Hierarchy Theorems for Property Testing*

Oded Goldreich† Michael Krivelevich‡ Ilan Newman§ Eyal Rozenberg¶

November 23, 2009

Abstract

Referring to the query complexity of property testing, we prove the existence of a rich hierarchy of corresponding complexity classes. That is, for any relevant function q, we prove the existence of properties that have testing complexity $\Theta(q)$. Such results are proven in three standard domains often considered in property testing: generic functions, adjacency predicates describing (dense) graphs, and incidence functions describing bounded-degree graphs. While in two cases the proofs are quite straightforward, the techniques employed in the case of the dense graph model seem significantly more involved. Specifically, problems that arise and are treated in the latter case include (1) the preservation of distances between graph under a blow-up operation, and (2) the construction of monotone graph properties that have local structure.

Keywords: Property Testing, Graph Properties, Monotone Graph Properties, Graph Blow-up, One-Sided versus Two-Sided Error, Adaptivity versus Non-adaptivity,

*A preliminary version has appeared in the proceedings of *RANDOM'09*.

†Faculty of Mathematics and Computer Science, Weizmann Institute of Science, Rehovot, ISRAEL. Email: oded.goldreich@weizmann.ac.il. Partially supported by the Israel Science Foundation (grant No. 1041/08).

‡School of Mathematical Sciences, Tel Aviv University, Tel Aviv 69978, ISRAEL. Email: krivelev@post.tau.ac.il. Partially supported by a USA-Israel BSF Grant, by a grant from the Israel Science Foundation, and by Pazy Memorial Award.

§Department of Computer Science, Haifa University, Haifa, ISRAEL. Email: ilan@cs.haifa.ac.il. Partially supported by an Israel Science Foundation (grant number 1011/06).

¶Department of Computer Science, Technion, Haifa, ISRAEL. Email: eyalroz@technion.ac.il

Contents

1 Introduction **1**
- 1.1 Our main results . . . 1
- 1.2 Our techniques . . . 2
- 1.3 Organization . . . 3

2 Properties of Generic Functions **3**

3 Graph Properties in the Bounded-Degree Model **4**

4 Graph Properties in the Adjacency Matrix Model **6**
- 4.1 The blow-up property Π . . . 7
- 4.2 Lower-bounding the query complexity of testing Π . . . 9
- 4.3 An optimal tester for property Π . . . 10

5 Revisiting the Adjacency Matrix Model: Monotone Properties **12**
- 5.1 The monotone property Π . . . 13
- 5.2 Lower-bounding the query complexity of testing Π . . . 15
- 5.3 An optimal tester for property Π . . . 17

6 Revisiting the Adjacency Matrix Model: One-Sided Error **21**
- 6.1 The (generalized) blow-up property Π . . . 22
- 6.2 Lower-bounding the query complexity of testing Π . . . 22
- 6.3 An optimal tester for property Π . . . 24

7 Hard-to-test Graph Properties in P **26**

8 Concluding Comments **29**

Bibliography **32**

1 Introduction

In the last decade, the area of property testing has attracted much attention (see, e.g., a couple of recent surveys [R1, R2]). Loosely speaking, property testing typically refers to sub-linear time probabilistic algorithms for deciding whether a given object has a predetermined property or is far from any object having this property. Such algorithms, called testers, obtain local views of the object by making adequate queries; that is, the object is seen as a function and the testers get oracle access to this function (and thus may be expected to work in time that is sub-linear in the length of the object).

Following most work in the area, we focus on the query complexity of property testing, measured as a function of the size of the object as well as the desired proximity (parameter). Interestingly, many natural properties can be tested in complexity that only depends on the proximity parameter; examples include linearity testing [BLR], and testing various graph properties in two natural models (e.g., [GGR, AFNS] and [GR1, BSS], respectively). On the other hand, properties for which testing requires essentially maximal query complexity were proved to exist too; see [GGR] for artificial examples in two models and [BHR, BOT] for natural examples in other models. In between these two extremes, there exist natural properties for which the query complexity of testing is logarithmic (e.g., monotonicity [EKK+, GGL+]), a square root (e.g., bipartitness in the bounded-degree model [GR1, GR2]), and possibly other constant powers (see [FM, PRR]).

1.1 Our main results

One natural question that arises is whether there exist properties of arbitrary query complexity. We answer this question affirmatively, proving the existence of a rich hierarchy of query complexity classes. Such hierarchy theorems are easiest to state and prove in the generic case (treated in Section 2): Loosely speaking, for every sub-linear function q, *there exists a property of functions over* $[n]$ *that is testable using* $q(n)$ *queries but is not testable using* $o(q(n))$ *queries.*

Similar hierarchy theorems are proved also for two standard models of testing graph properties: the adjacency representation model (of [GGR]) and the incidence representation model (of [GR1]). For the incidence representation model (a.k.a the bounded-degree graph model), we show (in Section 3) that, for every sub-linear function q, *there exists a property of bounded-degree* N*-vertex graphs that is testable using* $q(N)$ *queries but is not testable using* $o(q(N))$ *queries.* Furthermore, one such property corresponds to the set of N-vertex graphs that are 3-colorable and consist of connected components of size at most $q(N)$.

The bulk of this paper is devoted to hierarchy theorems for the adjacency representation model (a.k.a the dense graph model), where the complexity is stated as a function of the number of vertices (rather than as a function of the number of all vertex pairs, which is the representation size). Our main results for the adjacency matrix model are:

1. For every sub-quadratic function q, *there exists a graph property* Π *that is testable in* q *queries, but is not testable in* $o(q)$ *queries.* Furthermore, for "nice" functions q, it is the case that Π is in $\mathcal{P}$ and the tester can be implemented in poly(q)-time. (See Section 4.)

2. For every sub-quadratic function q, there exists a *monotone* graph property Π that is testable in $O(q)$ queries, but is not testable in $o(q)$ queries. (See Section 5.)

The adjacency representation model is further studied in Sections 6 and 7. (See Section 8 for further discussion.)

Conventions. For sake of simplicity, we state all results while referring to query complexity as a function of the object's size; that is, we consider a fixed (constant) value of the proximity parameter, denoted ϵ. In such cases, we sometimes use the term ϵ-testing, which refers to testing when the proximity parameter is fixed to ϵ. All our lower bounds hold for any sufficiently small value of the proximity parameter, whereas the upper bounds hide a (polynomial) dependence on (the reciprocal of) this parameter. In general, bounds that have no dependence on the proximity parameter refer to some (sufficiently small but) fixed value of this parameter.

A remotely related prior work. In contrast to the foregoing conventions, we mention here a result that refers to graph properties that are testable in (query) complexity that only depends on the proximity parameter. This result, due to [AS], establishes a (very sparse) hierarchy of such properties. Specifically, [AS, Thm. 4] asserts that for every function q there exists a function Q and a graph property that is ϵ-testable in $Q(\epsilon)$ queries but is *not* ϵ-testable in $q(\epsilon)$ queries. (We note that while Q depends only on q, the dependence proved in [AS, Thm. 4] is quite weak (i.e., Q is lower bounded by a non-constant number of compositions of q), and thus the hierarchy obtained by setting $q_i = Q_{i-1}$ for $i = 1, 2, ...$ is very sparse.)

1.2 Our techniques

The proofs of the hierarchy theorems for the generic case (treated in Section 2) and for the incidence representation graph model (treated in Section 3), are quite straightforward. In contrast, the treatment of the dense graph model is significantly more involved. We discuss the source of trouble next.

Given that properties of maximal query complexity are known in each of the testing models that we consider, a natural idea towards proving hierarchy theorems is to construct properties that correspond to repetitions of the original properties; that is, each object in the new property consists of an adequate number of objects, each belonging to the original property. Straightforward implementations of this idea work in the generic case and in the incidence representation graph model, but not in the dense graph model. The point is that a naive repetition of a graph, in this model, necessarily creates a graph that is not dense.

Nevertheless, the graph blow-up operation (see Section 4) does seem to be the adequate construction that we seek. Loosely speaking, the graph blow-up operation replaces each vertex by an independent set, and replaces edges by corresponding complete bipartite graphs. One source of trouble is that the blow-up operation does not necessarily preserve distances; indeed the relative distance between the blow-up of G_1 and G_2 is at most the relative distance between the original graphs, but the naive assumption that it may not be smaller is false. In Section 4 we overcome this difficulty by showing that for certain graphs, which we call dispersed, the blow-up does preserve the original distances (up to a constant factor). Thus, we first reduce the testing of the original property to testing a corresponding property that refers to dispersed graphs. (An n-vertex graph is called dispersed if the neighbor sets of any two vertices differ on at least $\Omega(n)$ elements.)

Using dispersed graphs also allows us to overcome another technical difficulty, which relates to the complexity of our tester. In particular, the use of dispersed graphs allows us to recover the canonical labeling of an unlabeled graph, which is helpful whenever a graph property (viewed as a set of labeled graphs) is obtained by a closure under isomorphism of some set of labeled graphs (cf. [GGR]). (For details see Section 7.)

When trying to obtain a result for monotone graph properties, we encounter another technical difficulty. The difficulty is that standard constructions of monotone graph properties (cf. [GT]) tend to lack any local structure, since the property should be preserved under arbitrary edge additions. We demonstrate that the latter conclusion is a bit hasty, by showing that a local structure can be essentially maintained as long as the edge density does not exceed some threshold, whereas we can include in the property all graphs that have edge density that exceeds this threshold. (For details see Section 5.)

A third type of difficulty arises in Section 6, where we use a different type of graph blow-up. Specifically, under the aforementioned blow-up each vertex is replaced by an independent set of the same size, whereas in Section 6 we used a generalized blow-up in which these independent sets may have different sizes.

1.3 Organization

Sections 2 and 3 present hierarchy theorems for the generic case and for the bounded-degree graph model, respectively. The bulk of this paper provides hierarchy theorems for graph properties in the adjacency matrix model. The basic hierarchy theorem regarding this model is presented in Section 4, whereas in Section 5 we obtain such a theorem for monotone graph properties.

In Section 6 we address a refined issue that has been ignored above. Specifically, we note that all our lower bounds refer to two-sided error testers, whereas the upper bounds in Sections 2 and 3 are demonstrated using one-sided error testers, which only make these separations stronger. In contrast, the upper bounds presented in Sections 4 and 5 use two-sided error testers. In Section 6 we modify the construction of Section 4 in order to obtain one-sided error testers (while the lower bounds still hold for two-sided error testers). However, the latter theorem loses some features of the former theorems; see Section 8 for further discussion.

We mention that our results for graph properties in the adjacency matrix model use the existence of graph properties that are in $\mathcal{P}$ and have maximal query complexity. This result is presented in Section 7, building on a prior construction of [GGR], which only asserted such properties in $\mathcal{NP}$.

2 Properties of Generic Functions

In the generic function model, the tester is given oracle access to a function over $[n]$, and distance between such functions is defined as the fraction of (the number of) arguments on which these functions differ. In addition to the input oracle, the tester is explicitly given two parameters: a size parameter, denoted n, and a proximity parameter, denoted ϵ.

Definition 1 *Let $\Pi = \bigcup_{n\in\mathbb{N}} \Pi_n$, where Π_n contains functions defined over the domain $[n] \stackrel{\rm def}{=} \{1, ..., n\}$. A* tester for a property Π *is a probabilistic oracle machine T that satisfies the following two conditions:*

1. *The tester accepts each $f \in \Pi$ with probability at least $2/3$; that is, for every $n \in \mathbb{N}$ and $f \in \Pi_n$* (and every $\epsilon > 0$), *it holds that $\Pr[T^f(n,\epsilon)=1] \geq 2/3$.*

2. *Given $\epsilon > 0$ and oracle access to any f that is ϵ-far from Π, the tester rejects with probability at least $2/3$; that is, for every $\epsilon > 0$ and $n \in \mathbb{N}$, if $f : [n] \to \{0,1\}^*$ is ϵ-far from Π_n, then $\Pr[T^f(n,\epsilon) = 0] \geq 2/3$, where g is* ϵ-far from Π_n *if, for every $g \in \Pi_n$, it holds that $|\{i \in [n] : f(i) \neq g(i)\}| > \epsilon \cdot n$.*

We say that the tester has one-sided error *if it accepts each $f \in \Pi$ with probability 1; that is, for every $f \in \Pi$ and every $\epsilon > 0$, it holds that $\Pr[T^f(n,\epsilon)=1] = 1$.*

When $\epsilon > 0$ is fixed, we refer to the residual oracle machine $T(\cdot,\epsilon)$ by the term ϵ-tester. We also use the corresponding term ϵ-testing Π.

Definition 1 does not specify the query complexity of the tester, and indeed an oracle machine that queries the entire domain of the function qualifies as a tester (with zero error probability...). Needless to say, we are interested in testers that have significantly lower query complexity. Recall that [GGR] asserts that in some cases such testers do not exist; that is, there exist properties that require linear query complexity. Building on this result, we show:

Theorem 2 *For every $q : \mathbb{N} \to \mathbb{N}$ that is onto and at most linear, there exists a property Π of Boolean functions that is testable* (with one-sided error) *in $q + O(1)$ queries, but is not testable in $o(q)$ queries* (even when allowing two-sided error).

Proof: We start with an arbitrary property Π' of Boolean functions for which testing is known to require a linear number of queries (even when allowing two-sided error). The existence of such properties was first proved in [GGR]. Given $\Pi' = \bigcup_{m\in\mathbb{N}} \Pi'_m$, we define $\Pi = \bigcup_{n\in\mathbb{N}} \Pi_n$ such that Π_n consists of "duplicated versions" of the functions in $\Pi'_{q(n)}$. Specifically, for every $f' \in \Pi'_{q(n)}$, we define $f(i) = f'(i \bmod q(n))$ and add f to Π_n, where $i \bmod m$ is (non-standardly) defined as the smallest *positive* integer that is congruent to i modulo m,

The query complexity lower bound of Π follows from the corresponding bound of Π'. Specifically, approximate membership of f' in Π'_m can be tested by emulating the testing of an imaginary function $f : [n] \to \{0,1\}$ defined such that $m = q(n)$ and $f(i) = f'(i \bmod m)$; that is, testing f' w.r.t Π'_m is performed by testing f w.r.t Π_n, while emulating oracle access to f by making corresponding queries to f'. Clearly, if $f' \in \Pi'_m$ then $f \in \Pi_n$, whereas if f' is ϵ-far from Π'_m then f is $\frac{\lfloor n/m\rfloor\cdot m}{n}\cdot\epsilon$-far from Π_n. Assuming without loss of generality that $q(n) \le n/2$, we have $\lfloor n/m\rfloor \cdot m \ge n/2$. Thus, a $o(q(n))$-query oracle machine that distinguishes the case that $f \in \Pi_n$ from the case that f is $(\epsilon/2)$-far from Π_n, yields a $o(m)$-query oracle machine that distinguishes the case that $f' \in \Pi'_m$ from the case that f' is ϵ-far from Π'_m. We conclude that an $\Omega(m)$ lower bound on ϵ-testing Π'_m implies an $\Omega(q(n))$ lower bound on $(\epsilon/2)$-testing Π_n.

The query complexity upper bound of Π follows by using a straightforward tester that essentially reconstructs the underlying function and checks whether it is in Π'. Specifically, on input n, ϵ and access to $f : [n] \to \{0,1\}$, we test whether f is a repetition of some function $f' : [q(n)] \to \{0,1\}$ in $\Pi'_{q(n)}$. This is done by conducting the following two steps:

1. Repeat the following basic check $O(1/\epsilon)$ times: Uniformly select $j \in [q(n)]$ and $r \in [\lfloor n/q(n)\rfloor - 1]$, and check whether $f(r \cdot q(n) + j) = f(j)$.

2. Using $q(n)$ queries, construct $f' : [q(n)] \to \{0,1\}$ such that $f'(i) \stackrel{\text{def}}{=} f(i)$, and check whether f' is in Π'. Note that checking whether f' is in Π' requires no queries, and that the corresponding computational complexity is ignored here.

Note that this (non-adaptive) oracle machine has query complexity $q(n) + O(1/\epsilon)$, and it accepts any $f \in \Pi$ with probability 1. On the other hand, if f is accepted with probability at least $2/3$, then the reconstructed f' must be in Π' (otherwise Step 2 would have rejected with probability 1) and f must be ϵ-close to the repetition of this f' (otherwise each iteration of Step 1 would have rejected with probability at least $\epsilon/2$, where we again use $q(n) \le n/2$). Thus, in this case f is ϵ-close to Π, which establishes the upper bound on the query complexity of testing Π. The theorem follows. ■

Comment. Needless to say, Boolean functions over $[n]$ may be viewed as n-bit long binary strings. Thus, Theorem 2 means that, for every sub-linear q, there are properties of binary strings for which the query complexity of testing is $\Theta(q)$. Given this perspective, it is natural to comment that such properties exist also in $\mathcal{P}$. This comment can be proved by starting with the hard-to-test property asserted in Theorem 7. Actually, starting with the hard-to-test property asserted in [LNS] (which is in $\mathcal{L}$), we obtain a hierarchy theorem for testing properties that are in $\mathcal{L}$.

