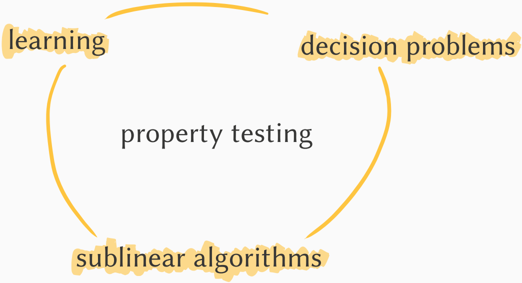
Property Testing of Graphs and the Role of Neighborhood Distributions

Hendrik Fichtenberger

February 11, 2020

Property Testing: in Context



Property Testing: in Context

property testing makes
approximative decisions

learning

decision problems

property testing

sublinear algorithms

Property Testing: in Context

property testing makes
approximative decisions

learning

decision problems

property testing

sublinear algorithms

property testing minimizes
the number of input bits read

Property Testing: in Context

proper PAC learning
implies property testing

property testing makes
approximative decisions

learning

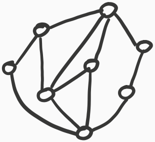
decision problems

property testing

sublinear algorithms

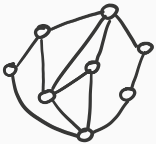
property testing minimizes
the number of input bits read

