\sim O((d/\varepsilon)^{O(d/\varepsilon)})$

Pick random sample of $\sim O(d/\varepsilon)$ vertices
For each vertex explore its neighborhood (of constant size)
If the input graph is ε-far:
the induced subgraph shouldn't satisfy the property!

Property testers

- One can make this idea to work to design property testers for planar graphs (of constant max-degree) **for all hereditary properties**

- Key property: every hereditary property can be characterized by a set of minimal forbidden induced subgraphs

  For example:
  no-bipartite = has a cycle of odd length
  no-chordal = has a cycle of length > 3

- Hence: we only have to check if these subgraphs don't exist in small components

Property testers

- One doesn't need planar graphs:
  - It's enough to have **some separator properties**
- Works for all **C-strongly non-expanding families** of graphs

Non-expanding graphs

- $G=(V,E)$ is a **λ-expander** if
  - $N(S) \geq \lambda\,|S|$ for all $S \subset V$ with $|S| \geq |V|/2$
- **Graph G is C-strongly non-expanding** if
  - every induced subgraph of G with at least C vertices is not a $(1/\log^2 n)$-expanders

Non-expanding graphs vs. separators

Let $G=(V,E)$ be a C-strongly non-expanding graph of maximum degree d. Let k be an arbitrary parameter, k>0.

If $n = |V| \geq \max\{2C, 2^{2/k^2}\}$ then one can partition V into V_1 and V_2 such that

- $|V_1|, |V_2| \geq n/4$ and
- $e(V_1,V_2) \leq kdn / \log^{1.5} n$.

Non-expanding graphs vs. separators

For every C-strongly non-expanding graph $G=(V,E)$ of maximum degree d there exists a positive constant c such that one can remove from G at most $\varepsilon dn/2$ edges such that

- their removal partitions G into connected components $C_1, C_2, \ldots$ of size at most $2^{c/\varepsilon^2}$ each,
- each connected component C_i is an induced subgraph of G, and
- no edge connects in G two non-trivial connected components C_i and C_j.

Tester

choose a random sample S of vertices

$|S| = O(1)$ – depending on d, ε, graph family, property to be tested

for each vertex v in S

let $N_r[v]$ be the r-th neighborhood of v

$r = O(1)$ – depending on d, ε, graph family, property to be tested

If the graph induced by $\bigcup_{v \in S} N_r[v]$ satisfies the property then ACCEPT

else REJECT

Complexity of the tester

- Complexity is O(1) for constant d and ε
- Dependency on d and ε is low
- But dependency on hereditary property/graph family might be large (but it's an absolute constant)
- All depend on the properties at hand
  - Testing planar graphs for "basic" hereditary properties (k-coloring, chordal, perfect, no induced subgraph H) in time $2^{(d/\varepsilon)^{O(1)}}$
- [Think: very fast when comparing to "constant-time" bounds for adjacency matrix model]

O(1) testing in adjacency list model?

- Very few properties known (for general graphs)
  - connectivity
  - k-connectivity
  - H-freeness
  - …
  - ~~very few more~~

This was the state of the art until 5-6 months ago.
Now: **Every minor-closed property is testable with O(1) queries**
Benjamini, Schramm, Shapira, STOC'2008

Constant time testing !

For example, there are graphs on $\omega(n)$ nodes with $\omega(1)$ girth

Testing planar graphs can be done with O(1) queries

- Why is it surprising?
- There are graphs G such that
  - any connected subgraph of G of constant size is planar
  - G is ε-far from planar

So: how come could we test planarity by checking only subgraphs of constant size?

For each subgraph of constant size,
check the number of its occurrences in G
No all frequencies are possible in planar graphs!

