Some recent advances in logic and graph property testing

Isolde Adler

Joint work with Frederik Harwath, Noleen Köhler, Pan Peng, Joseph Stimpson

FMT Workshop
Les Houches
27.5.2025

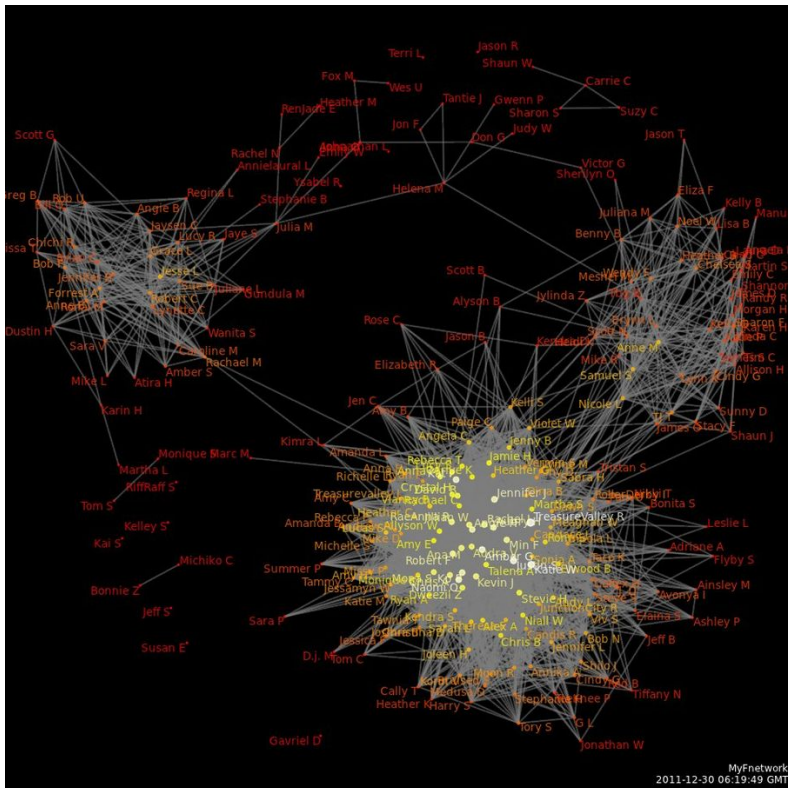
Welcome to the quest...

Contents

Property testing: motivation

‘Efficiency’ when the data set is huge:

Even reading the whole input just once can be too expensive.



Data visualisation of Facebook relationships

Author: Kencf0618, License: Creative Commons Attribution-Share Alike 3.0 Unported

Aim: Sublinear algorithms with local access to the input.

Graphs & logics

- We consider finite, simple, undirected graphs $G = (V(G), E(G))$.
- Class $\mathcal{C}$ of graphs has bounded degree, if there is a constant $d \in \mathbb{N}$ such that all graphs in $\mathcal{C}$ have degree $\leq d$.
- We use n to denote the number of vertices of G.
- All graph classes are closed under isomorphism, and a graph class is sometimes called a property.

FO := First-order logic

MSO := Monadic second-order logic =
first-order logic + quantification over subsets of the universe

Graphs & logics

- We consider finite, simple, undirected graphs $G = (V(G), E(G))$.
- Class $\mathcal{C}$ of graphs has bounded degree, if there is a constant $d \in \mathbb{N}$ such that all graphs in $\mathcal{C}$ have degree $\leq d$.
- We use n to denote the number of vertices of G.
- All graph classes are closed under isomorphism, and a graph class is sometimes called a property.

FO := First-order logic

MSO := Monadic second-order logic =
first-order logic + quantification over subsets of the universe

Graphs & logics

- We consider finite, simple, undirected graphs $G = (V(G), E(G))$.
- Class $\mathcal{C}$ of graphs has bounded degree, if there is a constant $d \in \mathbb{N}$ such that all graphs in $\mathcal{C}$ have degree $\leq d$.
- We use n to denote the number of vertices of G.
- All graph classes are closed under isomorphism, and a graph class is sometimes called a property.

FO := First-order logic

MSO := Monadic second-order logic =
first-order logic + quantification over subsets of the universe

Graphs & logics

- We consider finite, simple, undirected graphs $G = (V(G), E(G))$.
- Class $\mathcal{C}$ of graphs has bounded degree, if there is a constant $d \in \mathbb{N}$ such that all graphs in $\mathcal{C}$ have degree $\leq d$.
- We use n to denote the number of vertices of G.
- All graph classes are closed under isomorphism, and a graph class is sometimes called a property.

FO := First-order logic

MSO := Monadic second-order logic =
first-order logic + quantification over subsets of the universe

Graphs & logics

- We consider finite, simple, undirected graphs $G = (V(G), E(G))$.
- Class $\mathcal{C}$ of graphs has bounded degree, if there is a constant $d \in \mathbb{N}$ such that all graphs in $\mathcal{C}$ have degree $\leq d$.
- We use n to denote the number of vertices of G.
- All graph classes are closed under isomorphism, and a graph class is sometimes called a property.

FO := First-order logic

MSO := Monadic second-order logic =
first-order logic + quantification over subsets of the universe

Algorithms with oracle access

From now on: All graphs have degree $\leq d$.

- Input: the number n of vertices of G, and
- Oracle access to G
  - *Query:* v, for $v \in V(G)$
  - *Answer:* the 1-neighbourhood of vertex v
- The running time $=$ running time w.r.t. n.
- The query complexity $=$ number of oracle queries w.r.t. n.

We use the RAM model of computation.

Algorithms with oracle access

From now on: All graphs have degree $\leq d$.

- Input: the number n of vertices of G, and
- Oracle access to G
  - *Query:* v, for $v \in V(G)$
  - *Answer:* the 1-neighbourhood of vertex v
- The running time = running time w.r.t. n.
- The query complexity = number of oracle queries w.r.t. n.

We use the RAM model of computation.

Algorithms with oracle access

From now on: All graphs have degree $\leq d$.

- Input: the number n of vertices of G, and
- Oracle access to G
  - *Query:* v, for $v \in V(G)$
  - *Answer:* the 1-neighbourhood of vertex v
- The running time = running time w.r.t. n.
- The query complexity = number of oracle queries w.r.t. n.

We use the RAM model of computation.

Algorithms with oracle access

From now on: All graphs have degree $\leq d$.

- Input: the number n of vertices of G, and
- Oracle access to G
  - *Query:* v, for $v \in V(G)$
  - *Answer:* the 1-neighbourhood of vertex v
- The running time = running time w.r.t. n.
- The query complexity = number of oracle queries w.r.t. n.

We use the RAM model of computation.

Algorithms with oracle access

From now on: All graphs have degree $\leq d$.

- Input: the number n of vertices of G, and
- Oracle access to G
  - *Query:* v, for $v \in V(G)$
  - *Answer:* the 1-neighbourhood of vertex v
- The running time $=$ running time w.r.t. n.
- The query complexity $=$ number of oracle queries w.r.t. n.