Property Testing: in a Nutshell

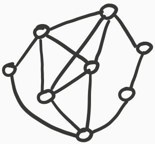


planar ✓

Property Testing: in a Nutshell

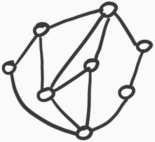


planar ✓

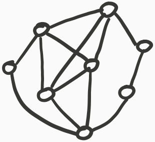


non-planar ✗

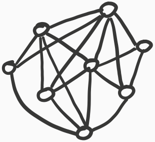
Property Testing: in a Nutshell



planar

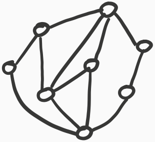


non-planar

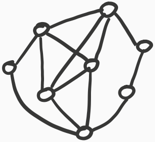


non-planar

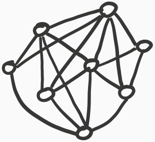
Property Testing: in a Nutshell



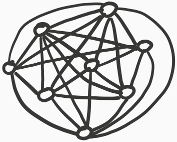
planar ✓



non-planar ✗

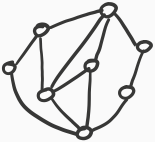


non-planar ✗

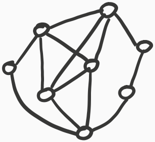


non-planar ✗

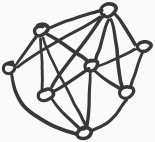
Property Testing: in a Nutshell



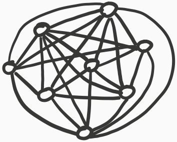
planar



non-planar



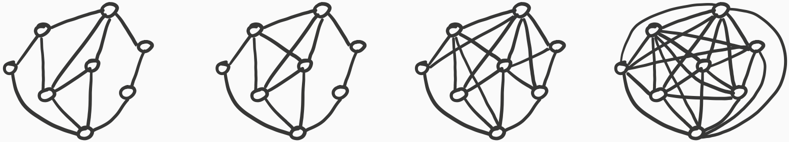
non-planar



non-planar

time complexity: $\Omega(|V|)$

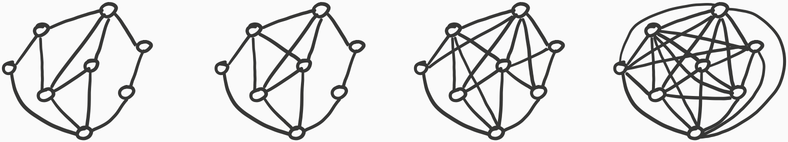
Property Testing: in a Nutshell



planar   non-planar   non-planar   non-planar

time complexity: $\Omega(|V|)$

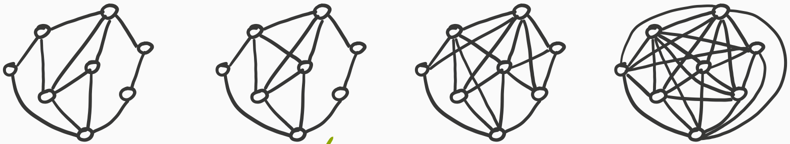
Property Testing: in a Nutshell



planar ✓ | slightly non-planar ✗ | quite non-planar ✗ | very non-planar ✗

time complexity: $\Omega(|V|)$

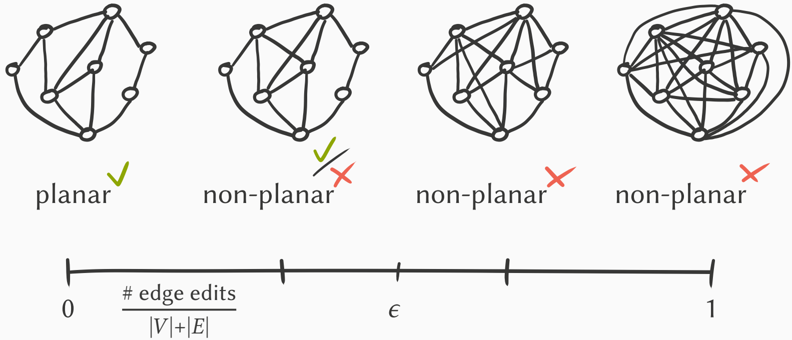
Property Testing: in a Nutshell



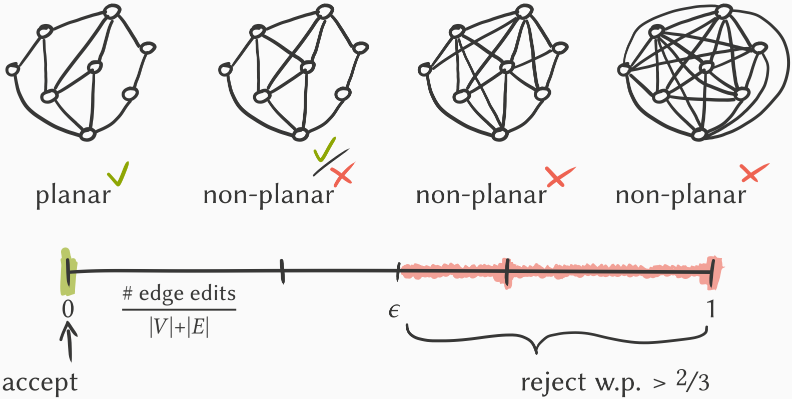
planar | slightly non-planar | quite non-planar | very non-planar

