Electronic Colloquium on Computational Complexity, Revision 3 of Report No. 160 (2020)

Non-adaptive vs Adaptive Queries in the Dense Graph Testing Model

Oded Goldreich[*] Avi Wigderson[†]

September 8, 2021

Abstract

We study the relation between the query complexity of adaptive and non-adaptive testers in the dense graph model. It has been known for a couple of decades that the query complexity of non-adaptive testers is at most quadratic in the query complexity of adaptive testers. We show that this general result is essentially tight; that is, there exist graph properties for which any non-adaptive tester must have query complexity that is almost quadratic in the query complexity of the best general (i.e., adaptive) tester.

More generally, for every $q : \mathbb{N} \to \mathbb{N}$ such that $q(n) \leq \sqrt{n}$ and constant $c \in [1, 2]$, we show a graph property that is testable in $\Theta(q(n))$ queries, but its non-adaptive query complexity is $\Theta(q(n)^c)$, omitting $\text{poly}(\log n)$ factors and ignoring the effect of the proximity parameter ϵ. Furthermore, the upper bounds hold for one-sided error testers, and are at most quadratic in $1/\epsilon$.

These results are obtained through the use of general reductions that transport properties of ordered structured (like bit strings) to those of unordered structures (like unlabeled graphs). The main features of these reductions are query-efficiency and preservation of distance to the properties. This method was initiated in our prior work (*ECCC*, TR20-149), and we significantly extend it here.

[*]Faculty of Mathematics and Computer Science, Weizmann Institute of Science, Rehovot, Israel. E-mail: `oded.goldreich@weizmann.ac.il`. Partially supported by the Israel Science Foundation (grant No. 1041/18) and by the European Research Council (ERC) under the European Unions Horizon 2020 research and innovation programme (grant agreement No. 819702).

[†]School of Mathematics, Institute for Advanced Study, Princeton, NJ 08540, USA. E-mail: `avi@ias.edu`. Research partially supported by NSF grant CCF-1900460.

ISSN 1433-8092

Contents

1 Introduction

The fundamental relation between adaptive and non-adaptive oracle machines has been studied in a variety of models. In particular, this relation has been studied also in the context of property testing in various settings. Specifically, in the setting of testing the satisfiability of linear constraints, it was shown that adaptivity offers absolutely no gain [2]. A similar result holds for testing monotonicity of sequences of positive integers [5]. In contrast, an exponential gap between the adaptive and non-adaptive complexities exists in the context of testing other properties of functions [5]. Lastly, we mention that an even more dramatic gap exists in the setting of testing graph properties in the bounded-degree model [18] (see also [6, Thm. 9.2]).

We follow [15, 12] in studying the relation between adaptive and non-adaptive oracle machines in the context of testing graph properties in the dense graph model. This is definitely a natural model, and the study is quite refined because it is known that, in this model, the gap between the query complexities of adaptive and non-adaptive machines is at most quadratic [1, 13]. Our results answer several natural open problems regarding possible relations between these complexities (cf. [6, Sec. 8.5.4]): Essentially, we show that any relation that is not ruled out by the quadratic upper bound is actually possible (i.e., occurs for some graph properties).

Our results are outlined in Section 1.2, following a brief review of the model (provided in Section 1.1). In Section 1.3 we describe some of the ideas used towards proving these results. We hint that the notion of robustly self-ordered graphs and local self-ordering procedures for them, introduced and studied in our prior work [14], play a central role in our proofs.

1.1 Property testing in the dense graph model

Property testing refers to algorithms of sublinear query complexity for *approximate decision*; that is, given oracle access to an object, these algorithms (called testers) distinguish objects that have a predetermined property from objects that are far from the property. Different models of property testing arise from different query access and different distance measures.

In the last couple of decades, the area of property testing has attracted significant attention (see, e.g., [6]). Much of this attention was devoted to testing graph properties in a variety of models including the dense graph model, introduced in [10] and surveyed in [6, Chap. 8]. In this model graphs are represented by their adjacency predicate and distances are measured as the ratio of the number of differing adjacencies to the maximal number of vertex-pairs.

Specifically, a (simple undirected) graph $G = ([n], E)$ is represented by the adjacency predicate $g : [n] \times [n] \to \{0, 1\}$ such that $g(u, v) = 1$ if and only if $\{u, v\} \in E$, and oracle access to a graph means oracle access to its adjacency predicate (equiv., adjacency matrix). The distance between the graphs $G = ([n], E)$ and $G' = ([n], E')$ is defined as the fraction of entries (in the adjacency matrix) on which the two graphs disagree. Lastly, recall that a graph property is a set of graphs that is closed under isomorphism; that is, if G is isomorphic to G' and Π is a graph property, then $G \in \Pi$ if and only if $G' \in \Pi$. (In other words, graph properties are actually properties of unlabeled graphs.)

Definition 1.1 (testing graph properties in the dense graph model): *A* tester *for a graph property Π is a probabilistic oracle machine that, on input parameters n and ϵ, and oracle access to an n-vertex graph $G = ([n], E)$ outputs a binary verdict that satisfies the following two conditions.*

1. *If $G \in \Pi$, then the tester accepts with probability at least* $2/3$.

2. *If G is ϵ-far from Π, then the tester accepts with probability at most $1/3$, where G is* ϵ-far *from Π if for every n-vertex graph $G' = ([n], E') \in \Pi$ the adjacency matrices of G and G' disagree on more than $\epsilon \cdot n^2$ entries.*

We say that the tester is non-adaptive *if it determines all its queries based on its explicit input parameters* (n and ϵ) *and its internal coin tosses, independently of answers provided to prior queries. Otherwise, we say that the tester is* adaptive.

The query complexity of a tester for Π is a function (of the parameters n and ϵ) that represents the number of queries made by the tester on the worst-case n-vertex graph, when given the proximity parameter $\epsilon > 0$. The dependency of the complexity on n is the primary concern, and one often views ϵ as an arbitrary small constant. (Definitely $\epsilon \geq n^{-2}$, although one often envisions $\epsilon > n^{-\Omega(1)}$ and even $\epsilon > 1/\log n$.) In light of these dispositions, when stating that the query complexity is $\Omega(q(n))$, we mean that this bound holds for all sufficiently small $\epsilon > 0$; that is, there exists a constant $\epsilon_0 > 0$ such that distinguishing between n-vertex graphs in Π and n-vertex graphs that are ϵ_0-far from Π requires $\Omega(q(n))$ queries.

1.2 Our main results

The focus of this paper is on the relation between the query complexity of adaptive and non-adaptive testers in the dense graph model. It has been known for a couple of decades that the query complexity of non-adaptive testers is at most quadratic in the query complexity of adaptive testers [1, 13], and the question of whether this relation is tight has been open since.

The first indication that non-adaptive testers may need more queries than adaptive ones was provided in [15] in the context of promise problems. In that context (of promise problems), an almost quadratic separation was proved in [12, Thm. 5.7].[1] In the standard context (i.e., no promise), separations were proved in [12], with the largest one being roughly a power of $3/2$; that is, a graph property was presented for which the general query complexity is $\widetilde{O}(q)$ but non-adaptive testers require $\Omega(q^{3/2})$ queries. Our main result is an almost quadratic separation in the standard context; that is, we prove the following –

Theorem 1.2 (an almost quadratic gap between non-adaptive and adaptive query complexities): *There exists a graph property of n-vertex graphs that is testable by an adaptive oracle machine that makes $O(\epsilon^{-1} \cdot \sqrt{n} \cdot \log n)$ queries but testing it non-adaptively requires $\Omega(n)$ queries.*

Theorem 1.2 asserts that the gap between the query complexity of non-adaptive testers and adaptive ones may be quadratic. Answering open problems raised in [12, Sec. 1.3] (see also [6, Sec. 8.5.4]), we prove that the gap can take the form of any function that is at most (almost) quadratic; that is, we prove the following –

Theorem 1.3 (generalizing Theorem 1.2 to smaller gaps between non-adaptive and adaptive query complexities): *For every function $g : \mathbb{N} \to \mathbb{N}$ such that $g(n) \leq \sqrt{n}$, there exists a graph property Π that satisfies the following two conditions:*

1. *There exists a general* (i.e., adaptive) *tester of* Π *that makes $O(\epsilon^{-1} \cdot \sqrt{n} \cdot \log n)$ queries, and any such tester must make $\Omega(\sqrt{n})$ queries.*

[1] Both the aforementioned results hold also in a model in which the graph property may depend on the proximity parameter ϵ.

2. *Any non-adaptive tester must make $\Omega(g(n) \cdot \sqrt{n})$ queries, and there exists such a tester that makes $O(\epsilon^{-2} \cdot g(n) \cdot \sqrt{n} \cdot \log n)$ queries.*

   Actually, the upper bound is $O(\epsilon^{-2} \cdot g(n) \cdot \sqrt{n} + \epsilon^{-1} \cdot \sqrt{n} \cdot \log n)$.

(Recall that n denotes the number of vertices in the tested graph.)

Note that in Theorem 1.2 there was no need to state the lower bound for adaptive testers and the upper bound for non-adaptive testers, since they roughly follow from the fact that the query complexity of non-adaptive testing is at most quadratic in the query complexity of adaptive testing [1, 13].

Theorem 1.3 refers to graph properties that are testable in query complexity $q(n)/\epsilon$ such that $q(n) = \widetilde{O}(n^{-0.5})$. It establishes a conjecture in [12, Sec. 1.3] (which referred to the case of $g(n) = q(n)^{1-\frac{2}{t}}$, for every constant $t \in \mathbb{N}$) as a special case. Theorem 1.3 also answers (negatively) an open problem in [12, Sec. 1.3] that asks whether the foregoing relations (i.e., g's of the form $g(n) = q(n)^{1-\frac{2}{t}}$ for some $t \in \mathbb{N}$) are the only possible ones. We further generalized the result to obtain analogous bounds for any lower level of query complexity; that is, we replace the special case of $q(n) = O(n^{-0.5} \cdot \log n)$ by $q(n) = O(f(n) \log n)$ for any $f(n) \leq n^{-0.5}$.

Theorem 1.4 (generalizing Theorem 1.3 to lower levels of query complexity): *For every functions $f, g : \mathbb{N} \to \mathbb{N}$ such that $f(n) \leq \sqrt{n}$ and $g(m) \leq m$, there exists a graph property Π that satisfies the following two conditions:*

1. *There exists a general* (i.e., adaptive) *tester of Π that makes $O(\epsilon^{-1} \cdot f(n) \cdot \log n)$ queries, and any such tester must make $\Omega(f(n))$ queries.*

2. *Any non-adaptive tester must make $\Omega(g(f(n)) \cdot f(n))$ queries, and there exists such a tester that makes $O(\epsilon^{-2} \cdot g(f(n)) \cdot f(n) \cdot \log n)$ queries.*

   Actually, the upper bound is $O(\epsilon^{-2} \cdot g(f(n)) \cdot f(n) + \epsilon^{-1} \cdot f(n) \cdot \log n)$.

We mention that Part 1 establishes a hierarchy theorem for testing graph properties in the dense graph model. This theorem is slightly weaker than [11, Thm. 4] (i.e. it applies to any $f(n) \leq \sqrt{n}$ rather than to any $f(n) \leq n$ and has a $\log n$ factor gap between its bounds), but its proof seems simpler and is definitely totally different.

Lastly, we prove that essentially the same upper bounds can be obtained by one-sided error testers (i.e., testers that always accept graphs in Π).

Theorem 1.5 (a one-sided error version of Theorem 1.4):[2] *For every functions $f, g : \mathbb{N} \to \mathbb{N}$ such that $f(n) \leq \sqrt{n}$ and $g(m) \leq m$, there exists a graph property Π that satisfies the following two conditions:*

1. *There exists a one-sided error tester of Π that makes $O(\epsilon^{-1} \cdot f(n) \cdot \log^3 n)$ queries, and any tester must make $\Omega(f(n))$ queries.*

2. *Any non-adaptive tester must make $\Omega(g(f(n)) \cdot f(n))$ queries, and there exists a one-sided error non-adaptive tester that makes $O(\epsilon^{-2} \cdot g(f(n)) \cdot f(n) \cdot \log^3 n)$ queries.*

We stress that the lower bounds hold also for two-sided error testers. Note that the upper bounds have extra poly$(\log n)$ factors (compared to Theorem 1.4).

[2]Actually, the upper bounds hold provided that $\epsilon > n^{-0.49}$. Otherwise, the trivial tester that checks all entries works using $O(1/\epsilon^{4.1})$ queries.

1.3 Techniques

The notion of robustly self-ordered graphs and local self-ordering procedures for them, introduced and studied in our prior work [14], play a central role in our proofs. These notions and the relevant results are reviewed in the preliminaries (i.e., Section 2).[3]

Loosely speaking, a *robustly self-ordered graph* is as far as possible from having even approximate automorphisms; that is, for any t, any permutation of the vertices that displaces t out of the n vertices must "displace" $\Omega(t \cdot n)$ edges. In other words, a graph $G = ([n], E)$ is called robustly self-ordered if for every permutation $\pi : [n] \to [n]$ it holds that G and $\pi(G)$ differ on $\Omega(n) \cdot |\{v \in [n] : \pi(v) \neq v\}|$ vertex-pairs, where $\pi(G) = ([n], \{\{\pi(u), \pi(v)\} : \{u, v\} \in E\})$; that is, if π has t non-fixed-points, then the symmetric difference between the (labeled) graphs is $\Omega(t \cdot n)$.

As for *local self-ordering procedure for* G, given oracle access to an arbitrary isomorphic copy of G, denoted G', and a vertex v in G', it is required to identify the name (or location) of v in G, while making few queries to G'. That is, a local self-ordering procedure for G is a randomized algorithm that, on input $v \in [n]$ and oracle access to $G' = \pi(G)$, makes $\text{poly}(\log n)$ queries to G' and outputs $\pi^{-1}(v)$ (with probability at least $1 - n^{-c}$, for any desired constant c). We stress that π is *a priori* uknown to this procedure, but indeed $\pi^{-1}(v)$ is partial information about π that is obtained by the procedure.

1.3.1 The basic ideas

Our first contribution is showing that local self-ordering procedures do exist for some robustly self-ordered graphs. Specifically, we use the fact that random graphs are robustly self-ordered [14, Prop. 7.1] and prove that they have a local self-ordering procedure (in a very strong sense).[4]

As we advocated in [14], working with robustly self-ordered graphs offers a way of embedding (naturally ordered) bit strings in *unlabeled* graphs such that relative distances are approximately preserved. As for local self-ordering procedures, they offer a way of sampling these embedded bits *along with their location*. Hence, the robust self-ordering allows to transport lower bounds regarding ordered objects to the domain of unlabeled graphs, whereas local self-ordering offer a way to transport upper bounds in the same direction.

The source of the quadratic gap. Let us first present a testing problem (on ordered structures – matrices) that is the source of our quadratic gap, while noting that robust self-ordering and local self-ordering will come into play when embedding this problem in an unordered structure (unlabeled graphs). The property consists of a pair of n-by-n matrices $A = (a_{i,j})$ and $B = (b_{i,j})$ coupled with bijections (on their rows and columns), denoted $\pi_{\mathtt{r}}, \pi_{\mathtt{c}}, \phi_{\mathtt{r}}, \phi_{\mathtt{c}} : [n] \to [n]$, such that $a_{\pi_{\mathtt{r}}(i),\pi_{\mathtt{c}}(j)} = b_{\phi_{\mathtt{r}}(i),\phi_{\mathtt{c}}(j)}$. In this case, we may write $\pi(A) = \phi(B)$. We are given oracle access both to entries in the two matrices and to the four bijections; that is, we can ask for the $(i,j)^{\text{th}}$ entry of each of the matrices as well as for the value of each of the bijections at any point of our choice. The testing task is to determine whether such a sextuple satisfies the property or is far from satsfying it.[5]

Note that a non-adaptive tester must make $\Omega(n)$ queries to the entries of the matrices, because otherwise it is unlikely that it will make a query (i_1, j_1) to matrix A and a query (i_2, j_2) to matrix

[3]The following formulations are slightly different from those used in [14].

[4]This strong sense allows us to incur a lower overhead than what would follow from using a generic local self-ordering procedure. Specifically, our overhead is $O(\log n)$ rather than $\text{poly}(\log n)$.

[5]The definition of distance will be weighted, making each oracle have equal weight. Alternatively, each of the bijections is repeated n times.

B such that $(\pi_{\mathtt{r}}(i_1), \pi_{\mathtt{c}}(j_1)) = (\phi_{\mathtt{r}}(i_2), \phi_{\mathtt{c}}(j_2))$. Failing to make such a pair of queries does not allow for distinguishing the case that $\pi(A) = \phi(B)$ is random from the case that A and B are random and independent of one another, where in both cases all bijections are random. Here the Birthday Paradox is (negatively) applied to the set of entries.

In contrast, an adaptive tester may make $O(\sqrt{n})$ queries to each of the bijections and find (w.h.p.) i_1 and i_2 such that $\pi_{\mathtt{r}}(i_1) = \phi_{\mathtt{r}}(i_2)$ as well as j_1 and j_2 such that $\pi_{\mathtt{c}}(j_1) = \phi_{\mathtt{c}}(j_2)$. In this case, it has obtained two pairs (i_1, j_1) and (i_2, j_2) such that $(\pi_{\mathtt{r}}(i_1), \pi_{\mathtt{c}}(j_1)) = (\phi_{\mathtt{r}}(i_2), \phi_{\mathtt{c}}(j_2))$. Then, it can make the corresponding queries to A and B, respectively, which is something that a non-adaptive algorithm cannot do. Note that the answers are different with probability that equals the relative distance between the permuted matrices $\pi(A)$ and $\phi(B)$. Here the Birthday Paradox is (positively) applied to the set of rows (and columns), which are quadratically smaller.

Embedding the two-matrix problem in a graph property. Loosely speaking, we embed the two matrices in the connection between two robustly self-ordered graphs that have local self-ordering procedures. In this embedding, the local self-ordering procedures play the role of the (inverses of the) foregoing bijections, which are not given to us explicitly in the setting of testing graph properties. This offers us a way to transport the foregoing adaptive tester into one that tests graphs in which the two matrices are embedded. Loosely speaking, we let the matrix determine a bipartite graph between two parts of a robustly self-ordered graph, and use the local self-ordering procedure to emulate the foregoing bijections. On the other hand, the robustness of the self-ordering of the graph guarantees that one must respect the structure of the matrix as embedded (rather than freely permute rows and columns in a way that is not consistent with the original bijections).

We stress that *local self-ordering procedures for robustly self-ordered graphs* play a central role in our testers, and proving their existence in the dense graph model is one of the contributions of this paper. We actually present two different procedures: One that carries a small failure probability on each input (see Theorem 2.6), and one that makes no failures on almost all inputs (see Claim 5.5).

1.3.2 Beyond basic ideas

As stated above, our proofs are based on embedding Boolean matrices in robustly self-ordered graphs, but the question is how exactly is this done. One way is proposed in our prior work [14]: We take two robustly self-ordered graphs that exhibit a *gap between their ranges of vertex degrees* and connect them by a bipartite graph that corresponds to the Boolean matrix. As argued implicitly in [14, Lem. 9.3], this yields a robustly self-ordered graph, no matter which matrix is embedded (equiv., bipartite graph is used).

