Chordal graphs are easily testable

Rémi de Joannis de Verclos

Presented by - Amit Roy

Spring School 2025

May 14, 2025

Graph Property Testing

Graph Property Testing

Graph Property

A **graph property** is a set of graphs that is closed under graph isomorphism. That is, Π is a graph property if, for every graph $G = (V, E)$ and every bijection $\pi : V \to V'$, it holds that $G \in \Pi$ if and only if $\pi(G) \in \Pi$, where $\pi(G)$ is the graph obtained from G by relabelling the vertices according to π; that is,

$$\pi(G) = (V', \pi(u), \pi(v) \mid \{u, v\} \in E)$$

Definition

ϵ-far from property $\mathcal{P}$

A graph G on n vertices is ϵ-far from satisfying a property $\mathcal{P}$ if one has to add or delete at least ϵn^2 edges to G to obtain a graph satisfying $\mathcal{P}$.

Definition

ϵ-tester

An algorithm $\mathcal{A}$ is an ϵ-tester for property $\mathcal{P}$ if it

1. accepts all $x \in \mathcal{P}$ with probability at least $2/3$ and
2. rejects all x that are ϵ-far from $\mathcal{P}$ with probability at least $2/3$

where the probability is taken over the internal coin tosses of $\mathcal{A}$.

Definition

ϵ-tester

An algorithm $\mathcal{A}$ is an ϵ-tester for property $\mathcal{P}$ if it

1. accepts all $x \in \mathcal{P}$ with probability at least $2/3$ and
2. rejects all x that are ϵ-far from $\mathcal{P}$ with probability at least $2/3$

where the probability is taken over the internal coin tosses of $\mathcal{A}$.

- The tester has one-sided error if it always accepts functions satisfying $\mathcal{P}$.

Definition

ϵ-tester

An algorithm $\mathcal{A}$ is an ϵ-tester for property $\mathcal{P}$ if it

1. accepts all $x \in \mathcal{P}$ with probability at least $2/3$ and
2. rejects all x that are ϵ-far from $\mathcal{P}$ with probability at least $2/3$

where the probability is taken over the internal coin tosses of $\mathcal{A}$.

- The tester has one-sided error if it always accepts functions satisfying $\mathcal{P}$.
- If the queries made by the algorithm do not depend on the answers to the previous queries, $\mathcal{A}$ is called a *nonadaptive* tester. Otherwise, it is an *adaptive* tester.

Definition

A hereditary class $\mathcal{P}$ of graphs is testable if for every fixed $\epsilon > 0$ there is a size m_ϵ such that the following holds.

Definition

A hereditary class $\mathcal{P}$ of graphs is testable if for every fixed $\epsilon > 0$ there is a size m_ϵ such that the following holds.

- If G is ϵ-far from $\mathcal{P}$ then a set $X \subseteq V(G)$ sampled uniformly at random among all subsets of $V(G)$ of size m_ϵ induces a graph $G[X]$ that is not in $\mathcal{P}$ with probability at least $2/3$.

Definition

A hereditary class $\mathcal{P}$ of graphs is testable if for every fixed $\epsilon > 0$ there is a size m_ϵ such that the following holds.

- If G is ϵ-far from $\mathcal{P}$ then a set $X \subseteq V(G)$ sampled uniformly at random among all subsets of $V(G)$ of size m_ϵ induces a graph $G[X]$ that is not in $\mathcal{P}$ with probability at least $2/3$.
- The property $\mathcal{P}$ is easily testable if moreover m_ϵ is a polynomial function of $\frac{1}{\epsilon}$. Otherwise, $\mathcal{P}$ is hard to test.

Motivation

Motivation

- Need to efficiently analyze large graphs using algorithms that provide approximate answers with minimal computational resources

Motivation

- Need to efficiently analyze large graphs using algorithms that provide approximate answers with minimal computational resources
- Property testing allows decisions based on a sublinear number of queries (e.g., inspecting only a random subset of edges or vertices)

Motivation

- Need to efficiently analyze large graphs using algorithms that provide approximate answers with minimal computational resources
- Property testing allows decisions based on a sublinear number of queries (e.g., inspecting only a random subset of edges or vertices)
- For example, testing whether a graph is bipartite can be done without examining all edges.

Motivation

- Need to efficiently analyze large graphs using algorithms that provide approximate answers with minimal computational resources
- Property testing allows decisions based on a sublinear number of queries (e.g., inspecting only a random subset of edges or vertices)
- For example, testing whether a graph is bipartite can be done without examining all edges.
- Testers can act as a fast preliminary check to avoid running slower exact algorithms on graphs far from the desired property

Previous Works

Previous Works

- Every hereditary property has a one-sided tester [Alon and Shapira 2008].