We use the RAM model of computation.

Algorithms with oracle access

From now on: All graphs have degree $\leq d$.

- Input: the number n of vertices of G, and
- Oracle access to G
  - *Query:* v, for $v \in V(G)$
  - *Answer:* the 1-neighbourhood of vertex v
- The running time $=$ running time w.r.t. n.
- The query complexity $=$ number of oracle queries w.r.t. n.

We use the RAM model of computation.

Algorithms with oracle access

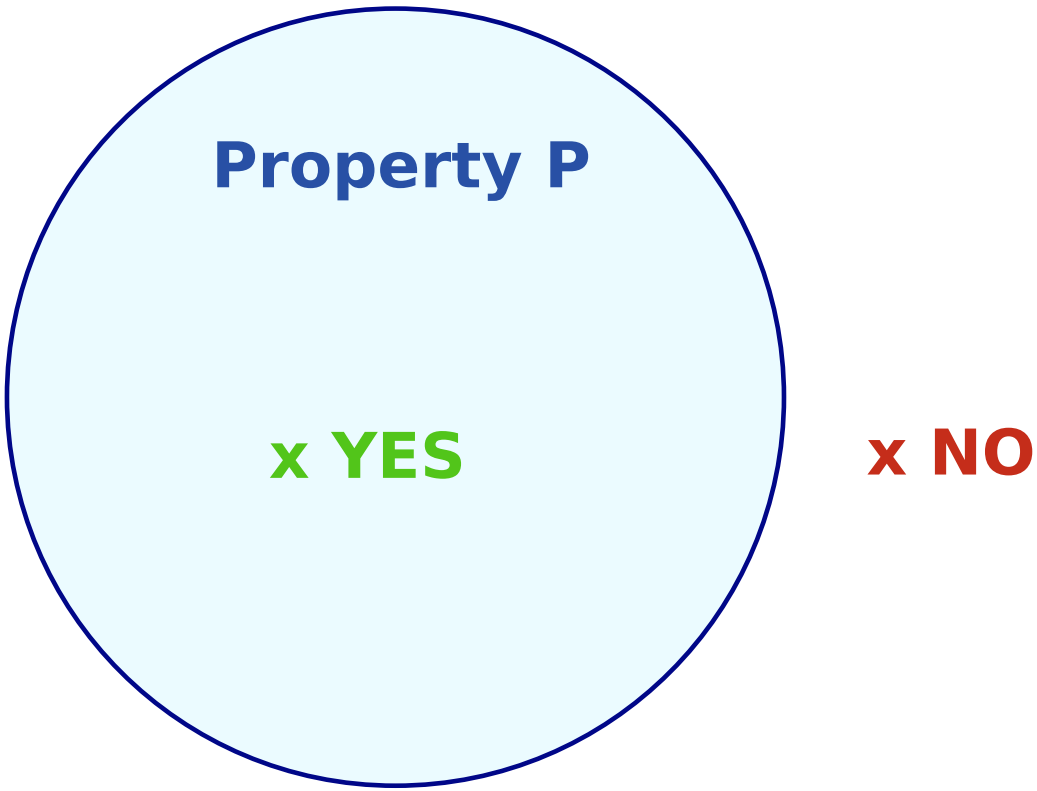
From now on: All graphs have degree $\leq d$.

- Input: the number n of vertices of G, and
- Oracle access to G
  - *Query:* v, for $v \in V(G)$
  - *Answer:* the 1-neighbourhood of vertex v
- The running time $=$ running time w.r.t. n.
- The query complexity $=$ number of oracle queries w.r.t. n.

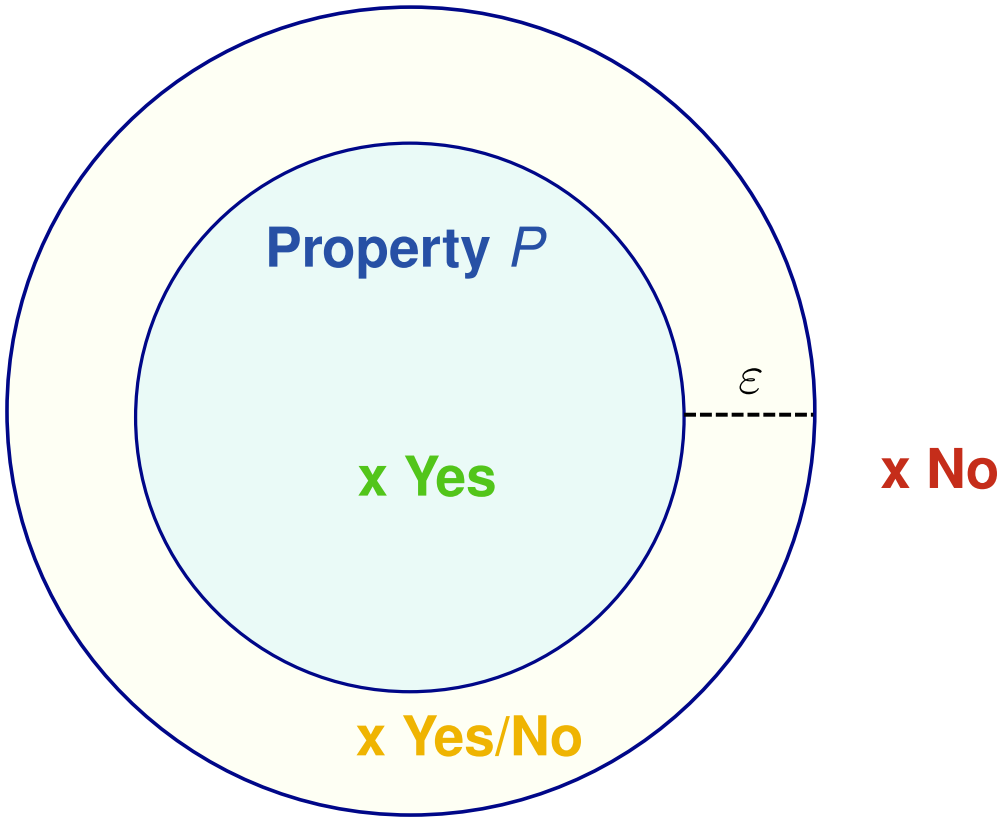
We use the RAM model of computation.

Decision Problems

Decision Problems



Property Testing = relaxation of decision problems



Distance (d-bounded degree model)

- Let $\varepsilon \in [0,1]$.
  Graphs G and H, both on n vertices, are ε-close, if we can make them isomorphic by modifying up to εdn edges of G or H.

  Edge modification = insertion/deletion

- If G, H are not ε-*close*, then they are ε-far.

- A graph G is ε-close to a class $\mathcal{C}$ if G is ε-close to some $H \in \mathcal{C}$.
  Otherwise, G is ε-far from $\mathcal{C}$.