Specifically, in [14, Const. 9.2], the two graphs had the same edge density, since they are taken from the same construction, but they have a different size (by a constant factor) and so their ranges of vertex degrees are sufficiently far apart.[6] We use this idea when proving Theorems 1.2–1.4, but it relies on the fact that when we sample vertices of the combined graph we are likely to get a proportional number of vertices in each of the original two graphs. Hence, this method is not adequate for one-sided error testing (i.e., proving Theorem 1.5).

The alternative, employed in our proof of Theorem 1.5, is using two robustly self-ordered graphs of vastly *different edge density* (e.g., one with edge-density 0.1 and one with edge density 0.9). Furthermore, each induced subgraph of logarithmic size in each of these graphs will have approximately

[6]Specifically, all vertex degrees in one graph are at most $1.1k$, all vertex degrees in another are at least $3.4k$. The first graph has $2k$ vertices and the second has $7k$ vertices, and we connect them by two k-by-k bipartite graphs.

the same edge density as the entire graph. Moreover, in order to maintain the degree separation also in the combined graph, we use only bipartite graphs of edge density 0.5 ± 0.1 (with induced subgraphs that maintain this density).[7]

The fact that each induced subgraph of logarithmic size in the combined graph provides a good approximation of the degrees of almost all vertices allows us to determine to which part almost each vertex belongs. Analogous requirements are imposed on the local self-ordering procedures that we use towards obtaining one-sided error testers: Each subset of logarithmic size allows for self-ordering almost all vertices of each graph that satisfies the tested property (i.e., with the exception of a logarithmic number of vertices). In both cases, random subsets are used to detect graphs that are far from the property, whereas no subset causes rejection of graphs in the property.

Needless to say, the local self-ordering procedures for these non-explicit graphs are non-explicit themselves, and their computational complexity is huge. What is small (i.e., logarithmic in the size of the graph) is the number of queries that they make. The same holds also for the local self-ordering procedure that we use for the two-sided error tester. Although we know of explicit constructions of robustly self-ordered graphs [14, Part II], we do not know of a local self-ordering procedures for them (not even in the query complexity sense that suffices here). Instead, in this case too, we use local self-ordering procedures that work on the non-explicit robustly self-ordered graphs.

1.4 Complexities that also depend on the proximity parameter

As stated in Section 1.1, the primary focus of property testing is on the dependence of the query complexity on the size parameter (i.e., n). Indeed, Theorems 1.2–1.5 follow this disposition, but once established they raise the question of what happens in the more general case when one is also interested in the complexities that may depend arbitrarily on the proximity parameter (i.e., ϵ). In this section, we address this question.

Specifically, Theorems 1.2–1.5 refer to the dependence of the query complexity on the size of the tested graph (i.e., n), and focus on the case that the proximity parameter is set to some small positive constant. Actually, the upper bounds do specify the (polynomial) dependence on the (reciprocal of the) proximity parameter, but the lower bounds do not. Furthermore, the results do not refer to *arbitrary dependence* of the query complexity on the proximity parameter. In this section, we rectify this state of affairs for Theorems 1.2–1.4. We start by stating a special of our general result: This special case refers to query complexity that (essentially) depends only on the proximity parameter.

Theorem 1.6 (size-oblivious complexities, a special case):[8] *For every monotonically non-increasing function $f' : (0,1] \to \mathbb{N}$ and every constant $c \in (1,2]$, there exists a graph property Π that satisfies the following two conditions:*

1. *There exists a general* (i.e., adaptive) *tester of Π that makes $\widetilde{O}(\epsilon^{-2} \cdot f'(\Omega(\epsilon)))$ queries, and any such tester must make* $\min(\Omega(f'(O(\epsilon))), \sqrt{\epsilon n})$ *queries.*

2. *Any non-adaptive tester of Π must make* $\min(\Omega(f'(O(\epsilon))), \sqrt{\epsilon n})^c$ *queries, and there exists such a tester that makes $\widetilde{O}(\epsilon^{-2} \cdot f'(\Omega(\epsilon))^c)$ queries.*

[7]We do not provide explicit constructions for these ingredients; for example, a bipartite graph with the foregoing parameters constitutes an optimal two-source extractor (for which explicit constructions are still unknown, cf. [19, 4]).

[8]Actually, if $f'(\Omega(\epsilon)) > \sqrt{\epsilon n}$, then we get a lower bound related to $f'(\Omega(\epsilon'))$ such that ϵ' is minimal such that $f'(\Omega(\epsilon')) \leq \sqrt{\epsilon' n}$.

In particular, for any constant $d > 0$, we get general query complexity $\widetilde{O}(\epsilon^{-(d+2)})$ and non-adaptive query complexity $\Omega(\min(\epsilon^{-2d}, \sqrt{n}))$.

Theorem 1.6 is a special case of the following result.

Theorem 1.7 (a generalization of Theorem 1.4): *For every functions $f, g : \mathbb{N} \times (0,1] \to \mathbb{N}$ that are monotonically non-decreasing in the first parameter and monotonically non-increasing in the second parameter and every $n \in \mathbb{N}$ and $\epsilon > 0$ such that $f(n,\epsilon) \leq \sqrt{\epsilon n}$ and $g(m,\epsilon) \leq m$, there exists a graph property Π that satisfies the following two conditions:*

1. *There exists a general* (i.e., adaptive) *tester of Π that makes $\widetilde{O}(\epsilon^{-2} \cdot f(n, \Omega(\epsilon)))$ queries, and any such tester must make $\Omega(f(n, O(\epsilon)))$ queries.*
2. *Any non-adaptive tester of Π must make $\Omega(g(f(n,O(\epsilon))) \cdot f(n,O(\epsilon)))$ queries, and there exists such a tester that makes $\widetilde{O}(\epsilon^{-2} \cdot g(f(n,\Omega(\epsilon)),\Omega(\epsilon)) \cdot f(n,\Omega(\epsilon)))$ queries.*

For example, for any $c \geq d > 0$ and every f, g as in Theorem 1.4, we get general query complexity $\widetilde{O}(\epsilon^{-(c+2)}) \cdot f(n)$ and non-adaptive query complexity $\Omega(\min(\epsilon^{-(c+d)} \cdot g(n) \cdot f(n), \sqrt{n}))$.

We point out that Theorem 1.7 differs from Theorem 1.4 also when considering query complexities that depend only on the size of the graph. This is the case because the $O(\log n)$ factor that appears in the upper bounds of Theorem 1.4 is replaced in Theorem 1.7 by an $O(\log f(n, \Omega(\epsilon)))$ factor. On the other hand, Theorem 1.7 poses additional (natural) conditions on the functions f and g (and also has a larger slackness in terms of ϵ).

Theorem 1.7 is proved in two steps. First, we use graph blow-up to derive a new version of Theorem 1.4 from Theorem 1.3, where in the new version the $O(\log n)$ factor (in the upper bounds) is replaced by an $O(\log f(n))$ factor. This step uses ideas and results of [11] as well as new ideas: The former are used in order to establish the lower bounds, whereas the latter are used in order to establish the upper bounds. Specifically, we address the problem of testing properties derived by graph blow-up, which was avoided in [11] (since it applied graph blow-up to properties of maximum complexity (i.e., $\Theta(n^2)$)), whereas we apply it to properties of intermediate complexity (i.e., $\widetilde{\Theta}(n^{1/2})$). In particular, we prove that the set of n/n'-factor blow-ups of the n'-vertex graphs (presented in the proof) of Theorem 1.3 can be tested using $\text{poly}(1/\epsilon) \cdot \widetilde{O}(\sqrt{n'})$ queries. Next, we apply the methodology of [7] to the result of the first step. In general, this methodology transforms existential results regarding graph properties of almost arbitrary size-dependent query complexity to results that support almost any query complexity. (In [7], this methodology was applied to the results of [11].)

We stress that the proof of Theorem 1.7 uses ideas that are significantly different from those that appeared in the proofs of Theorems 1.2–1.5. The tools used include graph blow-up, distribution testing (*sic*), and "packing" several differently padded properties into a single one. The new ideas appear in the first step, which is captured by Theorem 6.1. Lastly, we warn that the proof of Theorem 1.7 makes essential use of approximations of the density of subsets of vertices, and thus it yields a two-sided error tester.

1.5 Organization

In Section 2 we review the definitions of robustly self-ordered graphs and local self-ordering procedures for them, and prove the existence of the latter in random graphs (Theorem 2.6). These notions and the corresponding results play a central role in establishing our property testing results (i.e., Theorems 1.2–1.7). Our most basic result result (Theorem 1.2), which asserts an almost quadratic gap between non-adaptive and adaptive testers, is proved in Section 3.

The proof of Theorem 1.2 also serves as basis for modifications that establish Theorems 1.3–1.7. The modifications used to establish Theorems 1.3 and 1.4 are presented in Section 4. While these modifications are relatively simple, establishing Theorem 1.5 requires several additional ideas (see, e.g., the proof of Claim 5.5, the design of Construction 5.7, and the proof of Proposition 5.9). In fact, the proof of Theorem 1.5, presented in Section 5, is the most technically involved part of this paper. An overview of this proof is provided at the beginning of Section 5.

The proof of Theorem 1.7 appears in Section 6. We stress that Section 5 and Section 6 are independent of one another, and can be read in arbitrary order. Both sections build directly on Sections 3 and 4.

The appendices contain proofs that have appeared elsewhere. They are included for sake of self-containment and because we shall refer to sub-claims that appear in them. We also include some open problems (in Section 7).

2 Preliminaries

The following definitions and results are mostly reproduced from [14], with a few modifications, which will be spelled out. One of these modifications is merely in notation, and others are substantial. Unless explicitly stated differently, by graphs we mean labeled (simple) graphs.

Self-ordered graphs. For a graph $G=(V,E)$, and a bijection $\phi:V\to V'$, we denote by $\phi(G)$ the graph $G'=(V',E')$ such that $E'=\{\{\phi(u),\phi(v)\}:\{u,v\}\in E\}$, and say that G' is isomorphic to G. The set of automorphisms of the graph $G=(V,E)$, denoted $\mathsf{aut}(G)$, is the set of permutations that preserve the graph G; that is, $\pi\in\mathsf{aut}(G)$ if and only if $\pi(G)=G$. We say that a graph is asymmetric (equiv., self-ordered) if its set of automorphisms is a singleton, which consists of the trivial automorphism (i.e., the identity permutation). Following [14], we actually prefer the term *self-ordered*, because we take the perspective that is offered by the following equivalent definition.

Definition 2.1 (self-ordered (a.k.a asymmetric) graphs): *The graph $G=([n],E)$ is* self-ordered *if for every graph $G'=(V',E')$ that is isomorphic to G there exists a unique bijection $\phi:V'\to[n]$ such that $\phi(G')=G$.*

In other words, given an isomorphic copy $G'=(V',E')$ of a fixed graph $G=([n],E)$, there is a unique bijection $\phi:V'\to[n]$ that orders the vertices of G' such that the resulting graph (i.e., $\phi(G')$) is identical to G. Indeed, if $G'=G$, then this unique bijection is the identity permutation.

Robustly self-ordered graphs. In this work, we use a notion, introduced by us in [14] and called *robust self-ordering*, which is a quantitative version self-ordering. Unlike in [14, Part I], our focus in this work is on dense graphs with a high level of robust self-ordering. Hence, we say that a graph $G=([n],E)$ is robustly self-ordered if, for every permutation $\pi:[n]\to[n]$, the size of the symmetric difference between G and $\pi(G)$ (i.e., $|E\triangle\{\{\pi(u),\pi(v)\}:\{u,v\}\in E\}|$) is $\Omega(n)$ times the number of non-fixed-points under π (i.e., $|\{i\in[n]:\pi(i)\neq i\}|$).[9]

[9]This notation is a "scaled up" version of the notation in [14]. Specifically, in [14, Def. 1.2] a graph was defined to be robustly self-ordered if, for every permutation $\pi:[n]\to[n]$, the size of the symmetric difference between G

Definition 2.2 (robustly self-ordered graphs): *A graph $G = ([n], E)$ is said to be γ-*robustly self-ordered *if for every permutation $\pi : [n] \to [n]$ it holds that*

$$|E \triangle \{\{\pi(u), \pi(v)\} : \{u, v\} \in E\}| \ \geq \ \gamma \cdot n \cdot |\{i \in [n] : \pi(i) \neq i\}|. \qquad (1)$$

An infinite family of graphs $\{G_n = ([n], E_n)\}_{n \in \mathbb{N}}$ is called robustly self-ordered *if there exists a constant $\gamma > 0$ such that for every n the graph G_n is γ-robustly self-ordered.*

Note that $|E_n \triangle \{\{\pi(u), \pi(v)\} : \{u, v\} \in E_n\}| \leq n \cdot |\{i \in [n] : \pi(i) \neq i\}|$ always holds.

While our focus in [14, Part II] was on constructing robustly self-ordered graphs, here we are content with their mere existence. Actually, it will be beneficial for us to use the fact that, with high probability, a random graph is robustly self-ordered.

Theorem 2.3 (a random graph is robustly self-ordered [14, Prop. 7.1]): *A random n-vertex graph $G_n = ([n], E_n)$ is $\Omega(1)$-robustly self-ordered with probability $1 - \exp(-\Omega(n))$.*

For sake of self-containment, the proof is reproduced in Appendix A. We mention that the result is implicit in the proof of [16, Thm. 3.1].

Locally self-ordering a graph. By Definition 2.1 a graph $G = ([n], E)$ is called self-ordered if for every graph $G' = (V', E')$ that is isomorphic to G there exists a unique bijection $\phi : V' \to [n]$ such that $\phi(G') = G$. One reason for our preferring the term "self-ordered" over the classical term "asymmetric" is that we envision being given such an isomorphic copy $G' = (V', E')$ and asked to find its unique isomorphism to G, which may be viewed as ordering the vertices of G' according to (their name in) G. While the foregoing formulation is global in nature (i.e., one is given the entire graph and is asked to find the entire isomorphism), here we are interested in its *local* version: Given a vertex in G' (and oracle access to the adjacency predicate of G'), we wish to find the corresponding vertex in G while making $\text{poly}(\log n)$ many queries to G'.

We stress that, unlike in [14, Sec. 4.4], we *do not* require this task to be performed in $\text{poly}(\log n)$-time; we only bound the number of queries to G'. Another difference is that we allow randomized algorithms; these algorithms may fail (but not err) with small probability. We mention that, even under these relaxations, it is not clear whether graphs having local self-ordering procedures exist[10],

Definition 2.4 (locally self-ordering a self-ordered graph): *We say that a self-ordered graph $G = ([n], E)$ is* locally self-ordered *if there exists a randomized algorithm that, given a vertex v in any graph $G' = (V', E')$ that is isomorphic to G and oracle access to the adjacency predicate of G', makes* $\text{poly}(\log n)$ *queries to G', fail with probability at most $1/2$, and otherwise output $\phi(v) \in [n]$ for the unique bijection $\phi : V' \to [n]$ such that $\phi(G') = G$.*

Indeed, the isomorphism ϕ orders the vertices of G' in accordance with the original (or target) graph G. We stress that the foregoing algorithm may depend arbitrarily on the graph G and may not be efficient. Effectively, it is only required that, given $v \in V'$, with probability at least $1/2$, the answers given to the queries to G' determine $\phi(v) \in [n]$, and otherwise we get a failure notice

and $\pi(G)$ is $\Omega(1)$ times the number of non-fixed-points under π. Likewise, the parameter γ in [14, Def. 1.2] is the constant hidden in the $\Omega(1)$ notation (rather than in the $\Omega(n)$ notation).

[10] An indication to the non-triviality of this problem arises from the fact that most graphs (equiv., random graphs) do not have a deterministic local self-ordering procedure. This fact can be proved analogously to Footnote 12.

rather than a wrong answer. Clearly, the failure probability can be reduced to $n^{-\omega(1)}$ by repetitions, while maintaining a query complexity of $\text{poly}(\log n)$, but since we care about the specific bounds we define a stronger notion that we can achieve at low cost (see Theorem 2.6). This notion, called *reliable locators*, is a subset S of the vertices of G such that the subgraph of G induced by S is self-ordered and unique in G, and every other vertex has a unique "signature" with respect to S (i.e., its sequence of adjacencies with S is unique).

Definition 2.5 (reliable locator of a self-ordered graph): *A set of vertices $S \subset [n]$ is called a* reliable locator *of a graph $G = ([n], E)$ if the following two conditions hold*

1. *The subgraph of G induced by S is self-ordered and is not isomorphic to any other induced subgraph of G.*
2. *For every $v \in [n] \setminus S$, the adjacencies of v with S uniquely determine v; that is, for every $u \neq v$ in $[n] \setminus S$ there exists $s \in S$ such that $\{u, s\} \in E$ if and only if $\{v, s\} \notin E$.*

We stress that it is *not* required that all induced subgraphs of G are not isomorphic to one another; it is only required that none of the other induced subgraphs is isomorphic to the reliable locator. Hence, the first condition allows for identifying reliable locators in any graph G' that is isomorphic to G, since S' is a reliable locator of G' if and only if the subgraph of G' induced by S' is isomorphic to the subgraph of G induced by any of its reliable locators. The first condition also allows for self-ordering the vertices of S' (i.e., these vertices can be identified via structure of the (unlabeled) induced subgraph). The second condition then allows us to locally self-order each other given vertex in G', where a vertex outside S' is identified via its adjacencies to the identified vertices of S'.

We shall prove a stronger statement for random graphs, asserting that, in a random graph (whp), at least 99% of the *logarithmically-sized* sets are reliable locators. Note that the fact that reliable locators exist will not be good enough for our applications, where we shall be given oracle access to an isomorphic copy of G and need to find a reliable locator for it (rather than for G). In contrast, the fact that random subsets are reliable locators of G will allow us to find reliable locators in any graph that is isomorphic to G.

Theorem 2.6 (locally self-ordering a random graph):[11] *With probability $1 - (1/\text{poly}(n))$ over the uniform distribution of n-vertex graphs G_n, all but $1/\text{poly}(n)$ fraction of the $O(\log n)$-sized sets are reliable locators for G_n, where* poly *denotes any polynomial* (and the hidden constant in the O-notation is proportional to its degree).

We stress again that a reliable locator of G' that is isomorphic to G is uniquely identified by the unlabeled subgraph of G' induced by it, and that given this subgraph the set is recognized as a reliable locator by a thought experiment on G. That is, if $G' = (V', E')$ is isomorphic to G and $G = \phi(G')$ for some unknown to us bijection $\phi : V' \to [n]$, then given S' we can determine whether S' is a reliable locator of G' (equiv., $\phi(S')$ is a reliable locator of G) by inspecting the subgraph of G' induced by S' and checking whether this subgraph is isomorphic to the subgraph induced by some reliable locator of G. We stress that the latter check is a thought experiment that is based

[11] We mention that the same holds for graphs that are generated by a $O(\log n)^2$-wise independent process. This is the case since the proof takes a (careful) union bound over events that refer to subgraphs of the random n-vertex graph that are induced by vertex-sets of size $O(\log n)$.

on the fact that we know G, and so we know all its reliable locators and the subgraphs that they induce.