time complexity: $\Omega(|V|)$

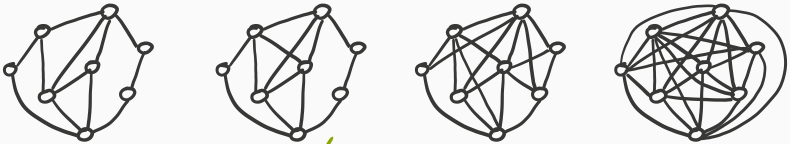
Property Testing: in a Nutshell

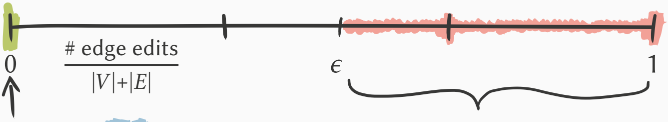


Property Testing: in a Nutshell



Property Testing: in a Nutshell



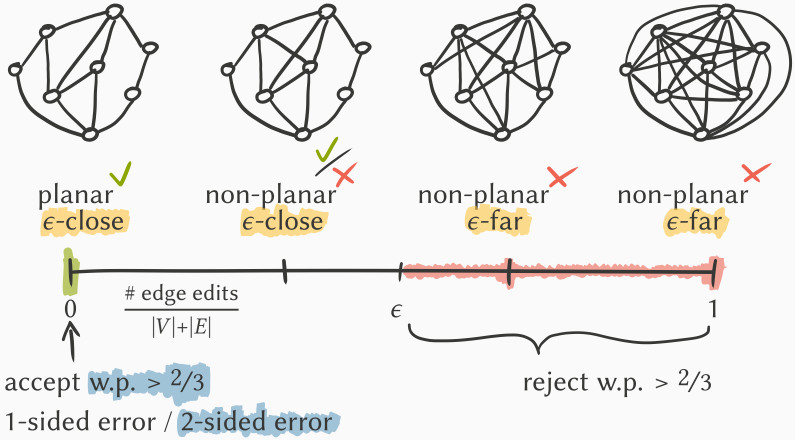


accept w.p. > 2/3

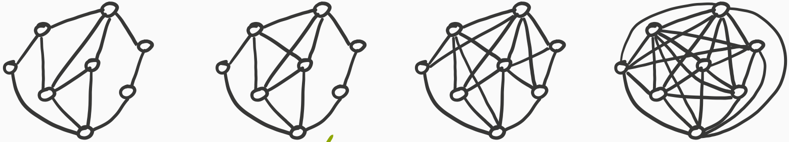
reject w.p. > 2/3

1-sided error / 2-sided error

Property Testing: in a Nutshell



Property Testing: in a Nutshell

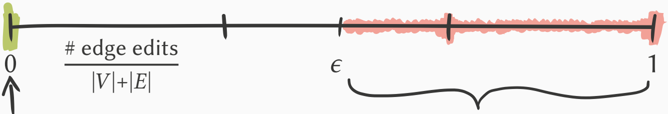


planar ϵ-close

non-planar ϵ-close

non-planar ϵ-far

non-planar ϵ-far

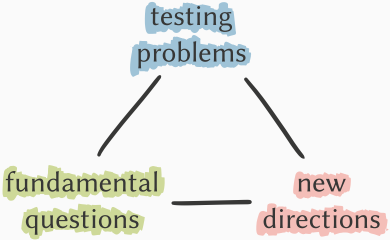


accept w.p. > $^2/_3$

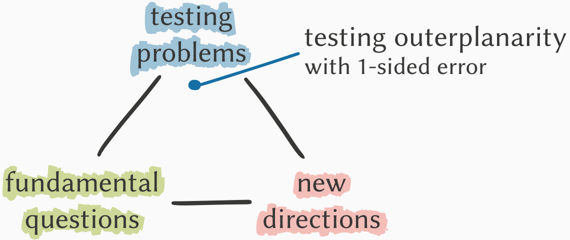
reject w.p. > $^2/_3$

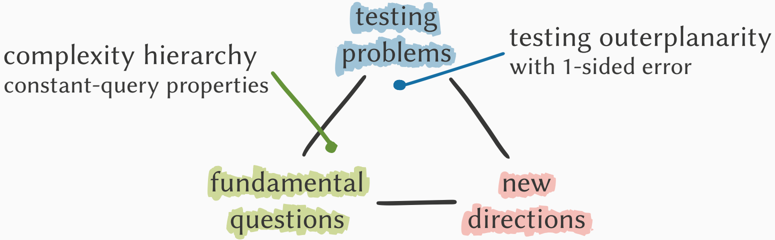
1-sided error / 2-sided error

complexity: # queries to data structure


testing problems
fundamental questions
new directions

Property Testing: in my Thesis




testing
problems
complexity hierarchy
constant-query properties
testing outerplanarity
with 1-sided error
fundamental
questions
new
directions

Property Testing: in my Thesis

testing problems

testing outerplanarity
with 1-sided error

complexity hierarchy
constant-query properties

fundamental questions

new directions

links between models
from random access to streams

Property Testing: in my Thesis

testing problems

complexity hierarchy
constant-query properties

testing outerplanarity
with 1-sided error

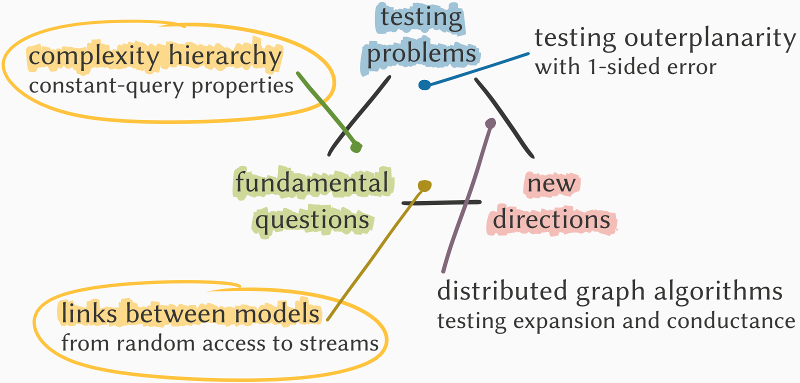
fundamental questions

new directions

links between models
from random access to streams

distributed graph algorithms
testing expansion and conductance

Property Testing: in my Thesis



The Bounded-Degree Model

- ☒ bounded-degree model: $\forall v \in V : d(v) \leq d, d \in O(1), n := |V|$
- ☒ input structure: adjacency lists (1 query $\hat{=}$ 1 entry)
- ☒ error: 2-sided

The Bounded-Degree Model

- ☒ bounded-degree model: $\forall v \in V : d(v) \leq d, d \in O(1), n := |V|$
- ☒ input structure: adjacency lists (1 query $\hat{=}$ 1 entry)
- ☒ error: 2-sided

$q(\epsilon, d)$ planar, degree-regular, cycle-free, subgraph-free, connected, minor-free, hyperfinite, ...

The Bounded-Degree Model

- ☒ bounded-degree model: $\forall v \in V : d(v) \leq d, d \in O(1), n := |V|$
- ☒ input structure: adjacency lists (1 query $\hat{=}$ 1 entry)
- ☒ error: 2-sided

$q(\epsilon, d)$ — planar, degree-regular, cycle-free, subgraph-free, connected, minor-free, hyperfinite, ...

$\Theta(\sqrt{n})$ — 2-colorability, expander

$\Omega(n)$ — 3-colorability

The Bounded-Degree Model

- ☒ bounded-degree model: $\forall v \in V : d(v) \leq d, d \in O(1), n := |V|$
- ☒ input structure: adjacency lists (1 query $\hat{=}$ 1 entry)
- ☒ error: 2-sided

$q(\epsilon, d)$ — planar, degree-regular, cycle-free, subgraph-free, connected, minor-free, hyperfinite, ...

no dependence on n

---

dependence on n

$\Theta(\sqrt{n})$ — 2-colorability, expander

$\Omega(n)$ — 3-colorability

The Bounded-Degree Model

- ☒ bounded-degree model: $\forall v \in V : d(v) \leq d, d \in O(1), n := |V|$
- ☒ input structure: adjacency lists (1 query $\hat{=}$ 1 entry)
- ☒ error: 2-sided

$q(\epsilon, d)$ — planar, degree-regular, cycle-free, subgraph-free, connected, minor-free, hyperfinite, ...

no dependence on n

why?