3 Graph Properties in the Bounded-Degree Model

The bounded-degree model refers to a fixed (constant) degree bound, denoted $d \ge 2$. An N-vertex graph $G = ([N], E)$ (of maximum degree d) is represented in this model by a function $g : [N] \times [d] \to \{0, 1, ..., N\}$ such that $g(v, i) = u \in [N]$ if u is the i^{th} neighbor of v and $g(v, i) = 0$ if v has less than i neighbors. (For simplicity, we assume here that the neighbors of v appear in arbitrary order in the sequence $g(v, 1), ..., g(v, \deg(v))$, where $\deg(v) \stackrel{\text{def}}{=} |\{i : g(v, i) \ne 0\}|$.) Distance between graphs is measured in terms of their aforementioned representation; that is, as the fraction of (the number of) different array entries (over dN). Graph properties are properties that are invariant under renaming of the vertices (i.e., they are actually properties of the underlying unlabeled graphs).

Recall that [BOT] proved that, in this model, testing 3-Colorability requires a linear number of queries (even when allowing two-sided error). Building on this result, we show:

Theorem 3 *In the bounded-degree graph model, for every $q : \mathbb{N} \to \mathbb{N}$ that is onto and at most linear, there exists a graph property Π that is testable* (with one-sided error) *in $O(q)$ queries, but is not testable in $o(q)$ queries* (even when allowing two-sided error). *Furthermore, this property is the set of N-vertex graphs of maximum degree d that are 3-colorable and consist of connected components of size at most $q(N)$.*

Proof: Actually, we may start with an arbitrary property Π' that satisfies the following two conditions:

1. Testing Π requires a linear number of queries (even when allowing two-sided error).
2. The property Π' is downward monotone; that is, if $G' \in \Pi'$ then any subgraph of G' is in Π'. In particular, the single-vertex graph is in Π'.

Indeed, by [BOT], 3-Colorability is such a property. Given $\Pi' = \bigcup_{n \in \mathbb{N}} \Pi'_n$, we define $\Pi = \bigcup_{N \in \mathbb{N}} \Pi_N$ such that Π_N is the set of all graphs that consist of connected components that are each in Π' and have size at most $q(N)$; that is, each connected component in any $G \in \Pi_N$ is in Π'_n for some $n \leq q(N)$ (i.e., n denotes this component's size).

The query complexity lower bound of Π follows from the corresponding bound of Π'. Specifically, approximate membership of the n-vertex graph G' in Π'_n can be tested by setting N such that $q(N) = n$ and emulating the testing of the N-vertex graph G obtained by taking $t \stackrel{\text{def}}{=} \lfloor N/q(N) \rfloor$ copies of G' (and additional $N - t \cdot q(N)$ isolated vertices). Clearly, if $G' \in \Pi'_n$ then $G \in \Pi_N$. On the other hand, if G' is ϵ-far from Π'_n then G is $\frac{t \cdot n}{N} \cdot \epsilon$-far from Π_N (because, by the downward monotonicity of Π', it suffices to consider the number of edges that must be omitted from G in order to obtain a graph in Π_N). As in the proof of Theorem 2, we may assume that $t \cdot n \geq N/2$, and conclude that in the latter case G is $(\epsilon/2)$-far from Π_N. Thus, a $o(q(N))$-query oracle machine that distinguishes the case that $G \in \Pi_N$ from the case that G is $(\epsilon/2)$-far from Π_N, yields a $o(n)$-query oracle machine that distinguishes the case that $G' \in \Pi'_n$ from the case that G' is ϵ-far from Π'_n. The desired $\Omega(q(N))$ lower bound follows.

The query complexity upper bound of Π is obtained by using a tester that selects at random a start vertex s in the input N-vertex graph and tests that s resides in a connected component that is in Π'_n for some $n \leq q(N)$. Specifically, on input N, ϵ and access to an N-vertex graph G, we repeat the following test $O(1/\epsilon)$ times.

1. Uniformly select a start vertex s, and explore its connected component untill either encountering $q(N) + 1$ vertices or discovering that the connected component has at most $q(N)$ vertices.
2. Denoting the number of encountered vertices by n, reject of $n > q(N)$. Similarly reject if the encountered graph is not in Π'_n.

The query complexity of this oracle machine is $O(d \cdot q(N)/\epsilon)$, which is $O(q(N))$ when both d and $\epsilon > 0$ are constants. Clearly, this oracle machine accepts any $G \in \Pi$ with probability 1. In analyzing its performance on graphs not in Π, we call a start vertex bad if it resides in a connected component that is either bigger than $q(N)$ or not in Π'. Note that if G has more than ϵN bad vertices, then the foregoing tester rejects with probability at least $2/3$. Otherwise (i.e., G has fewer than ϵN bad vertices), G is ϵ-close to Π, because we can omit all edges incident to bad vertices and obtain a graph in Π. The theorem follows. ■

Comment. The construction used in the proof of Theorem 3 is slightly different from the one used in the proof of Theorem 2: In the proof of Theorem 3 each object in Π_N corresponds to a sequence of (possibly different) objects in Π'_n, whereas in the proof of Theorem 2 each object in Π_N corresponds to multiple copies of a single object in Π'_n. While Theorem 2 can be proved using a construction that is analogous to one used in the proof of Theorem 3, the current proof of Theorem 2 provides a better starting point for the proof of the following Theorem 4.

4 Graph Properties in the Adjacency Matrix Model

In the adjacency matrix model, an N-vertex graph $G = ([N], E)$ is represented by the Boolean function $g : [N] \times [N] \to \{0,1\}$ such that $g(u,v) = 1$ if and only if u and v are adjacent in G (i.e., $\{u,v\} \in E$). Distance between graphs is measured in terms of their aforementioned representation; that is, as the fraction of (the number of) different matrix entries (over N^2). In this model, we state complexities in terms of the number of vertices (i.e., N) rather than in terms of the size of the representation (i.e., N^2). Again, we focus on graph properties (i.e., properties of labeled graphs that are invariant under renaming of the vertices).

Recall that [GGR] proved that, in this model, there exist graph properties for which testing requires a quadratic (in the number of vertices) query complexity (even when allowing two-sided error). It was further shown that such properties are in $\mathcal{NP}$. Slightly modifying these properties, we show that they can be placed in $\mathcal{P}$; see Section 7. Building on this result, we show:

Theorem 4 *In the adjacency matrix model, for every $q : \mathbb{N} \to \mathbb{N}$ such that $N \mapsto \lfloor\sqrt{N}\rfloor$ is onto and at most linear, there exists a graph property Π that is testable in q queries, but is not testable in $o(q)$ queries.* (Both the upper and lower bounds refer to two-sided error testers.) *Furthermore, if $N \mapsto q(N)$ is computable in* poly$(\log N)$*-time, then Π is in $\mathcal{P}$, and the tester is relatively efficient in the sense that its running time is polynomial in the total length of its queries.*

We stress that, unlike in the previous results, the positive part of Theorem 4 refers to a two-sided error tester. This is fair enough, since the negative side also refers to two-sided error testers. Still, one may seek a stronger separation in which the positive side is established via a one-sided error tester. Such a separation is presented in Theorem 6 (alas the positive side is established via a tester that is not relatively efficient).

Outline of the proof of Theorem 4. The basic idea of the proof is to implement the strategy used in the proof of Theorem 2. The problem, of course, is that we need to obtain graph properties (rather than properties of generic Boolean functions). Thus, the trivial "blow-up" (of Theorem 2) that took place on the truth-table (or function) level has to be replaced by a blow-up on the vertex level. Specifically, starting from a graph property Π' that requires quadratic query complexity, we consider the graph property Π consisting of N-vertex graphs that are obtained by a $(N/\sqrt{q(N)})$-factor blow-up of $\sqrt{q(N)}$-vertex graphs in Π', where G is a t-factor blow-up of G' if the vertex set of G can be partitioned into t-sized sets that correspond to the vertices of G' such that the edges between these sets represent the edges of G'; that is, if $\{i,j\}$ is an edge in G', then there is a complete bipartite between the i^{th} set and the j^{th} set, and otherwise there are no edges between this pair of sets. (In particular, there are no edges inside any set.)

Note that the notion of "graph blow-up" does not offer an easy identification of the underlying partition; that is, given a graph G that is as a t-factor blow-up of some graph G', it is not necessary easy to determine a partition of the vertex set of G (into t-sized sets) such that the edges between these (t-sized) sets represent the edges of G'. Things may become even harder if G is merely close to a t-factor blow-up of some graph G'. We resolve these as well as other difficulties by augmenting the graphs of the starting property Π'.

The proof of Theorem 4 is organized accordingly: In Section 4.1, we construct Π based on Π' by first augmenting the graphs and then applying graph blow-up. In Section 4.2 we lower-bound the query complexity of Π based on the query complexity of Π', while coping with the non-trivial question of *how does the blow-up operation affect distances between graphs.* In Section 4.3 we upper-bound the query complexity of Π, while using the aforementioned augmentations in order to obtain a tight result (rather than an upper bound that is off by a polylogarithmic factor).

4.1 The blow-up property Π

Our starting point is any graph property $\Pi' = \bigcup_{n\in\mathbb{N}} \Pi'_n$ for which testing requires quadratic query complexity. Furthermore, we assume that Π' is in $\mathcal{P}$. Such a graph property is presented in Theorem 7 (see Section 7, which builds on [GGR]).

The notion of graphs that have "substantially different vertex neighborhoods" is central to our analysis. Specifically, for a real number $\alpha > 0$, we say that a graph $G = (V, E)$ is α-dispersed if the neighbor sets of any two vertices differ on at least $\alpha \cdot |V|$ elements (i.e., for every $u \neq v \in V$, the symmetric difference between the sets $\{w : \{u, w\} \in E\}$ and $\{w : \{v, w\} \in E\}$ has size at least $\alpha \cdot |V|$). We say that a set of graphs is dispersed if there exists a constant $\alpha > 0$ such that every graph in the set is α-dispersed. (Our notion of dispersibility has nothing to do with the notion of dispersers, which in turn is a weakening of the notion of (randomness) extractors (see, e.g., [S]).)

The augmentation. We first augment the graphs in Π' such that the resulting graphs are dispersed, while the augmentation amounts to adding a linear number of vertices. The fact that these resulting graphs are dispersed will be useful for establishing both the lower and upper bounds. The augmentation is performed in two steps. First, setting $n' = 2^{\lceil \log_2(2n+1)\rceil} \in [2n + 1, 4n]$, we augment each graph $G' = ([n], E')$ by $n' - n$ isolated vertices, yielding an n'-vertex graph $H' = ([n'], E')$ in which every vertex has degree at most $n - 1$. Next, we augment each resulting n'-vertex graph H' by an n'-vertex clique and connect the vertices of H' and the clique vertices by a bipartite graph that corresponds to a Hadamard matrix; that is, the i^{th} vertex of H' is connected to the j^{th} vertex of the clique if and only if the inner product modulo 2 of $i - 1$ and $j - 1$ (viewed as $(\log_2 n')$-bit long strings) equals 1. Thus, each n'-vertex graph H' yields a $2n'$-vertex graph that contains H' one one side, a clique on the other side, and a "Hadamard-based" bipartite graph connecting them (see Figure 1).

We denote the resulting set of (unlabeled) graphs by Π'' (and sometimes refer to Π'' as the set of all labeled graphs obtained from these unlabeled graphs). We show that Π'' is dispersed and inherits the fundamental features of Π'.

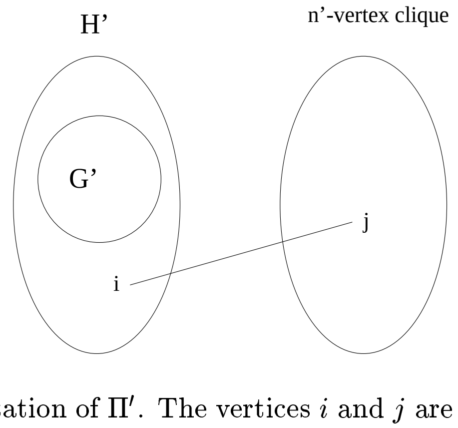

Figure 1: The two stage augmentation of Π'. The vertices i and j are connected if and only if the inner product modulo 2 of the binary representations of $i - 1$ and $j - 1$ equals 1.

Claim 4.1 *The graph property Π'' satisfies the following conditions.*

1. *The set Π'' is dispersed; that is, the resulting $2n'$-vertex graphs have vertex neighborhoods that differ on at least $n \geq n'/4$ vertices.*

2. *Testing Π'' requires a quadratic number of queries.*

3. *The set Π'' is in $\mathcal{P}$.*

Proof: We first show that the resulting $2n'$-vertex graphs have vertex neighborhoods that differ on at least $n \geq n'/4$ vertices. Consider the graph obtained by augmenting the n-vertex graph G', and let H' be the intermediate n'-vertex graph derived from G'. Then, vertices in H' neighbor (at most) $n'/2$ clique vertices, whereas vertices in the clique neighbor all other $n'-1$ clique vertices. Thus, these types of vertices differ on at least $(n'/2)-1 > n-1$ neighbors. As for any two vertices in H', by the use of the Hadamard bipartite graph, their neighborhood in the clique disagrees on $n'/2 > n$ vertices. An analogous claim holds with respect to any two vertices of the clique.

Proving that testing Π'' requires a quadratic number of queries is done by reducing testing Π' to testing Π''; specifically, ϵ-testing membership in Π'_n reduces to ϵ'-testing membership in $\Pi''_{2n'}$, where $n' \leq 4n$ and $\epsilon' = \epsilon/64$. The reduction merely emulates an $2n'$-vertex graph by making queries to to corresponding n-vertex graph (while answering some queries (i.e., those that are not confined to the original graph) according to the construction and without issuing any queries). Note that, since the original graph occupies an $n/2n' \geq 1/8$ fraction of the augmented graph, the relative distance to the property is reduced by a factor of at most 64.

Finally, note that the hypothesis that $\Pi' \in \mathcal{P}$ implies that Π'' is also in $\mathcal{P}$, because it is easy to distinguish the original graph from the vertices added to it, since the clique vertices have degree at least $n'-1$ whereas the vertices of G' have degree at most $(n-1)+(n'/2) < n'-1$ (and isolated vertices of H' have neighbors only in the clique). Once this is done, we can verify that the original graph is in Π (using $\Pi \in \mathcal{P}$), and that the additional edges correspond to a Hadamard matrix. $\square$

Applying graph blow-up. Next, we apply an (adequate factor) graph blow-up to the augmented set of graphs Π''. Actually, for simplicity of notation we assume, without loss of generality, that $\Pi' = \bigcup_{n\in\mathbb{N}} \Pi'_n$ itself is dispersed, and apply graph blow-up to Π' itself (rather than to Π''). Given a desired complexity bound $q : \mathbb{N} \to \mathbb{N}$, we first set $n = \sqrt{q(N)}$, and next apply to each graph in Π'_n an N/n-factor blow-up, thus obtaining a set of N-vertex graphs denoted Π_N. (Indeed, we assume for simplicity that both $n = \sqrt{q(N)}$ and N/n are integers.) Recall that G is a t-factor blow-up of G' if the vertex set of G can be partitioned into t-sized sets, called clouds, such that the edges between these clouds represent the edges of G'; that is, if $\{i,j\}$ is an edge in G', then there is complete bipartite between the i^{th} cloud and the j^{th} cloud, and otherwise there are no edges between this pair of clouds. This yields a graph property $\Pi = \bigcup_{N\in\mathbb{N}} \Pi_N$.

Let us first show that Π is in $\mathcal{P}$. The proof that the query complexity of testing Π indeed equals $\Theta(q)$ is undertaken in the next two sections.

Claim 4.2 *The graph property Π is in $\mathcal{P}$.*

Proof: The proof relies on the hypothesis that Π' is dispersed, or rather on the fact that each vertex in each $G' \in \Pi'$ has a distinct set of neighbors. This fact allows us to cluster vertices (in a graph resulting from a blow-up of any such G') according to their neighbor set. Specifically, given any graph N-vertex graph G, we first cluster its vertices according to their neighborhood, and check whether the number of clusters equals $n = \sqrt{q(N)}$. (Note that if $G \in \Pi_N$, then we obtain exactly n (equal sized) clusters, which correspond to the n clouds that are formed in the N/n-factor blow-up that yields G.) Next, we check that each cluster has size N/n and that the edges between these clusters correspond to the blow-up of some n-vertex graph, denoted G'. Finally, we check whether G' is in Π'_n, while relying on the fact that $\Pi' \in \mathcal{P}$. $\square$

4.2 Lower-bounding the query complexity of testing Π

In this section we prove that the query complexity of testing Π is $\Omega(q)$. The basic idea is reducing testing Π' to testing Π; that is, given a graph G' that we need to test for membership in Π'_n, we test its N/n-factor blow-up for membership in Π_N, where N is chosen such that $n = \sqrt{q(N)}$. This approach relies on the assumption that the N/n-factor blow-up of any n-vertex graph that is far from Π'_n results in a graph that is far from Π_N. (Needless to say, the N/n-factor blow-up of any graph in Π'_n results in a graph that is in Π_N.)