Proof: The driving observation is that, for a random graph, with overwhelmingly high probability, disjoint subsets of logarithmic size induce subgraphs that are not isomorphic to one another. An analogous claim holds also for non-disjoint subsets, but the probability is less overwhelming in this case. In particular, it is not true that *each* such subset induces a subgraph that is not isomorphic to any other induced subgraph.[12] But it is true that, in a random graph (with high probability), almost all subsets have this feature. This follows from the fact that, in a random graph, each subset has this feature with high probability.

Claim 2.6.1 (a fixed $O(\log n)$-subset in a random n-vertex graph satisfies Condition 1 of Definition 2.5): *For sufficiently large $\ell = O(\log n)$ it holds that for every fixed ℓ-subset S of $[n]$, with probability $\exp(-\Omega(\ell))$ over the choice of G_n, the subgraph of G_n induced by S is not isomorphic to the subgraph of G_n induced by any other ℓ-subset of $[n]$. Furthermore, with probability $\exp(-\Omega(\ell))$ over the choice of G_n, the subgraph of G_n induced by S is self-ordered.*

Note that the failure probability is not small enough to support a union bound over all ℓ-subsets of $[n]$. Claim 2.6.1 follows as a special case of [8, Clm. 8.2], which referred to the case that only the adjacencies of vertices in S are random. For sake of self-containment, the proof is reproduced in Appendix B.

Combining Claim 2.6.1 with the fact that, in a random graph G_n, with probability $\binom{n}{2} \cdot 2^{-\ell}$, the vertices in $[n] \setminus S$ have different neighborhood in S (i.e., satisfies Condition 2), we conclude that each ℓ-subset is a reliable locator of at least a $1 - \exp(-\Omega(\ell))$ fraction of the n-vertex graphs. The claim follows by an averaging argument, while recalling that we can pick $\ell = O(\log n)$ to be large enough so that $\exp(-\Omega(\ell)) = 1/\text{poly}(n)$. ■

Corollary 2.7 (by Theorems 2.3 and 2.6): *There exists a family of robustly self-ordered graphs $\{G_n = ([n], E_n)\}_{n\in\mathbb{N}}$ such that G_n is an n-vertex graph for which all but at most a $1/\text{poly}(n)$ fraction of the $O(\log n)$-sized sets are reliable locators. Furthermore, each vertex in G_n has degree $(0.5 \pm 0.01) \cdot n$.*

3 The Main Result

In this section we prove Theorem 1.2, while hinting that Theorems 1.3–1.5 will be proved (in Sections 4 and 5) by modifications to the proof presented here.

Theorem 3.1 (Theorem 1.2, restated): *There exists a graph property Π that is testable by adaptive algorithms that make $O(\epsilon^{-1} \cdot \sqrt{n} \cdot \log n)$ queries but testing it non-adaptively requires $\Omega(n)$ queries, where n denotes the number of vertices in the tested graph and ϵ denotes the proximity parameter.*

[12]Consider a collection $\mathcal{C}$ of $(\ell-1)$-subsets of $[n/2]$ such that the pairwise intersections are of size at most $\log_2 n < \ell/2$. Specifically, we may have $|\mathcal{C}| = \exp(\Omega(\log n)^2)$. Now, for each $S \in \mathcal{C}$ and $\{i,j\} \in \binom{\{(n/2)+1,\dots,n\}}{2}$, consider the 0-1 random variable $\zeta_{S,\{i,j\}} = \zeta_{S,\{i,j\}}(G)$ indicating whether the subgraphs of a random graph G that is induced by $S \cup \{i\}$ is identical to the subgraph of G induced by $S \cup \{j\}$. Note that $p = \mathrm{E}[\zeta_{S,\{i,j\}}] = 2^{-(\ell-1)} \gg 1/|\mathcal{C}|$, and that $\zeta_{S_1,\{i_1,j_1\}}$ and $\zeta_{S_2,\{i_2,j_2\}}$ are independent if $\{i_1,j_1\} \neq \{i_2,j_2\}$. Furthermore, for different sets $S_1, S_2 \in \mathcal{C}$, the co-variance of $\zeta_{S_1,\{i,j\}}$ and $\zeta_{S_2,\{i,j\}}$ is upper-bounded by $q = \Theta(2^{-|S_1 \cup S_2|}) = O(2^{-(2(\ell-1)-\log_2 n)}) = O(n \cdot p^2)$. Hence, the variance of the sum of all $\zeta_{S,\{i,j\}}$'s is $O(|\mathcal{C}| \cdot n^2 \cdot p) + O(|\mathcal{C}|^2 \cdot n^2 \cdot q)$, which equals $O(|\mathcal{C}|^2 \cdot n^3 \cdot p^2)$. Using Chebyshev's inequality, it follows that the probability that none of the events hold is $\frac{O(|\mathcal{C}|^2 \cdot n^3 \cdot p^2)}{(|\mathcal{C}| \cdot n^2 \cdot p)^2} = O(n^{-1})$.

Proof: For every n, we set $k = n/9$ and $\ell = O(\log k)$, and use a family of robustly self-ordered graphs $\{G_k = ([k], E_k) : k \in \mathbb{N}\}$ such that for each G_k all but an $\exp(-\Omega(\ell))$ fraction of the ℓ-sized sets are reliable locators. Furthermore, each vertex in G_k has degree $(0.5 \pm 0.01) \cdot k$. Recall that such graphs are provided by Corollary 2.7. Using these graphs we consider the following construction that consists of a copy of G_{2k} and a copy of G_{7k} such that the first (resp., last) k vertices of G_{2k} and the first (resp., last) k vertices of G_{7k} are connected according to some k-by-k Boolean matrix. These two matrices will be identical in the case of graphs in the property, and will be far apart in the construction used for showing the lower bound on non-adaptive testers.

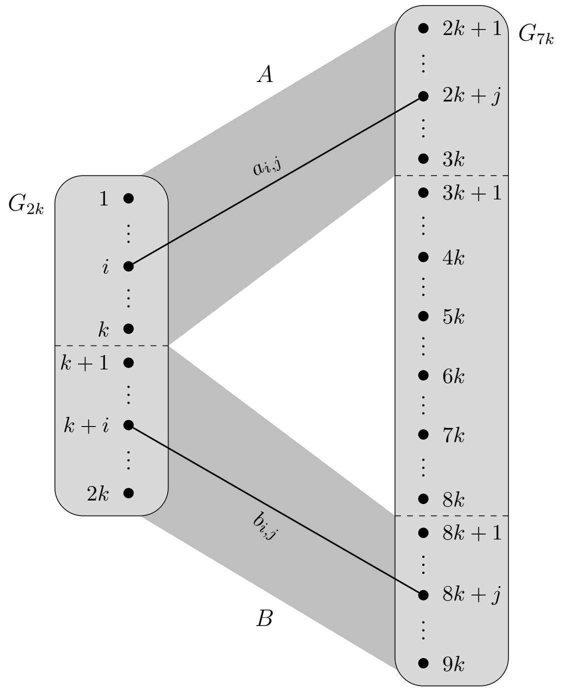

Figure 1: The graph $G_{A,B}$.

Construction 3.1.1 (the graph property Π): *Let $G_{2k} = ([2k], E_{2k})$ and $G_{7k} = ([7k], E_{7k})$ be as postulated in Corollary 2.7.*

- *For two k-by-k Boolean matrices $A = (a_{i,j})$ and $B = (b_{i,j})$, we define the graph $G_{A,B} = ([9k], E_{A,B})$ such that*

$$\begin{aligned} E_{A,B} &= E_{2k} \cup \{\{2k+i, 2k+j\} : \{i,j\} \in E_{7k}\} \\ &\cup \{\{i, 2k+j\} : i,j \in [k] \wedge a_{i,j} = 1\} \cup \{\{k+i, 8k+j\} : i,j \in [k] \wedge b_{i,j} = 1\}. \end{aligned} \quad (2)$$

  That is, $G_{A,B}$ consists of a copy of G_{2k} and a copy of G_{7k} that are connected by two bipartite graphs that are determined by A and B, respectively. The first bipartite graph connects $[k]$ to $\{2k+1, ..., 3k\}$ and the second bipartite graph connects $\{k+1, ..., 2k\}$ to $\{8k+1, ..., 9k\}$. See Figure 1.

- *We define the property* $\Pi = \bigcup_{n\in\mathbb{N}} \Pi_n$ *such that* Π_n *is the set of all graphs that are isomorphic to some* n*-vertex graph* $G_{A,A}$*; that is,*

$$\Pi_{9k} = \left\{ \pi(G_{A,A}) : A \in \{0,1\}^{k\times k} \wedge \pi \in \mathrm{Sym}_{9k} \right\} \tag{3}$$

where Sym_{9k} *denote the set of all permutations over* $[9k]$*.*

Note that, given a graph of the form $\pi(G_{A,A})$, the vertices of G_{2k} are easily identifiable (as having degree at most $0.51 \cdot 2k + 2k < 3.1k$).[13]

Claim 3.1.2 (lower bound for non-adaptively testing Π): *Any non-adaptive tester for* Π_{9k} *has query complexity* $\Omega(k)$*.*

Proof: We first show that $\Omega(k)$ non-adaptive queries are required in order to distinguish the following two distributions.

1. The uniform distribution over Π_{9k}; that is, the distribution generated by picking uniformly $A \in \{0,1\}^{k\times k}$ and $\pi \in \mathrm{Sym}_{9k}$, and outputting $\pi(G_{A,A})$.
2. The distribution generated by picking uniformly and independently $A, B \in \{0,1\}^{k\times k}$ and $\pi \in \mathrm{Sym}_{9k}$, and outputting $\pi(G_{A,B})$.

The key point here is that a non-adaptive machine determines its queries beforehand, and so it suffices to analyze the probability that a fixed sequence of q queries distinguishes the two distributions. Furthermore, the two q-long sequences of answers provided by these two distributions to a fixed q-long sequence of queries are distributed identically, unless the sequence of queries contains two queries (u_1, v_1) and (u_2, v_2) that refer to corresponding entries in the matrices A and B; that is, unless $(\pi^{-1}(u_1), \pi^{-1}(v_1) - 2k) \in [k]^2$ equals $(\pi^{-1}(u_2) - k, \pi^{-1}(v_2) - 8k)$.[14] However, the probability that this event occurs is upper-bounded by $\binom{q}{2} \cdot \frac{2k^2}{(9k)^2} \cdot \frac{1}{(9k)^2 - 1} < q^2/6000k^2$.

On the other hand, distinguishing these two distributions is essential for testing Π, since the first distribution is supported by Π, whereas (as shown next) graphs selected according to the second distribution are $\Omega(1)$-far from Π, with overwhelmingly high probability. To prove the latter claim, consider a random graph $\pi(G_{A,B})$ generated according to the second distribution and an arbitrary graph $\phi(G_{C,C}) \in \Pi$. Then, with probability at least $1 - \exp(-\Omega(k^2))$, the matrices A and B disagree on at least $k^2/3$ of their entries. Fixing such A and B (as well as π), we consider the following cases, where $\gamma \in (0,1]$ is a constant such that all G_k's are γ-robustly self-ordered (see Definition 2.2).

Case 1: $d \overset{\text{def}}{=} |\{i \in [9k] : \pi(i) \in [2k] \wedge \phi(i) \notin [2k]\}| \geq \gamma \cdot k/9$.

Using the different in degrees between the vertices in G_{2k} and G_{7k}, which is at least $0.3k$, it follows that the symmetric difference between $\pi(G_{A,B})$ and $\phi(G_{C,C})$ is at least $d \cdot 0.3k > 0.03\gamma \cdot k^2$.

[13]In contrast, the vertices of G_{7k} have degree at least $0.49 \cdot 7k > 3.4k$.

[14]Indeed, assuming that $\pi^{-1}(u_1), \pi^{-1}(u_2) - k \in [k]$ and $\pi^{-1}(v_1) - 2k, \pi^{-1}(v_2) - 8k \in [k]$, these queries are answered by the values $a_{\pi^{-1}(u_1),\pi^{-1}(v_1)-2k}$ and $b_{\pi^{-1}(u_2-k),\pi^{-1}(v_2)-8k}$, and these values are informative (towards distinguishing the two distributions) if and only if $(\pi^{-1}(u_1), \pi^{-1}(v_1) - 2k) = (\pi^{-1}(u_2 - k), \pi^{-1}(v_2) - 8k)$.

***Case 2*:** $d < \gamma \cdot k/9$ and $d' \stackrel{\rm def}{=} |\{i \in [9k] : \pi(i) \neq \phi(i)\}| \geq k/4$.

Using the γ-robust self-ordering of the graphs G_{2k} and G_{7k}, it holds that the symmetric difference between $\pi(G_{A,B})$ and $\phi(G_{C,C})$ is at least $(\gamma \cdot (d'-d) - d) \cdot 2k$, which is at least $(\gamma \cdot d' - 2d) \cdot 2k = \Omega(k^2)$.

(Letting $D_1 = \{i \in [9k] : \pi(i) \in [2k] \wedge \phi(i) \notin [2k]\}$ and $D_2 = \{i \in [9k] : \pi(i) \notin [2k] \wedge \phi(i) \in [2k]\}$, we only count on the contribution of the vertices in either $\{i \in [9k] \setminus D_1 : \pi(i) \in [2k] \setminus \{\phi(i)\}\}$ or $\{i \in [9k] \setminus D_2 : \pi(i) \in \{2k+1, ..., 9k\} \setminus \{\phi(i)\}\}$, and subtract for each contribution d units per their potential neighbours in either D_1 or D_2.)

***Case 3*:** $d' < k/4$.

In this case, we consider the set $D = \{i \in [9k] : \pi(i) \neq \phi(i)\}$, which is smaller than $k/4$, and the set

$$\Delta = \{(i,j) \in [k]^2 : a_{i,j} \neq b_{i,j} \ \& \ i, k+i \notin D \ \& \ 2k+j, 8k+j \notin D\}.$$

Recalling that $|\{(i,j) \in [k]^2 : a_{i,j} \neq b_{i,j}\}| \geq k^2/3$, it follows that $|\Delta| \geq \frac{k^2}{3} - |D| \cdot k = \Omega(k^2)$. Lastly note that each entry in Δ contributes one unit to the symmetric difference between $\pi(G_{A,B})$ and $\phi(G_{C,C})$, since the corresponding adjacencies in $\pi(G_{A,B})$ are different whereas these adjacencies are equal in $\phi(G_{C,C})$. We stress that $(i,j) \in \Delta$ contributes to the symmetric difference because this entry appears in the same location in both graphs (i.e., $\pi(i) = \phi(i)$, $\pi(k+i) = \phi(k+i)$, $\pi(2k+j) = \phi(2k+j)$, and $\pi(8k+j) = \phi(8k+j)$), whereas $a_{i,j} \neq b_{i,j}$ implies that either $a_{i,j} \neq c_{i,j}$ or $a_{i,j} \neq c_{i,j}$.

Hence, we have shown that in all cases the symmetric difference between $\pi(G_{A,B})$ and $\phi(G_{C,C})$ is $\Omega(k^2)$, whereas these are $O(k)$-vertex graphs. ■

Claim 3.1.3 (upper bound for adaptively testing Π): *There exists an adaptive tester for Π_{9k} of query complexity $O(\epsilon^{-1} \cdot \sqrt{k} \cdot \log k)$.*

Proof: The basic idea is for the tester to first identify the copies of G_{2k} and G_{7k} in the tested graph G', and then to index vertices in G' according to their location in G_{2k} and G_{7k} via the local self-ordering procedure (specifically via reliable locators). If these procedures fail, then we can definitely reject. More generally, we test whether G' is close to some graph of the form $\pi(G_{A,B})$ for some matrix pair (A,B) and a bijection $\pi : [9k] \to [9k]$. Assuming that this is the case, we generate samples of $O(\sqrt{k}/\epsilon)$ vertices in the two parts of G', and expect to obtain $\Omega(1/\epsilon)$ pairs of vertices (of G') that correspond to vertex-pairs $(i, k+i)$ in G_{2k} and $\Omega(1/\epsilon)$ pairs of vertices (of G') that correspond to vertex-pairs $(j, 6k+j)$ in G_{7k}. These pairs allow us to compare $\Omega(1/\epsilon)^2$ corresponding entries in the two matrices, since the $(i,j)^{\rm th}$ entry in A (resp., in B) is represented by the adjacency of the $i^{\rm th}$ vertex of G_{2k} and the $j^{\rm th}$ vertex of G_{7k} (resp., the $k+i^{\rm th}$ vertex of G_{2k} and the $6k+j^{\rm th}$ vertex of G_{7k}). Each of these entries is uniformly distributed in $[k] \times [k]$, and $\Omega(1/\epsilon)$ of them are independently distributed (and so we shall use them).

The crucial point is that we apply the Birthday paradox to the k rows (resp., columns) of these matrices rather than to their k^2 entries. Adaptivity is used in order to query the relevant entries (rather than querying all entries in the generalized rectangle spanned by the sampled rows and columns). We stress that the collisions (of rows and of columns, which are indexed by vertices in G_{2k} and G_{7k}, respectively) can be identified only after self-ordering the sampled vertices, and identification is performed using the local self-ordering procedure.

The actual tester. Recalling that $\ell = O(\log n)$ and following the foregoing outline, our tester first selects a sample that will be used to identify the G_{2k} and G_{7k} parts of G' and to index the vertices in these parts (see Steps 1 and 2). Then, an additional sample is chosen and it is used for testing that G' has the form $\pi(G_{A,B})$ for some matrices A and B and a permuation π (see Steps 3–5). This additional sample is large enough to cause collisions among rows (resp., among columns) of the two matrices, which in turn allow for testing the equality of A and B (in Step 6). Specifically, the tester proceeds as follows, on input $\epsilon > 0$ and oracle access to $G' = ([9k], E')$.

1. *Selecting a sample for degree estimation*: We select uniformly at random a set of 9ℓ vertices, denoted S, and inspect the subgraph of G' induced by S. If it contains less than 1.9ℓ or more than 2.1ℓ vertices of degree at most 3.1ℓ, then we halt and reject. (We may also reject if any of the other vertices has degree less than 3.4ℓ, but this is unuseful because this check is performed in Step 3 based on a larger sample.)

   In the sequel, we say that a vertex has low degree if it contains at most 3.1ℓ neighbors in S; otherwise, we say that it has high degree.

   (Recall that a graph in Π_{9k} has $2k$ vertices of degree at most $0.51 \cdot 2k + 2k < 3.1k$ and $7k$ vertices of degree at least $0.49 \cdot 7k > 3.4k$.)

