Hierarchy Theorems for Property Testing

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

November 14, 2008

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, Adaptivity vs Non-adaptivity,

*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

§Department of Computer Science, Technion, Haifa, ISRAEL. Email: eyalros@technion.ac.il

Contents

1 Introduction

In the last decade, the area of property testing has attracted much attention (see the surveys of [F, R], which are already somewhat out-of-date). 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, where the query complexity is 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]).

One natural problem that arises is whether there exist properties of arbitrary query complexity. We answer this question affirmative, 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 complexity is measured in terms of the number of vertices rather than the number of all vertex pairs. 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 treatment of the adjacency representation model raises several interesting problems, which are further discussed in Section 6. which are further discussed in Section 6.

Conventions. For sake of simplicity, we state all results while referring to query complexity as a function of the input 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.

Organization. Sections 2 and 3 present hierarchy theorems for the generic case and the bounded-degree graph model, respectively. The bulk of this paper provides hierarchy theorems for graph properties in the adjacency matrix model. Specifically, the focus of Section 4 is on the (standard) computational complexity of these properties, whereas the focus of Section 5 is on monotone properties. Combining both features is one of the open problems posed in Section 6. Appendices A and B also refer to the adjacency matrix model; they contain results that are not central to the main themes of this work (but are sufficiently related to it). In particular, in Appendix A we prove the existence of graph properties that are in $\mathcal{P}$ and have maximal query complexity (in the adjacency matrix model).

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) 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* $\Pr[T^f(n,\epsilon)=0] \geq 2/3$.

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

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 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 $f(i) = f'(i \bmod q(n))$, where $i \bmod m$ is (non-standardly) defined as the smallest positive integer that is congruent to i modulo m, and add f to Π_n.

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 $r \in \{0,1,...,(n/q(n))-1\}$, 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 $f \in \Pi$. On the other hand, if f is accepted with probability at least 2/3, then the reconstructed f' must be in Π' (otherwise the Step 2 would have rejected with probability 1) and f must be $(\epsilon/2)$-close to the repetition of this f' (otherwise each iteration of the Step 1 would have rejected with probability at least ϵ). 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 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 is proved by starting with the hard-to-test property asserted in Theorem 6.)

3 Testing 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.[1] 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 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*

[1] 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 $\deg(v) \stackrel{\text{def}}{=} |\{i : g(v,i) \neq 0\}|$.

graphs of maximum degree d that are 3-colorable and consist of connected components of size at most $q(N)$.

Proof: We start with an arbitrary property Π' for which testing is known to require a linear number of queries (even when allowing two-sided error). We further assume that Π' is downward monotone (i.e., if $G' \in \Pi'$ then any subgraph of G' 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 each graph in Π_N consists 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 $t \stackrel{\rm def}{=} \lfloor N/q(N) \rfloor$ 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 follows 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 while stopping after $q(N) + 1$ vertices are encountered.

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 $G \in \Pi$. 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 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 the proof of Theorem 2 each object in Π_N corresponds to multiples 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 Testing 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 Appendix A. Building on this result, we show:

Theorem 4 *In the adjacency matrix model, for every $q : \mathbb{N} \to \mathbb{N}$ that is at most quadratic, there exists a graph property Π that is testable in q queries, but is not testable in $o(q)$ queries.*[2] *Furthermore, if $N \mapsto q(N)$ is computable in $\mathrm{poly}(N)$-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, it would be nice to have a stronger separation in which the positive side is established via a one-sided error tester.

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 (equal size) 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.[3]

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 t-way partition of the vertex set of G such that the edges between these 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 all these 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 blown-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 6 (see Appendix A, which builds on [GGR]).

The notion of graphs that have "vastly 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

[2] Both the upper and lower bounds refer to two-sided error testers.

[3] In particular, there are no edges inside any set.