Distance (d-bounded degree model)

- Let $\varepsilon \in [0, 1]$.
  Graphs G and H, both on n vertices, are ε-close, if we can make them isomorphic by modifying up to εdn edges of G or H.

  Edge modification $=$ insertion/deletion

- If G, H are not ε-*close*, then they are ε-far.

- A graph G is ε-close to a class $\mathcal{C}$ if G is ε-close to some $H \in \mathcal{C}$.
  Otherwise, G is ε-far from $\mathcal{C}$.

Distance (d-bounded degree model)

- Let $\varepsilon \in [0, 1]$.
  Graphs G and H, both on n vertices, are ε-close, if we can make them isomorphic by modifying up to εdn edges of G or H.

  Edge modification $=$ insertion/deletion

- If G, H are not ε-*close*, then they are ε-far.

- A graph G is ε-close to a class $\mathcal{C}$ if G is ε-close to some $H \in \mathcal{C}$.
  Otherwise, G is ε-far from $\mathcal{C}$.

Distance (d-bounded degree model)

- Let $\varepsilon \in [0, 1]$.
  Graphs G and H, both on n vertices, are ε-close, if we can make them isomorphic by modifying up to εdn edges of G or H.

  Edge modification $=$ insertion/deletion

- If G, H are not ε-*close*, then they are ε-far.

- A graph G is ε-close to a class $\mathcal{C}$ if G is ε-close to some $H \in \mathcal{C}$.
  Otherwise, G is ε-far from $\mathcal{C}$.

Testability

Let $\mathcal{P}$ be a property.
An ε-tester for $\mathcal{P}$ is a probabilistic algorithm that, given oracle access to G and given $n := |V(G)|$ as input, does the following:

1. If $G \in \mathcal{P}$, then the tester accepts with probability $\geq \frac{2}{3}$,
2. if G is ε-far from $\mathcal{P}$, then the tester rejects with probability $\geq \frac{2}{3}$.

$\mathcal{P}$ is uniformly testable, if for each ε there is an ε-tester for $\mathcal{P}$ with constant query complexity.

$\mathcal{P}$ is (non-uniformly) testable, if for each ε and each n, there is a tester for $\mathcal{P}_n := \{G \in \mathcal{P} : |V(G)| = n\}$ with constant query complexity (independent of n).

Testability

Let $\mathcal{P}$ be a property.
An ε-tester for $\mathcal{P}$ is a probabilistic algorithm that, given oracle access to G and given $n := |V(G)|$ as input, does the following:

1. If $G \in \mathcal{P}$, then the tester accepts with probability $\geq \frac{2}{3}$,
2. if G is ε-far from $\mathcal{P}$, then the tester rejects with probability $\geq \frac{2}{3}$.

$\mathcal{P}$ is uniformly testable, if for each ε there is an ε-tester for $\mathcal{P}$ with constant query complexity.

$\mathcal{P}$ is (non-uniformly) testable, if for each ε and each n, there is a tester for $\mathcal{P}_n := \{G \in \mathcal{P} : |V(G)| = n\}$ with constant query complexity (independent of n).

Testability

Let $\mathcal{P}$ be a property.
An ε-tester for $\mathcal{P}$ is a probabilistic algorithm that, given oracle access to G and given $n := |V(G)|$ as input, does the following:

1. If $G \in \mathcal{P}$, then the tester accepts with probability $\geq \frac{2}{3}$,

2. if G is ε-far from $\mathcal{P}$, then the tester rejects with probability $\geq \frac{2}{3}$.

$\mathcal{P}$ is uniformly testable, if for each ε there is an ε-tester for $\mathcal{P}$ with constant query complexity.

$\mathcal{P}$ is (non-uniformly) testable, if for each ε and each n, there is a tester for $\mathcal{P}_n := \{G \in \mathcal{P} : |V(G)| = n\}$ with constant query complexity (independent of n).

Testability

Let $\mathcal{P}$ be a property.
An ε-tester for $\mathcal{P}$ is a probabilistic algorithm that, given oracle access to G and given $n := |V(G)|$ as input, does the following:

1. If $G \in \mathcal{P}$, then the tester accepts with probability $\geq \frac{2}{3}$,

2. if G is ε-far from $\mathcal{P}$, then the tester rejects with probability $\geq \frac{2}{3}$.

$\mathcal{P}$ is uniformly testable, if for each ε there is an ε-tester for $\mathcal{P}$ with constant query complexity.

$\mathcal{P}$ is (non-uniformly) testable, if for each ε and each n, there is a tester for $\mathcal{P}_n := \{G \in \mathcal{P} : |V(G)| = n\}$ with constant query complexity (independent of n).

Testability

Let $\mathcal{P}$ be a property.
An ε-tester for $\mathcal{P}$ is a probabilistic algorithm that, given oracle access to G and given $n := |V(G)|$ as input, does the following:

1. If $G \in \mathcal{P}$, then the tester accepts with probability $\geq \frac{2}{3}$,

2. if G is ε-far from $\mathcal{P}$, then the tester rejects with probability $\geq \frac{2}{3}$.

$\mathcal{P}$ is uniformly testable, if for each ε there is an ε-tester for $\mathcal{P}$ with constant query complexity.

$\mathcal{P}$ is (non-uniformly) testable, if for each ε and each n, there is a tester for $\mathcal{P}_n := \{G \in \mathcal{P} : |V(G)| = n\}$ with constant query complexity (independent of n).

Examples

On bounded degree graphs:

Uniformly testable with constant query complexity and running time:

- k-(edge-)connectivity
- being Eulerian
- subgraph-freeness
- induced subgraph-freeness

Not testable with constant query complexity:

- Bipartiteness, colourability
- Expander graphs
- Hamiltonicity

[Goldreich and Ron 2002; Yoshida and Ito 2010]

Examples

On bounded degree graphs:

Uniformly testable with constant query complexity and running time:

- k-(edge-)connectivity
- being Eulerian
- subgraph-freeness
- induced subgraph-freeness

Not testable with constant query complexity:

- Bipartiteness, colourability
- Expander graphs
- Hamiltonicity

[Goldreich and Ron 2002; Yoshida and Ito 2010]

Examples

On bounded degree graphs:

Uniformly testable with constant query complexity and running time:

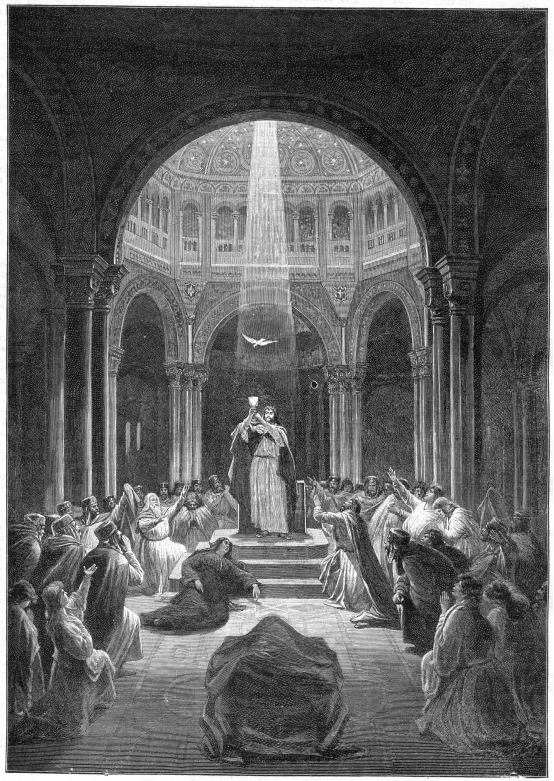
- k-(edge-)connectivity
- being Eulerian
- subgraph-freeness
- induced subgraph-freeness

Not testable with constant query complexity:

- Bipartiteness, colourability
- Expander graphs
- Hamiltonicity

[Goldreich and Ron 2002; Yoshida and Ito 2010]

Bounded-degree model: Holy Grail I



Open problem (Holy Grail I)

Characterise the properties that are (non-uniformly) testable with constant query complexity.