2. *Finding reliable locators in the initial sample*: Let S_1 be a random ℓ-subset of the low degree vertices selected in Step 1, and S_2 be a random ℓ-subset of the high degree vertices. (Recall that S contains more than ℓ low (resp., high) degree vertices.) If the subgraph of G' induced by S_1 (resp., by S_2) is not isomorphic to the subgraph of G_{2k} (resp., of G_{7k}) that is induced by some reliable locator of G_{2k} (resp., of G_{7k}), then we halt and reject.

   Otherwise, we let π_1 (resp., π_2) denote the unique isomorphism between the subgraph of G' induced by S_1 (resp., S_2) and the subgraph of G_{2k} (resp., of G_{7k}) induced by the (unique) reliable locator that we identified for G_{2k} (resp., for G_{7k}).

   (Note that no queries are made in this step: The subgraphs of G_{2k} and G_{7k} that are induced by the various ℓ-subsets are hard-wired in the tester.)

3. *Sampling the two parts of the graph and locating the sampled vertices*: We select uniformly at random a set of $O(\sqrt{k}/\epsilon)$ vertices, denoted R, and let R_1 and R_2 denote the low and high degree vertices, where these degrees are approximated according to S. That is, for each vertex $r \in R$, we check its neighborhood in S, by making $|S|$ queries.

   If R_2 contains a vertex of degree less than 3.4ℓ, then we reject.

   Using the local self-ordering procedure (based on the reliable locator S_1 (resp., S_2)), we determine the location in G_{2k} (resp., G_{7k}) of each vertex in R_1 (resp., R_2). Specifically, we determine that vertex v in R_1 corresponds to vertex i in G_{2k} if the sequence of adjacencies of v with S_1 (in G') match the adjacencies of i with $\pi_1(S_1)$ (in G_{2k}); that is, if for every $s \in S_1$ it holds that $\{v, s\} \in E'$ if and only if $\{i, \pi_1(s)\} \in E_{2k}$. Ditto for $v \in R_2$ (using S_2 and π_2).

   If any of these invocation fails, then we reject. Ditto if two different vertices were assigned the same location. Otherwise, we define $\pi_1 : R_1 \to [2k]$ and $\pi_2 : R_2 \to [7k]$ (or rather extend them from S_i to $S_i \cup R_i$) accordingly; that is, $\pi_1(v)$ (resp., $\pi_2(v)$) is the location of $v \in R_1$ (resp., $v \in R_2$) in G_{2k} (resp., G_{7k}).

4. *Using the sample to test isomorphism of each part of G' to the corresponding part in a generic $G_{A,B}$*: We test that the subgraph of G' induced by R_1 (resp., R_2) equals the subgraph of G_{2k} induced by $\pi_1(R_1)$ (resp., the subgraph of G_{7k} induced by $\pi_2(R_2)$). This test is performed by checking $O(1/\epsilon)$ vertex-pairs at random; that is, for every selected vertex-pair $(u,v) \in R_1{\times}R_1$, we check whether its adjacency in G' fits the adjacency of $(\pi_1(u),\pi_1(v))$ in G_{2k} (and ditto for R_2 and G_{7k} (via π_2)).

5. *Using the sample to complete a test of isomorphism of G' to a generic $G_{A,B}$*: We test that there are no edges between $R_{1,1} \stackrel{\rm def}{=} \{u \in R_1 : \pi_1(u) \in [k]\}$ and $R_{2,2} \stackrel{\rm def}{=} \{v \in R_2 : \pi_2(v) \in \{k+1,...,7k\}\}$, and ditto between $R_{1,2} \stackrel{\rm def}{=} \{u \in R_1 : \pi_1(u) \in \{k+1,...,2k\}\}$ and $R_{2,1} \stackrel{\rm def}{=} \{v \in R_2 : \pi_2(v) \in [6k]\}$. This test is performed by checking $O(1/\epsilon)$ vertex-pairs at random; that is, for every selected vertex-pair $(u,v) \in (R_{1,1} \times R_{2,2}) \cup (R_{1,2} \times R_{2,1})$, we reject if u is connected to v in G'.

6. *Testing that $A = B$*: We call a pair $(i,j) \in [k]^2$ a matrix-collision if there exists $u_1, v_1 \in R_1$ and $u_2, v_2 \in R_2$ such that $\pi_1(u_1) = i = \pi_1(v_1) - k$ and $\pi_2(u_2) = j = \pi_2(v_2) - 6k$. In such a case, we call i a row-collision, and call j a column-collision.

   (Note that, if $G' \in \Pi$, then we are likely to see $\Theta(1/\epsilon)$ row-collision (resp., column-collision). Furthermore, in this case, every matrix-collision (i,j) satisfies $\{\pi_1^{-1}(i), \pi_2^{-1}(j)\} \in E'$ if and only if $\{\pi_1^{-1}(k+i), \pi_2^{-1}(6k+j)\}$, since these edges correspond to the $(i,j)^{\rm th}$ entry in the matrix A such that G' is isomorphic to $G_{A,A}$.)

   Letting I (resp., J) denote the set of row-collisions (resp., column-collision), we select a random set of $\min(|I|,|J|,\Theta(1/\epsilon))$ disjoint pairs $P \subseteq I \times J$ (i.e., distinct $(i_1,j_1),(i_2,j_2) \in P$ satisfy both $i_1 \neq i_2$ and $j_1 \neq j_2$). For each $(i,j) \in P$, we query G' on the pairs $(\pi_1^{-1}(i), \pi_2^{-1}(j))$ and $(\pi_1^{-1}(k+i), \pi_2^{-1}(6k+j))$, and reject if the answers are different.

   If we did not reject so far, then we accept.

We stress that in Step 6 we only make $O(1/\epsilon)$ queries; these queries are determined adaptively based on the information gathered in Step 3 (i.e., the values $(\pi_1(v) : v \in R_1)$ and $(\pi_2(v) : v \in R_2)$ that were determined there). Hence, we only query $O(1/\epsilon)$ pairs out of all $|R_1| \cdot |R_2| = O(k/\epsilon^2)$ pairs.

Analysis of the foregoing algorithm. The query complexity of the algorithm is $O(\epsilon^{-1} \cdot \sqrt{k} \cdot \log k)$, where Step 3 dominates the number of queries made. Specifically, for each vertex in R, we invoked the local self-ordering procedure, which results in quering its neighborhood in S; that is, we queried all pairs in $R \times S$.

Let us first verify that graphs in Π are accepted with probability at least $1 - \exp(-\Omega(\ell))$, where the unlikely rejection events are solely due to Steps 1–3 (i.e., either a wrong approximation of a vertex's degree (in either Step 1 or Step 3) or failure to sample a reliable locator in Step 2).[15] In contrast, if all vertices of $G' \in \Pi$ are correctly categorized (based on their approximated degrees per their neighborhood in S) and a reliable locator was found, then π_1 (resp., π_2) equals the unique isomorphism between the subgraph of G' induced by the low degree vertices and G_{2k} (resp., the

[15]Note that a wrong approximation of a vertex degree may lead to placing it in the wrong R_i, which may lead to rejection at later steps. Failure to find a reliable locator in Step 2 may also be due to S containing too few low (resp., high) degree vertices, which is also highly unlikely. Specifically, a degree approximation error occurs with probability at most $n \cdot \exp(-\Omega(\ell)) = \exp(-\Omega(\ell))$, whereas failure to sample a reliable locator occurs with probability at most $\exp(-\Omega(\ell))$.

subgraph of G' induced by the high degree vertices and G_{7k}). In this case, all subsequent checks will be successful, and our tester will always accept.

On the other hand, suppose that $G' = ([9k], E')$ is ϵ-far from Π. As a warm-up, consider the case that $G' = \pi(G_{A,B})$, for some matrix pair (A, B) and a bijection $\pi : [9k] \to [9k]$. In this case, B must be 81ϵ-far from A, since the adjacencies determined by B constitute a $1/81$ fraction of all vertex-pairs. It follows that if we reach Step 6, then we reject with high probability, since we are likely to inspect $\Omega(1/\epsilon)$ disjoint matrix-collisions, which are uniformly and independently distributed in $[k]^2$. Intuitively, if G' is close enough to some $\pi(G_{A,B})$, then the same argument holds, and otherwise Steps 1–5 reject with high probability. The actual analysis follows.

Let V_1 denote the set of vertices of G' that have degree at most $3.2k$, and $V_2 \subseteq [9k] \setminus V_1$ denote the set of vertices of G' that have degree at least $3.3k$. We may assume that the degrees of all vertices are well approximated by their neighborhoods in the sample S (selected in Step 1), since this holds with probability at least $1 - \exp(-\Omega(\ell))$. It follows that all vertices in $V_1 \cup V_2$ are correctly categorized as low and high degree vertices. We may also assume that $|V_1 \cup V_2| \geq 9k - \epsilon' k$, since otherwise Step 3 rejects with high probability (because vertices of degree in $(3.2k, 3.3k)$ are most likely to be placed in R_2 and cause rejection per their too low degree).

Assuming that Steps 1 and 2 were completed successfully (i.e., without rejection), we define a function $\phi_1 : V_1 \to [2k] \cup \{\perp\}$ such that $\phi_1(v)$ denotes the answer of the local self-ordering (based on S_1) to the input v, which may be a failure symbol, denoted $\perp$, where failure may occur because the subgraph of G' induced by V_1 is not necessarily isomorphic to G_{2k}. Similarly, we define $\phi_2 : V_2 \to [7k] \cup \{\perp\}$. Note that ϕ_i agrees with π_i on R_i, where π_i is defined exactly in this manner in Step 3.

Letting $\epsilon' = \Omega(\epsilon)$, we may assume that ϕ_i does not evaluate to $\perp$ on more than $\epsilon' k$ points, since otherwise such a point is sampled (w.h.p.) by Step 3, leading it to reject. Likewise, ϕ_i does not have more than $\epsilon' k$ points that have an image with several pre-images under ϕ_i, where here a Birthday argument proves the claim (since the collision probability is at least $1/\epsilon' k$). In particular, it follows that $|V_1| \leq (2 + 2\epsilon') \cdot k$ (resp. $|V_2| \leq (7 + 2\epsilon') \cdot k$), because otherwise ϕ_1 (resp., ϕ_2) would have had more than $\epsilon' k$ points that have an image with several pre-images under it.

Denote by $V_i' \subseteq V_i$ the set of vertices that were not discarded above; that is, $V_i' = \{v \in V_i : \phi_i(v) \neq \perp \,\&\, |\phi_i^{-1}(\phi_i(v))| = 1\}$. Using $|V_1| \geq 9k - |V_2| - \epsilon' k \geq (2 - 3\epsilon') \cdot k$ (resp. $|V_2| \geq (7 - 3\epsilon') \cdot k$), we note that $|V_1'| \geq (2 - 5\epsilon') \cdot k$ and $|V_2'| \geq (7 - 5\epsilon') \cdot k$ must hold, and define ϕ_i' as the restriction of ϕ_i to V_i'. Lastly, let $\phi' : V_1' \cup V_2' \to [9k]$ such that $\phi'(v) = \phi_i'(v)$ if $v \in V_1'$ and $\phi'(v) = 2k + \phi_i'(v)$ otherwise, and let ϕ be an arbitrary extension of ϕ' to a permutation over $[9k]$.

Relying on Step 4 (and assuming that it does not reject w.h.p), we infer that the subgraph of $\phi(G')$ induced by $[2k]$ (resp., by $\{2k+1, ..., 9k\}$) is $6\epsilon'$-close to G_{2k} (resp., to G_{7k} (when relabeling its vertices by subtracting $2k$ to each label)). Relying on Step 5 (and assuming that it does not reject w.h.p), we infer that $\phi(G')$ is $7\epsilon'$-close to some $G_{A,B}$ for some matrices A and B. Recalling that G' is ϵ-far from Π (and using $\epsilon = 8\epsilon'$), we infer that A and B must disagree on more than $\epsilon' \cdot 81k^2$ entries, which means that they are $81\epsilon'$-far apart. We claim that in this case Step 6 rejects with high probability.

We stress that so far we have not conditioned the sets R_1 and R_2; the foregoing inferences that involve Steps 3–5 were thought experiments.[16] Recalling that $|V_1| = (2k \pm 0.3k)$, we observe that

[16]We argued that if certain conditions do not hold, then one of these steps would reject with high probability (when using random R_1 and R_2). However, we continued assuming that the conditions do hold, and did not condition the distribution of R_1 and R_2.

with high probability (over the choice of R_1 and R_2), Step 6 defines a set of at least $1/\epsilon$ disjoint matrix-collision pairs, and that these pairs are independently and uniformly distributed in $[k] \times [k]$. Such a pair (i,j) causes rejection if the following conditions hold:

1. A and B disagree on (i,j) (i.e., $a_{i,j} \neq b_{i,j}$);
2. $\phi_1^{-1}(i)$ and $\phi_1^{-1}(k+i)$ are in V_1', which implies $\pi_1(\phi_1^{-1}(i)) = i$ and $\pi_1(\phi_1^{-1}(k+i)) = k+i$;
3. $\phi_2^{-1}(j), \phi_2^{-1}(6k+j) \in V_2'$, which implies $\pi_2(\phi_2^{-1}(j)) = j$ and $\pi_2(\phi_2^{-1}(6k+j)) = 6k+j$.

The probability that this event holds is lower-bounded by $81\epsilon' - 5\epsilon' - 5\epsilon' > 8\epsilon$, and the claim follows. ■

Combining Claims 3.1.2 and 3.1.3, the theorem follows. ■

Digest. Note that the proof of Claim 3.1.2 only uses the hypothesis that the graphs $\{G_k\}_{k\in\mathbb{N}}$ are robustly self-ordered, whereas the proof of Claim 3.1.3 only uses the hypothesis that these graphs can be locally self-ordered (via a reliable locator of logarithmic size). The robust self-ordering is used to infer that if $\pi(G_{A,B})$ is far from $\pi'(G_{A',B'})$, then either π is far from π' or (A,B) is far from (A',B'). The local self-ordering procedure is used in order to associate vertices of $\pi(G_{A,B})$ with rows or columns of either A or B.

We note that, in contrast to the situation in the bounded-degree graph model (cf. [14, Thm. 4.7]), the local self-ordering procedures that we used are not computationally efficient. This is the reason that our testers are also not computationally efficient. The logarithmic gap between the square root of the lower bound on the query complexity of non-adaptive testers and the query complexity of our adaptive tester is due to the query complexity of our local self-ordering procedure, which we indeed tried to minimize. Lastly, we comment that "local reversed self-ordering" (cf. [14, Def. 4.8]) requires very high query complexity in the dense graph model (unlike in the bounded-degree graph model [14, Thm. 4.9]).

4 Generalizations

In this section we prove Theorems 1.3 and 1.4. The proofs are obtained by gradual modifications of the proof of Theorem 3.1, presented in Section 3. While these modifications are relatively simple, establishing Theorem 1.5 requires several additional ideas, which will be presented in Section 5.

4.1 Smaller complexity gaps

Theorem 3.1 asserts that the gap between the query complexity of non-adaptive testers and adaptive ones may be almost quadratic. Answering open problems raised in [12, Sec. 1.3] (see also [6, Sec. 8.5.4]), we next prove that the gap can take the form of any function that is at most (almost) quadratic. This exhibits a richer range of gaps than conjectured in [12].

Theorem 4.1 (Theorem 1.3, restated): *For every function $g : \mathbb{N} \to \mathbb{N}$ such that $g(n) \leq \sqrt{n}$, there exists a graph property Π that satisfies the following two conditions:*

1. *The general* (adaptive) *query complexity of testing Π lies between $\Omega(\sqrt{n})$ and $O(\epsilon^{-1} \cdot \sqrt{n} \cdot \log n)$.*

2. *The non-adaptive query complexity of testing* Π *lies between* $\Omega(g(n) \cdot \sqrt{n})$ *and* $O(\epsilon^{-2} \cdot g(n) \cdot \sqrt{n} + \epsilon^{-1} \cdot \sqrt{n} \cdot \log n)$.

Note that in Theorem 3.1 there was no need to state the non-adaptive upper bound and the adaptive lower bound, since they roughly follows from the fact that the non-adaptive complexity is at most quadratic in the general complexity [1, 13]; specifically, for $g(n) = \sqrt{n}$, Theorem 3.1 implies an upper bound of $O(O(\epsilon^{-1} \cdot \sqrt{n} \cdot \log n)^2) = O(\epsilon^{-2} \cdot n \cdot \log^2 n)$ on the query complexity of non-adaptive testers, and a lower bound of $\Omega(\Omega(n)^{1/2}) = \Omega(\sqrt{n})$ on the query complexity of adaptive testers.

Proof Sketch: We use the same property as in the proof of Theorem 3.1, except that we utilize matrices in which each column is repeated $\Theta(n/g(n)^2)$ times (in consecutive columns). Intuitively, this redundancy effectively shrinks the matrix size from n^2 (in Theorem 3.1) to $n \cdot g(n)^2$, thus reducing the complexity of non-adaptive tester from $O(\sqrt{n^2}) = O(n)$ to $O(\sqrt{g(n) \cdot n}) = O(g(n) \cdot \sqrt{n})$. On the other hand, a lower bound that is a square root of the effictive matrix-size still holds. Turning to the complexity of general (adaptive) testing, we observe that it remains a square root of the number of rows, which did not change (not even effectively). Details follow.

We first outline the modification to Construction 3.1.1. Setting $k = n/9$ (as before), we let $t = k/g(n)^2$ (be the number of repetitions), and modify the property Π as follows.

- For two k-by-k Boolean matrices $A = (a_{i,j})$ and $B = (b_{i,j})$, we define the graph $G_{A,B} = ([9k], E_{A,B})$ exactly as in Eq. (2).
- We say that a matrix $A = (a_{i,j})$ is t-column-redundant if $a_{i,j} = a_{i,j'}$ for every $i, j, j' \in [k]$ such that $\lceil j/t \rceil = \lceil j'/t \rceil$.

  We define the property $\Pi = \bigcup_{n \in \mathbb{N}} \Pi_n$ such that Π_n is the set of all graphs that are isomorphic to some n-vertex graph $G_{A,A}$ for some t-column-redundant matrix A. That is, the only difference between Π_n and its definition in Construction 3.1.1 is that A is t-column-redundant; each of its columns is repeated t times, in consecutive columns.

The proofs of Claims 3.1.2 and 3.1.3 remain almost intact under the relevant modifications. In particular, the *lower bound for non-adaptive testers* is still a square root of the actual information contents of the matrices, which in this case equals $\sqrt{k \cdot (k/t)}$, which in turn equals $\Omega(\sqrt{n \cdot g(n)^2}) = \Omega(g(n) \cdot \sqrt{n})$.

Specifically, we consider the problem of distinguishing the uniform distribution on Π_n and the distribution generated by picking two random and independent t-column-redundant matrices $A = (a_{i,j})$ and $B = (b_{i,j})$ and outputting $\pi(G_{A,B})$ for a random $\pi \in \mathrm{Sym}_n$. In this case, the probability that two specific queries correspond to entries (i,j) in A and (i,j') in B such that $\lceil j/t \rceil = \lceil j'/t \rceil$ is $\frac{1}{k} \cdot \frac{t}{k}$, which means that distinguishing these distributions requires $\Omega(\sqrt{k^2/t})$ non-adaptive queries. On the other hand, with overwhelmingly high probability, two random and independent t-column-redundant matrices disagree on more than $k^2/3$ of their entries.