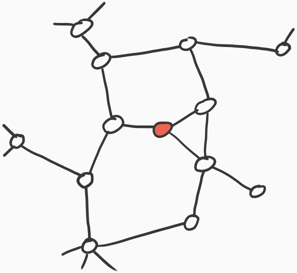
dependence on n

$\Theta(\sqrt{n})$ — 2-colorability, expander

$\Omega(n)$ — 3-colorability

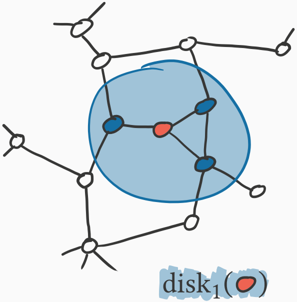
k-Disks / k-Hop Neighborhoods

$\text{disk}_k(v)$: subgraph induced by $\text{BFS}(v)$ of depth k

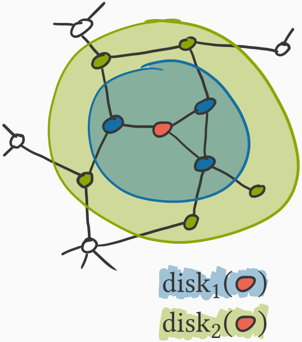


k-Disks / k-Hop Neighborhoods

$\text{disk}_k(v)$: subgraph induced by BFS(v) of depth k



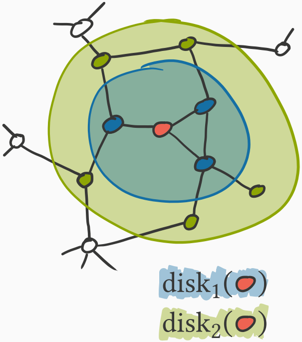
$\text{disk}_k(v)$: subgraph induced by $\text{BFS}(v)$ of depth k



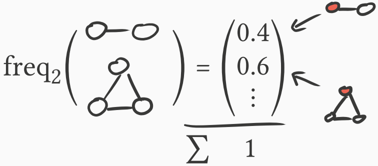
Goldreich, Ron, STOC'09

k-Disks / k-Hop Neighborhoods

$\mathrm{disk}_k(v)$: subgraph induced by $\mathrm{BFS}(v)$ of depth k

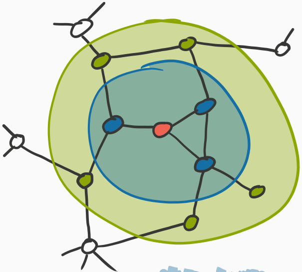


$\mathrm{freq}_k(G)$: for each k-disk isomorphism type calculate its share of vertices



k-Disks / k-Hop Neighborhoods

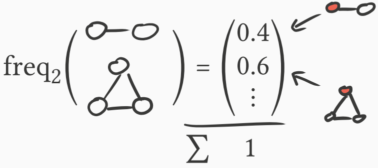
$\mathrm{disk}_k(v)$: subgraph induced by $\mathrm{BFS}(v)$ of depth k



$\mathrm{disk}_1(\ )$

$\mathrm{disk}_2(\ )$

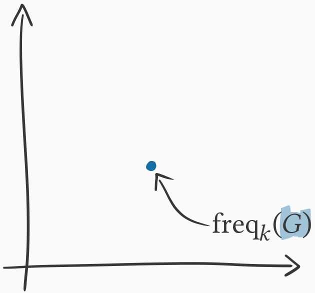
$\mathrm{freq}_k(G)$: for each k-disk isomorphism type calculate its share of vertices

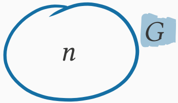


[GR'09]

Π constant-query testable iff $\mathrm{freq}_k(G)$ indicates membership

Small Frequency-Preserver Graphs



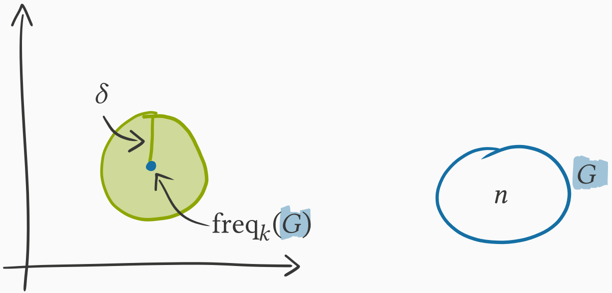


Theorem [Alon'11]

For every $\delta, k > 0$, there exists $M(\delta, k)$ such that for every G there exists H of size at most $M(\delta, k)$ and $\|\text{freq}_k(G) - \text{freq}_k(H)\|_1 < \delta$.

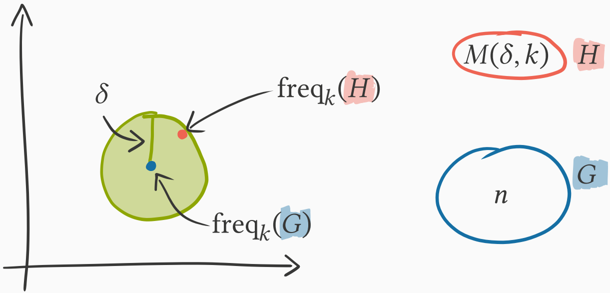
Alon, '11, see: Lovász, *Large Networks and Graph Limits*, Proposition 19.10

Small Frequency-Preserver Graphs



Theorem [Alon'11]

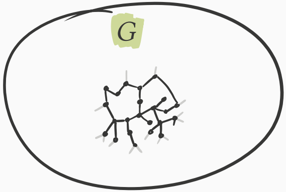
For every $\delta, k > 0$, there exists $M(\delta, k)$ such that for every G there exists H of size at most $M(\delta, k)$ and $\|\mathrm{freq}_k(G) - \mathrm{freq}_k(H)\|_1 < \delta$.