Holy Grail I: what is known

Theorem (Newman and Sohler, 2013 (Benjamini, Shramm, Shapira, Elek))

Every hyperfinite property is non-uniformly testable.

Theorem (Ito, Khoury, Newman 2020)

Characterisation of 1-sided error non-uniformly testable monotone/hereditary graph properties.

Theorem (A., Köhler, Peng 2021)

There is an FO*-property that is not testable on graphs of degree* ≤ 3.

Holy Grail I: what is known

Theorem (Newman and Sohler, 2013 (Benjamini, Shramm, Shapira, Elek))
Every hyperfinite property is non-uniformly testable.

Theorem (Ito, Khoury, Newman 2020)
Characterisation of 1-sided error non-uniformly testable monotone/hereditary graph properties.

Theorem (A., Köhler, Peng 2021)
There is an FO*-property that is not testable on graphs of degree* ≤ 3.

Holy Grail I: what is known

Theorem (Newman and Sohler, 2013 (Benjamini, Shramm, Shapira, Elek))

Every hyperfinite property is non-uniformly testable.

Theorem (Ito, Khoury, Newman 2020)

Characterisation of 1-sided error non-uniformly testable monotone/hereditary graph properties.

Theorem (A., Köhler, Peng 2021)

There is an FO*-property that is not testable on graphs of degree* ≤ 3.

Holy Grail I: what is known

Theorem (Newman and Sohler, 2013 (Benjamini, Shramm, Shapira, Elek))

Every hyperfinite property is non-uniformly testable.

Theorem (Ito, Khoury, Newman 2020)

Characterisation of 1-sided error non-uniformly testable monotone/hereditary graph properties.

Theorem (A., Köhler, Peng 2021)

There is an FO*-property that is not testable on graphs of degree* ≤ 3.

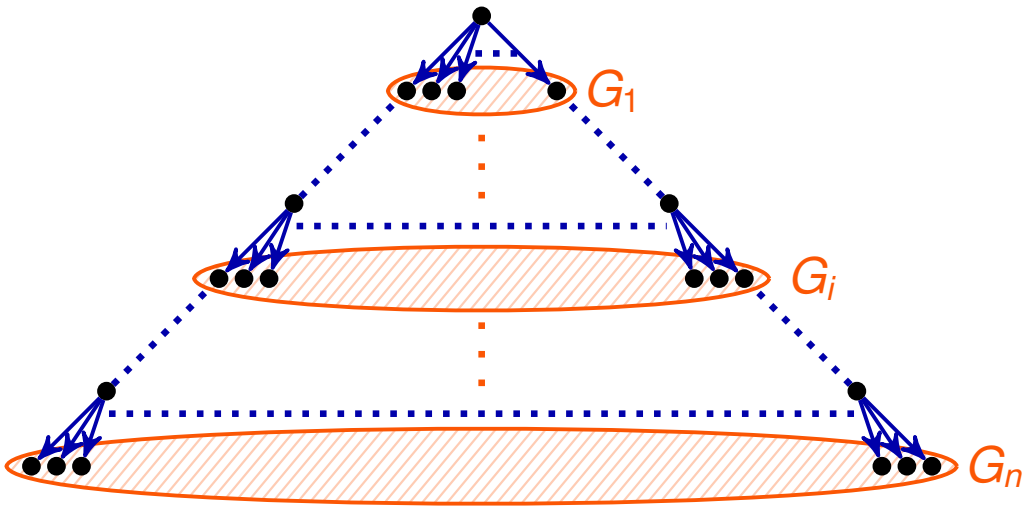
Non-testable property (proof idea)

Find an FO-formula φ whose models are expander graphs.

$(G_i)_{i\in\mathbb{N}}$: zig-zag construction [Rheingold, Vadhan, Wigdersen, Ann. of Math., 2002]

Non-testable property (proof idea)

Find an FO-formula φ whose models are expander graphs.



$(G_i)_{i\in\mathbb{N}}$: zig-zag construction [Rheingold, Vadhan, Wigdersen, Ann. of Math., 2002]

PT on hyperfinite graphs

Theorem (Newman and Sohler, 2013 (Benjamini, Shramm, Shapira, Elek))
Every hyperfinite property is non-uniformly testable.

Hyperfinite graph classes include
- bounded tree-width graphs,
- graphs excluding a fixed minor.

Observation
The theorem does not say anything about the running time.
$\exists$ undecidable properties (of edgeless graphs) that are testable with constant query complexity by the theorem.

Typically: Any natural property that is testable with constant query complexity is also testable with constant running time.

PT on hyperfinite graphs

Theorem (Newman and Sohler, 2013 (Benjamini, Shramm, Shapira, Elek))
Every hyperfinite property is non-uniformly testable.

Hyperfinite graph classes include
- bounded tree-width graphs,
- graphs excluding a fixed minor.

Observation
The theorem does not say anything about the running time.
$\exists$ undecidable properties (of edgeless graphs) that are testable with constant query complexity by the theorem.

Typically: Any natural property that is testable with constant query complexity is also testable with constant running time.

PT on hyperfinite graphs

Theorem (Newman and Sohler, 2013 (Benjamini, Shramm, Shapira, Elek))
Every hyperfinite property is non-uniformly testable.

Hyperfinite graph classes include
- bounded tree-width graphs,
- graphs excluding a fixed minor.

Observation
The theorem does not say anything about the running time.
$\exists$ undecidable properties (of edgeless graphs) that are testable with constant query complexity by the theorem.

Typically: Any natural property that is testable with constant query complexity is also testable with constant running time.

PT on hyperfinite graphs

Theorem (Newman and Sohler, 2013 (Benjamini, Shramm, Shapira, Elek))
Every hyperfinite property is non-uniformly testable.

Hyperfinite graph classes include
- bounded tree-width graphs,
- graphs excluding a fixed minor.