Unfortunately, as shown by Arie Matsliah, the aforementioned assumption does *not* hold in the strict sense of the word (i.e., it is not true that the blow-up of any graph that is ϵ-far from Π' results in a graph that is ϵ-far from Π).[1] However, for our purposes it suffices to prove a relaxed version of the aforementioned assumption that only asserts that *for any $\epsilon' > 0$ there exists an $\epsilon > 0$ such that the blow-up of any graph that is ϵ'-far from Π' results in a graph that is ϵ-far from* Π. Below we prove this assertion for $\epsilon = \Omega(\epsilon')$ and rely on the fact that Π' is dispersed. (We mention that in Appendix B of our technical report [GKNR], we present a more complicated proof that holds for arbitrary Π' (which need not be dispersed), but with $\epsilon = \Omega(\epsilon')^2$. Our result was superseded by Oleg Pikhurko, who showed that *the distance is actually preserved up to a factor of three* [P, Sec. 4].)

Lemma 4.3 *There exists a universal constant $c > 0$ such that the following holds for every n, ϵ', α and every pair of* (unlabeled) *n-vertex graphs, (G'_1, G'_2). If G'_1 is α-dispersed and ϵ'-far from G'_2, then for any t the* (unlabeled) *t-factor blow-up of G'_1 is $c\alpha \cdot \epsilon'$-far from the* (unlabeled) *t-factor blow-up of G'_2.*

Using Lemma 4.3 we infer that *if G' is ϵ'-far from Π' then its blow-up is $\Omega(\epsilon')$-far from* Π. This inference relies on the fact that Π' is dispersed (and on Lemma 4.3 when applied to $G'_2 = G'$ and every $G'_1 \in \Pi'$).

Proof: Let G_1 (resp., G_2) denote the (unlabeled) t-factor blow-up of G'_1 (resp., G'_2), and consider a bijection π of the vertices of $G_1 = ([t \cdot n], E_1)$ to the vertices of $G_2 = ([t \cdot n], E_2)$ that minimizes the size of the set (of violations)

$$\{(u,v) \in [t \cdot n]^2 : \{u,v\} \in E_1 \text{ iff } \{\pi(u), \pi(v)\} \notin E_2\}. \tag{1}$$

(Note that Eq. (1) refers to ordered pairs, whereas the distance between graphs refers to unordered pairs.) Clearly, if π were to map to each cloud of G_2 only vertices that belong to a single cloud of G_1 (equiv., for every u and v that belong to the same cloud of G_1 it holds that $\pi(u)$ and $\pi(v)$ belong to the same cloud of G_2), then G_2 would be ϵ'-far from G_1 (since the fraction of violations under such a mapping equals the fraction of violations in the corresponding mapping of G'_1 to G'_2). The problem, however, is that it is not clear that π behaves in such a nice manner (and so violations under π do not directly translate to violations in mappings of G'_1 to G'_2). Still, we show that things cannot be extremely bad.

Specifically, we call a cloud of G_2 **good** if at least $(t/2) + 1$ of its vertices are mapped to it (by π) from a single cloud of G_1. Letting 2ϵ denote the fraction of violations in Eq. (1) (i.e., the size of this set divided by $(tn)^2$), we first show that *at least $(1 - (6\epsilon/\alpha)) \cdot n$ of the clouds of G_2 are good.*

[1] Matsliah's proof refers to two 4-vertex graphs and their 2-factor blow-up. Specifically, let G be a 4-vertex graph that consists of a triangle and an isolated vertex, and H consists of a matching of size two, denoted $\{\{1,2\},\{3,4\}\}$. Then, the (absolute) distance between G and H is 3 edges (because at least two edges must be dropped from the triangle and one edge added to be incident the isolated vertex). On the other hand, it is not hard to see that the 2-factor blow-ups of G and H are at distance of at most $10 < 4 \cdot 3$ edges. For example, consider an mapping of the eight vertices, denoted $\{1', 1'', 2', 2'', 3', 3'', 4', 4''\}$, of the 2-factor blow-up of H to four clouds such that vertex i' is mapped to cloud i, whereas vertex $1''$ is mapped to the 1st cloud, vertex $2''$ is mapped to the 4th cloud, vertex $3''$ is mapped to the 2nd cloud, and vertex $4''$ is mapped to the 3rd cloud. Then, dropping the edges $\{3', 4''\}, \{3', 4'\}, \{3'', 4''\}$ and adding $12 - 5 = 7$ edges among the 1st, 2nd and 4th clouds, we obtain a 2-factor blow-up of G.

Assume, towards the contradiction, that G_2 contains more that $(6\epsilon/\alpha)\cdot n$ clouds that are not good. Considering any such a (non-good) cloud, we observe that it must contain at least $t/3$ disjoint pairs of vertices that originate in different clouds of G_1 (i.e., for each such pair (v,v') it holds that $\pi^{-1}(v)$ and $\pi^{-1}(v')$ belong to different clouds of G_1).[2] Recall that the edges in G_2 respect the cloud structure of G_2 (which in turn respects the edge relation of G_2'). But vertices that originate in different clouds of G_1 differ on at least $\alpha\cdot tn$ edges in G_1. Thus, every pair (v,v') (in this cloud of G_2) such that $\pi^{-1}(v)$ and $\pi^{-1}(v')$ belong to different clouds of G_1 contributes at least $\alpha\cdot tn$ violations to Eq. (1).[3] It follows that the set in Eq. (1) has size greater than

$$\frac{6\epsilon n}{\alpha}\cdot\frac{t}{3}\cdot\alpha tn \;=\; 2\epsilon\cdot(tn)^2$$

in contradiction to our hypothesis regarding π.

Having established that at least $(1-(6\epsilon/\alpha))\cdot n$ of the clouds of G_2 are good and recalling that a good cloud of G_2 contains a strict majority of vertices that originates from a single cloud of G_1, we consider the following bijection π' of the vertices of G_1 to the vertices of G_2: For each good cloud g of G_2 that contains a strict majority of vertices from cloud i of G_1, we map all vertices of the i^{th} cloud of G_1 to cloud g of G_2, and map all other vertices of G_1 arbitrarily.

The number of violations under π' is upper-bounded by four times the number of violations occuring under π between good clouds of G_2 (i.e., at most $4\cdot 2\epsilon\cdot(tn)^2$) plus at most $(6\epsilon/\alpha)\cdot tn\cdot tn$ violations created with the remaining $(6\epsilon/\alpha)\cdot n$ clouds. This holds, in particular, for a bijection π' that maps to each remaining cloud of G_2 vertices originating in a single cloud of G_1. This π', which maps complete clouds of G_1 to clouds of G_2, yields a mapping of G_1' to G_2' that has at most $(8\epsilon+(6\epsilon/\alpha))\cdot n^2$ violations. Recalling that G_1' is ϵ'-far from G_2', we conclude that $8\epsilon+(6\epsilon/\alpha)>2\epsilon'$, which implies $\epsilon>\alpha\epsilon'/7$. The claim follows (since ϵ is the minimal value such that G_1 is ϵ-close to G_2). □

Using Lemma 4.3, we are ready to establish the $\Omega(q)$ lower bound on the query complexity of testing Π.

Proposition 4.4 *Any tester for Π has query complexity $\Omega(q)$.*

Proof: Recall that Lemma 4.3 implies that if G' is ϵ'-far from Π', then its blow-up is $\Omega(\epsilon')$-far from Π. Using this fact, we conclude that ϵ'-testing of Π' reduces to $\Omega(\epsilon')$-testing of Π. Thus, a quadratic lower bound on the query complexity of ϵ'-testing Π_n' yields an $\Omega(n^2)$ lower bound on the query complexity of $\Omega(\epsilon')$-testing Π_N, where $n=\sqrt{q(N)}$. Hence, we obtain an $\Omega(q)$ lower bound on the query complexity of testing Π, for some constant value of the proximity parameter. □

4.3 An optimal tester for property Π

In this section we prove that the query complexity of testing Π is at most q (and that this can be met by a relatively efficient tester). We start by describing this (alleged) tester.

Algorithm 4.5 *On input N and proximity parameter ϵ, and when given oracle access to a graph $G=([N],E)$, the algorithm proceeds as follows:*

[2]This pairing is obtained by first clustering the vertices of the cloud of G_2 according to their origin in G_1. By the hypothesis, each cluster has size at most $t/2$. Next, observe that taking the union of some of these clusters yields a set containing between $t/3$ and $2t/3$ vertices. Finally, we pair vertices of this set with the remaining vertices. (A better bound of $\lfloor t/2\rfloor$ can be obtained by using the fact that a t-vertex graph of minimum degree $t/2$ contains a Hamiltonian cycle.)

[3]For each such pair (v,v'), there exist at least $\alpha\cdot tn$ vertices u such that exactly one of the (unordered) pairs $\{\pi^{-1}(u),\pi^{-1}(v)\}$ and $\{\pi^{-1}(u),\pi^{-1}(v')\}$ is an edge in G_1. Recalling that for every u, the pair $\{u,v\}$ is an edge in G_2 if and only if $\{u,v'\}$ is an edge in G_2, it follows that for at least $\alpha\cdot tn$ vertices u either $(\pi^{-1}(u),\pi^{-1}(v))$ or $(\pi^{-1}(u),\pi^{-1}(v'))$ is a violation.

1. *Setting* $\epsilon' \stackrel{\rm def}{=} \epsilon/3$ *and computing* $n \leftarrow \sqrt{q(N)}$.

2. *Finding* n *representative vertices; that is, vertices that reside in different alleged clouds, which corresponds to the* n *vertices of the original graph. This is done by first selecting* $s \stackrel{\rm def}{=} O(\log n)$ *random vertices, hereafter called the* signature vertices, *which will be used as a basis for clustering vertices* (according to their neighbors in the set of signature vertices). *Next, we select* $s' \stackrel{\rm def}{=} O(\epsilon^{-2} \cdot n \log n)$ *random vertices, probe all edges between these new vertices and the signature vertices, and cluster these* s' *vertices accordingly* (i.e., two vertices are placed in the same cluster if and only if they neighbor the same signature vertices). *If the number of clusters is different from* n*, then we reject. Furthermore, if the number of vertices that reside in each cluster is not* $(1 \pm \epsilon') \cdot s'/n$*, then we also reject. Otherwise, we select* (arbitrarily) *a vertex from each cluster, and proceed to the next step.*

3. *Note that the signature vertices* (selected in Step 2) *induce a clustering of all the vertices of* G*. Referring to this clustering, we check that the edges between the clusters are consistent with the edges between the representatives. Specifically, we select uniformly* $O(1/\epsilon)$ *vertex pairs, cluster the vertices in each pair according to the signature vertices, and check that their edge relation agrees with that of their corresponding representatives. That is, for each pair* (u, v)*, we first find the cluster to which each vertex belongs* (by making s queries per each vertex)*, determine the corresponding representatives, denoted* (r_u, r_v)*, and check* (by two queries) *whether* $\{u, v\} \in E$ *iff* $\{r_u, r_v\} \in E$*.* (Needless to say, if one of the newly selected vertices does not reside in any of the n existing clusters, then we reject.)

4. *Finally, using* $\binom{n}{2} < q(N)/2$ *queries, we determine the subgraph of* G *induced by the* n *representatives. We accept if and only if this induced subgraph is in* Π'.

Note that, for constant value of ϵ, the query complexity is dominated by Step 4, and is thus upper-bounded by $q(N)$. (In general, the query complexity is $o(q(N)/\epsilon^2) + (q(N)/2) = O(q(N)/\epsilon^2)$.) Furthermore, for constant ϵ, the above algorithm can be implemented in time $\text{poly}(n \cdot \log N) = \text{poly}(q(N) \cdot \log N)$. We comment that the Algorithm 4.5 is adaptive, and that a straightforward *non-adaptive* implementation of it has query complexity (that is dominated by) $\binom{s'}{2} = O(n \log n)^2 = \widetilde{O}(q(N))$.

Remark 4.6 *In fact, a* (non-adaptive) *tester of query complexity* $\widetilde{O}(q(N))$ *can be obtained by a simpler algorithm that selects a random set of* s' *vertices and accepts if and only if the induced subgraph is* ϵ'*-close to being a* (s'/n*-factor*) *blow-up of some graph in* Π'_n*. Specifically, we can cluster these* s' *vertices by using them also in the role of the signature vertices. Furthermore, these vertices* (*or part of them*) *can also be designated for use in Step 3. We note that the analysis of this simpler algorithm does not rely on the hypothesis that* Π' *is dispersed.*

We now turn to analyzing the performance of Algorithm 4.5. We note that the proof that this algorithm accepts, with very high probability, any graph in Π_N relies on the hypothesis that Π' is dispersed. (In contrast, the proof that Algorithm 4.5 rejects, with very high probability, any graph that is ϵ-far from Π_N does not rely on this hypothesis.)

Proposition 4.7 *Algorithm 4.5 constitutes a tester for* Π.

Proof: We first show that any graph in Π_N is accepted with very high probability. Suppose that $G \in \Pi_N$ is a N/n-factor blow-up of $G' \in \Pi'_n$. Relying on the fact that Π' is dispersed we note that, for every pair of vertices in $G' \in \Pi'_n$, with constant probability a random vertex has a different edge relation to the members of this pair. Therefore, with very high (constant) probability, a random set of $s = O(\log n)$ vertices yields n different neighborhood patterns for the n vertices of G'. It follows that, with the same high probability, the s signature vertices selected in Step 2 induced n (equal sized) clusters on the vertices of G, where each cluster contains the cloud of N/n vertices (of G) that replaces a single vertex of G'. Thus, with very high (constant) probability, the sample of $s' = O(\epsilon^{-2} \cdot n \log n)$ additional vertices selected in Step 2 hits each of these clusters (equiv., clouds) and furthermore has $(1 \pm \epsilon') \cdot s'/n$ hits in each cluster. We conclude that, with very high (constant) probability, Algorithm 4.5 does not reject G in Step 2. Finally, assuming that Step 2 does not reject (and we did obtain representatives from each cloud of G), Algorithm 4.5 never rejects $G \in \Pi$ in Steps 3 and 4.

We now turn to the case that G is ϵ-far from Π_N, where we need to show that G is rejected with high constant probability (say, with probability 2/3). We will actually prove that if G is accepted with sufficiently high constant probability (say, with probability 1/3), then it is ϵ-close to Π_N. We call a set of s vertices good if (when used as the set of signature vertices) it induces a clustering of the vertices of G such that n of these clusters are each of size $(1 \pm 2\epsilon') \cdot N/n$. Note that good s-vertex sets must exist, because otherwise Algorithm 4.5 rejects in Step 2 with probability at least $1 - \exp(\Omega(\epsilon^2/n) \cdot s') > 2/3$.

Fixing any good s-vertex set S, we call a sequence of n vertices $R = (r_1, ..., r_n)$ well-representing if (1) r_i resides in the i^{th} aforementioned cluster, (2) the subgraph of G induced by R is in Π'_n, and (3) when clustering the vertices of G according to S, at most ϵ' fraction of the vertex pairs of G have an edge relation that is inconsistent with the corresponding vertices in R. That is, condition (3) requires that at most ϵ' fraction of the vertex pairs in G violate the condition by which $\{u, v\} \in E$ if and only if $\{r_i, r_j\} \in E$, where u resides in the i^{th} cluster (w.r.t S) and v resides in the j^{th} cluster. Now, note that there must exist a good s-vertex set S that has a well-representing n-vertex sequence $R = (r_1, ..., r_n)$, because otherwise Algorithm 4.5 rejects with probability at least 2/3. (Specifically, if a ρ fraction of the s-vertex sets are good (but have no corresponding n-sequence that is well-representing), then Step 2 rejects with probability at least $(1 - \rho) \cdot 0.9$ and either Step 3 or Step 4 reject with probability $\rho \cdot \min((1 - (1 - \epsilon')^{\Omega(1/\epsilon)}), 1) > 0.9\rho$.)