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.[4]

The augmentation. We first augment the graphs in Π' such that the vertices in the resulting graphs are dispersed, while the augmentation amount 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 graph H' by a clique of n' vertices and connect the vertices of H' and the clique vertices by a bipartite graph that corresponds to a Hadamard matrix; that is, the $i^{\rm th}$ vertex of H' is connected to the $j^{\rm th}$ vertex of the clique if and only if the inner product modulo 2 of $i - 1$ and $j - 1$ (in $(\log_2 n')$-bit long binary notation) equals 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 first note that Π'' is indeed dispersed (i.e., the resulting $2n'$-vertex graphs have vertex neighborhoods that differ on at least $n \geq n'/4$ vertices).[5] Next note that testing Π'' requires a quadratic number of queries, because testing Π' can be reduced to testing Π'' (i.e., ϵ-testing membership in Π'_n reduces to ϵ'-testing membership in $\Pi''_{2n'}$, where $n' \leq 4n$ and $\epsilon' = \epsilon/64$). Finally, note 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).[6]

Applying graph blow-up. Next, we apply an (adequate factor) graph blow-up to the augmented set 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 the 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 G is a t-factor blow-up of G' if the vertex set of G can be partitioned into t (equal size) 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 $i^{\rm th}$ cloud and the $j^{\rm th}$ 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 note that Π is in $\mathcal{P}$. This fact follows from the hypothesis that Π' is dispersed: Specifically, given any graph N-vertex graph G, we can 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 G'. Finally, we check whether G' is in Π'_n (relying on the fact that $\Pi' \in \mathcal{P}$). Proving that the query complexity of testing Π indeed equals $\Theta(q)$ is undertaken in the next two sections.

[4]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.

[5]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', their neighborhood in the clique disagrees on $n'/2 > n$ vertices. An analogous claim holds with respect to any two vertices of the clique.

[6]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.

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.)

As shown by Arie Matsliah (see Appendix B), 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). 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. In Appendix B, we present a more complicated proof that holds for arbitrary Π' (which need not be dispersed), but with $\epsilon = \Omega(\epsilon')^2$.

Claim 4.1 *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'_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 Claim 4.1 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 Claim 4.1 is 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, v that belong to the same cloud of G_1 it holds that $\pi(u), \pi(v)$ 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. 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).[7] 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) such that $\pi^{-1}(v)$ and $\pi^{-1}(v')$ belong to different clouds of

[7]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.)

G_1 contributes at least $\alpha \cdot tn$ violations to Eq. (1).[8] 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 $8\epsilon + (6\epsilon/\alpha) \geq 2\epsilon'$, and the claim follows (with $c = 1/7$). $\square$

Recall that Claim 4.1 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 $n = \sqrt{q(N)}$. Thus, 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 4.2 *On input N and proximity parameter ϵ, and when given oracle access to a graph $G = ([N], E)$, the algorithm proceeds as follows:*

1. *Setting $\epsilon' \stackrel{\text{def}}{=} \epsilon/3$ 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 $s \stackrel{\text{def}}{=} O(\log n)$ 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 $s' \stackrel{\text{def}}{=} O(\epsilon^{-2} \cdot n \log n)$ 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*

[8] For each such pair (v, v'), there exists 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. Recall 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.

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 adequate 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 $q(N)$. Furthermore, in this case, the above algorithm can be implemented in time $\mathrm{poly}(n \cdot \log N) = \mathrm{poly}(q(N) \cdot \log N)$. We comment that the Algorithm 4.2 is adaptive, and that a straightforward non-adaptive implementation has query complexity $O(n \log n)^2 = \widetilde{O}(q(N))$. (In fact, a (non-adaptive) tester of query complexity $\widetilde{O}(q(N))$ can be obtained by a simpler algorithm that just 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.)[9]

We next verify 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 4.2 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 4.2 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 4.2 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) the subgraph of G induced by R is in Π'_n, and (2) at most ϵ' fraction of the vertex pairs of G have edge relation that is inconsistent with the corresponding vertices in R (i.e., 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 4.2 rejects with probability at least 2/3 (i.e., 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 $\rho \cdot \min((1-(1-\epsilon')^{\Omega(1/\epsilon)}), 1))$.

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

[9] 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.

$v \in [N] \setminus X$ 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.

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 here a hierarchy theorem for *monotone* graph properties (in the same model).

Theorem 5 *In the adjacency matrix model, for every $q : \mathbb{N} \to \mathbb{N}$ that is at most quadratic, 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).[10] 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 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 number of edges, is to augment the monotone closure of Π with all graphs that have significantly more edges, 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 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 allows to 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).

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:*

[10]Indeed, this is the approach used in the proof of [GT, Thm. 1].

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* $(0.75-\delta)\cdot n$.

Moreover, pairs of graphs in Π'_n *are related as follows:*

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.