Theorem [Alon'11]

For every $\delta, k > 0$, there exists $M(\delta, k)$ such that for every G there exists H of size at most $M(\delta, k)$ and $\|\mathrm{freq}_k(G) - \mathrm{freq}_k(H)\|_1 < \delta$.

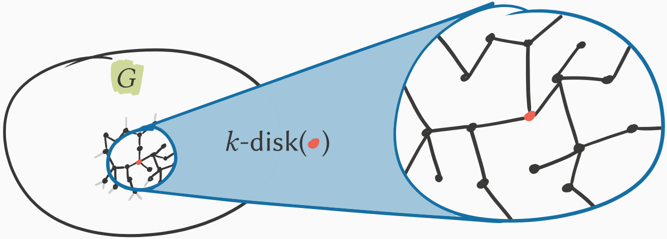
Alon, '11, see: Lovász, *Large Networks and Graph Limits*, Proposition 19.10



Theorem [with PS]

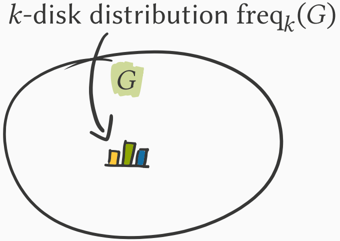
If all k-disks in G are cycle-free, there is a constant-time algorithm that constructs a frequency-preserver of size $M(\delta, k) \leq 10^4 d^{11k}/\epsilon^2\delta$.

Frequency-Preservers for Cycle-Free k-disks



Theorem [with PS]

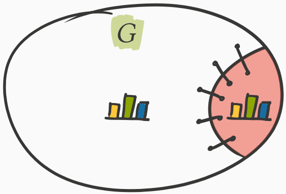
If all k-disks in G are cycle-free, there is a constant-time algorithm that constructs a frequency-preserver of size $M(\delta, k) \leq 10^4 d^{11k}/\epsilon^2\delta$.



Theorem [with PS]

If all k-disks in G are cycle-free, there is a constant-time algorithm that constructs a frequency-preserver of size $M(\delta, k) \leq 10^4 d^{11k}/\epsilon^2\delta$.

Frequency-Preservers for Cycle-Free k-disks

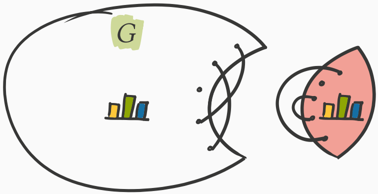


Theorem [with PS]

If all k-disks in G are cycle-free, there is a constant-time algorithm that constructs a frequency-preserver of size $M(\delta, k) \leq 10^4 d^{11k} / \epsilon^2 \delta$.

joint work with Peng, Sohler

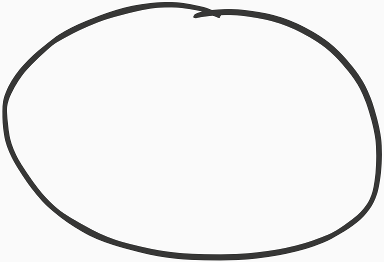
Frequency-Preservers for Cycle-Free k-disks



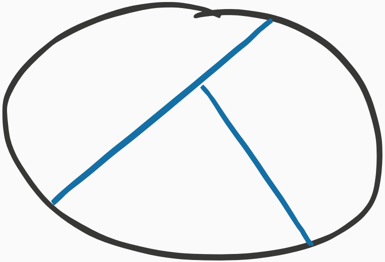
Theorem [with PS]

If all k-disks in G are cycle-free, there is a constant-time algorithm that constructs a frequency-preserver of size $M(\delta, k) \leq 10^4 d^{11k}/\epsilon^2\delta$.