Fixing any good s-vertex set S and any corresponding $R = (r_1, ..., r_n)$ that is well-representing, we consider the clustering induced by S, denoted $(C_1, ..., C_n, X)$, where X denotes the set of (untypical) vertices that do not belong to the n first clusters. Recall that, for every $i \in [n]$, it holds that $r_i \in C_i$ and $|C_i| = (1 \pm 2\epsilon') \cdot N/n$. Furthermore, denoting by $i(v)$ the index of the cluster to which vertex $v \in [N] \setminus X$ belongs, it holds that the number of pairs $\{u, v\}$ (from $[N] \setminus X$) that violate the condition $\{u, v\} \in E$ iff $\{r_{i(u)}, r_{i(v)}\} \in E$ is at most $\epsilon' \cdot \binom{N}{2}$. Now, observe that by modifying at most $\epsilon' \cdot \binom{N}{2}$ edges in G we can eliminate all the aforementioned violations, which means that we obtain n sets with edge relations that fit some graph in Π'_n (indeed the graph obtained as the subgraph of G induced by R, which was not modified). Recall that these sets are each of size $(1 \pm 2\epsilon') \cdot N/n$, and so we may need to move $2\epsilon' N$ vertices in order to obtain sets of size N/n. This movement may create up to $2\epsilon' N \cdot (N - 1)$ new violations, which can be eliminated by modifying at most $2\epsilon' \cdot \binom{N}{2}$ additional edges in G. Using $\epsilon = 3\epsilon'$, we conclude that G is ϵ-close to Π_N. The proposition follows. $\square$

Conclusion. We just showed that Algorithm 4.5 satisfies the upper bound requirements of Theorem 4; that is, it is a (relatively efficient) tester for Π and has query complexity $O(q)$. Recalling that Proposition 4.4 establishes a corresponding $\Omega(q)$ lower bound, we complete the proof of Theorem 4.

5 Revisiting the Adjacency Matrix Model: Monotone Properties

In continuation to Section 4, which provides a hierarchy theorem for generic graph properties (in the adjacency matrix model), we present in this section a hierarchy theorem for *monotone* graph properties (in the same model). We say that a graph property Π is monotone if adding edges to any graph that resides in Π yields a graph that also resides in Π. (That is, we actually refer to upward monotonicity, and an identical result for downward monotonicity follows by considering the complement graphs.)[4]

Theorem 5 *In the adjacency matrix model, for every $q : \mathbb{N} \to \mathbb{N}$ as in Theorem 4, there exists a* monotone *graph property Π that is testable in $O(q)$ queries, but is not testable in $o(q)$ queries.*

Note that Theorem 5 refers to two-sided error testing (just like Theorem 4). Theorems 4 and 5 are incomparable: the former provides graph properties that are in $\mathcal{P}$ (and the upper bound is established via relatively efficient testers), whereas the latter provides graph properties that are monotone.

Outline of the proof of Theorem 5. Starting with the proof of Theorem 4, one may want to apply a monotone closure to the graph property Π (presented in the proof of Theorem 4). (Indeed, this is the approach used in the proof of [GT, Thm. 1].) Under suitable tuning of parameters, this allows to retain the proof of the lower bound, but the problem is that the tester presented for the upper bound fails. The point is that this tester (i.e., Algorithm 4.5) relies on the structure of graphs obtained via blow-up, whereas this structure is not maintained by the monotone closure.

One possible solution, which assumes that all graphs in Π have approximately the same edge density, is to augment the monotone closure of Π with all graphs that have significantly larger edge density, where the corresponding threshold on the number of edges is denoted T. Intuitively, this way, we can afford accepting any graph that has more than T edges, and handle graphs with fewer edges by relying on the fact that in this case the blow-up structure is essentially maintained (because only few edges are added).

Unfortunately, implementing this idea is not straightforward: On the one hand, we should set the threshold high enough so that the lower bound proof still holds, whereas on the other hand such a setting may destroy the local structure of a constant fraction of the graph's vertices. The solution to this problem is to use an underlying property Π' that supports "error correction" (i.e., allows recovering the original structure even when a constant fraction of it is destroyed as above).

The proof of Theorem 5 copes with the aforementioned difficulties by a careful implementation of the stated ideas. In Section 5.1, we construct a monotone property Π by combining the blow-up operation with monotone (upward) closure and augmenting Π with all sufficiently dense graphs. In Section 5.2 we lower-bound the query complexity of Π by showing that for graphs of non-excessive maximal degree the distance to the property analyzed in Section 4.2 is linearly related to the distance to the monotone property Π. Finally, in Section 5.3, we upper-bound the query complexity of Π by analyzing the structure of the graphs in Π that are not too dense.

5.1 The monotone property Π

Our starting point is a graph property $\Pi' = \bigcup_{n\in\mathbb{N}} \Pi'_n$ for which testing requires quadratic query complexity. Furthermore, we assume that this property satisfies the additional conditions stated in the following claim.

Claim 5.1 *There exists a graph property $\Pi' = \bigcup_{n\in\mathbb{N}} \Pi'_n$ for which testing requires quadratic query complexity. Furthermore, for every constant $\delta > 0$ and all sufficiently large n, it holds that every graph $G' = ([n], E')$ in Π'_n satisfies the following two* (local) *conditions:*

1. *Every vertex has degree $(0.5 \pm \delta) \cdot n$; that is, for every $v \in [n]$ it holds that $\{u : \{v, u\} \in E'\}$ has size at least $(0.5 - \delta) \cdot n$ and at most $(0.5 + \delta) \cdot n$.*
2. *Every two different vertices neighbor at least $(0.75 - \delta) \cdot n$ vertices; that is, for every $v \neq w \in [n]$ it holds that $\{u : \{v, u\} \in E' \vee \{w, u\} \in E'\}$ has size at least $(0.75 - \delta) \cdot n$.*

[4] We stress that these notions of monotonicity are different from the notion of monotonicity considered in [AS], where a graph property Π is called monotone if any subgraph of a graph in Π is also in Π.

Moreover, pairs of graphs in Π'_n are related by the following two (global) *conditions:*

3. *Every two non-isomorphic graphs in Π'_n differ on at least $0.4 \cdot \binom{n}{2}$ vertex pairs; that is, if $G'_1, G'_2 \in \Pi'_n$ are not isomorphic, then G'_1 is 0.4-far from G'_2.*

4. *Graphs in Π'_n that are isomorphic via a mapping that fixes less than 90% of the vertices differ on at least $0.01 \cdot \binom{n}{2}$ vertex pairs; that is, if $G'_1, G'_2 \in \Pi'_n$ are isomorphic via π such that $|\{i \in [n] : \pi(i) \neq i\}| > 0.1n$, then G'_1 is 0.01-far from G'_2, where here we consider distance between labeled graphs* (or rather their adjacency matrices).

In addition, with probability $1 - o(1)$, a random n-vertex graph is 0.4-far from Π'_n.

Note that the graphs in Π' are $2 \cdot (0.25 - 2\delta)$-dispersed, because $|\Gamma(u) \setminus \Gamma(v)| = |\Gamma(u) \cup \Gamma(v)| - |\Gamma(v)| \geq (0.75 - \delta)n - (0.5 + \delta)n = (0.25 - 2\delta)n$.

Proof: The graph property presented in the proof of [GGR, Prop. 10.2.3.1] can be easily modified to satisfy the foregoing conditions. Recall that this property is obtained by selecting $K \stackrel{\text{def}}{=} \exp(\Theta(n^2))$ random graphs and considering the $n!$ isomorphic copies of each of these graphs. Note that each of the "basic" K graphs satisfies the two local conditions with probability at least $1 - n^2 \cdot \exp(-\Omega(\delta^2 n))$. Omitting the relatively few exceptional graphs (which violate either of these two conditions), we obtain a property that satisfies both local conditions and maintains the query-complexity lower bound. (Indeed, the query-complexity lower bound is not harmed, because it is established by considering the uniform distribution over the set of basic graphs (which hardly changes).)

Turning to the global conditions, which refer to the pairwise distances between graphs in Π'_n, we distinguish two cases. In the case that $G'_1, G'_2 \in \Pi'_n$ are not isomorphic, they arise from two independently selected basic graphs, and so with probability at least $1 - \exp(-\Omega(n^2)) > 1 - o(|\Pi'_n|^{-2})$ these two graphs are 0.4-far from one another. Applying the union bound (over all pairs in Π'_n), this establishes Condition 3.

It is left to consider pairs of graphs as in Condition 4 (i.e., graphs $G'_1, G'_2 \in \Pi'_n$ such that there exists an isomorphism π of G'_1 to G'_2 such that π fixes less than 90% of the vertices). Thus, we consider an arbitrary permutation π (over $[n]$) that fixes less than $0.9n$ of the domain (i.e., $|\{i \in [n] : \pi(i) \neq i\}| > 0.1n$). Next, we consider an arbitrary set $I \subset [n]$ such that $|I| = 0.05n$ and $\pi(I) = \{\pi(i) : i \in I\}$ does not intersect I. For a random n-vertex graph $G' = ([n], E')$, with probability at least $1 - \exp(-\Omega(n^2)) > 1 - o((n!^2 \cdot K)^{-1})$, the sets $\{(u,v) \in I \times ([n] \setminus (I \cup \pi(I)) : \{u,v\} \in E'\}$ and $\{(u,v) \in I \times ([n] \setminus (I \cup \pi(I)) : \{\pi(u), \pi(v)\} \in E'\}$ differ on at least $0.01n^2$ entries (since the expected difference is $0.05n \cdot 0.9n/2 > 0.02n^2$). Thus, the π-isomorphic copy of G' is 0.01-far from G' (where both are viewed as labeled graphs). Applying the union bound (over all the basic graphs and all choices of π and I), we establish Condition 4, and the claim follows. □

The monotone closure and augmentation (yielding Π). In the following description, we set $\Delta > 0$ to be a sufficiently small constant (e.g., smaller than 0.00001) such that the lower bound established in Theorem 4 holds for proximity parameter 100Δ (i.e., $\Delta \stackrel{\text{def}}{=} \epsilon_4/100$, where ϵ_4 is a value of the proximity parameter for which Theorem 4 holds). Needless to say, Π' satisfies the four conditions of Claim 5.1 also when we fix δ to equal Δ. Given a desired complexity bound $q : \mathbb{N} \to \mathbb{N}$, we set $n = \sqrt{q(N)}$ and define Π_N such that $G = ([N], E) \in \Pi_N$ if and only if (at least) one of the following two conditions holds:

(C1) The graph G has at least $(0.5 + 2\Delta) \cdot \binom{N}{2}$ edges.

(C2) Each vertex in G has degree at least $(0.5 - \Delta) \cdot N$ and G is an "approximate (monotone) blow-up" of some graph in Π_n; that is, there exists a partition of the vertex set of G (i.e., $[N]$) into n equal-sized sets, denoted $(V_1, ..., V_n)$, and a graph $G' = ([n], E') \in \Pi'_n$ such that for every $\{i,j\} \in E'$

and every $u \in V_i$ and $v \in V_j$ either $\{u,v\} \in E$ or the degree of either u or of v in G exceeds $0.52 \cdot N$.

Note that Condition (C2) mandates that each edge $\{i,j\} \in E'$ is replaced by a bipartite graph over $V_i \times V_j$ that contains all edges, with the possible exception of edges that are incident at vertices of degree exceeding $0.52 \cdot N$ (in G). We stress that Condition (C2) does *not* require that for $\{i,j\} \notin E'$ the bipartite graph over $V_i \times V_j$ is empty, but in the case that Condition (C1) does not hold these bipartite graphs will contain few edges (because the edges mandated by Condition (C2) leave room for few superfluous edges, when taking into account the upper bound on the number of edges that is implied by the violation of Condition (C1)).

Note that the property $\Pi = \bigcup_{N \in \mathbb{N}} \Pi_N$ is monotone (since Conditions (C1) and (C2) are each monotone). Also observe that Π_N contains the N/n-factor blow-up of any graph in Π'_n, because any such blow-up satisfies Condition (C2). (Indeed, such a blow-up does not satisfy Condition (C1), since each vertex in the blow-up has degree at most $(0.5 + \Delta) \cdot N$.)

On the constant Δ. Recall that Δ was fixed to be a small positive constant that is related to the constant hidden in Theorem 4 (i.e., the lower bound in this theorem should hold when the proximity parameter is set to any value that does not exceed 100Δ). In addition, we will assume that Δ is smaller than various specific constants (e.g., in the proof of Claim 5.2 we use $\Delta < 0.0001$). In general, setting $\Delta = 0.00001$ satisfies all these conditions. We also note that in our positive result (i.e., the analysis of the optimal tester) we will assume that the proximity parameter ϵ is significantly smaller than Δ (e.g., $\epsilon < \Delta/1000$).

5.2 Lower-bounding the query complexity of testing Π

In this section we prove that the query complexity of testing Π is $\Omega(q)$. We shall do this by building on [GGR, Prop. 10.2.3.1] and Section 4.2. Specifically, combining the approach of Section 4.2 with the analysis of [GGR, Prop. 10.2.3.1], we consider the following two distributions (D1) and (D2):

(D1) The distribution obtained by applying an $\frac{N}{n}$-factor blow-up to a random n-vertex graph (i.e., to a graph selected uniformly among all n-vertex graphs).

(D2) The distribution obtained by applying an $\frac{N}{n}$-factor blow-up to a graph selected uniformly in Π'_n, where Π'_n is as asserted in Claim 5.1 (with respect to $\delta = \Delta$).

Combining [GGR, Prop. 10.2.3.1] and Lemma 4.3, we claim that, *with high probability, a graph selected according to distribution* (D1) *is far* (i.e., 100Δ-far) *from the support of distribution* (D2), *whereas distinguishing the two distributions requires* $\Omega(q)$ *queries.* Specifically, we recall that, with high probability, a random n-vertex graph (as underlying distribution (D1)) is 0.4-far from any graph in Π'_n. By Lemma 4.3, the corresponding blow-ups preserve this distance (up to a constant factor), and thus (with high probability) a graph selected according to distribution (D1) is far from the support of distribution (D2). We also note that the proof of [GGR, Prop. 10.2.3.1] refers to these two underlying distributions on n-vertex graphs, and establishes that they are indistinguishable by $o(n^2)$ queries. It follows that the blow-up distributions (i.e., (D1) and (D2)) are indistinguishable by $o(q(N))$ queries.

Recalling that Π_N contains the support of distribution (D2), it suffices to show here that, with high probability, a graph selected according to distribution (D1) is far from Π_N (rather than merely far from the support of (D2)). This claim suffices, because by using it we obtain a distribution that is typically far from Π_N and yet is indistinguishable by $o(q(N))$ queries from a distribution on Π_N (indeed (D2) itself).

The claim that distribution (D1) is typically far from Π_N is proved by first observing that, with high probability, a graph selected in distribution (D1) has maximum degree smaller than $(0.5+\Delta) \cdot (N-1)$.

The proof is concluded by showing that if such a graph (i.e., of the foregoing degree bound) is 100Δ-far from the support of distribution (D2), then it is Δ-far from Π_N.

Claim 5.2 *Suppose that G has maximum degree smaller than $(0.5+\Delta)\cdot(N-1)$ and that G is Δ-close to Π_N. Then, G is 64Δ-close to the support of distribution* (D2).

Proof: Let C (standing for correct) be a graph in Π_N that is closest to G. Then, C has less than $\frac{N\cdot(0.5+\Delta)(N-1)}{2}+\Delta\cdot\binom{N}{2}=(0.5+2\Delta)\cdot\binom{N}{2}$ edges, and thus C must satisfy Condition (C2) in the definition of Π_N. Let $G'=([n],E')$ and $(V_1,...,V_n)$ be as required in Condition (C2), and let H denote the set of vertices that have degree greater than $0.52\cdot N$ in C.

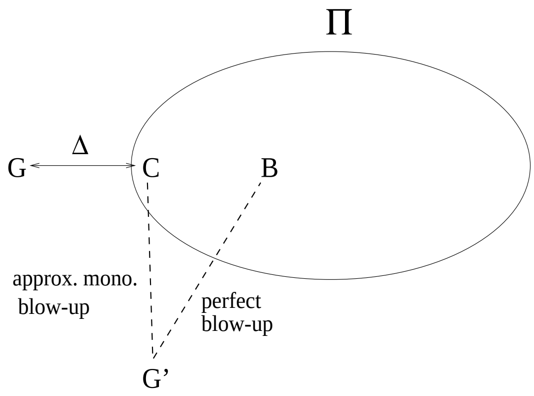

Figure 2: The graph G, its closest correction to Π, denoted C, and the corresponding perfect blow-up B.

Consider the distance between G and a blow-up of G', denoted B (standing for blow-up); see Figure 2. Each vertex in H contributes at most N units to this distance (between G and B), but its contribution to the distance between G and C is at least $0.52\cdot N-(0.5+\Delta)\cdot N>N/60$ (whereas the total distance between G and C is at most ΔN^2). Thus, the total contribution of vertices in H (to the distance between G and B) is less than $60\Delta N^2$. We stress that this count includes pairs of vertices that contain at least one element in H, and thus it remains to upper-bound the contribution of pairs that reside entirely within $[N]\setminus H$. We upper-bound the contribution of vertices in $[N]\setminus H$ (to the distance between G and B) by the sum of (1) their contribution to the distance between G and C (which is obviously upper-bounded by ΔN^2), and (2) their contribution to the distance between C and B.