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 $K \stackrel{\rm def}{=} \exp(\Theta(n^2))$ 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 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.[11]

Regarding the distance between graphs in Π'_n, we distinguish two cases. In the case that $G'_2, G'_2 \in \Pi'_n$ are not isomorphic, they arise from two independently selected graphs, and so with probability at least $1 - \exp(-\Omega(n^2)) > 1 - o(|\Pi'_n|^{-2})$ they are 0.4-far from one another. Applying the union bound, this establishes Condition 3. Turning to any pair of graphs that are isomorphic (and arise from isomorphic copies of the same "basic" graph), we consider the sub-case in which the isomorphism π (between G'_1 and G'_2) satisfies $|\{i \in [n] : \pi(i) \neq i\}| > 0.1n$ (i.e., as in Condition 4). Fixing any such permutation π, we consider disjoint sets $I \subset [n]$ and $\pi(I) = \{\pi(i) : i \in I\}$ such that $|I| \geq 0.05n$. For a random n-vertex graph $G' = ([n], E')$, with probability at least $1 - \exp(-\Omega(n^2)) > 1 - o(|\Pi'_n|^{-1})$, 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. The claim follows. □

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., $\Delta \stackrel{\rm def}{=} \epsilon_4/100$, where ϵ_4 is a value of the proximity parameter for which Theorem 4 holds). Needless to say, Π' satisfies the foregoing three conditions when setting $\delta = \Delta$. Given the 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 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 $0.52 \cdot N$.

[11] Indeed, the query-complexity lower bound is not harmed, because it is established by considering the uniform distribution over Π'_n (versus the uniform distribution over all n-vertex graphs).

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 $0.52 \cdot N$. 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. 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 above 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 we will assume that in our positive result (i.e., the analysis of the optimal tester) 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) the N/n-factor blow-up of random n-vertex graphs, and (D2) the N/n-factor blow-up of uniformly selected graph in Π'_n. Combining [GGR, Prop. 10.2.3.1] and Claim 4.1, it holds 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.

Recalling that Π_N contains the support of distribution (D2), it now suffices to show that, with high probability, a graph selected according to distribution (D1) is far from Π_N. This claim suffices because it yields a distribution on Π_N (indeed (D2) itself) and a distribution that is typically far from Π_N such that distinguishing these two distributions requires $\Omega(q)$ queries.

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 $(0.5+\Delta)\cdot N$. 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 $(0.5 + \Delta) \cdot N$ 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 $(0.5+2\Delta)\cdot\binom{N}{2}$ 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 at least $0.52 \cdot N$ in C.

Consider the distance between G and a blow-up of G', denoted B (standing for blow-up). Each vertex in H contributes at most N units to this distance, but its contribution to the distance between G and C is at least $0.52 \cdot N - (0.5 + \Delta) \cdot N > N/60$. 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 $(u,v) \in ([N] \setminus H)^2$ that is connected in B must 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 $(0.5-\Delta) \cdot N$. Note that the number of edges with at least one endpoint in H is larger in C than it is in B (by an additive term of $(0.02 - \Delta - 60\Delta)|H| \cdot N > 0.001|H| \cdot N$, which we do not use).[12] 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)). $\square$

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:

Claim 5.3 *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 neighborhood, 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 are empty.*

Proof: The mere existence of a partition $(V'_1, ..., V'_n)$ and of a graph G'' that satisfies the condition 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

[12] To justify this assertion we note that in C each vertex of H has degree at least $0.52 \cdot N$, whereas in B each vertex has degree at most $(0.5+\Delta) \cdot N$. Thus, the difference in the sum of degrees of vertices in H is at least $|H| \cdot (0.02 - \Delta) \cdot N$, but edges with both sides in H are counted twice (and thus a corrective term of at most $|H|^2 < 60\Delta|H| \cdot N$ is due, where $|H| < 60\Delta N$ was (implicitly) established above).

the same $V_i \setminus H$ may differ on at most $2 \cdot (0.52N - (0.5 - \Delta)N + |H|) < 0.05N$ of their neighbors, whereas vertices that reside in different $V_i \setminus H$'s must differ on at least $(0.5 - 4\Delta) \cdot N - 2 \cdot |H| > 0.45N$ neighbors. Also, noting that each V_i' has size at most N/n and 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 $[n]$.