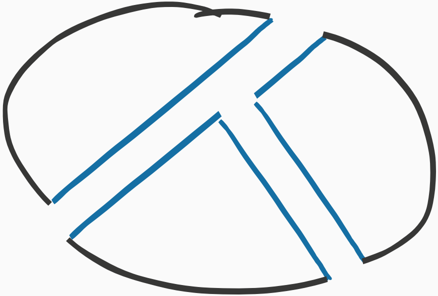
Hyperfinite Graphs



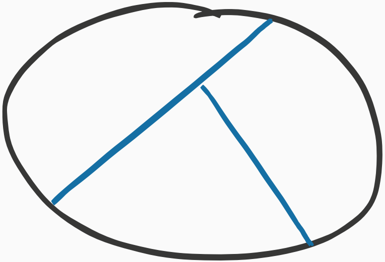
Hyperfinite Graphs



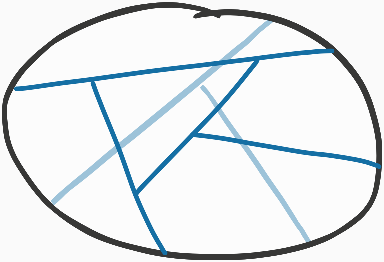
Hyperfinite Graphs



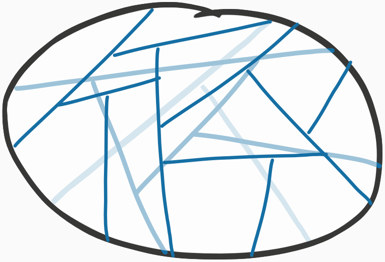
Hyperfinite Graphs



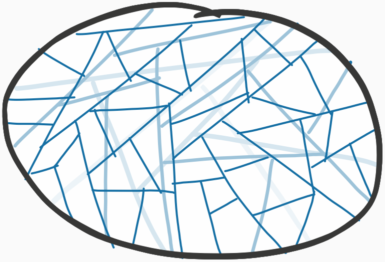
Hyperfinite Graphs



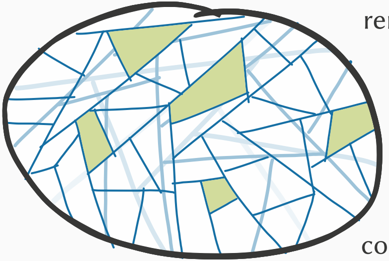
Hyperfinite Graphs



Hyperfinite Graphs

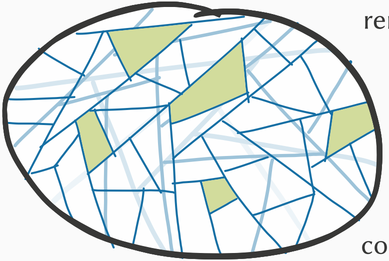


Hyperfinite Graphs



remove ϵdn edges for any $\epsilon > 0$

con. comp. of size $\leq \rho(\epsilon)$ for some ρ



remove ϵdn edges for any $\epsilon > 0$

con. comp. of size $\leq \rho(\epsilon)$ for some ρ

Theorem (informal) [BSS'08]

If graph G is $\rho(\epsilon)$-hyperfinite, then any graph H with $\mathrm{freq}(G) \approx \mathrm{freq}(H)$ is $\rho'(\epsilon)$-hyperfinite for some $\rho' \approx \rho$.

Constant-Query Properties

Π is ρ-hyperfinite

Π has constant query complexity

Constant-Query Properties

[NS'11]

Π is ρ-hyperfinite

Π has constant query complexity

Constant-Query Properties

$$\begin{tikzcd}[column sep=huge] \Pi \text{ is } \rho\text{-hyperfinite} \arrow[r, bend left, "{[\text{NS'11}]}"] & \begin{array}{l}\Pi \text{ has constant} \\ \text{query complexity}\end{array} \arrow[l, bend left, "?"] \end{tikzcd}$$

Newman, Sohler, STOC '11

Constant-Query Properties

[NS'11]

Π is ρ-hyperfinite

Π has constant query complexity

e.g. Π = connectivity

contains expanders

Constant-Query Properties

[NS'11]

Π is ρ-hyperfinite

Π has constant query complexity

e.g. Π = connectivity

contains expanders

$\exists\, \Pi' \subseteq \Pi, |\Pi'| = \infty$:

Π' is ρ'-hyperfinite

Theorem [with PS]

Every non-trivial, constant-query testable property of bounded-degree graphs contains an infinite hyperfinite subproperty.

The General Graph Model

- ☒ bounded-degree model
- ☒ input structure: adjacency lists
- ☒ error: 2-sided

The General Graph Model

- ~~bounded degree model~~ general graphs
- input structure: adjacency lists
- error: 2-sided

The General Graph Model

- [x] ~~bounded degree model~~ general graphs
- [x] input structure: adjacency lists
- [x] error: ~~2-sided~~ 1-sided

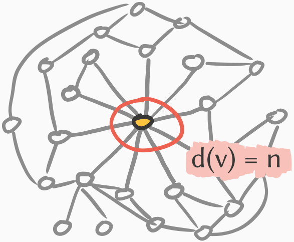
The General Graph Model

- ~~bounded-degree model~~ general graphs
- input structure: adjacency lists
- error: ~~2-sided~~ 1-sided

What can a constant-query property tester do?

The General Graph Model

- ~~bounded-degree model~~ general graphs
- input structure: adjacency lists
- error: ~~2-sided~~ 1-sided

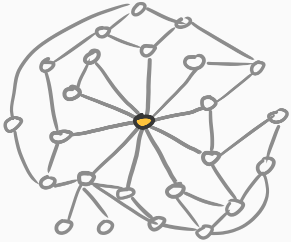


What can a constant-query property tester do?