In analyzing (2), we note that a pair $(u,v)\in([N]\backslash H)^2$ that is connected in B must also be connected in C, and so (2) counts the number of pairs $(u,v)\in([N]\setminus H)^2$ that are connected in C but not in B. Furthermore, the value of (2) equals the difference between the number of edges of the subgraph of B induced by $[N]\setminus H$ and the subgraph of C induced by $[N]\setminus H$. Recall that the average vertex degree of vertices in the graph C is at most $(0.5+\Delta)\cdot N+\Delta N=(0.5+2\Delta)\cdot N$, whereas in B vertices have degree at least $(0.5-\Delta)\cdot N$. If these averages were holding for the subgraphs induced by $[N]\setminus H$, then we could have upper bounded the value of (2) by $((0.5+2\Delta)\cdot N-(0.5-\Delta)\cdot N)\cdot(N-|H|)<3\Delta N^2$. Actually, as we shall see next, the gap between these averages may only increase when we move to the subgraphs induced by $[N]\setminus H$. Specifically, the number of edges with at least one endpoint in H is larger in C than it is in B, because the number of edges incident at any vertex of H is greater than $0.52\cdot N$ in C (by definition of H) and at most $(0.5+\Delta)\cdot N$ in B (by B's degree bound). Thus, the difference in the average degree between the subgraphs (of C and B) induced by $[N]\setminus H$ is at most $(0.5+2\Delta)\cdot N-(0.5-\Delta)\cdot N=3\Delta N$, and so the value of (2) is at most $3\Delta N^2$.

It follows that the total contribution (to both (1) and (2)) of vertices in $[N]\setminus H$ is at most $4\Delta N^2$. Hence, G is 64Δ-close to B, and the claim follows (because B is in the support of (D2)). $\square$

Proposition 5.3 *Any tester for* Π *has query complexity* $\Omega(q)$.

Proof: The claim follows by combing all facts stated above. Recall that, by [GGR, Prop. 10.2.3.1], distinguishing the distributions (D1) and (D2) requires $\Omega(q)$ queries. On the other hand, by [GGR, Prop. 10.2.3.1] and Lemma 4.3, with high probability, a graph selected according to distribution (D1) is 100Δ-far from the support of distribution (D2). With high probability, such a (random) graph has maximum degree smaller than $(0.5 + \Delta)(N - 1)$, and so by Claim 5.2 it is $(100\Delta/64)$-far from Π_N. Recalling that distribution (D2) resides on Π_N, it follows that any tester for Π must distinguish the distributions (D1) and (D2), and the proposition follows. $\square$

5.3 An optimal tester for property Π

In this section we prove that the query complexity of testing Π is $O(q)$. Before describing the (alleged) tester, we analyze the structure of graphs that satisfy Condition (C2) but do not satisfy Condition (C1). Denoting this set by $\Xi = \bigcup_{N\in\mathbb{N}} \Xi_N$, recall that Ξ_N contains N-vertex graphs that are in Π_N and have average degree below $(0.5 + 2\Delta) \cdot N$. Since these graphs have minimum degree at least $(0.5 - \Delta) \cdot N$, they may contain relatively few vertices of degree exceeding $0.52 \cdot N$ (i.e., the number of such vertices is at most $O(\Delta N)$). We call such vertices (i.e., of degree exceeding $0.52 \cdot N$) heavy. As we show next, the fact that almost all vertices in $G \in \Xi_N$ are not heavy implies that the edges among these non-heavy vertices (in any G) essentially determine a unique graph $G' \in \Pi'_n$ such that G is an approximate blow-up of G'. Moreover, this determines a unique partition of the non-heavy vertices of G to clouds that correspond to the vertices of G'. That is:

Lemma 5.4 *Let* $G = ([N], E) \in \Xi_N$ *and* H *denote the set of heavy vertices of* G (i.e., *vertices having degree that exceeds* $0.52 \cdot N$). *Then, up to a reordering of the indices in* $[n]$, *there exists a unique partition of* $[N] \setminus H$ *into* n *sets, denoted* $V'_1, ..., V'_n$, *and a unique graph* $G'' = (\{i \in [n] : V'_i \neq \emptyset\}, E'')$ *such that the following conditions hold:*

1. G'' *is an induced subgraph of some graph in* Π'_n (i.e., *there exists* $G' = ([n], E') \in \Pi'_n$ *such that* $\{i, j\} \in E''$ *if and only if* $V'_i \neq \emptyset$, $V'_j \neq \emptyset$ *and* $\{i, j\} \in E'$).
2. *For every* $\{i, j\} \in E''$ *and every* $u \in V'_i$ *and* $v \in V'_j$ *it holds that* $\{u, v\} \in E$.
3. *Vertices in the same* V'_i *differ on at most* $0.05N$ *of their neighborhoods, whereas vertices that reside in different* V'_i *differ on at least* $0.45N$ *neighbors.*
4. *Each* V'_i *has size at most* N/n, *and at most* $0.01n$ *sets* (i.e., V'_i's) *are empty.*

Furthermore, the claim holds even if G *has minimum degree that is only above* $(0.5 - 2\Delta) \cdot N$ (rather than above $(0.5 - \Delta) \cdot N$) *and its average degree is smaller than* $(0.5 + 3\Delta) \cdot N$ (rather than smaller than $(0.5 + 2\Delta) \cdot N$).

Proof: We shall focus on the main claim, and the furthermore part will follow by observing that the argument is actually insensitive to the value of Δ (as long as the latter is small enough).

The *mere existence* of a partition $(V'_1, ..., V'_n)$ and of a graph G'' that satisfies the foregoing four conditions follows from the fact that G satisfies Condition (C2). Specifically, let $(V_1, ..., V_n)$ and G' be as guaranteed by Condition (C2), and let $V'_i \stackrel{\text{def}}{=} V_i \setminus H$ for every $i \in [n]$. Then, $(V'_1, ..., V'_n)$ and the subgraph of G' that is induced by $\{i \in [n] : V'_i \neq \emptyset\}$ satisfy all the foregoing conditions. In particular, vertices in $[N] \setminus H$ have neighbors (in $[N] \setminus H$) as mandated by G', and may have at most $0.52N - ((0.5 - \Delta)N - |H|) < 0.021N$ additional neighbors. Thus, vertices in the same $V_i \setminus H$ may differ on at most $2 \cdot 0.021N + |H| < 0.05N$ of their neighbors, whereas vertices that reside in different $V_i \setminus H$'s must differ on at least $(0.5 - 4\Delta) \cdot N - (2 \cdot 0.021N + |H|) > 0.45N$ neighbors. Clearly, each V'_i has size at most N/n. Recalling that $|H| < 150\Delta N$ (since $|H| \cdot 0.52N + (N - |H|) \cdot (0.5 - \Delta)N < (0.5 + 2\Delta)N^2$), we conclude that at most $150\Delta \cdot n < 0.01n$ sets V_i' are empty.

Having established the existence of suitable objects, we now turn to establish their uniqueness; that is, we shall establish the uniqueness of both the partition of $[N] \setminus H$ and the graph G'', up to a reordering of the index set $[n]$.

We start by considering an arbitrary n-way partition of $[N] \setminus H$ that satisfies the four conditions of the claim. Referring to the foregoing partition $(V_1, ..., V_n)$, we show that two vertices $u, v \in [N] \setminus H$ can be placed in the same set of this n-wise partition (of $[N] \setminus H$) if and only if they reside in the same set V_i. This follows by the "clustering" condition asserted in Item 3 (since vertices in the same $V_i \setminus H$ may differ on at most $0.05N$ of their neighbors, whereas vertices that reside in different $V_i \setminus H$'s must differ on at least $0.45N$ neighbors). Thus, the partition of $[N] \setminus H$ is uniquely determined, up to a reordering of the index set $[n]$. Let us denote this partition by $(V_1', ..., V_n')$; indeed, the sequence $(V_1', ..., V_n')$ is a permutation of the sequence $(V_1 \setminus H, ..., V_n \setminus H)$, and here we arbitrarily fix such a permutation (ordering of $[n]$).

Note that so far we have only used the condition in Item 3, and this allowed us to uniquely determine the sequence $(V_1', ..., V_n')$ (up to reordering of $[n]$). Using the condition in Item 2, we show that this sequence uniquely determines the subgraph G'', which is an induced subgraph of some $G' \in \Pi_n'$.

Recall that, by Item 2, any unconnected pair of vertices $(u, v) \in V_i' \times V_j'$ mandates that the pair (i, j) cannot be connected in G'. Since there are at most $(0.5 + 2\Delta) \cdot \binom{N}{2}$ edges in G and at most $|H| \cdot N$ pairs that intersect H, we conclude that the number of unconnected pairs in $\bigcup_{i \neq j} V_i' \times V_j'$ is at least $(0.5 - 2\Delta) \cdot N^2 - |H| \cdot N \geq (0.5 - 152\Delta) \cdot N^2$. This forces at least $(0.5 - 152\Delta) \cdot n^2$ unconnected pairs in G'. Recalling that $G' \in \Pi_n'$ has average degree at most $(0.5 + \Delta) \cdot n$, this leaves us with slackness of at most $153\Delta \cdot n^2$ vertex pairs. Thus, any two graphs in Π_n' that satisfy Item 2, must be 153Δ-close. Recalling that non-isomorphic graphs in Π_n' are 0.4-far apart, this determines G' up to isomorphism. Actually, referring to the last condition in Claim 5.1, we conclude that G' is determined up to an isomorphism that fixes more than 90% of the vertices (because otherwise these graphs are 0.01-far). We shall show next that this (90%-fixing isomorphism) uniquely determines G''.

Suppose towards the contradiction that there exist two different graphs G_1'' and G_2'' that satisfy the conditions of the claim, and let i be a vertex in G_1'' that is mapped by the isomorphism to $j \neq i$ in G_2''. As we show next, this situation induces conflicting requirements on the neighbors of vertices in V_i' and V_j'; that is, it requires too many shared neighbors (when compared to the shared neighbors of i and j in G'). Specifically, by applying Item 2 to G_1'', the neighbors of each vertex in V_i' should contain all vertices in V_k' such that k is connected to i in G_1''. Similarly, by applying Item 2 to G_2'', the neighbors of each vertex in V_j' should contain all vertices in V_k' such that k is connected to j in G_2''. However, since the isomorphism fixes more than 90% of the vertices, it must be the case that for 90% of $k \in [n]$ it holds that i is connected to k in G_1'' iff j is connected to k in G_2''. It follows that each pair of vertices in $V_i' \times V_j'$ must share more than $((0.5 - O(\Delta)) \cdot n - 0.1n) \cdot (N/n) - |H| > 0.3N$ neighbors, which contradicts the postulate (regarding G' which implies) that each such pair can share at most $(2 \cdot (0.5 + \Delta) \cdot n - (0.75 - \Delta) \cdot n) \cdot (N/n) + |H| < 0.3N$ neighbors. The claim follows. □

Testing the property Π. Having established Lemma 5.4, we are now ready to present the (alleged) tester for Π. Intuitively, the tester first checks whether the input graph satisfies Condition (C1), and if the input is found to be $\Omega(\epsilon)$-far from satisfying Condition (C1) then it is tested for Condition (C2). Indeed, the core of this tester refers to the latter part (i.e., testing Ξ), and is obtained by suitable adaptations of Algorithm 4.5. In particular, since we cannot expect to identify representatives from all clouds (i.e., some sets V_i' in Lemma 5.4 may be too small or even empty), we settle for obtaining representatives from at least a $1 - \Omega(\epsilon)$ fraction of the identifiable clouds (which leads to using, as a basis, the simplified version of Algorithm 4.5 that is discussed in Remark 4.6).

Algorithm 5.5 *On input N and proximity parameter ϵ, and when given oracle access to a graph $G = ([N], E)$, the algorithm proceeds as follows, after setting $\epsilon' \stackrel{\text{def}}{=} \epsilon/10$ and $n \stackrel{\text{def}}{=} \sqrt{q(N)}$:*

1. Estimating the edge density of G. *Using a sample of $O(\epsilon^{-2})$ vertex pairs, we estimate the edge density of G and* accept *if this estimate exceeds $0.5 + 2\Delta - 2\epsilon'$. We proceed to the next steps only if the edge density of G is estimated to be less than $0.5 + 2\Delta - 2\epsilon'$, in which case we may assume that the edge density of G is actually less than $0.5 + 2\Delta - \epsilon'$.*

2. Estimating the minimum degree of G. *Using a sample of $\widetilde{O}(\epsilon^{-2})$ vertices, we estimate the minimum degree in G; that is, we pick $O(\epsilon^{-1})$ vertices and estimate their degrees using an auxiliary sample of $\widetilde{O}(\epsilon^{-2})$ vertices. If we find a vertex that we estimate to have degree less than $(0.5 - \Delta - \epsilon') \cdot N$, then we* reject. *We proceed to the next steps only if we failed to find such a vertex, in which case we may assume that all but at most $\epsilon' N$ vertices have degree exceeding $(0.5 - \Delta - 2\epsilon') \cdot N$.*

3. Finding representative vertices. *We start by selecting a sample, denoted S, of $s \stackrel{\text{def}}{=} O(\epsilon^{-2} n)$ random vertices, and estimating their individual degrees in G by their individual degrees in the subgraph induced by S. We let $S' \subseteq S$ denote the set of vertices for which the estimated degree is less than $(0.52 - \epsilon') \cdot N$. We proceed only if $|S'| > 0.99s$, and otherwise we halt and* reject.

   Next, we cluster the vertices in S' as follows. Probing all $\binom{|S'|}{2}$ possible edges between these vertices, we cluster these vertices such that each cluster contains vertices having neighbor sets that differ on at most $0.06s$ vertices in S'. Specifically, we associate to each vertex an $|S'|$-dimensional Boolean vector that indicates whether or not it neighbors each of the vertices in S', and consider the metric defined by Hamming distance between these vectors. Scanning the vertices of S', we put the current vertex in an existing cluster if it is 0.06-close to the center of this cluster, and open a new cluster with the current vertex as its center otherwise (i.e., *if the current vertex cannot be fit to any existing cluster*).

   If the number of clusters, denoted n', is greater than n, then we reject. *Otherwise, we select at random a representative from each cluster, and denote by r_i the representative of the i^{th} cluster.*

4. Determining an adequate subgraph of a graph in Π'_n. *Let $R = \{r_i : i \in [n']\}$ and let G_R denote the subgraph of G induced by R* (i.e., by the set of representatives selected above). *We try to determine a graph $G' \in \Pi'_n$ such that the subgraph of G' induced by $[n']$, denoted $G'' = ([n'], E'')$, is consistent with G_R in the sense that if $\{i, j\} \in E''$ then $\{r_i, r_j\} \in E$* (equiv., the pair (r_i, r_j) is connected by an edge in G_R). *If either such a graph G' does not exist or G'' is not uniquely determined, then we halt and* reject.

5. Testing the induced clustering of the vertices of G. *The set R suggests a clustering of the vertices of G according to their neighbors in the set R. Specifically, for any vertex $v \in [N]$ of degree at most $0.52 \cdot N$, we let $\iota(v) = i$ if v is 0.06-close to the representative r_i and is 0.4-far from all other representatives. Otherwise* (i.e., if no such i exists), *then $\iota(v) = \perp$. Referring to this clustering* (i.e., the clustering according to ι), *we check whether it is indeed adequate* (both with respect to size and fitting G'') *by taking a sample of $t = O(\epsilon^{-2} n \log(1/\epsilon))$ vertices.*

   In the following sub-steps we refer to estimates of the degrees of individual vertices that are obtained by an auxiliary sample of size $O(\epsilon^{-2} \log t)$.

   (a) *We check that all but at most an ϵ' fraction of the vertices that have degree at most $0.52 \cdot N$ are uniquely clustered and that each of these vertices resides in a cluster that has size at most $(1 + \epsilon')N/n$. That is, using an auxiliary sample of $O(\epsilon^{-2} n \log(1/\epsilon))$ vertices, we check that for each such vertex v that is estimated to have degree at most $(0.52 - \epsilon') \cdot N$, it holds* *that $\iota(v) \in [n']$, and that at least a $1-\epsilon'$ fraction of these vertices are clustered so that for every $i \in [n']$ at most $(1+\epsilon')/n$ fraction of the vertices v satisfy $\iota(v) = i$. We* reject *if either such an unclusterable vertex v is found or some cluster is too big.*

(b) *We check that the edges between the clusters are consistent with the edges between the corresponding vertices of G''. Specifically, we select uniformly $O(1/\epsilon)$ vertex pairs, cluster the vertices in each pair according to ι, and check that their edge relation agrees with that of their corresponding representatives in the sense that each vertex pair must be connected if the corresponding pair of ι-indices is connected in G''. That is, for each pair (u,v), we first estimate the degree of each vertex and proceed only if both estimates are below $(0.52-\epsilon')\cdot N$. Next, we find the cluster to which each vertex belongs, and* reject *if $\{\iota(u), \iota(v)\} \in E''$ holds but $\{u,v\} \notin E$.*

(The condition in Step 5b may be interpreted as not rejecting if either $\iota(u) = \perp$ or $\iota(v) = \perp$; but we can also reject in this case, as in Step 5a.)