Checking planarity in constant time

Let F_{dk} be the family of all connected graphs of maximum degree d on at most k vertices

Let $F_{dk}[G]$ be the characteristic vector of length $|F_{dk}|$ such that if H is the ith element of F_{dk} then the ith element of the vector equals the number of occurrences of H as an induced subgraph of G

Theorem: If G is ε-far from planar then its vector $F_{dk}[G]$ significantly differ from $F_{dk}[G']$ any planar graph G'

Testing planarity ~ checking the characteristic vector $F_{dk}[G]$

We don't need to know the exact values of the vector: approximation is enough

Checking planarity in constant time

Let F_{dk} be the family of all connected graphs of maximum degree d on at most k vertices

Let $F_{dk}[G]$ be the characteristic vector of length $|F_{dk}|$ such that if H is the ith element of F_{dk} then the ith element of the vector equals the number of occurrences of H as an induced subgraph of G

Testing planarity ~ checking the characteristic vector $F_{dk}[G]$

We don't need to know the exact values of the vector: approximation is enough

Randomly sample O(1) vertices
For each sampled vertex v
run BFS from v of O(1) depth
let H_v be the obtained graph
Accept or reject G using only graphs H_v to estimate $F_{dk}[G]$

Extension: all minor-closed properties

- Every minor-closed property can be tested in a similar way
- Minor-closed properties include:
  - Planar,
  - Outer-planar,
  - Series-parallel,
  - Bounded-genus,
  - bounded tree-width,
  - …

- Minor = obtained by edge/vertex removal + edge contractions
- P is minor-closed if every minor of a graph in P is also in P

These techniques don't work for arbitrary-degree graphs

Testing planarity in arbitrary degree graphs requires $\Omega(n^{1/2})$ time

Open problem: can it be done in $O(n^{1/2})$ time?

Future of Property Testing

We need general results

Relation to

- approximation algorithms
- distributed algorithms
- streaming algorithms

Conclusions

- Modern applications need very fast algorithms
- Property testing:
  - Framework to study graph/network properties
  - Can be used to design some very fast testers
- Key questions:
  - Which problems/properties can be tested efficiently?
- Beautiful and nontrivial mathematics behind

References

Surveys:

- E. Fischer. The art of uninformed decisions: A primer to property testing. *Bulletin of the EATCS*, 75: 97–126, October 2001.
- O. Goldreich. Property testing in massive graphs. In J. Abello, P. M. Pardalos, and M. G. C. Resende, editors, *Handbook of Massive Data Sets*, pp. 123–147. Kluwer Academic Publishers, 2002.
- R. Kumar and R. Rubinfeld. Sublinear time algorithms. *SIGACT News*, 34: 57–67, 2003.
- D. Ron. Property testing. In P. M. Pardalos, S. Rajasekaran, J. Reif, and J. D. P. Rolim, editors, *Handobook of Randomized Algorithms*, volume II, pp. 597–649. Kluwer Academic Publishers, 2001.
- A. Czumaj and C. Sohler. Sublinear-time algorithms. *Bulletin of the EATCS*, 89: 23–47, June, 2006.

References

Key papers:

- N. Alon and A. Shapira. A characterization of the (natural) graph properties testable with one-sided error. *SIAM Journal on Computing*, 37(6): 1703-1727, 2008.
- I. Benjamini, O. Schramm, and A. Shapira. Every minor-closed property of sparse graphs is testable. *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC)*, pp. 393–402, 2008.
- A. Czumaj and C. Sohler. On testable properties in bounded degree graphs. *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 494–501, 2007.
- O. Goldreich, S. Goldwasser, and D. Ron. Property testing and its connection to learning and approximation. *Journal of the ACM*, 45(4): 653–750, 1998.
- O. Goldreich and D. Ron. Property testing in bounded degree graphs. *Algorithmica*, 32(2): 302–343, 2002.
- N. Alon, E. Fischer, I. Newman, and A. Shapira. A combinatorial characterization of the testable graph properties: it's all about regularity. *Proceedings of the 38th Annual ACM Symposium on Theory of Computing (STOC)*, pp. 251–260, 2006.
- A. Czumaj and C. Sohler. Abstract combinatorial programs and efficient property testers. *SIAM Journal on Computing*, 34(3): 580–615, 2005.
- O. Goldreich and D. Ron. A sublinear bipartiteness tester for bounded degree graphs. *Combinatorica*, 19(3):335–373, 1999.