BFS

The General Graph Model

- ~~bounded-degree model~~ general graphs
- input structure: adjacency lists
- error: ~~2-sided~~ 1-sided



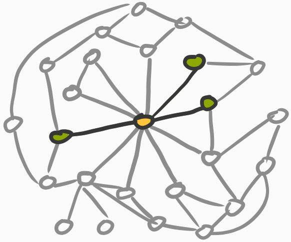
What can a constant-query property tester do?

~~BFS~~

random / subsampling BFS

The General Graph Model

- ~~bounded degree model~~ general graphs
- input structure: adjacency lists
- error: ~~2-sided~~ 1-sided



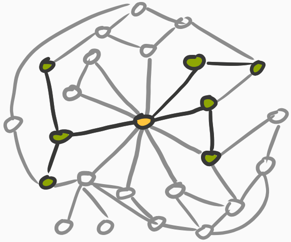
What can a constant-query property tester do?

B~~F~~S

random / subsampling BFS

The General Graph Model

- ~~bounded degree model~~ general graphs
- input structure: adjacency lists
- error: ~~2-sided~~ 1-sided



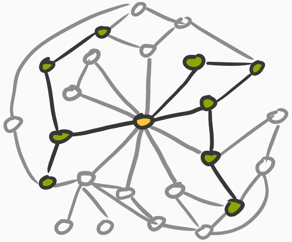
What can a constant-query property tester do?

BFS

random / subsampling BFS

The General Graph Model

- ~~bounded degree model~~ general graphs
- input structure: adjacency lists
- error: ~~2 sided~~ 1-sided

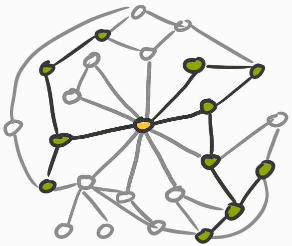


What can a constant-query property tester do?

B~~F~~S

random / subsampling BFS

- ~~bounded degree model~~ general graphs
- input structure: adjacency lists
- error: ~~2-sided~~ 1-sided



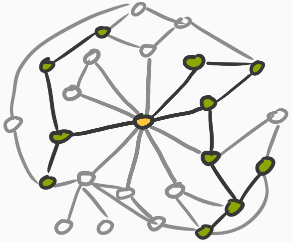
What can a constant-query property tester do?

B~~F~~S

random / subsampling BFS

The General Graph Model

- ~~bounded degree model~~ general graphs
- input structure: adjacency lists
- error: ~~2-sided~~ 1-sided



What can a constant-query property tester do?

~~BFS~~

random / subsampling BFS

Theorem (informal) [with CPS]

Every constant-query property tester for general graphs that queries adjacency lists can be reduced to (multiple) random BFS.

The Streaming Model

- ☒ general graphs
- ☒ input structure: adjacency lists
- ☒ error: 1-sided

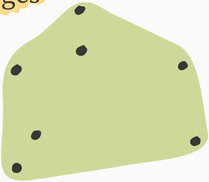
The Streaming Model

- ☒ general graphs
- ☒ input structure: ~~adjacency lists~~ stream of edges
- ☒ error: 1-sided

The Streaming Model

- ☒ general graphs
- ☒ input structure: ~~adjacency lists~~ stream of edges
- ☒ error: 1-sided

planar?

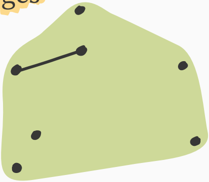


Monemizadeh, Muthukrishnan, Peng, Sohler, ICALP'17; Peng, Sohler, SODA'18

The Streaming Model

- ☒ general graphs
- ☒ input structure: ~~adjacency lists~~ stream of edges
- ☒ error: 1-sided

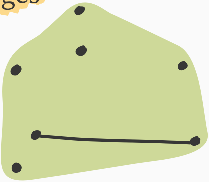
planar?



The Streaming Model

- ☒ general graphs
- ☒ input structure: ~~adjacency lists~~ stream of edges
- ☒ error: 1-sided

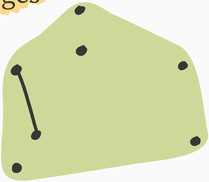
planar?



The Streaming Model

- general graphs
- input structure: ~~adjacency lists~~ stream of edges
- error: 1-sided

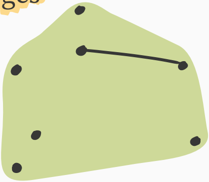
planar?



The Streaming Model

- ☒ general graphs
- ☒ input structure: ~~adjacency lists~~ stream of edges
- ☒ error: 1-sided

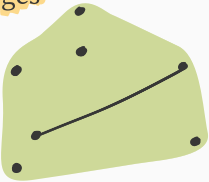
planar?



The Streaming Model

- general graphs
- input structure: ~~adjacency lists~~ stream of edges
- error: 1-sided

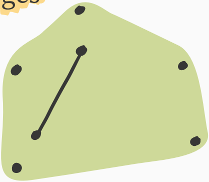
planar?



The Streaming Model

- ☒ general graphs
- ☒ input structure: ~~adjacency lists~~ stream of edges
- ☒ error: 1-sided

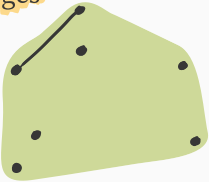
planar?