Previous Works

- Every hereditary property has a one-sided tester [Alon and Shapira 2008].
- *Query complexity* - tower of towers of exponentials of size polynomial in ϵ^{-1}

Previous Works

- Every hereditary property has a one-sided tester [Alon and Shapira 2008].
- *Query complexity* - tower of towers of exponentials of size polynomial in ϵ^{-1}
- Alon and Shapira also showed that class H- FREE of graphs without induced copy of H is hard to test when H is different from P_2, P_3, P_4, C_4 and different from complement of one of these graphs.

Previous Works

- Every hereditary property has a one-sided tester [Alon and Shapira 2008].
- *Query complexity* - tower of towers of exponentials of size polynomial in ϵ^{-1}
- Alon and Shapira also showed that class H- FREE of graphs without induced copy of H is hard to test when H is different from P_2, P_3, P_4, C_4 and different from complement of one of these graphs.
- This class is known to be easily testable when $H \in \{P_2, P_3, P_4\}$.

Previous Works

- Every hereditary property has a one-sided tester [Alon and Shapira 2008].
- *Query complexity* - tower of towers of exponentials of size polynomial in ϵ^{-1}
- Alon and Shapira also showed that class H- FREE of graphs without induced copy of H is hard to test when H is different from P_2, P_3, P_4, C_4 and different from complement of one of these graphs.
- This class is known to be easily testable when $H \in \{P_2, P_3, P_4\}$.
- H for which the class H-free is easily testable are known except when H is C_4 or its complement $C_4 = 2K_2$.

Previous Works

Is C_4-FREE easily testable?

Previous Works

Is C_4-FREE easily testable?

Gishboliner and Shapira 2018

Every graph that is ϵ-far from being C_4-FREE contains at least $\frac{n^4}{2^{(1/\epsilon)^c}}$ induced copies of C_4 for some constant c, implying C_4-FREE can be tested with query complexity $2^{(1/\epsilon)^c}$.

This Paper

Chordal Graphs

A graph G is chordal if it does not contain any induced cycle of length at least four; i.e., any (≥ 4)-cycle in G has a chord (an edge between non-consecutive vertices of the cycle).

This Paper

Chordal Graphs

A graph G is chordal if it does not contain any induced cycle of length at least four; i.e., any (≥ 4)-cycle in G has a chord (an edge between non-consecutive vertices of the cycle).

- Chordal graphs is an important and natural subclass of C_4-FREE.

This Paper

Chordal Graphs

A graph G is chordal if it does not contain any induced cycle of length at least four; i.e., any (≥ 4)-cycle in G has a chord (an edge between non-consecutive vertices of the cycle).

- Chordal graphs is an important and natural subclass of C_4-FREE.
- From Gishboliner and Shapira's work, this class is testable with query complexity $2^{(1/\epsilon)^c}$.

This Paper

Chordal Graphs

A graph G is chordal if it does not contain any induced cycle of length at least four; i.e., any (≥ 4)-cycle in G has a chord (an edge between non-consecutive vertices of the cycle).

- Chordal graphs is an important and natural subclass of C_4-FREE.
- From Gishboliner and Shapira's work, this class is testable with query complexity $2^{(1/\epsilon)^c}$.
- They also conjectured that this bound can be further improved to a polynomial in $\frac{1}{\epsilon}$.

This Paper

- **The class of chordal graph is easily testable.**

Theorem

The class of chordal graph is testable with query complexity $\mathcal{O}(\epsilon^{-37})$.

Testing Connectedness

Input : A graph $G = (V, E)$ of degree at most d, represented by adjacency lists.

Testing Connectedness

Input : A graph $G = (V, E)$ of degree at most d, represented by adjacency lists.
Goal : An ϵ-tester for connectedness (that accepts all connected graphs with probability $\geq 2/3$, and rejects with probability$\geq 2/3$ all graphs that are ϵ-far from connected.

Testing Connectedness

Input : A graph $G = (V, E)$ of degree at most d, represented by adjacency lists.
Goal : An ϵ-tester for connectedness (that accepts all connected graphs with probability $\geq 2/3$, and rejects with probability$\geq 2/3$ all graphs that are ϵ-far from connected.
Idea : Graphs that are far from connected have many connected components, and therefore they also have many "small" connected components.

Testing Connectedness

Algorithm

Algorithm 1: Tester for connectedness(n, d, ϵ, query access to G).

1. Pick $s = \frac{8}{\epsilon d}$ nodes from V independently and uniformly at random.
2. For every selected node v, determine whether v is in a small connected component: do a BFS until either the connected component is exhausted or $\frac{4}{\epsilon d}$ new nodes are visited.
3. Reject if at least one small connected component was found, otherwise accept.

Testing Connectedness

Algorithm

Algorithm 1: Tester for connectedness(n, d, ϵ, query access to G).