We accept *if and only if none of the foregoing checks led to rejection.*

Note that, for constant value of ϵ, the query complexity is dominated by Step 3, which uses $\binom{|S'|}{2} = O(\epsilon^{-2}n)^2 = O(\epsilon^{-4}q(N))$ queries. (In contrast, the number of queries made in Step 5 is $(t+O(1/\epsilon)) \cdot (n' + O(\epsilon^{-2}\log t)) = O(\epsilon^{-4}n^2\log^2(1/\epsilon))$, while a better bound of $o(\epsilon^{-4}n^2)$ holds when $\epsilon \gg 1/\sqrt{n}$). We now turn to analyzing the performance of Algorithm 5.5.

Proposition 5.6 *Algorithm 5.5 constitutes a tester for* Π.

Proof: We first verify that any graph in Π_N is accepted with very high probability. Note first that if $G \in \Pi_N$ satisfies Condition (C1), then Step 1 accepts with very high probability. The same holds if G has average degree at least $(0.5+2\Delta-\epsilon')N$. Thus, we focus on the case that $G \in \Xi_N$, and furthermore assume that G has average degree less than $(0.5+2\Delta-\epsilon')N$. Needless to say, Step 2 is unlikely to reject G (because G has minimum degree at least $(0.5-\Delta)N$).

Regarding the sample S taken in Step 3, with very high probability, the degree of each sample vertex in G is approximated up-to an relative term of $\pm\epsilon'$ by this vertex's degree in the subgraph induced by S. The same holds with respect to the number of neighbors that each such vertex has in any n fixed sets, where we care about the sets V_i associated with $G \in \Xi_N$. Letting H, $(V_1',...,V_n')$ and G'' be as in Lemma 5.4, we note that with high probability the sample S' taken in Step 3 is clustered accordingly (i.e., the i^{th} cluster consists of $V_i' \cap S'$, where here we consider a possible reordering of the sequence of clusters). (Here we allow also empty clusters in order to obtain a sequence of n clusters.) Furthermore, the subgraph G'' fits the induced graph G_R in the sense that G'' satisfies the condition in Step 4 (i.e., if $\{i,j\}$ is an edge in G'' then $\{r_i, r_j\} \in E$). Moreover, with high probability, G'' passes the checks in Step 5b (because the auxiliary samples taken in Step 5 are also clustered according to the V_i's). Thus, Steps 3 and 5 are unlikely to reject G (because, with probability at least $1-\epsilon$, the i^{th} cluster is assigned a $(N^{-1}\cdot|V_i'| \pm \epsilon')/n$ fraction of the vertices sampled in Step 3 and in Step 5).

To show that Step 4 is also unlikely to reject G, we need to show that, with high probability, the graph G'' is the only adequate graph that fits the set R. The latter is proved by considering an (imaginary) set I selected at random such that I includes a single uniformly distributed element from each set V_i. Observe that, with high probability (i.e., probability at least $1-\exp(-\Omega(n))$) over the choice of I, the N/n-factor blow-up of the subgraph G_I is approximately in Ξ_N, and so applying (the furthermore part of) Lemma 5.4 to this blow-up of G_I guarantees the uniqueness of G'' (with respect to G_I). That is, G'' is the only n'-vertex graph that is an induced subgraph of some graph in Π_n' and fits G_I (i.e., if $\{i,j\}$ is an edge in G'' then either the corresponding edge is in G_I or at least one of its endpoints is in H). We now consider R as being derived from I by replacing each vertex in $V_i \cap H$ by a uniformly chosen vertex in $V_i' = V_i \setminus H$, and note that this replacement may only increase the set of constraints involved in the "fitting condition" (i.e., if some graph did not fit G_I, then it will also not fit the corresponding G_R).[5] It follows that G'' is the only graph that fits G_R, and so Step 4 is unlikely to reject G. We conclude that G is unlikely to be rejected by any step, and thus it is accepted (with high probability).

We now turn to the case that G is ϵ-far from Π, where we need to show that G is rejected with, say, probability 2/3. We will actually prove that if G is accepted with probability 1/3, then it is ϵ-close to Π_N. We may assume that G has average degree below $(0.5+2\Delta-\epsilon)N$, since otherwise the claim follows easily. Thus, with high probability, the graph G is not accepted by Step 1, and so we may use the fact that G is accepted by virtue of not violating the subsequent checks (of Steps 2–5). In particular, by virtue of Step 2, we may assume that at most $\epsilon' N$ vertices of G have degree below $(0.5 - \Delta - 2\epsilon')N$, which means that we can meet the degree lower bound (of Ξ) by adding at most $3\epsilon' N^2$ edges. Let S', $r_1, ..., r_{n'}$ and G'' be as determined in Steps 3 and 4. (Indeed, here we use the fact that G'' is uniquely determined by S, and that the same holds with respect to the distribution of $r_1, ..., r_{n'}$.) Then, by virtue of Step 5, we obtain a clustering of at least $(1 - \epsilon')N$ vertices that approximately fits the graph G'' in the sense that they reside in clusters that have each size at most $(1 + 2\epsilon')N/n$ and the number of missing edges between these clusters is at most $\epsilon' N^2$. By moving $m \stackrel{\rm def}{=} 3\epsilon' N$ vertices and adding at most $mN + \epsilon' N^2$ edges, we obtain a partition of the vertices into n equal sized sets that perfectly fit G'', and it follows that G is $(3 + 4) \cdot \epsilon'$-close to Ξ_N. $\square$

Conclusion. We just showed that Algorithm 5.5 satisfies the upper bound requirements of Theorem 5; that is, it is a tester for Π and has query complexity $O(q)$. Recalling that Proposition 5.3 establishes a corresponding $\Omega(q)$ lower bound, we complete the proof of Theorem 5.

6 Revisiting the Adjacency Matrix Model: One-Sided Error

In continuation to Section 4, which provides a hierarchy theorem for two-sided error testing of graph properties (in the adjacency matrix model), we present in this section a hierarchy theorem that refers to one-sided error testing. Actually, the lower bounds will hold also with respect to two-sided error, but the upper bounds will be established using a tester with one-sided error.

Theorem 6 *In the adjacency matrix model, for every $q : \mathbb{N} \to \mathbb{N}$ as in Theorem 4, there exists a graph property Π that is testable with one-sided error in $O(q)$ queries, but is not testable in $o(q)$ queries even when allowing two-sided error. Furthermore, Π is in $\mathcal{P}$.*

Theorems 4 and 6 are incomparable: in the former the upper bound is established via relatively efficient testers (of two-sided error), whereas in the latter the upper bound is established via one-sided error testers (which are not relatively efficient). (Unlike Theorem 5, both Theorems 4 and 6 do not provide monotone properties.)

Outline of the proof of Theorem 6. Starting with the proof of Theorem 4, we observe that the source of the two-sided error of the tester is in the need to approximate set sizes. This is unavoidable when considering graph properties that are blow-ups of some other graph properties, where blow-up is defined by replacing vertices of the original graph by *equal-size* clouds. The natural solution is to consider a *generalized* notion of blow-up in which each vertex is replaced by a (non-empty) cloud of arbitrary size. That is, G is a (generalized) blow-up of $G' = ([n], E')$ if the vertex set of G can be

[5]The key observation is that all constraints that refer to a vertex in H (present in I) are trivially satisfied. Thus, it does not matter which vertices are used to replace those in H, and it does not matter if we just omit these vertices (as in case that $V_i \setminus H = \emptyset$).

partitioned into n non-empty sets (of arbitrary sizes) that correspond to the n vertices of G' such that the edges between these sets represent the edges of G'; that is, if $\{i,j\}$ is an edge in G' (i.e., $\{i,j\} \in E'$), then there is a complete bipartite between the i^{th} set and the j^{th} set, and otherwise (i.e., $\{i,j\} \notin E'$) there are no edges between this pair of sets.

The proof of Theorem 6 builds on the proof of Theorem 4 (while deviating from it in some places). In Section 6.1, we construct Π based on Π' by applying the generalized graph blow-up operation. In Section 6.2 we lower-bound the query complexity of Π based on the query complexity of Π', while coping with the non-trivial question of how does the *generalized* (rather than the standard) blow-up operation affect distances between graphs. In Section 6.3 we upper-bound the query complexity of Π with respect to one-sided error testers.

6.1 The (generalized) blow-up property Π

Our starting point is any graph property $\Pi' = \bigcup_{n\in\mathbb{N}} \Pi'_n$ for which testing requires quadratic query complexity. Actually, we start with a graph property Π' for which distinguishing a random n-vertex graph from a graph uniformly distributed in Π'_n requires $\Omega(n^2)$ queries (cf. Section 7, which builds on [GGR]). Furthermore, we assume that Π' is in $\mathcal{P}$ (as in Section 4.1).

Given a desired complexity bound $q : \mathbb{N} \to \mathbb{N}$, we first set $n = \sqrt{q(N)}$, and define Π_N as the set of all N-vertex graphs that are (generalized) blow-ups of graphs in Π'_n; that is, the N-vertex graph G is in Π_N if and only if G is a (generalized) blow-up of some graph in Π'_n.

We note that, as in Section 4, if $\Pi' \in \mathcal{P}$ (and each vertex in each graph in Π' has distinct neighbor set), then $\Pi \in \mathcal{P}$. We comment that the latter implication relies on the fact that the definition of (generalized) blow-up requires that each vertex (of the original graph) is replaced by a *non-empty* cloud. For further discussion see Remark 6.5.

6.2 Lower-bounding the query complexity of testing Π

In this section we prove that the query complexity of testing Π is $\Omega(q)$. As in Section 4.2, the basic idea is reducing testing Π' to testing Π; that is, given a graph G' that we need to test for membership in Π'_n, we test its N/n-factor blow-up for membership in Π_N, where N is chosen such that $n = \sqrt{q(N)}$. (Needless to say, the N/n-factor blow-up of any graph in Π'_n results in a graph that is in Π_N.) Note that we still use the "balanced" blow-up operation in our reduction, although Π_N contains any generalized blow-up (of any graph in Π'_n). Indeed, this reduction relies on the assumption that the N/n-factor blow-up of any n-vertex graph that is far from Π'_n results in a graph that is far from Π_N (and not only from graphs obtained from Π'_n by a "balanced" blow-up).

Recall that in Section 4.2 we proved that for any $\epsilon' > 0$ there exists an $\epsilon > 0$ such that *the N/n-factor blow-up of any graph that is ϵ'-far from Π'_n is ϵ-far from the N/n-factor blow-up of any graph in Π'_n*. Here we show that, in the relevant (for us) case, the former graph is ϵ-far from Π_N (i.e., far from any *generalized* blow-up of any graph in Π'_n). Specifically, since the lower bound regarding testing Π' refers to distinguishing a random n-vertex graph from a graph that is uniformly distributed in Π'_n, it suffices to consider the case that G' is random. In particular, in this case, with high probability G' is dispersed.

Lemma 6.1 *There exists a universal constant $c > 0$ such that the following holds for every n, ϵ', α and* (unlabeled) *n-vertex graphs G'_1, G'_2. If G'_2 is α-dispersed and ϵ'-far from G'_1, then for any t any tn-vertex graph that is obtained by a generalized blow-up of G'_1 is $c\alpha^2 \cdot \epsilon'$-far from the t-factor blow-up of G'_2.*

Using Lemma 6.1 we conclude that *if G' is α-dispersed and ϵ'-far from Π', then its generalized blow-up is $\Omega(\epsilon')$-far from* Π. Specifically, applying Lemma 6.1 to $G'_2 = G'$ and every $G'_1 \in \Pi'$, we conclude that with high probability the N/n-factor blow-up of a random n-vertex graph G' is $\Omega(\epsilon')$-far from Π (because such a random graph is likely to be dispersed and ϵ'-far from Π').

We comment that the use of a balanced blow-up on one of the original graphs (in Lemma 6.1) is essential to the validity of the "approximate distance preservation" claim (of Lemma 6.1). In contrast, note that the generalized blow-ups of an n-vertex clique and an n-vertex independent set may be relatively close to one another. Similarly, the dispersity condition is also essential (e.g., a balanced blow-up of an n-vertex independent set is relatively close to some generalized blow-up of an n-vertex clique).

Proof: By Lemma 4.3, for a suitable constant c_1, it holds that the t-factor blow-up of G'_1 is $c_1\alpha \cdot \epsilon'$-far from the t-factor blow-up of G'_2, denoted G_2. Let G_1 be an arbitrary (generalized) blow-up of G'_1 (see Figure 3). We need to prove that G_1 is $c\alpha^2 \cdot \epsilon'$-far from G_2. We consider two cases regarding the amount of imbalance in the blow-up underlying G_1, where G_1 is called a δ-imbalanced blow-up of G'_1 if the variation distance between the relative densities of the various clouds in G_1 and the uniform sequence of densities is at most δ (i.e., $\sum_{i=1}^{n} |\rho_i - (1/n)| \leq 2\delta$, where ρ_i is the relative size of the i^{th} cloud in G_1). (Indeed, 0-imbalance corresponds to the case of a t-factor blow-up (of G'_1), and any generalized blow-up of G'_1 is 1-imbalanced.)

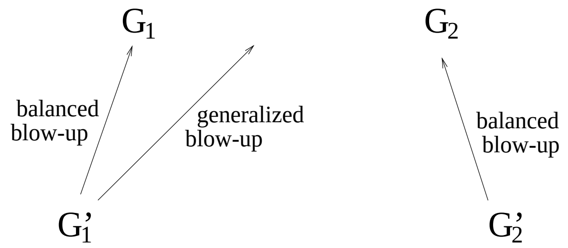

Figure 3: G'_1, G'_2, and their blow-ups.

Case 1: G_1 is a δ-imbalanced blow-up of G'_1, where $\delta = c_1\alpha\epsilon'/3$. In this case G_1 is $(c_1\alpha \cdot \epsilon' - 2\delta)$-far from G_2, because G_1 is 2δ-close to a t-factor blow-up of G'_1 (whereas the t-factor blow-up of G'_1 is $c_1\alpha \cdot \epsilon'$-far from G_2).

Note that, by the choice of δ, it holds that $c_1\alpha \cdot \epsilon' - 2\delta = \delta$.

Case 2: G_1 is not a δ-imbalanced blow-up of G'_1. In this case, using the fact that G_2 is a t-factor blow-up of an α-dispersed graph, we prove that G_1 is far from G_2.

Let ρ_i denote the relative size of the i^{th} cloud in G_1, and $I = \{i \in [n] : \rho_i > 1/n\}$ denote the set of clouds that are larger than in the uniform case. Then, $\sum_{i \in I} \rho_i > (|I|/n) + \delta$. We consider the most edge-fitting bijection π of the vertices of G_2 to the vertices of G_1, and lower-bound the number of vertex pairs that do not preserve the edge relation (i.e., pairs (u, v) such that $\{u, v\} \in E_2$ iff $\{\pi(u), \pi(v)\} \notin E_1$). Observe that, for each $i \in I$, the i^{th} cloud of G_1 must contain at least $(\rho_i - (1/n)) \cdot N/2$ pairs of vertices such that each pair consists of vertices that reside in different clouds of G_2 (because this cloud of G_1 contains at most N/n vertices that reside in the same cloud of G_2).[6] Each such pair contributes at least $\alpha \cdot N$ units to the (absolute) distance between G_1 and G_2 (because these vertices must have the same neighbors in G_1 whereas their neighborhoods in G_2 must have a disagreement rate of at least α). Thus, we lower-bound this

[6]The lower bound on the number of pairs follows from the following claim: *For $b \leq t$, if an urn contains t balls such that at most b balls have the same color, then the urn must contain at least $(t - b)/2$ disjoint pairs of mixed-colored balls.* The claim is proved by induction on t. In the induction step (assuming $b < t$), we take an arbitrary pair of balls with different colors, and are left with $t - 2$ balls (and at most b have the same color).

(absolute) distance by

$$\sum_{i\in I} \frac{(\rho_i - (1/n))N}{2} \cdot \alpha N \;=\; \frac{\delta \cdot \alpha}{2} \cdot N^2$$

and it follows that G_1 is $(\delta\alpha/2)$-far from G_2.

The claim follows by setting $c = c_1/6$ and noting that $\min(\delta, \delta\alpha/2) = c_1\alpha^2\epsilon'/6$. $\square$

Proposition 6.2 *Any tester for* Π *has query complexity* $\Omega(q)$.

Proof: Recall that Lemma 6.1 implies that if G' is dispersed and ϵ'-far from Π', then its (balanced) blow-up is $\Omega(\epsilon')$-far from Π. Since a random n-vertex graph G' is very likely to be dispersed, we infer that the distance of the (balanced) blow-up of G' from Π is linearly related to the distance of G' from Π'. Using this fact, we conclude that distinguishing graphs that are uniformly distributed in Π'_n from random n-vertex graphs reduces to testing membership in Π_N, where $n = \sqrt{q(N)}$. Specifically, if $G' \in \Pi'_n$ then its (balanced) blow-up is in Π_N, whereas if G' is ϵ-far from Π'_n (as is likely to be the case when it is a random n-vertex graph) then its (balanced) blow-up is $\Omega(\epsilon)$-far from Π_N. Thus, a quadratic lower bound on the query complexity of the distinguishing task regarding Π' implies an $\Omega(q)$ lower bound on the query complexity of testing Π, for some constant value of the proximity parameter. $\square$

6.3 An optimal tester for property Π

In this section we prove that the query complexity of testing Π is at most $O(q)$ and that this can be met by a one-sided error tester. In fact, essentially, we will use a straightforward tester, which selects uniformly a sample of $O(\sqrt{q})$ vertices and accepts if and only if the induced subgraph is a (generalized) blow-up of some graph in Π'. Actually, since some clouds of the tested graph may not be represented in the sample, we shall use a relaxed version of (generalized) blow-up that allows empty clouds. Equivalently, we shall check whether the induced subgraph is a (generalized) blow-up of an induced subgraph of some graph in Π'.