The Streaming Model

- ☒ general graphs
- ☒ input structure: ~~adjacency lists~~ stream of edges
- ☒ error: 1-sided

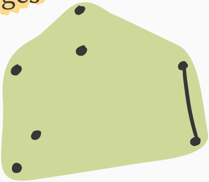
planar?



The Streaming Model

- ☒ general graphs
- ☒ input structure: ~~adjacency lists~~ stream of edges
- ☒ error: 1-sided

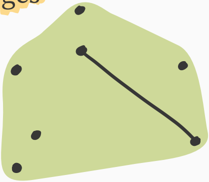
planar?



The Streaming Model

- general graphs
- input structure: ~~adjacency lists~~ stream of edges
- error: 1-sided

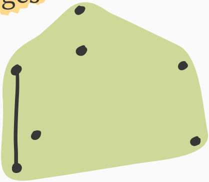
planar?



The Streaming Model

- ☒ general graphs
- ☒ input structure: ~~adjacency lists~~ stream of edges
- ☒ error: 1-sided

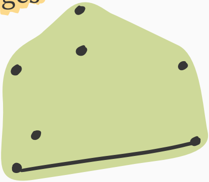
planar?



The Streaming Model

- general graphs
- input structure: ~~adjacency lists~~ stream of edges
- error: 1-sided

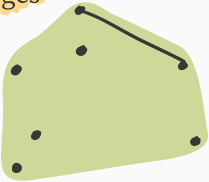
planar?



The Streaming Model

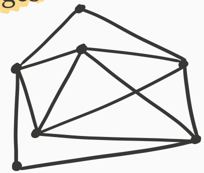
- ☒ general graphs
- ☒ input structure: ~~adjacency lists~~ stream of edges
- ☒ error: 1-sided

planar?



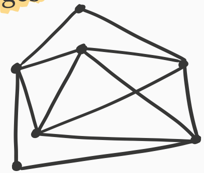
The Streaming Model

- ☒ general graphs
- ☒ input structure: ~~adjacency lists~~ stream of edges
- ☒ error: 1-sided



The Streaming Model

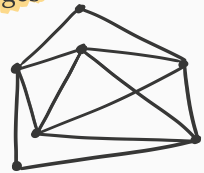
- general graphs
- input structure: ~~adjacency lists~~ stream of edges
- error: 1-sided



objective: $o(n)$ space

The Streaming Model

- ☒ general graphs
- ☒ input structure: ~~adjacency lists~~ stream of edges
- ☒ error: 1-sided

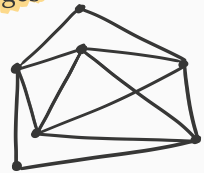


objective: $o(n)$ space

- some problems $\Omega(n)$ in adversarial-order streams

The Streaming Model

- ☒ general graphs
- ☒ input structure: ~~adjacency lists~~ stream of edges
- ☒ error: 1-sided



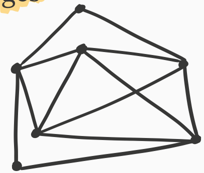
objective: $o(n)$ space

- some problems $\Omega(n)$ in adversarial-order streams
- trivial if number of edges is $O(n)$

Monemizadeh, Muthukrishnan, Peng, Sohler, ICALP'17; Peng, Sohler, SODA'18

The Streaming Model

- [x] general graphs
- [x] input structure: ~~adjacency lists~~ stream of edges
- [x] error: 1-sided



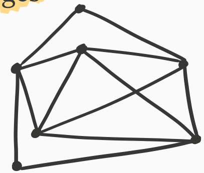
objective: $o(n)$ space

- some problems $\Omega(n)$ in adversarial-order streams
- trivial if number of edges is $O(n)$
- recent model: random-order streams

Monemizadeh, Muthukrishnan, Peng, Sohler, ICALP'17; Peng, Sohler, SODA'18

The Streaming Model

- general graphs
- input structure: ~~adjacency lists~~ stream of edges
- error: 1-sided



objective: $o(n)$ space

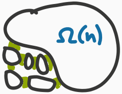
- some problems $\Omega(n)$ in adversarial-order streams
- trivial if number of edges is $O(n)$
- recent model: random-order streams

Theorem (informal) [with CPS]

One-sided error constant-query testers that query adjacency lists admit a $O(\log n)$-space random-order streaming tester.

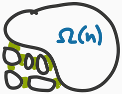
Open Problems

- characterize constant-query properties
    - role of small connected components / cuts



Open Problems

- characterize constant-query properties
  - role of small connected components / cuts
  - relate k-disk vectors and combinatorial properties



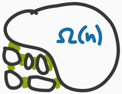
Open Problems

- characterize constant-query properties
    - role of small connected components / cuts

    - relate k-disk vectors and combinatorial properties

- reduce stronger models to streaming setting
    - degree / adjacency matrix queries

- characterize constant-query properties
  - role of small connected components / cuts



  - relate k-disk vectors and combinatorial properties

- reduce stronger models to streaming setting
  - degree / adjacency matrix queries

  - 2-sided error