Referring to the foregoing partition $(V_1, ..., V_n)$, we claim that two vertices $u, v \in [N] \setminus H$ can be placed in the same set of an 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. 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 $(V_1 \setminus H, ..., V_n \setminus H)$.

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 $153\Delta \cdot n^2$ pairs. 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. We shall show next that this 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 both V_i' and V_j' must share more than $(0.5 - O(\Delta)) \cdot N - 0.1N > 0.3N$ neighbors, which contradicts the postulate (regarding G' which implies) that each such pair can share at most $(0.25 + 3\Delta) \cdot N + |H| < 0.3N$ neighbors. The claim follows. □

Having established Claim 5.3, 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 4.2. In particular, since we cannot expect to identify representatives from all clouds (i.e., some sets V_i' in Claim 5.3 may be too small or even empty), we settle for obtaining representatives from at least a $1 - \epsilon'$ fraction of the identifiable clouds (which leads to using, as a basis, the version of Algorithm 4.2 that is discussed in Footnote 9).

Algorithm 5.4 *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. *Using a sample of $O(\epsilon^{-2})$ vertex pairs, we first 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 less than $0.5 + 2\Delta - \epsilon'$.*

2. *Next, 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 degress is 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 them 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. *Note that the set R suggests a clustering of the vertices of G according to their neighbors in the set R. Referring to this clustering, we check whether it is indeed adequate. 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 $\iota(v) = \perp$. In the following sub-steps we refer to estimates of the degrees of individual vertices that are obtained by an auxiliary sample of size $O(\epsilon^{-2}\log t)$, where $t = O(\epsilon^{-2} n \log(1/\epsilon))$ denotes the number of vertices for which we need a degree estimate.*

   (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 $\iota(v) = i$.*

   (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 representatives is connected. 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 $\{u,v\} \notin E$.*

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 $(t+O(1/\epsilon)) \cdot (\epsilon^{-2}\log t + n) = O(\epsilon^{-4}n^2\log^2(1/\epsilon))$, where a better bound of $o(\epsilon^{-4}n^2)$ holds when $\epsilon \gg 1/\sqrt{n}$).

We next 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 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 degree in the subgraph induced by S. The same holds with respect to the number of neighbors that each such vertex has in n designated sets (i.e., the sets V_i associated with $G \in \Xi_N$). Letting H, $(V_1', ..., V_n')$ and G'' be as in Claim 5.3, 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 and allow also empty clusters to obtain a sequence of length n). Furthermore, the induced graph G_R fits the subgraph G'' in the sense that it passes the checks in Step 5b. 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 a N/t-factor blow-up of the subgraph G_I is likely to be in Ξ_N, and so applying Claim 5.3 to this blow-up of G_I guarantees the uniqueness of G'' (with respect to G_I). The uniqueness of G'' with respect to G_R follows by observing that, with high probability, the constraints on G'' imposed by the subgraph G_R are a superset of the constraints on G'' imposed by the subgraph G_I, because R can be viewed as obtained from I by replacing some vertices of H by vertices not in H (i.e., a vertex in $V_i \cap H$ is replaced by some vertex in $V_i' = V_i \setminus H$).[13] 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. 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. 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 $m \stackrel{\text{def}}{=} 3\epsilon' N$ 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.

6 Summary of Open Problems That Arise

Theorems 4 and 5 (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 and 5.

[13] Indeed, the remaining vertices of H are viewed (in R) as non-existing).

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

   Recall that Claim 4.1 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 7 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 beginning of Appendix B). A natural question that arises is whether the distance between any two unlabeled graphs is preserved up to a constant factor by any blow-up, and if so then what is the minimal such constant.

2. *One-sided versus two-sided error testers*: Theorem 4 refers to testing with two-sided error. However, following the analogy with prior sections, one may hope to see a hierarchy theorem that confronts two-sided error lower bounds and one-sided error upper bounds.[14] As an intermediate question, how about a hierarchy theorem for one-sided testing (i.e., a result, analogous to Theorem 4, that would refer to one-sided error lower and upper bounds)? Needless to say, the same applies to Theorem 5.

3. *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?

4. *Hard-to-test monotone graph property in $\mathcal{P}$*: Indeed, before addressing Problem 3, one should ask whether a result analogous to Theorem 6 holds for a monotone graph property? Recall that [GT, Thm. 1] provides a monotone graph property in $\mathcal{NP}$ that is hard-to-test.