The *adaptive tester* is as in the proof of Claim 3.1.3, except that we slightly modify Step 6; specifically, the pair $(i,j) \in [k]^2$ is considered a matrix-collision if there exists $u_1, v_1 \in R_1$ and $u_2, v_2 \in R_2$ such that $\pi_1(u_1) = i = \pi_1(v_1) - k$ and $\lceil \pi_2(u_2)/t \rceil = \lceil j/t \rceil = \lceil (\pi_2(v_2) - 6k)/t \rceil$. (In this case, as in the proof of Claim 3.1.3, for each selected matrix-collision (i,j), we query G' on the pairs $(\pi_1^{-1}(i), \pi_2^{-1}(j))$ and $(\pi_1^{-1}(k+i), \pi_2^{-1}(6k+j))$, and reject if the answers are different.)

The query complexity of this modified algorithm is $O(\epsilon^{-1} \cdot \sqrt{k} \cdot \log k)$ as before, and the analysis of Step 6 is extended to show that if $\phi(G')$ is far from Π but close to some $G_{A,B}$, then, for any

t-column-redundant matrix C it holds that either A or B is far from C. To see that, in this case, the modified Step 6 rejects (w.h.p) consider $\alpha, \beta \in \{0,1\}^t$ such that for every $\gamma \in \{0^t, 1^t\}$ either α or β disagrees with γ on at least d entries. Then, the probability that a random entry in α disagrees with a random entry in β is at least d/t.[17]

We now turn to the two remaining claimed bounds. The *upper bound on non-adaptive testers* follows by considering a non-adaptive version of the foregoing adaptive tester, with a crucial variation in Step 6. First note that Steps 1–5 use adaptivity only in their identification of the S_i's and R_i's. We can avoid this adaptivity by querying the corresponding samples; specifically, denoting the sample taken in Step 1 (resp., Step 3) by S (resp., R), we query $S \times R$ (rather than querying $\bigcup_{i\in\{1,2\}}(S_i \times R_i)$), and use the identification of the S_i's and R_i's only in the interpretation of the answers. Note that $|S| \cdot |R| = O(\ell \cdot \sqrt{k}/\epsilon) = O(\epsilon^{-1}\sqrt{n}\log n)$. (In particular, we benefit from the fact that the local self-ordering procedure, which relies on a reliable locator, is non-adaptive.)[18]

As for Step 6, the key observation is that we can select a random subset R' of $O(g(n)/\epsilon)$ vertices of R, and make non-adaptive queries to all pairs in $R \times R'$, which means making $O(g(n) \cdot \sqrt{n}/\epsilon^2)$ queries. The point is that the number of non-repeated columns is $k/t = g(n)^2$, and R' is likely to yield $\Theta(1/\epsilon)$ column-collisions, where j is a column-collision if there exists $u_2, v_2 \in R_2$ such that $\lceil \pi_2(u_2)/t \rceil = \lceil j/t \rceil = \lceil (\pi_2(v_2) - 6k)/t \rceil$. Again, the identification of the S_i's and R_i's as well as of the matrix-collisions takes place only in the interpretation of the answers.

Lastly, we turn the *lower bound on adaptive testers*. This bound follows merely by considering collisions among samples of the k rows of the matrix. Specifically, we used the same distributions as in the proof of the lower bound for non-adaptive testers (i.e., the modified Claim 3.1.2), and observe that $\Omega(\sqrt{k})$ adaptive queries are needed in order to find vertices of G' that are mapped (by π) to location i and $i+k$, for some $i \in [k]$. ■

4.2 Lower levels of complexity

Using the idea of redundancy, we can get gaps at lower levels of complexity. Specifically, starting from Theorem 4.1, we replace $\sqrt{n}$ by $f(n) \le \sqrt{n}$ (and replace $g(n)$ by $g(f(n))$).[19]

Theorem 4.2 (Theorem 1.4, restated): *For every functions $f, g : \mathbb{N} \to \mathbb{N}$ such that $f(n) \le \sqrt{n}$ and $g(m) \le m$, there exists a graph property Π that satisfies the following two conditions:*

1. *The general* (adaptive) *query complexity of testing Π lies between $\Omega(f(n))$ and $O(\epsilon^{-1} \cdot f(n) \cdot \log n)$.*

2. *The non-adaptive query complexity of testing Π lies between $\Omega(g(f(n)) \cdot f(n))$ and $O(\epsilon^{-2} \cdot g(f(n)) \cdot f(n) + \epsilon^{-1} \cdot f(n) \cdot \log n)$.*

[17] Denoting the fraction of 1-entries in α (resp., β) by p (resp., q), observe that the probability that random entries disagree equals $p(1-q) + (1-p)q = p + q - 2pq$. Assuming, w.l.o.g., that $p + q \le 1$, we have $p + q - 2pq \ge (p+q)/2$. On the other hand, the relative distance of α from a constant string is at least $\min(p, 1-p)$, and ditto for β. Hence, $\min(p, 1-p, q, 1-q) \ge d/t$, and $(p+q)/2 \ge d/t$ follows.

[18] This is not so crucial given that the query complexity of this procedure is polylogarithmic, and such a bound would have been preserved under the transformation from adaptive to non-adaptive algorithms. Still, this saves us a polylogarithmic factor.

[19] Indeed, it would have been more consistent to replace $g(n)$ by $g(f(n)^2)$, but doing so would have made the current statement slightly more complex.

Proof Sketch: As in the proof of Theorem 4.1, we use the same property as in the proof of Theorem 3.1, except that we utilize matrices in which each row is repeated $\Theta(n/f(n)^2)$ times and each column is repeated $\Theta(n/(g(f(n))\cdot f(n))^2)$ times. Specifically, we set $k = n/9$, $m = f(n)$, $t' = k/m^2$ and $t = k/g(m)^2 = t'\cdot m^2/g(m)^2 \geq t'$, and modify the definition of $\Pi = \bigcup_{n\in\mathbb{N}}\Pi_n$ such that Π_n is the set of all graphs that are isomorphic to some n-vertex graph $G_{A,A}$ such that $A = (a_{i,j})$ satisfies $a_{i,j} = a_{i,j'}$ for every $i, i', j, j' \in [k]$ such that $\lceil i/t'\rceil = \lceil i'/t'\rceil$ and $\lceil j/t\rceil = \lceil j'/t\rceil$. We call such matrices (t', t)-redundant

The proofs of all four bounds remain almost intact under the relevant modifications. Starting with the *lower bound for non-adaptive testers*, we observe that it is still a square root of the actual information contents of the matrices, which in this case equals $\sqrt{(k/t')\cdot(k/t)}$, which in turn equals $\Omega(\sqrt{f(n)^2\cdot g(f(n))^2}) = \Omega(g(f(n))\cdot f(n))$. Specifically, we consider the problem of distinguishing the uniform distribution on Π_n and the distribution generated by picking two random and independent (t', t)-redundant matrices $A = (a_{i,j})$ and $B = (b_{i,j})$ and outputting $\pi(G_{A,B})$ for a random $\pi \in \mathrm{Sym}_n$. In this case the probability that two specific queries correspond to entries (i, j) in A and (i', j') in B such that $\lceil i/t'\rceil = \lceil i'/t'\rceil$ and $\lceil j/t\rceil = \lceil j'/t\rceil$ is $\frac{t'}{k}\cdot\frac{t}{k}$, which means that distinguishing these distributions requires $\Omega(\sqrt{k^2/t't})$ non-adaptive queries. On the other hand, with high probability, two random and independent (t', t)-redundant matrices disagree on more than $k^2/3$ of their entries.[20]

The *adaptive tester* is again a modification of the tester presented in the proof of Claim 3.1.3, where the modification is confined to taking a smaller sample in Step 3, and adapting Step 6 to the current redundancy; specifically, in Step 3 we take a sample of $O(f(n)/\epsilon)$ vertices (i.e., $|R| = O(f(n)/\epsilon)$), and in Step 6 we consider the pair $(i, j) \in [k]^2$ to be a matrix-collision if there exists $u_1, v_1 \in R_1$ and $u_2, v_2 \in R_2$ such that $\lceil \pi_1(u_1)/t'\rceil = \lceil i/t'\rceil = \lceil(\pi_1(v_1) - k)/t'\rceil$ and $\lceil\pi_2(u_2)/t\rceil = \lceil j/t\rceil = \lceil(\pi_2(v_2) - 6k)/t\rceil$. The query complexity of this modified algorithm is $O(\epsilon^{-1}\cdot f(n)\cdot\log n)$, and the analysis of Step 6 is extended to show that if $\phi(G')$ is far from Π but close to some $G_{A,B}$, then, for any (t', t)-redundant matrix C it holds that either A or B is far from C.[21]

The *upper bound on non-adaptive testers* follows by considering a non-adaptive version of the foregoing adaptive tester. The observations and modifications regarding this matter that were made in the proof of Theorem 4.1 apply here (without any change). Specifically, recall that for Step 6 we can select a random subset R' of $O(g(f(n))/\epsilon)$ vertices of R, and make non-adaptive queries to all pairs in $R \times R'$, which means making $O(g(f(n))\cdot f(n)/\epsilon^2)$ queries. The point is that the number of non-repeated rows (resp., columns) is $k/t' = f(n)^2$ (resp., $k/t = g(f(n))^2$), and R (resp., R') is likely to yield $\Theta(1/\epsilon)$ row-collisions (resp., column-collisions), where i is a row-collision if there exists $u_1, v_1 \in R_1$ such that $\lceil\pi_1(u_1)/t'\rceil = \lceil i/t'\rceil = \lceil(\pi_1(v_1) - k)/t'\rceil$ and j is a column-collision if there exists $u_2, v_2 \in R_2$ such that $\lceil\pi_2(u_2)/t\rceil = \lceil j/t\rceil = \lceil(\pi_2(v_2) - 6k)/t\rceil$.

Lastly, we turn the *lower bound on adaptive testers*. This bound follows merely by considering collisions (in the revised sense) among samples of the k rows of the matrix. Specifically, we used the same distributions as in the proof of the lower bound for non-adaptive testers, and observe that $\Omega(\sqrt{k/t'}) = \Omega(f(n))$ adaptive queries are needed in order to find vertices of G' that are mapped (by π) to location i and $i' + k$, for some $i, i' \in [k]$ such that $\lceil i/t'\rceil = \lceil i'/t'\rceil$. ■

[20]This holds provided that $t' = o(n)$, and otherwise the claims hold vacuously.

[21]To see that, in this case, the modified Step 6 rejects (w.h.p) consider $\alpha, \beta \in \{0,1\}^{t'\times t}$ such that for every $\gamma \in \{0^{t'\times t}, 1^{t'\times t}\}$ either α or β disagrees with γ on at least d entries. Then, as detailed in Footnote 17, the probability that a random entry in α equals a random entry in β is at least $d/t't$.

5 Accommodating one-sided error testers

Recall that a tester for a graph property Π is said to have one-sided error probability if it accepts every graph in Π with probability 1; otherwise, we say that it has two-sided error probability.

We show that a (significant) variant of the graph property defined in the proof of Theorem 4.2 can be tested with one-sided error probability within almost the same query complexity. In particular, although we shall define some new notions and establish some new results before getting to the lower bounds, the proof of the lower bounds will use simple modifications of the prior proofs. The new notions and results will be mostly used in the description and analysis of the testers, which are quite complex. (Of course, some of these modifications will appear in the graph property itself, and the proof of the lower bound will need to deal with them, but doing so will not be difficult.) In any case, our goal is to prove the following –

Theorem 5.1 (Theorem 1.5, restated): *For every functions $f, g : \mathbb{N} \to \mathbb{N}$ such that $f(n) \leq \sqrt{n}$ and $g(m) \leq m$, there exists a graph property Π that satisfies the following two conditions:*

1. *The general* (adaptive) *query complexity of testing Π lies between $\Omega(f(n))$ and $O(\epsilon^{-1} \cdot f(n) \cdot \log^3 n)$, where the upper bound holds for a one-sided error tester and the lower bound holds for two-sided error testers.*

2. *The non-adaptive query complexity of testing Π lies between $\Omega(g(f(n)) \cdot f(n))$ and $O(\epsilon^{-2} \cdot g(f(n)) \cdot f(n) \cdot \log^2 n + \epsilon^{-1} \cdot f(n) \cdot \log^3 n)$, where the lower bound holds for two-sided error testers and the upper bound holds for a one-sided error tester.*

Actually, the upper bounds hold provided that $\epsilon > n^{-0.49}$.

Otherwise (i.e., if $\epsilon \leq n^{-0.49}$), the trivial tester that checks all entries works using $O(1/\epsilon^{4.1})$ queries.

Overview of the proof of Theorem 5.1. The issues that we shall address are rooted in the proof of Theorem 3.1, but we deal with them in a way that also accommodates the modification made to derive Theorem 4.2. Hence, we refer to the proof of Theorem 4.2 rather than to that of Theorem 3.1.

As stated above, we shall use a variant of graph property Π that was used in the proof of Theorem 4.2, and so the lower bounds remain valid (although we shall have to show this too). Furthermore, we will need to modify the testers (which are based on the tester presented in the proof of Claim 3.1.3), which evidently have two-sided error probability. This error probability (on graphs in Π) has three sources:

1. A wrong categorization of vertices (w.r.t the S_i's and R_i's) caused by a wrong approximation of their degrees (which takes place in Steps 1 and 3, respectively).

2. Failure to sample (in Step 1) a proportional number of vertices on each side of the graph, where we refer to the copies of G_{2k} and G_{7k} used in the construction of graphs in Π_{9k}.

3. Failure to find a reliable locator (for each side) in Step 2.

These errors seem unavoidable. For starters, when taking a small sample of the graph vertices, it may happen with (small) positive probability that the sample does not provide a correct estimate for the degrees of some vertices in the graph (e.g., all sampled vertices may happen to be neighbors of some vertex). Likewise, it may happen (although rarely) that all vertices sampled in Step 1 hit one side of the graph. And, similarly, it may happen that a small sample (as taken in Step 2) does not contain a reliable locator, because not all polylogarithmic-sized subsets of the vertex-set are reliable locators.

Faced with these problems, we take two steps. First, we move away from notions that require correct performance with respect to all the graph's vertices, but only guarantee it with high probability. Instead, we devise notions that allow for few exceptional vertices, but guarantee such performance with probability 1 (when the graph is in the property). For example, rather than seeking a sample that estimates correctly the degrees of all vertices in a graph, we seek samples that (for graphs in the property) always estimate correctly the degrees of all but logarithmically many vertices. (Needless to say, these samples will provide good estimations, with high probability, for any vertex in any graph.) An analogue relaxation is employed to the notion of a reliable locator. Furthermore, we show that both these relaxations can be achieved.

Second, given that we cannot rely on Step 1 to hit the two sides of the graph in proportion to their size, we use a different way of obtaining two robustly self-ordered graphs with significantly different vertex degrees. Specifically, we use random graphs with different edge density. Note that we can detect the case that a sample is extremely biased towards vertices of one of the two sides (or graphs), and in this case we proceed without rejecting (rather than reject based on statistic evidence). In this case, graphs that are far from the property will be rejected by the subsequent steps, which will produce (with high probability) an absolute witness for violation of the property.

Organization of the proof. In Section 5.1, we develop the tools that will be used in the actual proof. These include a relaxed notion of reliable locators, called *pseudo-locators*, that allows for location failures on very few vertices of the graph (Definition 5.4), and a proof that in a random graph all logarithmically-sized subsets are pseudo-locators (Claim 5.5). The actual proof is presented in Section 5.2. Its core is the descriptions of the modified graph property (Construction 5.7) and of the modified adaptive tester (presented in the proof of Proposition 5.9).

5.1 Developing tools for the proof

As noted above, in previous sections, distinguishing the vertices in the two sides of the graph (i.e., in copies of G_{2k} and G_{7k}) was based on the fact that these vertices have different degrees, but this difference is detected based on the assumption that the sample taken in Step 1 hits each side in proportion to its size. Although this happens with high probability, it does not happen with probability 1, and so we cannot rely on this reasoning for one-sided error testers. The solution is to use different edge densities in the two sides, which means that we need an alternative to Construction 3.1.1. Specifically, for a small constant $p > 0$ (e.g., $p = 0.1$), we replace the graphs G_{2k} and G_{7k} used there, with two $2k$-vertex graphs, one of edge density p, and the other of edge density $1 - p$. Both graphs will be robustly self-ordered and their edge density will be preserved by any induced subgraph of size $\Omega(\log k)$. Hence, we first prove the following –

Claim 5.2 (robust self-ordering of $\mathcal{G}(n,p)$ and vertex degrees in it): *For any constant $p \in (0,1)$, let $\mathcal{G}(n,p)$ be the Erdos–Renyi random n-vertex graph in which each pair of vertices is connected with probability p independently of all other vertex-pairs. Then:*

1. *With probability $1 - \exp(-\Omega(n))$, the graph is robustly self-ordered.*

2. *For sufficiently large $\ell = O(\log n)$, with probability $1 - \exp(-\Omega(\ell^2))$, the following two conditions hold:*
   (a) *Every set of ℓ vertices induces a subgraph of edge density $(1 \pm 0.01) \cdot p$.*
   (b) *For each set of ℓ vertices S there are at least $n - 2\ell$ vertices $v \in [n] \setminus S$ such that the number of neighbors that v has in S is $(1 \pm 0.01) \cdot p \cdot \ell$.*

We mention that Part 1 is implicit in the proof of [16, Thm. 3.1]. We highlight the fact that *all* ℓ-subsets provide a good approximation of the degrees of *almost all* vertices (Part 2b). Indeed, for every vertex there exists ℓ-subsets that fail to provide a good approximation to its degree, but no ℓ-subset fails for more than ℓ vertices (outside it).[22]

Proof Sketch: Part 2a follows by the fact that the probability that any fixed ℓ-subset induces a subgraph with deviating edge density is $\exp(-\Omega(\min(p, 1-p) \cdot \ell^2))$, which is $\exp(-\Omega(\ell^2))$ since p is constant.[23] Hence, the effect of p is hidden by the Ω-notation, and the same phenomenon will happen in the later parts.

Part 2b follows by the fact that, for each ℓ-set S and each $v \in [n] \setminus S$, the probability that the number of neighbors that v has in S is not $(1 \pm 0.01) \cdot p \cdot \ell$ is $\exp(-\Omega(\ell)$. Applying a union bound, it follows that the probability that there exists an ℓ-set for which there are $\ell + 1$ deviating vertices is at most

$$\binom{n}{\ell} \cdot \binom{n}{\ell+1} \cdot \exp(-\Omega(\ell))^{\ell+1},$$

which is $\exp(-\Omega(\ell^2))$. (We highlight the fact that allowing ℓ exceptional vertices, which means that violation requires $\ell + 1$ vertices rather than 1, raises the probability to the power of $\ell + 1$, which in turn enables the application of a union bound over all ℓ-sets of $[n]$.)