Problems for students

Using Szemeredi lemma

Szemeredi Regularity Lemma:

For any δ, any graph G can be partitioned into k, $1/\delta \leq k \leq T(\delta)$, subsets $V_1,\ldots,V_k$ of equal size, such that all but at most δk^2 of the pairs (V_i, V_j) are δ-regular

Find a partition of V into $V_1,\ldots,V_k$ with $k<f(\varepsilon)$ and $k \gg 1/\varepsilon$, such that all but at most of the pairs are δ-regular for some constant $\delta=\delta(\varepsilon) \ll \varepsilon$

Edge $e=(x,y)$ with $x \in V_i$ and $y \in V_j$, is **useful** if

- $V_i \neq V_j$,
- (V_i,V_j) is δ-regular, and
- the density between V_i and V_j is at least $\varepsilon/15$

Lemma: There are less than εn^2 non-useful edges

THE UNIVERSITY OF WARWICK

DIMAP

Using Szemeredi lemma

- Let G be ε–far from triangle free
- Remove all non-useful edges to define graph G'
- Since G has less than εn^2 non-useful edges, G' must has at least one triangle ➔
  - There are three useful edges (x,y), (y,z), (z,x) with $x \in V_i$, $y \in V_j$, $z \in V_s$, such that
    - all sets V_i, V_j, V_s are distinct,
    - all sets V_i, V_j, V_s are pairwise δ-regular, and
    - the density between each pair V_i, V_j, V_s is at least $\varepsilon/15$.

There are $\Theta(n^3)$ triangles between V_i, V_j, V_s

Non-expanding graphs vs. separators

- $G=(V,E)$ is a **λ-expander** if
  - $N(S) \geq \lambda\,|S|$ for all $S \subset V$ with $|S| \geq |V|/2$

- **Graph G is C-strongly non-expanding** if
  - every induced subgraph of G with at least C vertices is not a $(1/\log^2 n)$-expanders

Let $G=(V,E)$ be a C-strongly non-expanding graph of maximum degree d. Let k be an arbitrary parameter, k>0.

If $n = |V| \geq \max\{2C, 2^{2/k^2}\}$ then one can partition V into V_1 and V_2 such that

- $|V_1|, |V_2| \geq n/4$ and
- $e(V_1,V_2) \leq kdn / \log^{1.5} n$.

Non-expanding graphs vs. separators

For every C-strongly non-expanding graph G=(V,E) of maximum degree d there exists a positive constant c such that one can remove from G at most $\varepsilon dn/2$ edges such that

- their removal partitions G into connected components $C_1, C_2, \ldots$ of size at most $2^{c/\varepsilon^2}$ each,
- each connected component C_i is an induced subgraph of G, and
- no edge connects in G two non-trivial connected components C_i and C_j.

Testing connectivity

- In the bounded-degree model with adjacency lists, design a property testing algorithm for connectivity with the running time

$$O(\varepsilon^{-1} \text{polylog}(\varepsilon^{-1}/d))$$

MST

- Let $G=(V,E)$ be an edge-weighted graph and suppose that all edges are in $\{1,2\}$.
- Let $c(i)$ = #connected components of the subgraph of G induced by edges of weight at most i
- Show that $MST(G) = n-2+c(1)$

MST

- Let $G=(V,E)$ be an edge-weighted graph and suppose that all edges are in $\{1,2,\dots,W\}$.
- Let $c(i)$ = #connected components of the subgraph of G induced by edges of weight at most i
- Show that $MST(G) = n-W+c(1)+c(2)+\dots+c(W-1)$
- How could you use this approach to estimate the cost of $MST(G)$?

Approximating #connected components

choose vertices $u_1, \ldots, u_s$ at random

for each vertex i do

- choose X according to $\Pr[X \geq k] = 1/k$
- Run BFS starting at u_i until either
  1. Entire connected component containing u_i has been explored, or
  2. X vertices have been explored
- If BFS stopped in case 1 then $b_i = 1$
- else $b_i = 0$

Output est $= n/s \sum_i b_i$

- Compute $E[b_i]$ and $Var[b_i]$
- Compute E[est] and Var[est]
- Compute $\Pr[|est-E[est]| \leq \lambda n]$
- Use this to estimate the cost of MST