Observation
The theorem does not say anything about the running time.
$\exists$ *undecidable properties (of edgeless graphs) that are testable with constant query complexity by the theorem.*

Typically: Any natural property that is testable with constant query complexity is also testable with constant running time.

PT on hyperfinite graphs

Theorem (Newman and Sohler, 2013 (Benjamini, Shramm, Shapira, Elek))
Every hyperfinite property is non-uniformly testable.

Hyperfinite graph classes include
- bounded tree-width graphs,
- graphs excluding a fixed minor.

Observation
The theorem does not say anything about the running time.
$\exists$ *undecidable properties (of edgeless graphs) that are testable with constant query complexity by the theorem.*

Typically: Any natural property that is testable with constant query complexity is also testable with constant running time.

Bounded-degree model: Holy Grail II

Open problem (Holy Grail II)

Characterise the properties that are uniformly testable with constant running time.

Holy Grail II: what is known

Theorem (Benjamini, Schramm, Shapira 2008)

Every hyperfinite, monotone property is testable with constant running time.

Theorem (Czumaj, Shapira, Sohler 2009)

Every hyperfinite, hereditary property is testable with constant running time.

Holy Grail II: what is known

Theorem (Benjamini, Schramm, Shapira 2008)

Every hyperfinite, monotone property is testable with constant running time.

Theorem (Czumaj, Shapira, Sohler 2009)

Every hyperfinite, hereditary property is testable with constant running time.

Contents

CMSO *on bounded tw with sublinear running time*

Theorem (A., Harwath 2018)

Let $\mathcal{C}_d^t$ be the class of all t-bounded tree-width graphs of degree $\leq d$.

Every MSO-definable property $\mathcal{P} \subseteq \mathcal{C}_d^t$ is uniformly testable with constant query complexity and polylogarithmic running time.

- Open: can it be improved to constant running time?

CMSO *on bounded tw with sublinear running time*

Theorem (A., Harwath 2018)

Let $\mathcal{C}_d^t$ be the class of all t-bounded tree-width graphs of degree $\leq d$.

Every MSO-definable property $\mathcal{P} \subseteq \mathcal{C}_d^t$ is uniformly testable with constant query complexity and polylogarithmic running time.

- Open: can it be improved to constant running time?

Example: FO on degree ≤ 1

We consider graphs of degree ≤ 1.

Example

Let

$$\varphi = \neg\exists x_1 x_2 x_3 \Big(\bigwedge_{i \neq j} x_i \neq x_j \wedge \forall y (\neg E x_1 y) \wedge E x_2 x_3 \Big).$$

The only graphs satisfying φ are of the form

or

Let $G =$

Let $\varepsilon = 1/3$. Then G is ε-far from satisfying φ. However, w.h.p. we will not see this by sampling a constant number of vertices from G.

But φ is still testable: check if n is even or odd.

Example: FO on degree ≤ 1

We consider graphs of degree ≤ 1.

Example

Let

$$\varphi = \neg\exists x_1 x_2 x_3 \Big(\bigwedge_{i \neq j} x_i \neq x_j \wedge \forall y (\neg E x_1 y) \wedge E x_2 x_3 \Big).$$

The only graphs satisfying φ are of the form

or

Let $G =$

Let $\varepsilon = 1/3$. Then G is ε-far from satisfying φ. However, w.h.p. we will not see this by sampling a constant number of vertices from G.

But φ is still testable: check if n is even or odd.

Example: FO on degree ≤ 1

We consider graphs of degree ≤ 1.

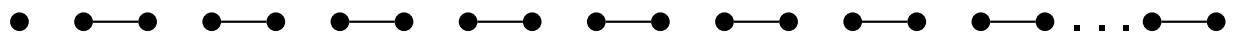
Example

Let

$$\varphi = \neg\exists x_1 x_2 x_3 \Big(\bigwedge_{i \neq j} x_i \neq x_j \wedge \forall y (\neg E x_1 y) \wedge E x_2 x_3 \Big).$$

The only graphs satisfying φ are of the form

or

Let $G =$ 

Let $\varepsilon = 1/3$. Then G is ε-far from satisfying φ. However, w.h.p. we will not see this by sampling a constant number of vertices from G.

But φ is still testable: check if n is even or odd.

Example: FO on degree ≤ 1

We consider graphs of degree ≤ 1.

Example

Let

$$\varphi = \neg\exists x_1 x_2 x_3 \Big(\bigwedge_{i \neq j} x_i \neq x_j \wedge \forall y (\neg E x_1 y) \wedge E x_2 x_3 \Big).$$

The only graphs satisfying φ are of the form

or

Let $G =$ 

Let $\varepsilon = 1/3$. Then G is ε-far from satisfying φ. However, w.h.p. we will not see this by sampling a constant number of vertices from G.

But φ is still testable: check if n is even or odd.

Example: FO on degree ≤ 1

We consider graphs of degree ≤ 1.

Example

Let

$$\varphi = \neg\exists x_1 x_2 x_3 \Big(\bigwedge_{i \neq j} x_i \neq x_j \wedge \forall y (\neg E x_1 y) \wedge E x_2 x_3 \Big).$$

The only graphs satisfying φ are of the form

or

Let $G =$ 

Let $\varepsilon = 1/3$. Then G is ε-far from satisfying φ. However, w.h.p. we will not see this by sampling a constant number of vertices from G.

But φ is still testable: check if n is even or odd.

Example: FO on degree ≤ 1

We consider graphs of degree ≤ 1.

Example

Let

$$\varphi = \neg\exists x_1 x_2 x_3 \Big(\bigwedge_{i \neq j} x_i \neq x_j \wedge \forall y (\neg E x_1 y) \wedge E x_2 x_3 \Big).$$

The only graphs satisfying φ are of the form

or

Let $G =$ 

Let $\varepsilon = 1/3$. Then G is ε-far from satisfying φ. However, w.h.p. we will not see this by sampling a constant number of vertices from G.

But φ is still testable: check if n is even or odd.

FO *on finitary graphs*

Let $b \in \mathbb{N}$. Class $\mathcal{C}$ is b-finitary, if every component of a graph in $\mathcal{C}$ has $\leq b$ vertices.

Theorem (A., Stimpson 2025+)

Let $\mathcal{C}$ be b-finitary. Then every FO*-definable property is testable on $\mathcal{C}$ with constant running time.*

Let $C_1, C_2, \ldots, C_t$ be an enumeration of all isomorphism types of components of size $\leq b$.

Definition

Every b-finitary graph G has a component histogram vector $\bar{v}_G$ where

$$\bar{v}_G[i] := \#\text{ occurrences of } C_i \text{ in } G.$$

FO *on finitary graphs*

Let $b \in \mathbb{N}$. Class $\mathcal{C}$ is b-finitary, if every component of a graph in $\mathcal{C}$ has $\leq b$ vertices.

Theorem (A., Stimpson 2025+)

Let $\mathcal{C}$ be b-finitary. Then every FO*-definable property is testable on $\mathcal{C}$ with constant running time.*

Let $C_1, C_2, \ldots, C_t$ be an enumeration of all isomorphism types of components of size $\leq b$.