Part 1 follows by a simple adaptation of the proof of Theorem 2.3, which is presented in Appendix A. Specifically, the probability that two different vertex-pairs disagree on whether or not they form an edge is $q = 2p(1-p)$ rather than 1/2, and Eq. (9) is modified accordingly (i.e., $n \cdot |T|/20$ is replaced by $n \cdot |T| \cdot q/10$). ■

Connecting graphs sampled from $\mathcal{G}(n, 0.1)$ and $\mathcal{G}(n, 0.9)$. For any constant $p < 1/2$, part 2b of Claim 5.2 offers a way of distinguishing vertices in (a graph sampled from) $\mathcal{G}(n, p)$ from vertices in (a graph sampled from) $\mathcal{G}(n, 1-p)$, but this way may be frustrated if the two graphs are connected arbitrarily (or almost so) as in Construction 3.1.1. We solve this problem by connecting these two graphs by a bipartite graph in which the edge density is $0.5 \pm p$, for say $p = 0.1$, where the slackness offers enough room to encode information (of the embedded matrices). Furthermore, we need every ℓ-by-ℓ bipartite subgraph to have the same density up to a factor of 1 ± 0.01. Actually, we use the following claim, which is formulated in terms of matrices rather than in terms of bipartite graphs, since this allows to capture the slackness (reflected by $*$-entries) in a more elegant manner.

Claim 5.3 (all logarithmically sized submatrices are balanced in a random p-slacked matrix): *For any constant $p \in [0, 0.5)$, let $\mathcal{M}(n, p)$ be the distribution on n-by-n matrices over $\{0, 1, *\}$ such*

[22] The choice of ℓ as the number of exceptional vertices is immaterial; we could have chosen $c\ell$ for any small or larger positive constant c. (Making c smaller can be supported by making $\ell = O(\log n)$ larger, whereas our applications can tolerate $O(\ell)$ exceptional vertices with no real effect.) Analogous comments apply to our relaxation of the notion of a reliable locator (see Definition 5.4 and Claim 5.5).

[23] Using a union bound over all $\binom{n}{\ell}$ subsets, the claim follows.

that each entry is 0 (resp., 1) *with probability $q = (1-p)/2$, independently of all other entries, and equals $*$ otherwise. Then, for sufficiently large $\ell = O(\log n)$, with probability $1 - \exp(-\Omega(\ell^2))$, every ℓ-by-ℓ submatrix has $(1 \pm 0.01) \cdot q \cdot \ell^2$ entries of value 0, and ditto for value 1. Furthermore, each row* (resp., column) *in the matrix has $(1 \pm 0.01) \cdot q \cdot n$ entries of value 0, and ditto for value 1.*

The proof of Claim 5.3 is by a straightforward counting argument (as used in the proof of Part 2a of Claim 5.2).[24] We shall use an arbitrary matrix M that satisfies the condition of Claim 5.3.

Intuitively, for some small constant $p > 0$ (e.g., $p = 0.1$), the graphs in our revised property Π will consist of a pair of graphs, denoted $G_{2k}^{(p)}$ and $G_{2k}^{(1-p)}$, drawn from $\mathcal{G}(2k, p)$ and $\mathcal{G}(2k, 1-p)$ respectively, that are connected by biparite graphs that are determined by two matrices in which the $*$-entries of M are replaced by Boolean values. But before presenting this construction, we introduce another feature that we wish the graphs $G_{2k}^{(\cdot)}$'s to possess.

The foregoing modifications to Construction 3.1.1 are aimed at removing the sources of errors of Types 1 & 2 discussed in the overview of the proof. We are now going to deal with errors of Type 3 (i.e., failure to sample a reliable locator).

Relaxing the notion of a locator. Recall that the remaining source of error (on graphs in Π) in our original tester is the failure to sample a reliable locator (for each side) in Step 2. The problem is that not all ℓ-subsets are reliable locators (although almost all of them are). We resolve this problem by defining a relaxed notion of a locator, and showing that, with high probability, in both $\mathcal{G}(2k, p)$ and $\mathcal{G}(2k, 1-p)$, all ℓ-subset satisfy this relaxed definition. The nature of the relaxation is allowing the locator to correctly identify *all but very few* of the vertices (as opposed to *all* in the original definition). When using the following definition, the threshold parameter τ will be set to $p \cdot (1-p)$.

Definition 5.4 (pseudo-locators): *A set of vertices $S \subset [n]$ is a* pseudo-locator (for threshold τ) *of a graph $G = ([n], E)$ if the following two conditions hold*

1. *The subgraph of G induced by S is not isomorphic to any subgraph of G that is induced by any $|S|$-set that misses more than $\tau \cdot |S|$ vertices of S* (i.e., any $|S|$-set S' such that $|S' \cap S| < (1-\tau) \cdot |S|$).

2. *There exists a set $U \subseteq [n] \setminus S$ of $n - 2 \cdot |S|$ vertices such that, for every $v \in U$, the adjacencies of v with any subset that induces the same unlabeled subgraph as S uniquely determine v; that is, for every $u \neq v$ in U and every S' such that the subgraph of G induced by S' is isomorphic via some $\phi : [n] \to [n]$ to the subgraph of G induced by S, there exists $s \in S$ such that $\{u, s\} \in E$ if and only if $\{v, \phi(s)\} \notin E$.*

   In other words: For every $u \neq v$ in U and every S', let H (resp., H') *denote the subgraph of G induced by S* (resp., S')*, and suppose that $\phi(H) = H'$. Then, there exists $s \in S$ such that $\{u, s\} \in E$ if and only if $\{v, \phi(s)\} \notin E$.* (U stands for *unexceptional.*)

(In light of Condition 1, it suffices to consider in Condition 2 sets S' such that $|S' \cap S| \geq (1-\tau) \cdot |S|$).

Definition 5.4 relaxes both conditions of Definition 2.5: In Condition 1 of Definition 5.4 we requires that only $|S|$-subsets that intersect S in less than $(1-\tau) \cdot |S|$ vertices induce a subgraph that is not

[24]Indeed, for $p = 0$, Claim 5.3 yields a non-explicit two-source extractor (cf. [19]) with almost optimal parameters.

isomorphic to the one induced by S; that is, we discard $|S|$-subsets that have a larger intersection with S and do not require that the subgraph induced by S is asymmetric. In light of the fact that Condition 1 was relaxed, the revised Condition 2 refers not only to the adjacencies with S but also to the adjacencies with any subset that induces the same unlabeled subgraph as S. On the other hand, in the revised Condition 2 we allow for $|S|$ exceptional vertices (even in case $S' = S$, which is the only case considered in Definition 2.5). Using both relaxations, we shall prove that, in a random graph, all logarithmically-sized subsets are pseudor-locators (Claim 5.5). But before proving this claim, let us spell out how a pseudo-locator will be used.

Suppose that $G' = (V', E')$ is isomorphic to $G = ([n], E)$; that is, $G = \phi(G')$ for some unknown to us bijection $\phi : V' \to [n]$. Now, if S' is a pseudo-locator of G' (equiv., $\phi(S')$ is a pseudo-locator of G), then, given $v \in V' \setminus S'$ we can identify $\phi(v)$ by inspecting the subgraph of G' induced by $S' \cup \{v\}$. Specifically, we look in G for an $|S'|$-subset $S \subset [n]$ and a vertex $i \in [n] \setminus S$ such that the subgraph of G induced by $S \cup \{i\}$ is isomorphic to the subgraph of G' induced by $S' \cup \{v\}$ and this isomorphism maps i to v. (In other words, we seek a bijection $\pi : S \cup \{i\} \to S' \cup \{v\}$ such that $\pi(i) = v$ and for every $j, j' \in S \cup \{i\}$ it holds that $\{j, j'\} \in E$ if and only if $\{\pi(j), \pi(j')\} \in E'$.) If G' is indeed isomorphic to G and v is not one of the $2 \cdot |S|$ exceptional vertices, then there exists a single i that satisfies this condition (although S need not be unique). If G' is not isomorphic to G, then anything may happen, but if the number of i's satisfying the foregoing condition is not one, then we shall announce failure. We stress that, while we actually query G' for the subgraph induced by $S' \cup \{v\}$, looking for suitable S and i in G is a thought experiment (since G is fixed and known to us).

Claim 5.5 (pseudo-locators in a random graph): *For any constant $p \in (0, 1)$ and sufficiently large $\ell = O(\log n)$, with probability $1 - \exp(-\Omega(\ell^2))$ over the choice of $\mathcal{G}(n, p)$, all ℓ-subsets of the vertex-set are pseudo-locators for threshold $p \cdot (1 - p)$.*

Proof: Fixing an arbitrary ℓ-subset, S, we shall upper-bound by $\exp(-\Omega(\ell^2))$ the probability that S is not a pseudo-locator of $\mathcal{G}(n, p)$ (for threshold $p \cdot (1 - p)$). This probability is low enough so to support a union bound over all ℓ-subsets; specifically, $\binom{n}{\ell} \cdot \exp(-\Omega(\ell^2)) = \exp(-\Omega(\ell^2))$, for sufficiently large $\ell = O(\log n)$.

For $p = 1/2$, Condition 1 of being a pseudo-locator (i.e., sufficiently different ℓ-subsets induce non-isomorphic subgraphs) is implicit in the proof of Claim 2.6.1, which is presented in Appendix B. Specifically, the probability that the subgraphs of a random graph induced by S and S' are isomorphic was upper-bounded in Eq. (10)&(11) by

$$\frac{\ell!}{|S \cap S'|!} \cdot 2^{-\Omega((\ell - |S \cap S'|) \cdot \ell)}. \tag{4}$$

Using $|S \cap S'| < 3\ell/4$, the foregoing is upper-bounded by $\exp(-\Omega(\ell^2))$, which allows us to apply a union bound (over all relevant ℓ-subsets S''s) and get an upper bound of $\binom{n}{\ell} \cdot \exp(-\Omega(\ell^2)) = \exp(-\Omega(\ell^2))$. As for a general $p \in (0, 1)$, it is handled by observing that all that changes is the probability of disagreement, which is $q = 2p(1 - p)$ rather than $1/2$, and the threshold (which is $p(1 - p)$ rather than $1/4$). Specifically, as in the proof of Part 2 of Claim 5.2, the probability that two different vertex-pairs disagree is $q = 2p(1 - p)$ rather than $1/2$, and Eq. (4) is modified accordingly (cf. Eq. (6) below), while noting that the derived bound is

$$\frac{\ell!}{|S \cap S'|!} \cdot q^{\Omega((\ell - |S \cap S'|) \cdot \ell)} \; < \; \ell^{0.5q\ell} \cdot q^{\Omega(q \cdot \ell^2)} \; = \; \exp(-\Omega(\ell^2)), \tag{5}$$

where we use $|S \cap S'| < (1-0.5q)\cdot\ell$ and $q=\Omega(1)$. This establishes Condition 1 (of being pseudo-locator).

We seize the opportunity to state a related fact that will be used for establishing Condition 2. We claim that, for every ℓ-subset S' and a bijection $\pi : S \to S'$, the probability that the subgraph of $\mathcal{G}(n,p)$ induced by S is isomorphic via π to the subgraph induced by $\pi(S)$ (i.e., $\{w,w'\}$ is an edge in the first subgraph iff $\{\pi(w),\pi(w')\}$ is an edge in the second subgraph) is upper-bounded by

$$\min\left(q^{|\mathtt{FP}(\pi)|\cdot(\ell-|\mathtt{FP}(\pi)|)/3}, q^{\binom{(\ell-|\mathtt{FP}(\pi)|)/3}{2}}\right) \tag{6}$$

where $\mathtt{FP}(\pi) \stackrel{\rm def}{=} \{v \in S : \pi(v)=v\}$. This fact, which generalizes Eq. (10), is actually established *en route* to establishing Eq. (5). Note that if $|\mathtt{FP}(\pi)| > 0.1q\cdot\ell$, then Eq. (6) is upper-bounded by $q^{\Omega(q\cdot\ell^2)} = \exp(-\Omega(\ell^2))$.

Before proceeding to Condition 2 (of being pseudo-locator), we highlight the fact that the $\Omega(\ell)$ lower bound on the symmetric difference between the subsets S and S', which yields a probability upper bound of $\exp(-\Omega(\ell^2))$, enables the application of a union bound on all ℓ-subsets. Analogously, as we shall see, allowing for ℓ exceptions in localization, which means that failure requires more that ℓ mistakes (rather than one), implies that the probability that an ℓ-subset is not a pseudo-locator vanishes exponentially with ℓ^2 (i.e., another ℓ factor on top of what we get for one mistake). Again, such an upper bound enables the application of a union bound over all ℓ-subsets.

Turning to Condition 2 of Definition 5.4, we stress that we shall show that S can be used to locate almost all vertices in $[n]\setminus S$, although the vertices of S itself may not be all uniquely located. Still, using Eq. (6) it follows that most vertices in S are uniquely located, and we shall show that almost all vertices in $[n]\setminus S$ have significantly different adjacencies in S. Hence, the uncertainty regarding the location of few vertices in S is compensated by the large difference between the adjacencies in S of almost all vertices in $[n]\setminus S$. Details follow, where we start with a description of a process that identifies the exceptional vertices in $[n]\setminus S$.

We consider an iterative process that starts with $U=[n]\setminus S$ and searches for a pair of vertices in U such that their adjacencies to the set S disagree on at most $0.9\cdot q\cdot\ell$ entries; that is, we look for $w\neq w'$ in U such that $|\{s\in S : \{w,s\}\in E \Leftrightarrow \{w',s\}\notin E\}| \leq 0.9\cdot q\cdot\ell$. If such a pair does not exist, then we halt and output U, which is considered a success. Otherwise, we omit both vertices in the pair from U, and proceed to the next iteration. (We may halt without output after $\ell/2$ unsuccessful iterations).

Vertices in U are correctly located by S. We show that if the process outputs U (after at most $\ell/2$ iterations), then U constitutes a set as required in Condition 2. Recall that here we consider arbitrary $u\neq v$ in U and any bijection $\phi : S\cup\{u\}\to S'\cup\{v\}$ (for any S') such that $\phi(u)=v$ and the subgraphs induced by S and by $S'=\phi(S)$ are isomorphic via ϕ. Using Condition 1, it suffices to consider sets S' such that $|S'\cap S|\geq(1-0.5q)\cdot\ell$, since the other sets induce subgraphs that are not isomorphic to the one induced by S. Furthermore, using Eq. (6) and a union bound, we may assume that $\phi(s)=s$ for at least $(1-0.6q)\cdot\ell$ of the vertices $s\in S\cap S'$, because the probability that this does not hold for some S' and ϕ is $\exp(-\Omega(\ell^2))$ (over the choice of $\mathcal{G}(n,p)$).[25]

[25]Specifically, using $|S\cap S'|\geq(1-0.5q)\cdot\ell$, we show that (with probability at least $1-\exp(-\Omega(\ell^2))$) it holds that $|\{s\in S\cap S' : \phi(s)=s\}|\geq(1-0.6q)\cdot\ell$. To see this note that if at least $0.1\cdot q\cdot\ell$ of the vertices in $S\cap S'$ are not fixed-points of ϕ, then the subgraphs of $\mathcal{G}(n,p)$ induced by S and $\phi(S)$ are equal (i.e., for every $w,w'\in S$ it holds that $\{w,w'\}$ is an edge iff $\{\phi(w),\phi(w')\}$ is an edge) with probability at most $q^{\binom{0.1q\ell/3}{2}}$. Using a union-bound over all

The punchline is that, by construction of U, *the neighborhoods of different vertices u and v in U disagree on more than $0.9 \cdot q \cdot \ell$ entries* (i.e., $|\{s \in S : \{u,s\} \in E \Leftrightarrow \{v,s\} \notin E\}| > 0.9q\ell$), and (using $|\{s \in S : \phi(s) \neq s\}| \leq 0.6q \cdot \ell$) this implies that *there exists $s \in S \cap S'$ such that $\{u,s\} \in E$ if and only if $\{v,\phi(s)\} \notin E$*, where we also used $\phi(s) = s$.

Hence, we have shown that, with probability at least $1 - \exp(-\Omega(\ell^2))$ over the choice of $\mathcal{G}(n,p)$, if the iterative process outputs U, then, for every $u \neq v \in U$ and any $\phi : S \cup \{u\} \to S' \cup \{v\}$, the hypothesis that $\phi(u) = v$ and the subgraphs induced by S and by $S' = \phi(S)$ are isomorphic via ϕ (i.e., $\{w, w'\}$ is an edge in the first subgraph iff $\{\phi(w), \phi(w')\}$ is an edge in the second subgraph) implies that there exists $s \in S$ such that $\{u,s\} \in E$ if and only if $\{v,\phi(s)\} \notin E$.

The process is successful with overwhelmingly high probability. It remains to prove that, with probability at least $1 - \exp(-\Omega(\ell^2))$ over $\mathcal{G}(n,p)$, the foregoing process is successful (i.e., does halt with output in at most $\ell/2$ iterations). This is the case because the probability that the first $\ell/2$ iterations find pairs of the type sought (equiv., that $\ell/2$ such disjoint pairs exist), in $\mathcal{G}(n,p)$, is upper-bounded by

$$\binom{n}{2}^{\ell/2} \cdot \exp(-\Omega(q \cdot \ell))^{\ell/2} = \exp(-\Omega(\ell^2)), \tag{7}$$

where each fixed pair of vertices disagrees on at most $0.9 \cdot q\ell$ of their neighbors with probability at most $\exp(-\Omega(q \cdot \ell))$, because they differs on neighboring any other vertex with probability q. Again, the bound in Eq. (7) is small enough for allowing the application of a union bound over all $\binom{n}{\ell}$ sets (i.e., the S's). ■

Corollary 5.6 (by Claims 5.2 and 5.5): *For every $p \in (0,1)$ and sufficiently large $\ell = O(\log n)$, there exists a family of robustly self-ordered graphs $\{G_n^{(p)} = ([n], E_n^{(p)})\}_{n \in \mathbb{N}}$ such that any ℓ-subset $S \subset [n]$ of its vertices satisfies the following four conditions.*

1. *The subgraph of $G_n^{(p)}$ induced by S has $(1 \pm 0.01) \cdot p \cdot \binom{\ell}{2}$ edges.*
2. *For at least $n - 2\ell$ vertices $v \in [n] \setminus S$, the number of neighbors that v has in S is $(1 \pm 0.01) \cdot p\ell$.*
3. *The set S is a pseudo-locator for threshold $p \cdot (1-p)$ of $G_n^{(p)}$.*

Furthermore, each vertex in $G_n^{(p)}$ has degree $(1 \pm 0.01) \cdot p \cdot n$.

Combining Corollary 5.6 with Claim 5.3, we are finally ready to present the (revised) graph property Π, and proceed to the actual proof.