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 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.

[14] We note that the "blow-up" properties used in the proof of Theorem 4 do not allow one-sided testers with sub-linear query complexity (because such testers may fail to find evidence that a graph is far from a blow-up of any graph).

References

[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.

[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.

[BSS] I. Benjamini, O. Schramm, and A. Shapira. Every Minor-Closed Property of Sparse Graphs is Testable. *ECCC*, TR08-010, 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.

[F] E. Fischer. The art of uninformed decisions: A primer to property testing. *Bulletin of the European Association for Theoretical Computer Science*, Vol. 75, pages 97–126, 2001.

[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.

[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.

[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.

[R] D. Ron. Property testing. In *Handbook on Randomization, Volume II*, pages 597–649, 2001. (Editors: S. Rajasekaran, P.M. Pardalos, J.H. Reif and J.D.P. Rolim.)

[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.

APPENDICES

Appendix A: Hard-to-test Properties in P

In this appendix 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 $\mathcal{P}$).[15] 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 graph properties that we obtain in this appendix are not monotone). Furthermore, our techniques seem incompatible with monotonicity. The result we prove is stated next.

Theorem 6 *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 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

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

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].[16]

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 that 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 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).

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,

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

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 $\mathcal{P}$.

[16] 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).

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 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). We shall first show that, with high probability, R_N is 0.01-far from the the set of final graphs.

Claim 6.1 *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 less than $2^{0.1\cdot\binom{3N+1}{2}} \ll 2^{(0.9+o(1))\cdot\binom{N}{2}}$ graphs, and so (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. $\square$

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

Claim 6.2 *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 (i.e., in such cases adaptive queries offer no advantage). $\square$

Theorem 6 follows by combining Claims 6.1 and 6.2 (with the fact that the set of final graphs is in $\mathcal{P}$).

Appendix B: A General Analysis of the Effect of Graph Blow-Up

A natural question is whether the distance between any two unlabeled graphs is *perfectly preserved* by any blow-up. This question was answered negatively by Arie Matsliah, and we start this appendix with a presentation of his proof. The 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 4 clouds such that i' is mapped to cloud i, whereas $1''$ is mapped to the 1st cloud, $2''$ is mapped to the 4th cloud, $3''$ is mapped to the 2nd cloud, and $4''$ is mapped to the 3rd cloud (see Figure 1). 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.

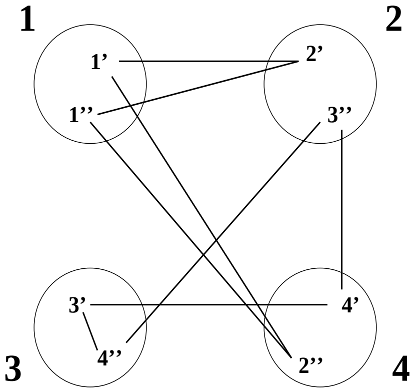

Figure 1: A mapping of the 2-factor blow-up of H to four clouds that fit the 2-factor blow-up of G

Recall that Claim 4.1 has established that for dispersed graphs the blow-up operation maintains distances up to a constant factor (depending on the dispersing parameter of one of the two graphs). Here we prove a weaker preservation that refers to all pairs of graphs (i.e., we waive the dispersing condition).

Theorem 7 *There exists a universal constant $c > 0$ such that the following holds for every n, ϵ' and* (unlabeled) *n-vertex graphs G_1', G_2'. If G_1' is ϵ'-far from G_2', then for any t the* (unlabeled) *t-factor blow-up of G_1' is $c \cdot (\epsilon')^2$-far from the the* (unlabeled) *t-factor blow-up of G_2'.*

Proof: The current proof builds on the ideas underlying the proof of Claim 4.1. However, in the current case we have no bound on the dispersing parameter of G_1', and so the argument is more refined.

Again, we 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 (as defined in Eq. (1)). Intuitively, it may be that π maps to each cloud of G_2 vertices that originate in different clouds of G_1, but we shall show that on the average these vertices do not have very different neighborhoods and hence they can be moved to obtain homogeneous clouds in G_2 without creating too many violations.