Algorithm 6.3 *On input* N *and proximity parameter* ϵ*, and when given oracle access to a graph* $G = ([N], E)$*, the algorithm proceeds as follows:*

1. *The algorithm sets* $\epsilon' \stackrel{\text{def}}{=} \epsilon/3$ *and computes* $n \leftarrow \sqrt{q(N)}$.
2. *The algorithm selects uniformly a set of* $O(n/\epsilon)$ *vertices, denoted* S*, and inspects the subgraph of* G *induced by* S*; that is, for every* $u, v \in S$*, the algorithm checks whether* $\{u, v\} \in E$.
3. *The algorithm accepts* G *if and only if the subgraph viewed in Step 2 is a generalized blow-up of some induced subgraph of some graph in* Π'_n.

We stress that Step 3 does not require the subgraph viewed in Step 2 to be a generalized blow-up of some graph $G' \in \Pi'_n$, but rather allows the former graph to be a generalized blow-up of any induced subgraph of such G'. In other words, Step 3 refers to the following relaxation of the notion of a generalized blow-up: the graph G is a relaxed blow-up of G' if the vertex set of G can be partitioned into sets (of arbitrary sizes) that correspond to vertices of G' such that the edges between these sets represent the edges of G'. We stress that some of these sets may be empty (and, needless to say, in such a case no edges are incident at these empty sets).

The query complexity of Algorithm 6.3 is $\binom{O(n/\epsilon)}{2} = O(q(N)/\epsilon^2)$. Note that this algorithm may not be relatively efficient, since we do not know of an efficient implementation of Step 3 (even if $\Pi' \in \mathcal{P}$; see Remark 6.5). Clearly, Algorithm 6.3 accepts any graph in Π with probability 1, because being a relaxed blow-up of any graph G' is hereditary (i.e., if G is a relaxed blow-up of G', then any induced subgraph of G is a relaxed blow-up of G'). It is left to show that Algorithm 6.3 rejects with probability 2/3 any graph that is ϵ-far from Π.

Let G be an arbitrary N-vertex graph that is ϵ-far from Π_N, and let us consider the sample S as being drawn in $2n$ iterations such that at each iteration $O(1/\epsilon)$ random vertices are selected. We denote by S_i the sample taken in iteration i, and by G_i the subgraph of G that is induced by $S^{(i)} \stackrel{\rm def}{=} \bigcup_{j\in[i]} S_j$. We refer to the clustering of the vertices of G_i according to their neighbor sets such that two vertices are in the same cluster if and only if they have exactly the same set of neighbors. We shall show (see the following Claim 6.4) that in each iteration, with high constant probability, either the number of clusters increases or we obtain a subgraph that is not a relaxed blow-up of any graph in Π'_n. It follows that, with overwhelmingly high probability, after $2n$ iterations we obtain a subgraph that is not a relaxed blow-up of any graph in Π'_n.

Claim 6.4 *Let G be an arbitrary N-vertex graph that is ϵ-far from Π_N, and $G_{S'}$ be the subgraph of G induced by S'. Let m denote the number of clusters in $G_{S'}$ and suppose that $m \leq n$. Further suppose that $G_{S'}$ is a relaxed blow-up of some graph in Π'_n. Then, for a randomly selected pair of vertices $u, v \in [N]$, with probability $\Omega(\epsilon)$, the number of clusters in the subgraph induced by $S' \cup \{u, v\}$ is greater than m.*

Note that if $G_{S'}$ is not a relaxed blow-up of any graph in Π'_n, then neither is the subgraph induced by $S' \cup \{u, v\}$. On the other hand, by Claim 6.4, if $G_{S'}$ is a relaxed blow-up of some graph in Π'_n and we augment S' with $O(1/\epsilon)$ random vertices, then, with probability at least 2/3, the number of clusters in the resulting induced subgraph is greater than m. Finally, note that if the number of clusters in a graph (e.g., G_S) is greater than n, then this graph cannot be a relaxed blow-up of any n-vertex graph (e.g., any graph in Π'_n). It follows that, with overwhelmingly high probability, the induced subgraph G_S is not a relaxed blow-up of any graph in Π'_n.

Proof: By the hypothesis regarding $G_{S'}$, there exists $G' \in \Pi'_n$ such that $G_{S'}$ is a relaxed blow-up of G'. We consider a partition of the vertex set of $G_{S'}$ to clouds that correspond to vertices of G' and denote by C_v the cloud that corresponds to vertex v. Clearly, the vertices in each cloud must belong to the same cluster, because otherwise the (relaxed) blow-up condition is violated. Thus, the clouds are a refinement of the partition of the vertex set of $G_{S'}$ into clusters. On the other hand, without loss of generality, all the vertices of each cluster may belong to a single cloud, because if C_v and C_w are clouds of the same cluster then we can move vertices of C_w to C_v while maintaining clouds that correspond to vertices of G'. We conclude that, without loss of generality, the collection of m clusters equals the collection of non-empty clouds of $G_{S'}$, which correspond to an induced subgraph of G', denoted $G'' = (V'', E'')$. Without loss of generality, we assume that $V'' = [m]$.

We now consider a clustering of the vertices of the entire graph G according to their neighbors in the set S'; that is, we cluster the vertices of G according to their S'-neighborhood, where the S'-neighborhood of v equals $\Gamma_{S'}(v) \stackrel{\rm def}{=} \{w \in S' : \{v, w\} \in E\}$. Note that some of these clusters extend the foregoing C_v's, whereas the other clusters, called new, contain vertices that have S'-neighborhood that are different from the S'-neighborhoods of all vertices in S'. If the number of vertices that are placed in new clusters exceeds $\epsilon N/4$, then such a vertex is selected with probability at least $\epsilon/4$ and the claim follows immediately. Otherwise, we consider an m-way partition, denoted $(V_1, ..., V_m)$, of the vertices that have the same S'-neighborhood as some vertices of S' such that

$$V_i = \{v : (\forall u \in C_i)\ \Gamma_{S'}(v) = \Gamma_{S'}(u)\}.$$

By the hypothesis that G is ϵ-far from Π_N and $\left|\bigcup_{i\in[m]} V_i\right| \geq (1 - (\epsilon/4)) \cdot N$ (and $n/N < \epsilon/4$), it must be the case that $(\epsilon/2) \cdot N^2$ vertex pairs in $\bigcup_{i,j\in[m]} V_i \times V_j$ have edge relations that are inconsistent with

G'' (i.e., for such a pair $(u,v) \in V_i \times V_j$ it holds that $\{u,v\} \in E$ iff $\{i,j\} \notin E''$).[7] Hence, these pairs have edge relations that are inconsistent with the edges between the corresponding C_i's (because the vertices in $\bigcup_{i,j\in[m]} C_i \times C_j$ have edge relations that are consistent with G''). Thus, with probability at least $\epsilon/2$, for a random pair of vertices $\{u,v\}$ the edge relation between u and v does not fit the edge relation between C_i and C_j, where i and j are the designated clouds of u and v (i.e., $u \in V_i$ and $v \in V_j$). It follows that the $\{u\} \cup C_i$ (and $\{v\} \cup C_j$) must be split when clustering the vertex set $S' \cup \{u,v\}$ according to the neighborhoods in $S' \cup \{u,v\}$. Thus, the claim follows also in this case. □

Remark 6.5 *Recall that the property Π was obtained by a generalized blow-up of Π', whereas Step 3 in Algorithm 6.3 refers to relaxed blow-ups of Π'. Denoting the set of relaxed blow-ups of Π' by $\widehat{\Pi}$, we note that $\Pi' \in \mathcal{P}$ implies $\widehat{\Pi} \in \mathcal{NP}$, but it is not clear whether $\widehat{\Pi} \in \mathcal{P}$ even when $\Pi' \in \mathcal{P}$. In fact, for some $\Pi' \in \mathcal{P}$, deciding membership in the corresponding $\widehat{\Pi}$ is NP-complete.*[8]

Conclusion. We just showed that Algorithm 6.3 satisfies the upper bound requirements of Theorem 6; that is, it is a one-sided error tester for Π and has query complexity $O(q)$. Recalling that Proposition 6.2 establishes a corresponding $\Omega(q)$ lower bound (also for two-sided testing), we complete the proof of Theorem 6.

7 Hard-to-test Graph Properties in P

In this section we strengthen the hardness results of [GGR] that refer to the existence of properties that are hard to test. These properties were shown to be in $\mathcal{NP}$. Here we modify the constructions in order to obtain such properties in $\mathcal{P}$. The aforementioned results refer both to the model of generic functions and to the model of testing graph properties in the adjacency matrix model (a.k.a dense model).

Let us first comment on the reasons that the original properties were only known to be in $\mathcal{NP}$ (rather than in $\mathcal{P}$).[9] In the first case (i.e., the case of generic functions), the reason is the complexity of recognizing possible outputs of an adequate pseudorandom generator (which becomes easy when given an adequate seed as an NP-witness). In the second case (i.e., the case of graph properties), an additional reason stems from the fact that "closure under isomorphism" is applied to the basic construction, and so the problem of recognizing graphs that are isomorphic to graphs in a particular set arises (and becomes easy when given an adequate isomorphism as an NP-witness). Below, we shall avoid the use of NP-witnesses by augmenting the basic construction in adequate ways.

We comment that the additional monotone closure used in [GT, Sec. 3] (in order to obtain monotone graph properties) introduces additional difficulties, which we were not able to resolve (and thus the

[7] Otherwise, we can obtain an m-way partition that is consistent with G'' by changing the edge relation of at most ϵN^2 vertex pairs (i.e., at most $(\epsilon/2) \cdot N^2$ vertex pairs in $\bigcup_{i,j\in[m]} V_i \times V_j$ and at most all pairs with one element not in $\bigcup_{i\in[m]} V_i$). Similarly, we can obtain an n-way partition that is consistent with G' (by creating $n-m$ new singleton clusters and using $n-m < \epsilon N/4$). This violates the hypothesis that G is ϵ-far from Π_N.

[8] For any NP witness relation R, we show how to reduce membership in $S_R \stackrel{\rm def}{=} \{x : \exists w\, (x,w) \in R\}$ to testing whether a constructed graph is an induced subgraph (or a relaxed blow-up) of some graph in an adequate set Π'. We use the graphs in Π' to encode pairs in R, and use the constructed graph to encode an input x that we need to check for membership in S_R. Each graph in Π'_n will correspond to a pair $(x,w) \in \{0,1\}^{n+m}$ such that the graph will consist of (1) a clique of $2(n+m)$ vertices, (2) a sequence of $n+m$ pairs of vertices such that the $i^{\rm th}$ pair is connected iff the $i^{\rm th}$ bit in xw equals 1, and (3) edges connecting the $i^{\rm th}$ vertex in Part (2) to the i first vertices of the clique. On input $x \in \{0,1\}^n$, we construct a $(2(n+m)+2n)$-vertex graph G_x essentially as above, except that we do not include the m last pairs of Part (2). (Indeed, given x, we cannot construct the corresponding m pairs, since we don't know w.) Note that G_x is an induced subgraph (or a relaxed blow-up) of some graph in Π'_n if and only if $x \in S_R$.

[9] The current description is intended for readers who have some recall of the aforementioned result. A self-contained description follows.

graph properties that we obtain in this section are not monotone). Furthermore, our techniques seem incompatible with monotonicity. The result we prove is stated next.

Theorem 7 *There exists a graph property in $\mathcal{P}$ for which, in the adjacency matrix model, every tester must query a constant fraction of the representation of the graph* (even when invoked with constant proximity parameter).

Background: the GGR construction and two difficulties. The graph property for which a quadratic query complexity lower bound is proved in [GGR, Prop. 10.2.3.2] is defined in two steps.

1. First, it is shown that certain sample spaces yield a collection of Boolean functions (i.e., a property of Boolean functions) that is hard to test (i.e., any tester must inspect at least a constant fraction of the function's values).

   On the one hand, the sample space is relatively sparse (and thus a random function is far from any function in the resulting collection), but on the other hand it enjoys a strong pseudorandom feature (and so its projection on any constant fraction of the coordinates looks random). Thus, the functions in the class (which must be accepted with high probability) look random to any tester that inspect only a small constant fraction of the function's values, whereas random functions are far from the class (and should be rejected with high probability). This yields a contradiction to the existence of a tester that inspect only a small constant fraction of the function's values.

2. Next, the domain of the functions is associated with the set of unordered pairs of elements in $[N]$, and the collection of functions is "closed" under graph isomorphism (i.e., if a certain function on $\binom{[N]}{2}$ is in the collection, then so is any function obtained from it by a relabeling of the elements of $[N]$).

   The closure operation makes this collection correspond to a graph property (since it is now preserved under isomorphism). The parameters are such that the resulting collection (although likely to be $N!$ times bigger than the original one) is still sparse enough (and so a random graph is far from it). On the other hand, the indistinguishability feature is maintained.

The two difficulties discussed above correspond to these two steps. Firstly, while the (support of the) sample space used in the proof of [GGR, Prop. 10.2.3.2] is in $\mathcal{NP}$, it is not clear whether it is in $\mathcal{P}$. Secondly, while NP-witnesses can be provided to prove that a given graph is isomorphic to a graph obtained in Step 1, it is not clear how to efficiently verify such a claim without an NP-witness.

Resolving the two difficulties (overview). The first difficulty is resolved by using an adequate pseudorandom generator for which membership in the corresponding sample space can be decided in polynomial time. Specifically, we shall use an adequate $\Omega(n)$-wise independence generator of n-bit long sequences rather than using a quite generic small-biased sample space as done in the proof of [GGR, Prop. 10.2.3.2]. (We mention that an alternative construction may be based on a *specific* small-biased generator; specifically, on the first small-biased generator of [AGHP] (i.e., the one based on LFSR sequences).)

The second difficulty is resolved by augmenting the graphs (constructed in Step 1) in a way that makes the original graph easy to recover from any relabeling of the resulting graph. Thus, applying Step 2 to these augmented graphs yields a class of graphs that is easy to recognize (by first recovering the original graph and then checking whether it corresponds to a string in the sample space).

The actual construction. For every N, we start by considering an efficiently constructible d-wise independent sample space over n-bit long strings, where $n \stackrel{\text{def}}{=} \binom{N}{2}$ and $d \stackrel{\text{def}}{=} \Omega(n)$. Specifically, for some constant $\delta > 0$, we use an explicitly constructible linear code mapping $0.01n$-bit long strings to n-bit strings such that every δn positions in a generic codeword are linearly independent (see [ABI]). Such a code is constructed by constructing a party-check matrix that spans a $0.99n$-dimensional vector space (called the "dual code") in which each vector has Hamming weight at least δn. We will use the parity-check matrix of the (primary) code in order to check membership in this code.

For each sequence $s = (s_1, ..., s_n) \in \{0,1\}^n$, we define a graph $G_s = ([N], E_s)$ by letting $\{i,j\} \in E_s$ if and only if the $(i,j)^{\text{th}}$ bit of s equals 1, where we consider any fixed (efficiently computable) order of the elements in $\{(i,j) : 1 \le i < j \le N\}$. We call the graph G_s good if s is in the aforementioned sample space, and call it bad otherwise. We refer to each such graph as basic; that is, the set of basic graphs includes all good and bad graphs (and indeed includes all N-vertex graphs). We highlight the fact that the set of good graphs is recognizable in polynomial-time, because the support of the aforementioned sample space is recognizable in polynomial-time (and the set of all N-vertex graphs is in 1-1 correspondence to the set of all n-bit strings).

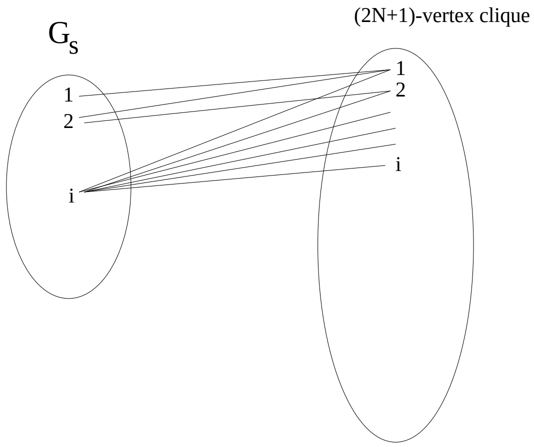

Figure 4: From G_s to G'_s.