5.2 The actual proof

We shall use a graph property that is a variant of the property used in Construction 3.1.1, as adapted in Section 4.2. Here we start with the graphs $G_{2k}^{(\delta)}$ and $G_{2k}^{(1-\delta)}$ provided by Corollary 5.6, and connect them using four different bipartite graphs that are derived from a fixed k-by-k matrix M (provided by Claim 5.3) and two arbitrary k-by-k matrices A and B. The latter matrices will be identical in the case of graphs in the property, but will be random and independent of one another when proving the lower bounds. Unlike in Construction 3.1.1, only some of the entries of A and B

ℓ-sets S' and $\phi : S \to S'$, we get a probability bound of $n^\ell \cdot \exp(-\Omega(\ell^2)) = \exp(-\Omega(\ell^2))$.

(i.e., those that correspond to $*$-entries of M) will be encoded in the construction, but these entries constitute a constant fraction of all entries and so the proofs of the lower bounds can handle this change. Recall that this complication is introduced so to maintain a gap between the degrees of the vertices that belong to the different sides of the constructed graph.

In accordance with the foregoing, we present the following (revised) graph property Π, which is a variant of the property used in Construction 3.1.1, as adapted in Section 4.2 for any functions $f, g : \mathbb{N} \to \mathbb{N}$ such that $f(n) \leq \sqrt{n}$ and $g(m) \leq m$. Indeed, the case of $f(n) = \sqrt{n}$ and $g(m) = m$ corresponds to Construction 3.1.1 itself, where no redundancy is used, and it suffices for establishing a quadratic gap (between adaptive one-sided error testers and non-adaptive two-sided error testers).[26]

Construction 5.7 (the revised graph property Π): *For any $\delta \in (0, 0.5)$ and sufficiently large $\ell = O(\log k)$, let $G_{2k}^{(\delta)} = ([2k], E_{2k}^{(\delta)})$ and $G_{2k}^{(1-\delta)} = ([2k], E_{2k}^{(1-\delta)})$ be as postulated in Corollary 5.6, and let M be a k-by-k matrix that satisfies the conditions of Claim 5.3.*

- *For two k-by-k Boolean matrices $A = (a_{i,j})$ and $B = (b_{i,j})$, we define the graph $G_{A,B} = ([4k], E_{A,B})$ such that*

$$\begin{array}{rcl} E_{A,B} & = & E_{2k}^{(\delta)} \cup \{\{2k+i, 2k+j\} : \{i,j\} \in E_{2k}^{(1-\delta)}\} \\ & & \cup \{\{i, 2k+j\} : i,j \in [k] \ \wedge \ m_{i,j} \odot a_{i,j} = 1\} \\ & & \cup \{\{k+i, 3k+j\} : i,j \in [k] \ \wedge \ m_{i,j} \odot b_{i,j} = 1\} \\ & & \cup \{\{i, 3k+j\}, \{k+i, 2k+j\} : i,j \in [k] \ \wedge \ m_{i,j} = 1\} \end{array} \qquad (8)$$

  where $m_{i,j} \odot \sigma = m_{i,j}$ if $m_{i,j} \in \{0,1\}$ and $m_{i,j} \odot \sigma = \sigma$ otherwise (i.e., if $m_{i,j} = *$).

  That is, $G_{A,B}$ consists of a copy of $G_{2k}^{(\delta)}$ and a copy of $G_{2k}^{(1-\delta)}$ that are connected by four bipartite graphs that is determined by the matrices $M \odot A$, $M \odot B$ and M, respectively, where $\odot$ is defined such that Boolean entries of M dominate and entries of the other matrix are used only for entries in which M is undetermined (as indicated by $*$). *In other words, only entries of A and B that correspond to $*$-entries of M are actually encoded in this construction.*[27]

  The first bipartite graph is determined by $M \odot A$ and connects $[k]$ to $\{2k+1, ..., 3k\}$, the second bipartite graph is determined by $M \odot B$ and connects $\{k+1, ..., 2k\}$ to $\{3k+1, ..., 4k\}$, whereas the other two bipartite graphs are determined by M and connect $[k]$ to $\{3k+1, ..., 4k\}$ and $\{k+1, ..., 2k\}$ to $\{2k+1, ..., 3k\}$.[28] *The first two bipartite graphs, which will be called* informative, *encode entries of A and B that correspond to $*$-entries of M.*

- *For $t', t \in [k]$, we say that a k-by-k Boolean matrix $A = (a_{i,j})$ is (t', t)*-redundant *if $a_{i,j} = a_{i,j'}$ for every $i, i', j, j' \in [k]$ such that $\lceil i/t' \rceil = \lceil i'/t' \rceil$ and $\lceil j/t \rceil = \lceil j'/t \rceil$.*

[26]In such a case, the second item may be skipped, and the third item may be simplified by allowing any k-by-k matrix A.

[27]As noted above, these entries constitute a constant fraction of all entries, and this will suffice for our purpose: Specifically, for uniformly and independently distributed A and B, with very high probability, the pair $(M \odot A, M \odot B)$ is far from any pair of the form $(M \odot C, M \odot C)$.

[28]The latter two bipartite graphs, which are determined solely by M, are used in order to simplify the identification of vertices based on their adjacencies in a small set T. Specifically, in this case, $|T \cap [2k]|/|T|$ determines the expected number of neighbours that $v \in [2k]$ (resp., $v \in \{2k+1, ..., 4k\}$) has in T upto $\pm 2\delta \cdot |T|$. Not using these two bipartite graphs would have required either knowledge of $|T \cap \{(i-1)k+1, ..., ik\}|/|T|$ for $i = 0, 1, 2$ or using far more crude approximations.

- *For $f, g : \mathbb{N} \to \mathbb{N}$ such that $f(n) \leq \sqrt{n}$ and $g(m) \leq m$, we let $k = n/4$, $m = f(n)$, $t' = k/m^2$ and $t = k/g(m)^2$, and define the property $\Pi^{(f,g)} = \Pi = \bigcup_{n \in \mathbb{N}} \Pi_n$ such that Π_n is the set of all graphs that are isomorphic to some n-vertex graph $G_{A,A}$ such that A is (t', t)-redundant.*

Note that, given a graph of the form $\pi(G_{A,A})$, the vertices in the copy of $G_{2k}^{(\delta)}$ are easily identifiable as having degree (in $\pi(G_{A,A})$) at most $1.01 \cdot \delta \cdot 2k + (1.01 \cdot (1-\delta)/2 + 1.01 \cdot \delta) \cdot 2k = 1.01 \cdot (1+3\delta) \cdot k < 1.4k$, where we use $\delta \leq 0.1$. (In contrast, the vertices in the copy of $G_{2k}^{(1-\delta)}$ have degree (in the $4n$-vertex graph $\pi(G_{A,A})$) at least $0.99 \cdot (1-\delta) \cdot 2k + (0.99 \cdot (1-\delta)/2) \cdot 2k = 0.99 \cdot 3 \cdot (1-\delta) \cdot k > 2.6k$.)

Claim 5.8 (lower bound for non-adaptively testing Π): *Any non-adaptive tester for Π_n has query complexity $\Omega(g(f(n)) \cdot f(n))$.*

Proof Sketch: Following the proof of Claim 3.1.2 (modulo the adaptations presented in Sections 4.1 and 4.2), we observe that the lower bound is still a square root of the actual information contents of the matrices, which yields a bound of $\Omega(\sqrt{(k/t') \cdot (k/t)}) = \Omega(f(n) \cdot g(f(n)))$.

Specifically, we consider the problem of distinguishing the uniform distribution on Π_n and the distribution generated by picking two random and independent (t', t)-redundant matrices $A = (a_{i,j})$ and $B = (b_{i,j})$ and outputting $\pi(G_{A,B})$ for a random $\pi \in \mathrm{Sym}_n$. In this case, the probability that two specific queries correspond to entries (i, j) in $M \odot A$ and (i', j') in $M \odot B$ such that $\lceil i/t' \rceil = \lceil i'/t' \rceil$ and $\lceil j/t \rceil = \lceil j'/t \rceil$ is $\frac{t'}{k} \cdot \frac{t}{k}$, which means that distinguishing these distributions requires $\Omega(\sqrt{k^2/t})$ non-adaptive queries. (This ignore the fact that distinguishing requires that $m_{i,j} = *$, which only acts in our favor.) On the other hand, with high probability, for two random and independent (t', t)-redundant matrices A and B, the matrices $M \odot A$ and $M \odot B$ disagree on more than $\delta \cdot k^2/3$ of their entries. This is because the entries in A and B that correspond to the $*$-entries of M are random and sufficiently independent (i.e., entries in different t'-by-t submatrices are independent)[29] and their number is at least $0.99 \cdot \delta \cdot k^2$. Note that a small δ merely means that the lower bound holds for a smaller value of the proximity parameter. ■

Proposition 5.9 (upper bound for adaptively testing Π): *There exists an adaptive tester for Π_n of query complexity $O(\epsilon^{-1} \cdot f(n) \cdot \log^3 n)$ if $\epsilon > n^{-0.49}$ and $O(1/\epsilon^{4.1})$ otherwise.*

The alternative tester (for $\epsilon \leq n^{-0.49}$) is trivial: In this case we can afford exploring the entire graph, since $n^2 \leq \epsilon^{-4.1}$.[30]

Proof: Our starting point is the tester used in the proof of Claim 3.1.2 (modulo the adaptations presented in Sections 4.1 and 4.2), but we need to significantly modify it so to avoid the (small) error probability on graphs in Π. As outlined above, this is done by avoiding any rejection decision that is based on statistical evidence. The main modifications are the following (whereas the actual description of the tester will be provided later).

- *A modification of Step 1*: We select uniformly at random a sample of size $\ell' = O(\ell)$, denoted T, to be used in all degree estimations (but not as a pool that contains the two locators, S_1 and S_2).[31] In particular, we shall use the edge density in the subgraph induced by T in order

[29] In the case of $f(n) = \sqrt{n}$ and $g(m) = m$ (equiv., $t' = t = 1$), the entries are totally independent.

[30] Indeed, 0.49 stands for any constant smaller than 0.5, and $4.1 > 2/0.49$.

[31] Recall that in the original tester, the random sample S was used both for degree estimation and as a pool for the selection of reliable locators (in Step 2).

to estimate how many vertices of T reside in the copy of $G_{2k}^{(\delta)}$, and determine the thresholds that distinguishes low and high degree vertices accordingly.

We stress that, in contrast to the original Step 1, we perform no checks in this modified step (and, needless to say, never reject in it).

Actually, we may (and do) pick T arbitrarily (rather than at random).

- *Branching and modification of Step 2*: Since T may provide a wrong degree estimation for few vertices, the identification of locators in the original Step 2 may be wrong, and lead to subsequent errors (in locating vertices), which in turn may lead to rejecting graphs in Π. We overcome the problem by branching to many executions, which use different samples for the locators, such that in almost all branches the locators are correctly identified (based on correct estimation of the vertices' degrees). We shall reject only if most branches reject (or rather if most sub-branches (introduced later) reject).

  Specifically, we branch to ℓ' executions, using the same degree approximation sample, T, in all of them. In the p^{th} branch we select, almost independently of other branches, a sample $S^{(p)}$, identify its low and high degree vertices (according to T), and select random ℓ-sets, $S_1^{(p)}$ and $S_2^{(p)}$, as in Step 2. These sets will be used as pseudo-locators (for threshold $\delta \cdot (1-\delta)$). If $S^{(p)}$ contains less that ℓ vertices of one type, we reset $S_i^{(p)}$ to be empty, but do not reject. We do reject, *in this branch*, if $S^{(p)}$ contains at least ℓ vertices of type i, but the selected $S_i^{(p)}$ is not a pseudo-locator, where from this point onwards whenever we mention a pseudo-locator we means a pseudo-locator for threshold $\delta \cdot (1-\delta)$.

  We make sure that the samples $S^{(p)}$ taken in the different branches are dispersed in the sense that any set of ℓ vertices is intersected by $O(\ell)$ samples. This guarantees that the (at most) ℓ vertices that were wrongly categorised based on their degree (as approximated by T) only affect the execution of $O(\ell) < \ell'/100$ of the branches.

- *Secondary branching and adaptation of subsequent steps*: Since a pseudo-locator may wrongly locate few vertices, the outcomes provided by the original Step 3 may be wrong, and lead to errors in the tests performed in subsequent steps, which in turn may lead to rejecting graphs in Π. We overcome the problem by branching to many executions, which use different samples for the subsequent steps, such that in almost all sub-branches all vertices are correctly located.

  Specifically, in the p^{th} foregoing branch, we branch again to ℓ' executions, all using the same pseudo-locators $S_1^{(p)}$ and $S_2^{(p)}$. In the q^{th} sub-branch we select, almost independently of other sub-branches, a sample $R^{(p,q)}$, identify its low and high degree vertices, denoted $R_1^{(p,q)}$ and $R_2^{(p,q)}$, and locally self-order $R_i^{(p,q)}$ using $S_i^{(p)}$. The later local self-ordering is performed analogously to Step 3, although the $S_i^{(p)}$'s may only be pseudo-locators. (If $S_i^{(p)}$ is empty, we define $R_i^{(p,q)}$ as empty.)

  We stress that if one of the $S_i^{(p)}$ is empty, then we invoke the locally self-order only for the other $R_{i'}^{(p,q)}$; in this case, if no failures or collision occurs, then we halt and accept. In any case, if some failure or collision occurs when performing local self-ordering, then we halt and reject *in this sub-branch*. (Recall that failure means that the procedure returns a failure symbol on some vertex in $R_{i'}^{(p,q)}$, and collision means that two different vertices in $R_{i'}^{(p,q)}$ were assigned the same location.)

We make sure that the samples $R^{(p,q)}$ taken in the different sub-branches (referring to $S^{(p)}$) are dispersed in the very sense defined above. This guarantees that the (at most) ℓ vertices that were wrongly located by $S_i^{(p)}$ only affect the execution of $O(\ell) < \ell'/100$ of the sub-branches.

The remaining steps are performed almost as in the original testers, where rejection in them will mean rejection in the corresponding sub-branch. The only significant modification is that Step 5 is replaced by a testing of the fixed part of the bipartite graphs (i.e., the part determined by the Boolean entries in the matrix M).

At the end, we reject if a majority of the sub-branches rejected; otherwise, we accept.

We first observe that the modified procedure never rejects a graph in Π. Intuitively, this holds because we either eliminated the sources of rejection (i.e., in Step 1) or confined them to few branches and sub-branches. Specifically, the *disperseness* condition guarantees that the ℓ exceptional vertices of $G^{(\delta)}$ whose degree is badly approximated affect the execution of $O(\ell)$ branches. Ditto for the ℓ exceptional vertices of $G^{(1-\delta)}$. Analogously, the ℓ exceptional vertices of $G^{(\delta)}$ that are incorrectly located by $S_1^{(p)}$ affect the execution of $O(\ell)$ sub-branches of the p^{th} branch. Hence, in total, only $O(\ell) \cdot \ell' + (\ell' - O(\ell)) \cdot O(\ell) = O(\ell \cdot \ell')$ of the sub-branches are affected, whereas a vast majority (e.g., more than 99%) of the other sub-branches vote for acceptance (and ℓ' is picked to dominate $O(\ell)$). We shall also have to show that the resulting algorithm still rejects graphs that are far from Π. In fact, we shall show that almost all sub-branches do so. But, first, let us spell out the algorithm.

The algorithm (claimed to be a one-sided error tester of Π). On input $\epsilon > n^{-0.49}$ and oracle access to $G' = ([4k], E')$, we proceed as follows.

1. *Selecting a sample for degree estimation*: We pick an arbitrary set of $\ell' = O(\ell)$, denoted T, and inspect the subgraph of G' induced by T. Let d denote the average degree of vertices in this subgraph, and let $\widetilde{\rho} = 1 - (d/\ell')$ be a rough approximation of the fraction of T that resides in the copy of $G_{2k}^{(\delta)}$ (assuming $G' \in \Pi$).

   Indeed, if $G' \in \Pi$ and ρ denotes the (actual) fraction of vertices of T that resides in the copy of $G_{2k}^{(\delta)}$, then $d = (1 \pm 0.01) \cdot ((1-\rho) \pm \delta) \cdot \ell' \pm 2\ell$. Furthermore, in this case, almost all vertices that reside in $G_{2k}^{(\delta)}$ have $(1 \pm 0.01) \cdot (0.5 - 0.5\rho + (\rho \pm 0.5) \cdot \delta) \cdot \ell'$ neighbors in T, whereas the other vertices have $(1 \pm 0.01) \cdot (1 - 0.5\rho - (1 \pm 0.5) \cdot \delta) \cdot \ell'$ neighbors in T.

   In light of the foregoing, we shall rule that a vertex has low degree if it has less than $(0.75 - 0.5 \cdot \widetilde{\rho}) \cdot \ell'$ neighbors in T.

2. *Finding reliable locators for each branch*: As stated above, we branch to ℓ' executions, using the same degree approximation sample, T, in all of them. In the p^{th} branch we select, almost independently of other branches (see below), a sample $S^{(p)}$ of ℓ' vertices (in $[4k] \setminus T$), and identify its low and high degree vertices (according to T).

   If $S^{(p)}$ contains less that ℓ vertices of low (resp., high) degree, then we set $S_1^{(p)}$ (resp., $S_2^{(p)}$) to be empty, but do not reject; otherwise (i.e., $S^{(p)}$ contains at least ℓ vertices of low (resp., high) degree), we select $S_1^{(p)}$ (resp., $S_2^{(p)}$) uniformly at random among all ℓ-subsets of $S^{(p)}$ that consists of vertices of low (resp., high) degree.

   If the subgraph of G' induced by $S_1^{(p)} \neq \emptyset$ (resp., by $S_2^{(p)} \neq \emptyset$) is not isomorphic to the subgraph of $G_{2k}^{(\delta)}$ (resp., of $G_{2k}^{(1-\delta)}$) that is induced by some pseudo-locator of $G_{2k}^{(\delta)}$ (resp., of $G_{2k}^{(1-\delta)}$), then we halt and *reject*.

Recall that we make sure that the samples $S^{(p)}$ taken in the different branches are dispersed in the sense that any set of ℓ vertices is intersected by $O(\ell)$ samples. As argued below (see paragraph on *bounding the deviation caused by the disperseness condition*), this can be done using $S^{(p)}$'s that are distributed *almost independently* of one another.

3. *Sampling the two sides of the graph in each sub-branch and locating the sampled vertices*: In the p^{th} foregoing branch, we branch again to ℓ' executions, all using the same pseudo-locators $S_1^{(p)}$ and $S_2^{(p)}$. In the q^{th} sub-branch we select, almost independently of other sub-branches, a sample $R^{(p,q)}$ of $O(f(n)/\epsilon)$ vertices (in $[4k] \setminus (T \cup S^{(p)})$), and identify its low and high degree vertices (according to T). Denoting the corresponding sets by $R_1^{(p,q)}$ and $R_2^{(p,q)}$, we locally self-order the vertices of $R_i^{(p,q)}$ using $S_i^{(p)}$. If $S_i^{(p)}$ is empty, then we reset $R_i^{(p,q)}$ to be empty.