1. Pick $s = \frac{8}{\epsilon d}$ nodes from V independently and uniformly at random.
2. For every selected node v, determine whether v is in a small connected component: do a BFS until either the connected component is exhausted or $\frac{4}{\epsilon d}$ new nodes are visited.
3. Reject if at least one small connected component was found, otherwise accept.

- **Query Complexity** - $\mathcal{O}(\frac{1}{\epsilon^2 d})$
- Running Time - same

Proof

Claim 1

If G is ϵ-far from connected, it has at least $\frac{\epsilon dn}{2}$ connected components.

Proof

Claim 1

If G is ϵ-far from connected, it has at least $\frac{\epsilon dn}{2}$ connected components.

Claim 2

If G is ϵ-far from connected, it has at least $\frac{\epsilon dn}{4}$ connected components of size at most $\frac{4}{\epsilon d}$.

Proof

Claim 1

If G is ϵ-far from connected, it has at least $\frac{\epsilon dn}{2}$ connected components.

Claim 2

If G is ϵ-far from connected, it has at least $\frac{\epsilon dn}{4}$ connected components of size at most $\frac{4}{\epsilon d}$.

Proof of Query Complexity:

- From Claim 2 we have, at least $\frac{\epsilon dn}{4}$ nodes are in small connected components (of size $\leq \frac{4}{\epsilon d}$).
- At least $\frac{\epsilon d}{4}$ fraction of the nodes of the graph "witness" that it is not connected.
- It suffices to look at $\frac{2}{\epsilon d/4} = \frac{8}{\epsilon d}$ random samples to get at least one witness with probability $\geq \frac{2}{3}$.

Chordal Graphs

Algorithm:

1. Sample a set of $s = \mathcal{O}(\epsilon^{-37})$ vertices X u.a.r.
2. If $G[X]$ is chordal then Return "YES"
3. Else Return "NO"

Chordal Graphs

Algorithm:

1. Sample a set of $s = \mathcal{O}(\epsilon^{-37})$ vertices X u.a.r.
2. If $G[X]$ is chordal then Return "YES"
3. Else Return "NO"

- **Query Complexity** - $\mathcal{O}(\epsilon^{-37})$
- **Time Complexity** - $poly(\epsilon^{-37}))$

Chordal Graphs

Properties

- A vertex of a graph is *simplicial* if its neighbourhood is a clique.

Chordal Graphs

Properties

- A vertex of a graph is *simplicial* if its neighbourhood is a clique.
- Chordal graphs are stable by addition of simplicial vertices.
- Chordal graphs are exactly the graphs that can be obtained from the empty graph by iteratively adding simplicial vertices.

Chordal Graphs

Properties

- A vertex of a graph is *simplicial* if its neighbourhood is a clique.
- Chordal graphs are stable by addition of simplicial vertices.
- Chordal graphs are exactly the graphs that can be obtained from the empty graph by iteratively adding simplicial vertices.

For a vertex v of a graph G, let $p_G(v)$ be the number of non-edges in the neighborhood of v.

Chordal Graphs

Properties

- A vertex of a graph is *simplicial* if its neighbourhood is a clique.
- Chordal graphs are stable by addition of simplicial vertices.
- Chordal graphs are exactly the graphs that can be obtained from the empty graph by iteratively adding simplicial vertices.

For a vertex v of a graph G, let $p_G(v)$ be the number of non-edges in the neighborhood of v.

- v is simplicial if and only if $p_G(v) = 0$.

Nearly Simplicial Vertices

Theorem

Let $\epsilon > 0$ and $n \geq \epsilon^{-1}$. Let G be a graph with vertex partition $X \cup Y$ such that $G[Y]$ is chordal and $p_G(v) \leq \epsilon n^2$ for every $v \in X$. Then G is $6\epsilon^{1/2}$-close from a chordal graph.

Main Theorem - Proof Overview

Bird's View

1. Use the *Simplicial* Theorem, with a parameter δ such that $6\delta^{1/2} = \frac{\epsilon}{2}$.
2. Partition the vertex set into two sets (vertices whose neighborhoods have few non-edges) and Y (the rest).
3. Using testability of k-coloring and the representation of chordal graphs as intersection graphs of subtrees, show that if $G[Y]$ is far from chordal, this can be detected with high probability from a random sample.
4. With a union bound argument we can prove that $\mathcal{O}(\epsilon^{-37})$ vertices are sufficient to test for chordalness.

Thanks

References

Verclos, Rémi de Joannis de. “Chordal graphs are easily testable.” arXiv: Combinatorics (2019): n. pag.

Noga Alon and Asaf Shapira. 2006. A Characterization of Easily Testable Induced Subgraphs. Comb. Probab. Comput. 15, 6 (November 2006), 791–805. https://doi.org/10.1017/S0963548306007759

Gishboliner, L., Shapira, A. Efficient Removal Without Efficient Regularity. Combinatorica 39, 639–658 (2019). https://doi.org/10.1007/s00493-018-3899-6