Definition

Every b-finitary graph G has a component histogram vector $\bar{v}_G$ where

$$\bar{v}_G[i] := \#\text{ occurrences of } C_i \text{ in } G.$$

FO *on finitary graphs*

Let $b \in \mathbb{N}$. Class $\mathcal{C}$ is b-finitary, if every component of a graph in $\mathcal{C}$ has $\leq b$ vertices.

Theorem (A., Stimpson 2025+)

Let $\mathcal{C}$ be b-finitary. Then every FO*-definable property is testable on $\mathcal{C}$ with constant running time.*

Let $C_1, C_2, \ldots, C_t$ be an enumeration of all isomorphism types of components of size $\leq b$.

Definition

Every b-finitary graph G has a component histogram vector $\bar{v}_G$ where

$$\bar{v}_G[i] := \#\text{ occurrences of } C_i \text{ in } G.$$

FO *on finitary graphs*

Let $b \in \mathbb{N}$. Class $\mathcal{C}$ is *b*-finitary, if every component of a graph in $\mathcal{C}$ has $\leq b$ vertices.

Theorem (A., Stimpson 2025+)

Let $\mathcal{C}$ be b-finitary. Then every FO*-definable property is testable on $\mathcal{C}$ with constant running time.*

Let $C_1, C_2, \ldots, C_t$ be an enumeration of all isomorphism types of components of size $\leq b$.

Definition

Every b-finitary graph G has a component histogram vector $\bar{v}_G$ where

$$\bar{v}_G[i] := \#\ \textit{occurrences of } C_i \textit{ in } G.$$

Capped component histograms

Let $C_1, C_2, \ldots, C_t$ be an enumeration of all isomorphism types of components of size $\leq b$.

Definition

For $k \in \mathbb{N}$, a k-capped component histogram vector is a vector $\bar{v}$ indexed by the C_i where every entry $\bar{v}[i]$ is either

- *a number $m \in \mathbb{N}$ with $0 \leq m < k$ ('rare components'), or*
- *'$\geq k$' ('frequent components').*

Example

Let $k = 10$, $\bar{v} = (0, 2, 0, \geq 10, 0, 0, 3, \geq 10, 0, 1, 0, 6, 0, \geq 10, \geq 10, 0, \ldots, 0)$.

Lemma (Corollary of Hanf's Theorem)

Let G be a b-finitary graph G and $\varphi \in$ FO. Checking whether $G \models \varphi$ amounts to checking whether $\bar{v}_G$ satisfies one of finitely many capped component histogram vectors.

Capped component histograms

Let $C_1, C_2, \ldots, C_t$ be an enumeration of all isomorphism types of components of size $\leq b$.

Definition

For $k \in \mathbb{N}$, a k-capped component histogram vector is a vector $\bar{v}$ indexed by the C_i where every entry $\bar{v}[i]$ is either

- *a number $m \in \mathbb{N}$ with $0 \leq m < k$ ('rare components'), or*
- *'$\geq k$' ('frequent components').*

Example

Let $k = 10$, $\bar{v} = (0, 2, 0, \geq 10, 0, 0, 3, \geq 10, 0, 1, 0, 6, 0, \geq 10, \geq 10, 0, \ldots, 0)$.

Lemma (Corollary of Hanf's Theorem)

Let G be a b-finitary graph G and $\varphi \in$ FO. Checking whether $G \models \varphi$ amounts to checking whether $\bar{v}_G$ satisfies one of finitely many capped component histogram vectors.

Capped component histograms

Let $C_1, C_2, \ldots, C_t$ be an enumeration of all isomorphism types of components of size $\leq b$.

Definition

For $k \in \mathbb{N}$, a k-capped component histogram vector is a vector $\bar{v}$ indexed by the C_i where every entry $\bar{v}[i]$ is either

- *a number $m \in \mathbb{N}$ with $0 \leq m < k$ ('rare components'), or*
- *'$\geq k$' ('frequent components').*

Example

Let $k = 10$, $\bar{v} = (0, 2, 0, \geq 10, 0, 0, 3, \geq 10, 0, 1, 0, 6, 0, \geq 10, \geq 10, 0, \ldots, 0)$.

Lemma (Corollary of Hanf's Theorem)

Let G be a b-finitary graph G and $\varphi \in$ FO. Checking whether $G \models \varphi$ amounts to checking whether $\bar{v}_G$ satisfies one of finitely many capped component histogram vectors.

Capped component histograms

Let $C_1, C_2, \ldots, C_t$ be an enumeration of all isomorphism types of components of size $\leq b$.

Definition

For $k \in \mathbb{N}$, a k-capped component histogram vector is a vector $\bar{v}$ indexed by the C_i where every entry $\bar{v}[i]$ is either

- *a number $m \in \mathbb{N}$ with $0 \leq m < k$ ('rare components'), or*
- *'$\geq k$' ('frequent components').*

Example

Let $k = 10$, $\bar{v} = (0, 2, 0, \geq 10, 0, 0, 3, \geq 10, 0, 1, 0, 6, 0, \geq 10, \geq 10, 0, \ldots, 0)$.

Lemma (Corollary of Hanf's Theorem)

Let G be a b-finitary graph G and $\varphi \in \mathrm{FO}$. Checking whether $G \models \varphi$ amounts to checking whether $\bar{v}_G$ satisfies one of finitely many capped component histogram vectors.

FO *on finitary graphs*

Theorem (A., Stimpson 2025+)

Let $\mathcal{C}$ be b-finitary. Then every* FO*-definable property φ is testable on $\mathcal{C}$ with constant running time.

Proof.

Assume φ is given by a capped component histogram $\bar{v}$.
Let $n_0 := \sum_{C \text{ rare component}} \bar{v}[C] \cdot |C|$ and
$g := \gcd\{|C| : C \text{ frequent component}\}$.
Fix ε. Given oracle access to G and $n = |V(G)|$:

- Sample $s = s(\varepsilon)$ vertices from G and explore their b-neighbourhood & obtain good approximation $\bar{v}$ of G's b-neighbourhood distribution.
- If a rare component is found, reject.
- Otherwise, if $g|(n - n_0)$, accept, else reject.

Correctness: Assume n is large and $\bar{v}$ is good.
Show: If no rare comp/s found and $g|(n - n_0)$, then G can be turned into a graph satisfying φ by a small number of modifications ('Frobenius coin problem'; Newman-Sohler). □

FO *on finitary graphs*

Theorem (A., Stimpson 2025+)

Let $\mathcal{C}$ be b-finitary. Then every FO*-definable property φ is testable on $\mathcal{C}$ with constant running time.*

Proof.

Assume φ is given by a capped component histogram $\bar{v}$.
Let $n_0 := \sum_{C \text{ rare component}} \bar{v}[C] \cdot |C|$ and
$g := \gcd\{|C| : C \text{ frequent component}\}$.
Fix ε. Given oracle access to G and $n = |V(G)|$:

- Sample $s = s(\varepsilon)$ vertices from G and explore their b-neighbourhood & obtain good approximation $\bar{v}$ of G's b-neighbourhood distribution.
- If a rare component is found, reject.
- Otherwise, if $g|(n - n_0)$, accept, else reject.