Letting 2ϵ denote the fraction of violation in Eq. (1), we say that two vertices in G_1' are similar if the neighbor sets of these two vertices differ on at most $\sqrt{\epsilon} \cdot n$ elements (i.e., vertices u and v are similar if the symmetric difference between the sets $\{w : \{u, w\} \in E\}$ and $\{w : \{v, w\} \in E\}$ has size at most $\sqrt{\epsilon} \cdot n$). Similarly, we say that two vertices in G_1 are similar if the neighbor sets of these two vertices differ on at most $\sqrt{\epsilon} \cdot tn$ elements. Indeed, a pair of vertices of G_1 is similar if and only if these vertices reside in clouds of G_1 that correspond to vertices that are similar in G'_1.

We consider a maximal pairing of vertices of G_2 that consists of disjoint pairs of vertices such that each pair consists of vertices that reside in the same cloud of G_2 but are not similar (w.r.t G_1). We first show that this pairing may contain at most $2\sqrt{\epsilon} \cdot tn$ pairs. As in the proof of Claim 4.1, this holds because every pair contributes at least $\sqrt{\epsilon} \cdot tn$ violations to Eq. (1), whereas the total number of violations is bounded by $2\epsilon \cdot (tn)^2$.

We call a vertex of G_2 free if it does not appear in the aforementioned maximal pairing, and recall that the number of free vertices is at least $(1 - 4\sqrt{\epsilon}) \cdot tn$. Note that any two free vertices that reside in the same cloud of G_2 are similar with respect to G_1 (i.e., they have similar neighborhoods in G_1). A cloud of G_2 is called good if at least $t/2$ of its vertices are free. We note that at least $(1-8\sqrt{\epsilon}) \cdot n$ of the clouds of G_2 are good, and that these clouds contain at least $(1 - 4\sqrt{\epsilon}) \cdot tn - 8\sqrt{\epsilon}n \cdot t/2 = (1 - 8\sqrt{\epsilon}) \cdot tn$ free vertices, which are called super-free.

Consider an auxiliary t-regular bipartite graph with clouds of G_1 on one side, clouds of G_2 on the other side, and edges representing the mapping π (i.e., the i^{th} cloud of G_1 is connected to the j^{th} cloud of G_2 if some vertex that resides in the i^{th} cloud of G_1 is mapped by π to the j^{th} cloud of G_2). Consider a t-coloring of the edges of this bipartite graph, and note that this t-coloring induces a t-partition of the pairs $\{(v, \pi(v)) : v \in [tn]\}$ such that each part contains n pairs that in turn contain a single vertex from each cloud of G_1 and a single vertex from each cloud of G_1. Thus, each part induces an injection of the clouds of G_1 to n vertices that belong to different clouds of G_2, and the t sets of vertices appearing in the range of these t injections constitute a partition of the set of vertices of G_2. It follows that one of these injections, denoted $\phi : [n] \to [tn]$, contains in its range at least $(1 - 8\sqrt{\epsilon}) \cdot n$ super-free vertices. We call the (good) clouds of G_2 containing these (super-free) vertices very good.

Using the foregoing injection ϕ, we define a new bijection π' of the vertices of G_1 to the vertices of G_2. The bijection π' maps all vertices in each cloud of G_1 to some cloud of G_2 such that the i^{th} cloud of G_1 is mapped to the $\phi'(i)^{\text{th}}$ cloud of G_1, where $\phi'(i)$ denote the cloud of G_2 (under π) that contains the vertex $\phi(i)$. The number of violations created by π' is upper-bounded as follows. The number of violations between the very good clouds is upper-bounded by $4(\epsilon + \sqrt{\epsilon}) \cdot (tn)^2$, which represents four times the number of violations occuring under π between the very good clouds plus the slackness between similar vertices residing in these clouds (specifically, between the vertex $\phi(i)$ and the other super-free vertices in this cloud). The number of violations involving clouds that are not very good is upper-bounded by $8\sqrt{\epsilon} \cdot (tn)^2$ (since the number of such clouds is $8\sqrt{\epsilon} \cdot n$). Thus, π' yields a mapping of G'_1 to G'_2 that has at most $(4\epsilon + 12\sqrt{\epsilon}) \cdot n^2$ violations. Recalling that G'_1 is ϵ'-far from G'_2, we conclude that $4\epsilon + 12\sqrt{\epsilon} \geq \epsilon'$, and the claim follows (with $c = 1/256$). ■