Note that the set of good graphs is not likely to be closed under isomorphism, and thus this collection does not constitute a graph property. Following [GGR], we wish to consider the "closure" of the set of good graphs under isomorphism, but before applying this operation we augment the graphs in a way that makes it easy to reconstruct their original labeling. Specifically, for each graph $G_s = ([N], E_s)$, we consider the augmented graph $G'_s = ([3N+1], E'_s)$ obtained by adding a clique of size $2N+1$ to G_s and connecting the i^{th} vertex of G_s to the first i vertices in the clique; that is,

$$E'_s = E_s \cup \{\{u,v\} : u,v \in \{N+1, ..., 3N+1\}\} \cup \{\{i, N+j\} : i \in [N] \wedge j \in [i]\}. \tag{2}$$

(See Figure 4.) Now, we consider the set of final graphs obtained by "closing" the set of augmented graphs under isomorphism. That is, for every s in the sample space (equiv., an augmented graph G'_s obtained from a good graph G_s) and every permutation π over $[3N+1]$, we consider the final graph $G'_{s,\pi} = ([3N+1], E_{s,\pi})$ that is defined so that $\{\pi(u), \pi(v)\} \in E_{s,\pi}$ iff $\{u,v\} \in E'_s$. By construction, the set of final graphs is closed under isomorphism, and so this collection does constitute a graph property. Furthermore, as is shown next, the augmentation guarantees that the set of final graphs is in $\mathcal{P}$.

To test whether a graph $G = ([3N+1], E)$ is in the set of final graphs, we first attempt to reconstruct the corresponding basic graph. We use the fact that given a final graph it is easy to determine which vertex belongs to the basic graph (since these vertices have degree at most $(N-1)+N = 2N-1$, whereas each clique vertex has degree at least $2N$). Next, we determine the label of each vertex in the basic graphs by counting the number of its neighbors in the clique. (Needless to say, if this reconstruction fails, then G is not a final graph and we just reject it.) Finally, we check whether the resulting basic graph belongs to the set of good graphs (and whether the rest of the graph indeed fits the augmentation procedure).

Showing that the final graphs are hard to test. Our aim is to show that the property of being a final $(3N+1)$-vertex graph cannot be tested using $o(N^2)$ queries. We shall prove this claim by presenting two distributions on $(3N+1)$-vertex graphs such that a tester of final graphs must distinguish these two distributions whereas no machine that makes $o(N^2)$ queries can distinguish these two distributions. The first distribution is confined to final graphs, whereas with high probability graphs in the second distribution are 0.01-far from any final graph. Specifically, the first distribution, denoted G_N, is obtained by uniformly selecting a good N-vertex graph and augmenting it to an $(3N+1)$-graph (as done above). The second distribution, denoted R_N, is obtained by uniformly selecting a N-vertex graph and augmenting it to a $(3N+1)$-graph (again, as done above, except that here we apply this augmentation to all graphs). Throughout the rest of the argument, we associate the two distributions with random graphs drawn from them. We shall first show that, with high probability, R_N is 0.01-far from the set of final graphs.

Claim 7.1 *The probability that R_N is 0.01-close to some final $(3N+1)$-vertex graph is $o(1)$.*

Proof: The key observation is that the set of final graphs is very sparse. Specifically, each good graph gives rise to at most $(3N+1)!$ final graphs, whereas the number of good graphs is $2^{0.01n} = 2^{0.01\cdot\binom{N}{2}}$. Thus, the number of final graphs is at most $2^{(0.01+o(1))\cdot\binom{N}{2}}$. Each such graph is 0.01-close to at most $2^{H_2(0.01)\cdot\binom{3N+1}{2}} \ll 2^{0.1\cdot\binom{3N+1}{2}} \ll 2^{(0.9+o(1))\cdot\binom{N}{2}}$ graphs, where H_2 denotes the binary entropy function (and $H(0.01) < 0.1$). Thus (for all sufficiently large N), the total number of graphs that are 0.01-close to the set of final graphs is smaller than $2^{0.92\cdot\binom{N}{2}}$. Since R_N is uniformly distributed on a set of $2^{\binom{N}{2}}$ graphs, the claim follows. □

Next, we show that $o(N^2)$ queries do not allow distinguishing R_N from G_N.

Claim 7.2 *Let M be a probabilistic oracle machine that makes at most $d = \delta n \approx \delta N^2/2$ queries. Then,* $\Pr[M^{R_N}(N) = 1] = \Pr[M^{G_N}(N) = 1]$.

Proof: Since both distributions are obtained by applying the same fixed augmentation to some preliminary distributions, it suffices to consider queries to the preliminary distributions. Specifically, let us denote by G'_N the uniform distribution over good N-vertex graphs, and let R'_N denote the uniform distribution over all N-vertex graphs. Indeed, G_N (resp., R_N) is obtained by applying the (fixed) augmentation of Eq. (2) to G'_N (resp., R'_N), and each query to G_N (resp., R_N) can be answered either by using a constant value or by making a single query to the corresponding G'_N (resp., R'_N). Thus, it suffices to show that a machine that makes at most d queries cannot distinguish R'_N from G'_N.

We identify $\binom{N}{2}$-bit long strings with N-vertex graphs (obtained as in the first stage of the construction). Recall that G'_N denote a graph uniformly selected among all graphs in the sample space; that is, it corresponds to a d-wise independent sequence of length $n = \binom{N}{2}$. So the claim reduces to asserting that using d queries one cannot distinguish between a d-wise independent sequence and a uniformly distributed sequence, which follows easily from the definition of d-wise independent sample spaces (since in such cases adaptive queries offer no advantage). □

Theorem 7 follows by combining Claims 7.1 and 7.2 (with the fact that the set of final graphs is in $\mathcal{P}$).

8 Concluding Comments

Figure 5 provides a bird's eye view of the various hierarchy theorems proved in this work, where bd-graphs denote bounded-degree graphs. The third column specifies the type of error made by the tester that establishes the upper bound; recall that in all cases the lower bound refers to two-sided error testers. The fourth column specifies the computational complexity of the tester as a function of its query complexity (e.g., poly-time means running time that is polynomial in the total length of the queries made by the tester). The monotone graph properties are listed as monotone in the direction shown in the main text; we mention that monotonicity in the opposite direction can be shown too (see comment below). The weaker features of the results are indicated by italic (or by '–').

	property's domain	tester's error	tester's complexity	property's monotonicity
Thm 2	generic	one-sided	log-space	–
Thm 3	bd-graphs	one-sided	*in* $\mathcal{NP}$	downwards
Thm 4	dense graph	*two-sided*	poly-time	–
Thm 5	dense graph	*two-sided*	*in* $\mathcal{NP}$	upwards
Thm 6	dense graph	one-sided	*in* $\mathcal{NP}$	–

Figure 5: All hierarchy theorems at a glance

Reversing the direction of monotonicity. As stated above, the direction of monotonicity of graph properties can be reversed while preserving the query complexity of testing. Three simple ways of obtaining this effect are discussed next.

1. The simplest method, which is only applicable to the dense graph model, consists of considering the complement graphs. That is, for a graph property Π, we consider the graph property $\Pi^{\mathsf{c}} = \{G^{\mathsf{c}} : G \in \Pi\}$, where for $G = ([N], E)$ it holds that $G^{\mathsf{c}} = ([N], \{\{u, v\} : \{u, v\} \notin E)$.

2. For any fixed $\epsilon_0 > 0$, we may consider the property of being ϵ_0-far from the original property. That is, for a property Π and $\epsilon_0 > 0$, we consider the property $\texttt{far}_{\epsilon_0}(\Pi)$ that consists of all objects that are ϵ_0-far from Π. This notion is applicable to all models, and it holds that $\Pi \subseteq \texttt{far}_{\epsilon_0}(\texttt{far}_{\epsilon_0}(\Pi))$ (but equality does not necessarily hold). Thus, a lower bound for ϵ_0-testing Π does imply a corresponding lower bound for (ϵ_0-testing) $\texttt{far}_{\epsilon_0}(\Pi)$. Consequently, we can obtain a hierarchy theorem for upwards monotone properties graph in the bounded-degree model by starting with the property of being 0.01-far from 3-Colorability. However, in such a case, the upper bounds will be established by two-sided error testers.

3. An alternative method, which is only applicable to the bounded-degree graph model, consists of considering only the graphs with a maximum number of edges, which means that upwards monotonicity holds vacuously. But we need to make sure that this set of graphs yields a property that maintains the complexity of the original one. Specifically, starting with 3-Colorability, we consider the set of 3-colorable d-regular graphs (with an even number of vertices). To see that this property is hard to test consider a transformation of arbitrary graphs of maximum degree d into d-regular graphs that (approximately) preserves the distance from 3-Colorability. For example, consider taking two copies of the original graph and connecting copies of each vertex v of degree d_v by $d - d_v$ gadgets, where each consists of the complete bipartite graph $K_{d,d}$ with one edge omitted (so that the free endpoints can be used for connecting).

Some open problems. An immediate gap, conspicuous in Figure 5, refers to the fact that Theorem 3 refers to graph properties that are unlikely to be in $\mathcal{P}$ (and so they are unlikely to admit a relatively efficient tester). In fact, we do not know of a graph property in $\mathcal{P}$ that has maximal testing complexity in the bounded-degree model.

Many more natural open problems arise in the dense graph model. In particular, Theorems 4, 5 and 6 (and their proofs) raise several natural open problems, listed next. We stress that all questions refer to the adjacency matrix graph model considered in Sections 4–6 (and Section 7).

1. *Combining the features of all three hierarchy theorems*: Theorems 4, 5 and 6 provide incomparable hierarchy theorems, each having an additional feature that the others lack. Specifically, Theorem 4 refers to properties in $\mathcal{P}$ (and testing, in the positive part, is relatively efficient), Theorem 5 refers to monotone properties, and Theorem 6 provides one-sided testing (in the positive part). Is it possible to have a single hierarchy theorem that enjoys all three additional feature? Intermediate goals include the following:

   (a) *Hierarchy of monotone graph properties in $\mathcal{P}$*: Recall that Theorem 4 is proved by using non-monotone graph properties (which are in $\mathcal{P}$), while Theorem 5 refers to monotone graph properties that are not likely to be in $\mathcal{P}$. Can one combine the good aspects of both results?

   (b) *Hard-to-test monotone graph property in $\mathcal{P}$*: Indeed, before addressing Problem 1a, one should ask whether a result analogous to Theorem 7 holds for a monotone graph property. Recall that [GT, Thm. 1] provides a monotone graph property in $\mathcal{NP}$ that is hard-to-test.

   (c) *One-sided versus two-sided error testers*: Recall that the positive part of Theorem 6 refers to testing with one-sided error, but these testers are not relatively efficient. In contrast, the positive part of Theorem 4 provides relatively efficient testers, but these testers have two-sided error. Can one combine the good aspects of both results?

2. *Determining the exact effect of the blow-up operation on the distance between graphs*: Recall that the proof of Theorem 4 relies on the preservation of distances between graphs under the blow-up operation. While the partial results obtained in this work (regarding this matter) suffice for the proof of Theorem 4, the problem seems natural and of independent interest.

   Recall that Lemma 4.3 asserts that in some cases the distance between two unlabeled graphs is preserved *up to a constant factor* by any blow-up (i.e., "linear preservation"), whereas Theorem 8 of our technical report [GKNR] asserts a quadratic preservation for any pair of graphs. Also recall that it is not true that the distance between any two unlabeled graphs is *perfectly preserved* by any blow-up (see Footnote 1).

   In earlier versions of this work we raised the natural question of whether the distance between any two unlabeled graphs is preserved up to a constant factor by any blow-up. This question has been recently resolved by Oleg Pikhurko, who showed that *the distance is indeed preserved up to a constant factor, specifically a factor of three* [P, Sec. 4]. Note that Arie Matsliah's counterexample to perfect preservation (presented in Footnote 1) shows that the said constant factor cannot be smaller than 6/5. Indeed, *determining the true constant factor remains an open problem.*

Acknowledgments

We are grateful to Ronitt Rubinfeld for asking about the existence of hierarchy theorems for the adjacency matrix model. Ronitt raised this question during a discussion that took place at the Dagstuhl 2008 workshop on sub-linear algorithms. We are also grateful to Arie Matsliah, Dana Ron, and Yoav Tzur for helpful discussions. In particular, we thank Arie Matsliah for providing us with a proof that the blow-up operation does not preserve distances in a perfect manner.

References

[ABI] N. Alon, L. Babai and A. Itai. A fast and Simple Randomized Algorithm for the Maximal Independent Set Problem. *J. of Algorithms*, Vol. 7, pages 567–583, 1986.

[AFKS] N. Alon, E. Fischer, M. Krivelevich and M. Szegedy. Efficient Testing of Large Graphs. *Combinatorica*, Vol. 20, pages 451–476, 2000.

[AFNS] N. Alon, E. Fischer, I. Newman, and A. Shapira. A Combinatorial Characterization of the Testable Graph Properties: It's All About Regularity. In *38th STOC*, pages 251–260, 2006.

[AGHP] N. Alon, O. Goldreich, J. Hastad, and R. Peralta. Simple constructions of almost k-wise independent random variables. *Journal of Random structures and Algorithms*, Vol. 3 (3), pages 289–304, 1992.

[AS] N. Alon and A. Shapira. Every Monotone Graph Property is Testable. *SIAM Journal on Computing*, Vol. 38, pages 505–522, 2008.

[BSS] I. Benjamini, O. Schramm, and A. Shapira. Every Minor-Closed Property of Sparse Graphs is Testable. In *40th STOC*, pages 393–402, 2008.

[BLR] M. Blum, M. Luby and R. Rubinfeld. Self-Testing/Correcting with Applications to Numerical Problems. *JCSS*, Vol. 47, No. 3, pages 549–595, 1993.

[BHR] E. Ben-Sasson, P. Harsha, and S. Raskhodnikova. 3CNF Properties Are Hard to Test. *SIAM Journal on Computing*, Vol. 35 (1), pages 1–21, 2005.

[BOT] A. Bogdanov, K. Obata, and L. Trevisan. A lower bound for testing 3-colorability in bounded-degree graphs. In *43rd FOCS*, pages 93–102, 2002.

[EKK+] F. Ergun, S. Kannan, S. R. Kumar, R. Rubinfeld, and M. Viswanathan. Spot-checkers. *JCSS*, Vol. 60 (3), pages 717–751, 2000.

[FM] E. Fischer and A. Matsliah. Testing Graph Isomorphism. In *17th SODA*, pages 299–308, 2006.

[GGL+] O. Goldreich, S. Goldwasser, E. Lehman, D. Ron, and A. Samorodnitsky. Testing Monotonicity. *Combinatorica*, Vol. 20 (3), pages 301–337, 2000.

[GGR] O. Goldreich, S. Goldwasser, and D. Ron. Property testing and its connection to learning and approximation. *Journal of the ACM*, pages 653–750, July 1998.

[GKNR] O. Goldreich, M. Krivelevich, I. Newman, and E. Rozenberg. Hierarchy Theorems for Property Testing. *ECCC*, TR08-097, 2008.

[GR1] O. Goldreich and D. Ron. Property Testing in Bounded Degree Graphs. *Algorithmica*, Vol. 32 (2), pages 302–343, 2002.

[GR2] O. Goldreich and D. Ron. A Sublinear Bipartitness Tester for Bounded Degree Graphs. *Combinatorica*, Vol. 19 (3), pages 335–373, 1999.

[GT] O. Goldreich and L. Trevisan. Three theorems regarding testing graph properties. *Random Structures and Algorithms*, Vol. 23 (1), pages 23–57, August 2003.

[LNS] O. Lachish, I. Newman, and A. Shapira. Space Complexity vs. Query Complexity. *Computational Complexity*, Vol. 17, pages 70–93, 2008.

[NN] J. Naor and M. Naor. Small-bias Probability Spaces: Efficient Constructions and Applications. *SIAM J. on Computing*, Vol 22, 1993, pages 838–856.

[PRR] M. Parnas, D. Ron and R. Rubinfeld. Testing Membership in Parenthesis Laguages. *Random Structures and Algorithms*, Vol. 22 (1), pages 98–138, 2003.

[P] O. Pikhurko. An Analytic Approach to Stability. Manuscript, 2009. Available from `http://arxiv.org/abs/0812.0214`

[R1] D. Ron. Property Testing: A Learning Theory Perspective. *Foundations and Trends in Machine Learning*, Vol. 1 (3), pages 307–402, 2008.

[R2] D. Ron. Algorithmic and Analysis Techniques in Property Testing. *Foundations and Trends in TCS*, to appear.

[RS] R. Rubinfeld and M. Sudan. Robust characterization of polynomials with applications to program testing. *SIAM Journal on Computing*, 25(2), pages 252–271, 1996.

[S] R. Shaltiel. Recent Developments in Explicit Constructions of Extractors. In *Current Trends in Theoretical Computer Science: The Challenge of the New Century, Vol 1: Algorithms and Complexity*, World scietific, 2004. Preliminary version in *Bulletin of the EATCS 77*, pages 67–95, 2002.