Correctness: Assume n is large and $\bar{v}$ is good.
Show: If no rare comp/s found and $g|(n - n_0)$, then G can be turned into a graph satisfying φ by a small number of modifications ('Frobenius coin problem'; Newman-Sohler). □

FO *on finitary graphs*

Theorem (A., Stimpson 2025+)

Let $\mathcal{C}$ be b-finitary. Then every FO*-definable property φ is testable on $\mathcal{C}$ with constant running time.*

Proof.

Assume φ is given by a capped component histogram $\bar{v}$.
Let $n_0 := \sum_{C \text{ rare component}} \bar{v}[C] \cdot |C|$ and
$g := \gcd\{|C| : C \text{ frequent component}\}$.
Fix ε. Given oracle access to G and $n = |V(G)|$:

- Sample $s = s(\varepsilon)$ vertices from G and explore their b-neighbourhood & obtain good approximation $\bar{v}$ of G's b-neighbourhood distribution.
- If a rare component is found, reject.
- Otherwise, if $g|(n - n_0)$, accept, else reject.

Correctness: Assume n is large and $\bar{v}$ is good.
Show: If no rare comp/s found and $g|(n - n_0)$, then G can be turned into a graph satisfying φ by a small number of modifications ('Frobenius coin problem'; Newman-Sohler). □

FO *on finitary graphs*

Theorem (A., Stimpson 2025+)

Let $\mathcal{C}$ be b-finitary. Then every FO*-definable property φ is testable on $\mathcal{C}$ with constant running time.*

Proof.

Assume φ is given by a capped component histogram $\bar{v}$.
Let $n_0 := \sum_{C \text{ rare component}} \bar{v}[C] \cdot |C|$ and
$g := \gcd\{|C| : C \text{ frequent component}\}$.
Fix ε. Given oracle access to G and $n = |V(G)|$:

- Sample $s = s(\varepsilon)$ vertices from G and explore their b-neighbourhood & obtain good approximation $\bar{v}$ of G's b-neighbourhood distribution.
- If a rare component is found, reject.
- Otherwise, if $g|(n - n_0)$, accept, else reject.

Correctness: Assume n is large and $\bar{v}$ is good.
Show: If no rare comp/s found and $g|(n - n_0)$, then G can be turned into a graph satisfying φ by a small number of modifications ('Frobenius coin problem'; Newman-Sohler). □

FO *on finitary graphs*

Theorem (A., Stimpson 2025+)

Let $\mathcal{C}$ be b-finitary. Then every FO*-definable property φ is testable on $\mathcal{C}$ with constant running time.*

Proof.

Assume φ is given by a capped component histogram $\bar{v}$.
Let $n_0 := \sum_{C \text{ rare component}} \bar{v}[C] \cdot |C|$ and
$g := \gcd\{|C| : C \text{ frequent component}\}$.
Fix ε. Given oracle access to G and $n = |V(G)|$:

- Sample $s = s(\varepsilon)$ vertices from G and explore their b-neighbourhood & obtain good approximation $\bar{v}$ of G's b-neighbourhood distribution.
- If a rare component is found, reject.
- Otherwise, if $g|(n - n_0)$, accept, else reject.

Correctness: Assume n is large and $\bar{v}$ is good.
Show: If no rare comp/s found and $g|(n - n_0)$, then G can be turned into a graph satisfying φ by a small number of modifications ('Frobenius coin problem'; Newman-Sohler). □

FO *on finitary graphs*

Theorem (A., Stimpson 2025+)

Let $\mathcal{C}$ be b-finitary. Then every FO*-definable property φ is testable on $\mathcal{C}$ with constant running time.*

Proof.

Assume φ is given by a capped component histogram $\bar{v}$.
Let $n_0 := \sum_{C \text{ rare component}} \bar{v}[C] \cdot |C|$ and
$g := \gcd\{|C| : C \text{ frequent component}\}$.
Fix ε. Given oracle access to G and $n = |V(G)|$:

- Sample $s = s(\varepsilon)$ vertices from G and explore their b-neighbourhood & obtain good approximation $\bar{v}$ of G's b-neighbourhood distribution.
- If a rare component is found, reject.
- Otherwise, if $g|(n - n_0)$, accept, else reject.

Correctness: Assume n is large and $\bar{v}$ is good.
Show: If no rare comp/s found and $g|(n - n_0)$, then G can be turned into a graph satisfying φ by a small number of modifications ('Frobenius coin problem'; Newman-Sohler). □

FO *on finitary graphs*

Theorem (A., Stimpson 2025+)

Let $\mathcal{C}$ be b-finitary. Then every FO*-definable property φ is testable on $\mathcal{C}$ with constant running time.*

Proof.

Assume φ is given by a capped component histogram $\bar{v}$.
Let $n_0 := \sum_{C \text{ rare component}} \bar{v}[C] \cdot |C|$ and
$g := \gcd\{|C| : C \text{ frequent component}\}$.
Fix ε. Given oracle access to G and $n = |V(G)|$:

- Sample $s = s(\varepsilon)$ vertices from G and explore their b-neighbourhood & obtain good approximation $\bar{v}$ of G's b-neighbourhood distribution.
- If a rare component is found, reject.
- Otherwise, if $g|(n - n_0)$, accept, else reject.

Correctness: Assume n is large and $\bar{v}$ is good.
Show: If no rare comp/s found and $g|(n - n_0)$, then G can be turned into a graph satisfying φ by a small number of modifications ('Frobenius coin problem'; Newman-Sohler). □

FO *on finitary graphs*

Theorem (A., Stimpson 2025+)

Let $\mathcal{C}$ be b-finitary. Then every FO*-definable property φ is testable on $\mathcal{C}$ with constant running time.*

Proof.

Assume φ is given by a capped component histogram $\bar{v}$.
Let $n_0 := \sum_{C \text{ rare component}} \bar{v}[C] \cdot |C|$ and
$g := \gcd\{|C| : C \text{ frequent component}\}$.
Fix ε. Given oracle access to G and $n = |V(G)|$:

- Sample $s = s(\varepsilon)$ vertices from G and explore their b-neighbourhood & obtain good approximation $\bar{v}$ of G's b-neighbourhood distribution.
- If a rare component is found, reject.
- Otherwise, if $g|(n - n_0)$, accept, else reject.

Correctness: Assume n is large and $\bar{v}$ is good.
Show: If no rare comp/s found and $g|(n - n_0)$, then G can be turned into a graph satisfying φ by a small number of modifications ('Frobenius coin problem'; Newman-Sohler). □

FO *on finitary graphs*

Theorem (A., Stimpson 2025+)

Let $\mathcal{C}$ be b-finitary. Then every FO*-definable property φ is testable on $\mathcal{C}$ with constant running time.*

Proof.

Assume φ is given by a capped component histogram $\bar{v}$.
Let $n_0 := \sum_{C \text{ rare component}} \bar{v}[C] \cdot |C|$ and
$g := \gcd\{|C| : C \text{ frequent component}\}$.
Fix ε. Given oracle access to G and $n = |V(G)|$:

- Sample $s = s(\varepsilon)$ vertices from G and explore their b-neighbourhood & obtain good approximation $\bar{v}$ of G's b-neighbourhood distribution.
- If a rare component is found, reject.
- Otherwise, if $g|(n - n_0)$, accept, else reject.

Correctness: Assume n is large and $\bar{v}$ is good.
Show: If no rare comp/s found and $g|(n - n_0)$, then G can be turned into a graph satisfying φ by a small number of modifications ('Frobenius coin problem'; Newman-Sohler). □

Outlook

- Generalise approach to larger classes and logics
- Testing FO in the bounded-degree model: what is the query-complexity?

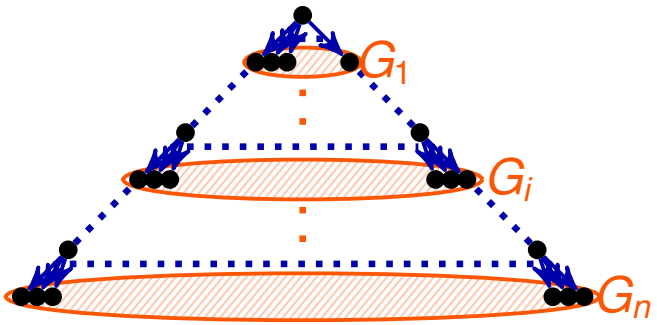
Theorem (Goldreich 2024)

Testing our property has query complexity $\Omega(\log\log\log\log n)$.

- Find the holy grails.

Outlook

- Generalise approach to larger classes and logics
- Testing FO in the bounded-degree model: what is the query-complexity?



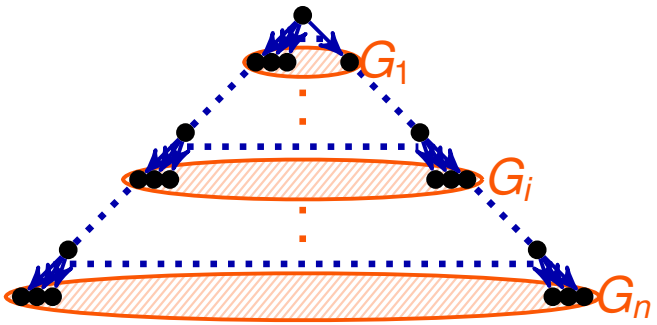
Theorem (Goldreich 2024)

Testing our property has query complexity $\Omega(\log\log\log\log n)$.

- Find the holy grails.

Outlook

- Generalise approach to larger classes and logics
- Testing FO in the bounded-degree model: what is the query-complexity?



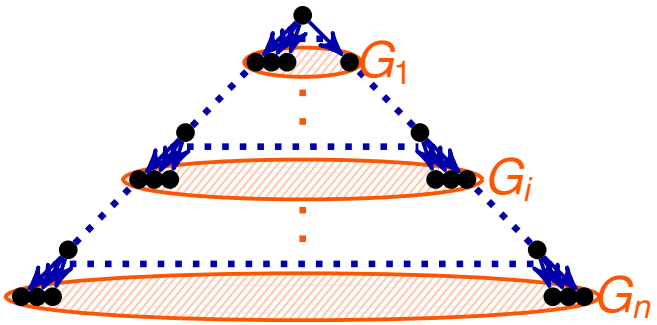
Theorem (Goldreich 2024)

Testing our property has query complexity $\Omega(\log\log\log\log n)$.

- Find the holy grails.

Outlook

- Generalise approach to larger classes and logics
- Testing FO in the bounded-degree model: what is the query-complexity?



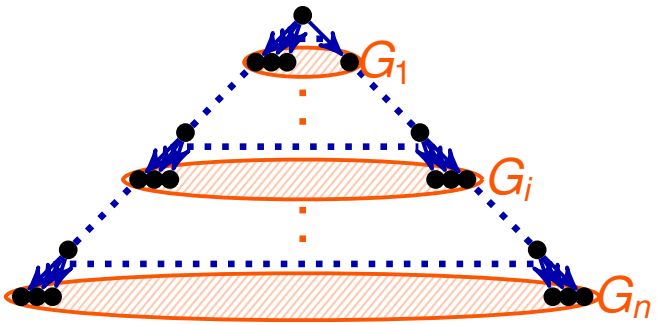
Theorem (Goldreich 2024)

Testing our property has query complexity $\Omega(\log\log\log\log n)$.

- Find the holy grails.

Outlook

- Generalise approach to larger classes and logics
- Testing FO in the bounded-degree model: what is the query-complexity?



Theorem (Goldreich 2024)

Testing our property has query complexity $\Omega(\log\log\log\log n)$.

- Find the holy grails.

Merci beaucoup!

Hyperfinite graphs

Let $0 \leq \varepsilon \leq 1$ and $k \in \mathbb{N}$.

- G is (ε, k)-hyperfinite if one can remove εn edges from G and obtain a graph whose connected components have size $\leq k$.
- Fix a function $\rho : \mathbb{R}^+ \to \mathbb{N}$.
  G is ρ-hyperfinite if G is $(\varepsilon, \rho(\varepsilon))$-hyperfinite for every $\varepsilon > 0$.
- A graph class $\mathcal{C}$ is hyperfinite if there is a function ρ such that every $G \in \mathcal{C}$ is ρ-hyperfinite.

Hyperfinite graphs

Let $0 \leq \varepsilon \leq 1$ and $k \in \mathbb{N}$.

- G is (ε, k)-hyperfinite if one can remove εn edges from G and obtain a graph whose connected components have size $\leq k$.
- Fix a function $\rho : \mathbb{R}^+ \to \mathbb{N}$.
  G is ρ-hyperfinite if G is $(\varepsilon, \rho(\varepsilon))$-hyperfinite for every $\varepsilon > 0$.
- A graph class $\mathcal{C}$ is hyperfinite if there is a function ρ such that every $G \in \mathcal{C}$ is ρ-hyperfinite.

Hyperfinite graphs

Let $0 \leq \varepsilon \leq 1$ and $k \in \mathbb{N}$.

- G is (ε, k)-hyperfinite if one can remove εn edges from G and obtain a graph whose connected components have size $\leq k$.
- Fix a function $\rho : \mathbb{R}^+ \to \mathbb{N}$.
  G is ρ-hyperfinite if G is $(\varepsilon, \rho(\varepsilon))$-hyperfinite for every $\varepsilon > 0$.
- A graph class $\mathcal{C}$ is hyperfinite if there is a function ρ such that every $G \in \mathcal{C}$ is ρ-hyperfinite.