   We locally self-order the vertices of $R_1^{(p,q)}$ using $S_1^{(p)}$ by querying G' on all pairs $R_1^{(p,q)} \times S_1^{(p)}$ and looking at our hard-wired copy of $G_{2k}^{(\delta)}$. Specifically, letting $S_1^{(p)} = \{s_1, ..., s_\ell\}$ and $s_0 = v \in R_1^{(p,q)} \times S_1^{(p)}$ be the vertex that we wish to locate, we look for an ℓ-subset $W = \{w_1, ..., w_\ell\}$ and a vertex $w_0 \in [2k]$ such that $\{s_j, s_{j'}\} \in E'$ if and only if $\{w_j, w_{j'}\} \in E^{(\delta)}$ for every $j, j' \in \{0, 1, ..., \ell\}$. If the foregoing condition identifies a unique w_0, then we assign v to location w_0; otherwise, we consider the local self-ordering as failing on v. Ditto for $v \in R_2^{(p,q)}$ (using $S_2^{(p)}$).

   If any of these invocation fails, then we *vote for rejection* in the corresponding sub-branch and suspend its execution. Ditto if two different vertices were assigned the same location. Otherwise, we define a bijection $\pi_i^{(p,q)} : R_i^{(p,q)} \to [2k]$ according to the local self-ordering procedure; that is, $\pi_i^{(p,q)}(v)$ is the location of $v \in R_i^{(p,q)}$ in $G_{2k}^{(c_i)}$, where $c_1 = \delta$ and $c_2 = 1 - \delta$.

   Recall that we make sure that the samples $S^{(p)}$ taken in the different sub-branches are dispersed in the same sense as in Step 2.

4. *Testing isomorphism of each side of G' to the corresponding side in a generic $G_{A,B}$*: For every p and q, we test that the subgraph of G' induced by $R_1^{(p,q)}$ equals the subgraph of $G_{2k}^{(\delta)}$ induced by $\pi_1^{(p,q)}(R_1^{(p,q)})$, and ditto for $R_2^{(p,q)}$ and $G_{2k}^{(\delta)}$. As in the original tester, this test is performed by checking $O(1/\epsilon)$ vertex-pairs at random. Again, a failed check leads to *voting for rejection* in the corresponding sub-branch (and suspension of its execution).

5. *Completing a test of isomorphism of G' to a generic $G_{A,B}$*: For every p and q, we test that the subgraph of G' induced by $R_1^{(p,q)} \times R_2^{(p,q)}$ fits the Boolean entries of the matrix M. Specifically, we pick $O(1/\epsilon)$ random vertex-pairs (u, v) in $R_1^{(p,q)} \times R_2^{(p,q)}$ and check whether the adjacency of u and v fits the $(\pi_1^{(p,q)}(u), \pi_2^{(p,q)}(v))^{\text{th}}$ entry in the matrix M; that is, letting $(i, j) \stackrel{\text{def}}{=} (\pi_1^{(p,q)}(u), \pi_2^{(p,q)}(v))$, we *vote for rejection* in the current sub-branch if the relevant condition among the following four conditions holds:

   (a) For $i, j \in [k]$: if either $m_{i,j} = 0$ and $\{u, v\} \in E'$ or $m_{i,j} = 1$ and $\{u, v\} \notin E'$.

   (b) For $i, j \in \{k + 1, ..., 2k\}$: if either $m_{i-k,j-k} = 0$ and $\{u, v\} \in E'$ or $m_{i-k,j-k} = 1$ and $\{u, v\} \notin E'$.

   (c) For $i \in [k]$ and $j \in \{k + 1, ..., 2k\}$: if either $m_{i,j-k} \neq 1$ and $\{u, v\} \in E'$ or $m_{i,j-k} = 1$ and $\{u, v\} \notin E'$.

(d) For $i \in \{k+1,...,2k\}$ and $j \in [k]$: if either $m_{i-k,j} \neq 1$ and $\{u,v\} \in E'$ or $m_{i-k,j} = 1$ and $\{u,v\} \notin E'$.

The first (resp., second) condition refers to edges in the first (resp., second) informative bipartite graph of a generic $G_{A,B}$, where a bipartite graph is called informative if it depends on the matrix A (resp., B). The third (resp., fourth) condition refers to edges in the first (resp., second) non-informative bipartite graph. Note that in the non-informative case $m_{i,j} = *$ mandates a non-edge, whereas in the informative case determining the adjacency depends on $a_{i,j}$ (resp., $b_{i,j}$), and this will be checked by Step 6.

6. *Testing that $A = B$*: For every p and q, we call a pair $(i,j) \in [k]^2$ a matrix-collision if there exists $u_1, v_1 \in R_1^{(p,q)}$ and $u_2, v_2 \in R_2^{(p,q)}$ such that $\lceil \pi_1^{(p,q)}(u_1)/t' \rceil = \lceil i/t' \rceil = \lceil (\pi_1^{(p,q)}(v_1) - k)/t' \rceil$ and $\lceil \pi_2^{(p,q)}(u_2)/t \rceil = \lceil j/t \rceil = \lceil (\pi_2^{(p,q)}(v_2) - k)/t \rceil$. In such a case, we call i a row-collision, and call j a column-collision.

   Letting I (resp., J) denote the set of row-collisions (resp., column-collision), we select a random set of $\min(|I|, |J|, \Theta(1/\epsilon))$ disjoint pairs $P \subseteq I \times J$ (i.e., distinct $(i_1,j_1),(i_2,j_2) \in P$ satisfy both $i_1 \neq i_2$ and $j_1 \neq j_2$). For each $(i,j) \in P$, we query G' on the corresponding pairs (u_1,u_2) and (v_1,v_2), where $\lceil \pi_1^{(p,q)}(u_1)/t' \rceil = \lceil i/t' \rceil = \lceil (\pi_1^{(p,q)}(v_1) - k)/t' \rceil$ and $\lceil \pi_2^{(p,q)}(u_2)/t \rceil = \lceil j/t \rceil = \lceil (\pi_2^{(p,q)}(v_2) - k)/t \rceil$, and *vote for rejection* in this sub-branch if *all* the following conditions hold.

   (a) The first pair (i.e., (u_1,u_2)) corresponds to a non-fixed entry of M;
   that is, $m_{\pi_1^{(p,q)}(u_1), \pi_2^{(p,q)}(u_2)} = *$.

   (b) The second pair (i.e., (v_1,v_2)) corresponds to a non-fixed entry of M;
   that is, $m_{\pi_1^{(p,q)}(v_1)-k, \pi_2^{(p,q)}(v_2)-k} = *$.

   (c) The answers to these two queries disagree;
   that is, $\{u_1,u_2\} \in E'$ if and only if $\{v_1,v_2\} \notin E'$.

   (Note that in case one of the first conditions does not hold, we can check consistency of the answer with the relevant Boolean entry of M. However, this is unnecessary in light of Step 5.)

If the majority of all sub-branches vote for rejection, then we reject. Otherwise we accept.

The query complexity of the algorithm. Step 3 makes $\sum_{i\in[2],p,q\in[\ell']} |R_i^{(p,q)} \times S_i^{(p)}| = O(\epsilon^{-1} \cdot f(n) \cdot \log^3 n)$ queries, and dominates the total number of queries made by the foregoing algorithm. The extra $O(\log^2 n)$ factor (in comparison to the Theorem 4.2) is due to the branching.

Analysis of the execution on graphs in Π. The key point in proving that graphs in Π are accepted with probability 1 is showing that the categorization of vertices to low and high degree based on T is always correct (i.e., it is correct for all but at most 2ℓ vertices outside T, no matter which set T is chosen in Step 1). This calls for an analysis of the categorization rule defined in Step 1, which we undertake next.

Let $G' = \pi(G_{A,A})$ and $T_1 = T \cap \pi([2k])$. Then, the sum of the number of neighbors that vertices in T_1 have in T is

$$|T_1| \cdot \left((1 \pm 0.01) \cdot \delta \cdot (|T_1| - 1) + ((1 \pm 0.01) \cdot \frac{1 \pm \delta}{2} \cdot |T_2| \right) \pm 2\ell \cdot |T|$$

$$\approx \quad (1 \pm 0.01) \cdot \left(\rho \cdot \delta + (1-\rho) \cdot \frac{1 \pm \delta}{2} \right) \cdot |T| \cdot |T_1| \pm 2\ell \cdot |T|,$$

where $\rho \overset{\text{def}}{=} |T_1|/|T|$. Recalling that $|T| = \ell'$, this means that the average degree (in the subgraph induced by T) of vertices in T_1 is approximately $(1 \pm 0.01) \cdot ((0.5 - 0.5\rho) + (\rho \pm 0.5(1-\rho)) \cdot \delta) \cdot \ell' \pm (2\ell/\rho)$. Similarly, for vertices in $T_2 = T \setminus T_1$ the average degree is approximately $(1 \pm 0.01) \cdot ((1 - 0.5\rho) - (1 - \rho \pm 0.5\rho) \cdot \delta) \cdot \ell' \pm (2\ell/(1-\rho))$. Hence, using $\ell' > 400\ell$, the average degree of vertices in the subgraph induced by T is $(1 \pm 0.02) \cdot (1 - \rho + (2\rho - 1 \pm \rho(1-\rho))\delta) \cdot |T|$, which means that $\widetilde{\rho} \approx \rho$ (where $\widetilde{\rho}$ is the estimate of ρ that was determined in Step 1). The foregoing calculations presume that $|T_1|, |T_2| \geq \ell$, but the claims are trivial if this condition does not hold (because in that case $\min(\rho, 1-\rho) \ll \delta$). Hence, a threshold of $(0.75 - 0.5 \cdot \widetilde{\rho}) \cdot \ell'$ for the number of neighbors in T correctly distinguishes all but ℓ of the vertices in $\pi([2k]) \setminus T$ from those in $([4k] \setminus \pi([2k])) \setminus T$.

Using the hypothesis that the $S^{(p)}$'s are dispersed in the sense that the 2ℓ exceptional vertices appear in at most $O(\ell)$ of the ℓ' branches, it follows that in the other branches the vertices of $S^{(p)}$'s and all corresponding $R^{(p,q)}$ are categorized correctly. In these branches, all but at most 2ℓ of the vertices in $[n]$ are correctly located, which means that in at least $\ell' - O(\ell)$ sub-branches all vertices in $R^{(p,q)}$ are correctly located. (Recall that whenever the vertices of $S^{(p)}$ are categorized correctly, we obtain corresponding pseudo-locators.) Setting the hidden constant in the O-notation to 300 (see below), it follows that at least $\ell' \cdot (\ell' - 600\ell)$ of the sub-branches will not reject, which implies that the tester accepts (provided that $\ell' > 1200\ell$).

Bounding the deviation caused by the disperseness condition. The foregoing analysis relies on the disperseness of the collection of $S^{(p)}$'s and $R^{(p,q)}$'s. We show that almost independent sampling of the $S^{(p)}$'s and $R^{(p,q)}$'s satisfies this condition while not affecting the distribution of the sequence of sub-branches too much. This is indeed the case per our choice of the constant 300 and the hypothesis that $\epsilon > n^{-0.49}$ (and $f(n) \leq n^{0.5}$). Specifically, the probability that, when selecting ℓ' subsets of size $O(f(n)/\epsilon) = O(n^{0.99})$ in $[n] \setminus T$ (resp., in $[n] \setminus (T \cup S^{(p)})$), there exists ℓ vertices that occur in 150ℓ sets is upper-bounded by

$$\binom{n}{\ell} \cdot \binom{\ell'}{150\ell} \cdot \left(\frac{\ell \cdot O(n^{0.99})}{n - \ell - \ell'} \right)^{150\ell} = n^{-(0.5 - o(1))\ell}$$

where $\frac{\ell \cdot O(n^{0.99})}{n - \ell - \ell'}$ represents the probability of one of the ℓ vertices appearing in a random $O(n^{0.99})$-subset.

Analysis of the execution on graphs that are far from Π. We now turn to the case that G' is ϵ-far from Π, and show that each sub-branch rejects with high probability (while recalling that these executions are almost independent). As a warm-up, consider the case that $G' = \pi(G_{A,B})$, for some pair (A, B) of (t', t)-redundant matrices and a bijection $\pi : [4k] \to [4k]$. In this case, B must be $15\delta^{-1} \cdot \epsilon$-far from A, since the adjacencies determined by A and B constitute a $(1 \pm 0.01) \cdot \delta/16$ fraction of all vertex-pairs. It follows that Step 6 rejects with high probability, since it is likely to inspect $\Omega(1/\epsilon)$ disjoint matrix-collisions, which are uniformly and independently distributed in $[k]^2$.[32] Intuitively, if G' is close enough to some $\pi(G_{A,B})$, then the same argument holds, and otherwise Steps 2–5 reject with high probability. The actual analysis follows.

Let V_1 denote the set of vertices of G' that are categorized as being of low degree (according to T), and $V_2 = [4k] \setminus V_1$. We consider a single sub-branch, and simplify notations accordingly; that

[32]Indeed, if $G' = \pi(G_{A,B})$, then Step 6 is always reached.

is, we omit the superscripts (p) and (p,q). The following argument repeats some of the arguments made in the proof of Claim 3.1.3, but these repetitions are called for since some of the objects (i.e., T and the V_i's) are different here.

Assuming that Steps 1–2 were completed successfully (i.e., without rejection), we define a function $\phi_1 : V_1 \to [2k] \cup \{\perp\}$ such that $\phi_1(v)$ denotes the answer of the local self-ordering (based on S_1) to the input v, which may be a failure symbol, denoted $\perp$, where failure may happen because the subgraph of G' induced by V_1 is not necessarily isomorphic to $G_{2k}^{(\delta)}$. Similarly, we define $\phi_2 : V_2 \to [2k] \cup \{\perp\}$ based on S_2. Note that ϕ_i agrees with π_i on R_i.

Letting $\epsilon' = \Omega(\epsilon)$, we may assume that ϕ_i does not evaluate to $\perp$ on more than $\epsilon' k$ points, since otherwise such a point is sampled (w.h.p.) by Step 3, leading it to reject. Likewise, ϕ_i does not have more than $\epsilon' k$ points that have an image with several pre-images under ϕ_i, where here a Birthday argument proves the claim (since the collision probability is at least $1/\epsilon' k$). In particular, it follows that $|V_i| \leq (2+2\epsilon')\cdot k$, because otherwise ϕ_i would have had more than $\epsilon' k$ points that have an image with several pre-images under it. (The latter assertion also uses the randomness of S, which guarantees that each V_i is sampled proportionally by S, which implies that if $|V_i| > (2+2\epsilon')\cdot k$ then (w.h.p.) S_i is not empty.) Denote by $V'_i \subseteq V_i$ the set of vertices that were not discarded above; that is, $V'_i = \{v \in V_i : \phi_i(v) \neq \perp \,\&\, |\phi_i^{-1}(\phi_i(v))| = 1\}$. Using $|V_i| \geq (2-2\epsilon')\cdot k$, we note that $|V'_i| \geq (2-4\epsilon')\cdot k$ must hold, and define ϕ'_i as the restriction of ϕ_i to V'_i. Lastly, let $\phi' : V'_1 \cup V'_2 \to [4k]$ such that $\phi'(v) = \phi'_i(v)$ if $v \in V'_1$ and $\phi'(v) = 2k + \phi'_i(v)$ otherwise, and let ϕ be an arbitrary extension of ϕ' to a permutation over $[4k]$.

Relying on Step 4 (and assuming that it does not reject w.h.p), we infer that the subgraph of $\phi(G')$ induced by $[2k]$ (resp., by $\{2k+1, ..., 4k\}$) is $5\epsilon'$-close to $G_{2k}^{(\delta)}$ (resp., to $G_{2k}^{(1-\delta)}$, when relabeling its vertices by subtracting $2k$ to each label). Relying on Step 5 (and assuming that it does not reject w.h.p), we infer that $\phi(G')$ is $6\epsilon'$-close to some $G_{A,B}$ for some matrices A and B (which are not necessarily (t',t)-redundant). Recalling that G' is ϵ-far from Π (and using $\epsilon' = \epsilon/7$), we infer that A and B must disagree on more than $\epsilon' \cdot 16 \cdot k^2$ entries that are not fixed in M. We claim that in this case Step 6 rejects with high probability.

Recalling that $|V_i| = (2k \pm 0.2k)$, we observe that with high probability (over the choice of R_1 and R_2), Step 6 defines a set of at least $10/\epsilon$ disjoint matrix-collision pairs, and that these pairs are independently and uniformly distributed in $[k] \times [k]$. Such a pair (i,j) causes rejection if there exists $u_1, v_1 \in R_1$ and $u_2, v_2 \in R_2$ such that the following conditions hold:

1. The corresponding queries, (u_1, u_2) and (v_1, v_2), are not fixed in M; that is, $m_{\pi_1(u_1),\pi_2(u_2)} = m_{\pi_1(v_1),\pi_2(v_2)} = *$.

2. These entries that are supposed to be equal in A and B; that is, $\lceil \pi_1(u_1)/t' \rceil = \lceil i/t' \rceil = \lceil (\pi_1(v_1)-k)/t' \rceil$ and $\lceil \pi_2(u_2)/t \rceil = \lceil j/t \rceil = \lceil (\pi_2(v_2)-k)/t \rceil$.

3. Yet, the foregoing entries are not identical in A and B; that is, $a_{\pi_1(u_1),\pi_2(u_2)} \neq b_{\pi_1(v_1),\pi_2(v_2)}$.

The probability that this event holds (i.e., all three conditions hold) is lower-bounded by $15\epsilon' - 4\epsilon' - 4\epsilon' = \epsilon$, and the claim follows. ■

Claim 5.10 (upper bound for non-adaptively testing Π): *There exists a non-adaptive tester for* Π_n *of query complexity* $O(\epsilon^{-2} \cdot g(f(n)) \cdot f(n) \cdot \log^2 n + \epsilon^{-1} \cdot f(n) \cdot \log^3 n)$, *if* $\epsilon > n^{-0.49}$ *and* $O(1/\epsilon^{4.1})$ *otherwise.*

Proof Sketch: We use a non-adaptive version of the adaptive tester presented in the proof of Proposition 5.9. The observations and modifications regarding this matter that were made in the proof of Theorem 4.1 apply here (without any change). Specifically, for Step 6, we select a random subset R' of $O(g(f(n))/\epsilon)$ vertices of each relevant $R = R^{(p,q)}$, and make non-adaptive queries to all pairs in $R \times R'$, which means making $O(g(f(n)) \cdot f(n)/\epsilon^2)$ queries.[33] (This quantity has to be multiplied by the number of sub-branches, whereas the $O(\epsilon^{-1} \cdot f(n) \cdot \log^3 n)$ term arises from Step 3.) ■

Claim 5.11 (lower bound for adaptively testing Π): *Any non-adaptive tester for Π_n has query complexity $\Omega(f(n))$.*

Proof Sketch: This bound follows merely by considering matrix-collisions among samples of the k rows of the matrix. Specifically, we used the same distributions as in the proof of the lower bound for non-adaptive testers, and observe that $\Omega(\sqrt{k/t'}) = \Omega(f(n))$ adaptive queries are needed in order to find vertices of G' that are mapped (by π) to location i and $i' + k$, for some $i, i' \in [k]$ such that $\lceil i/t' \rceil = \lceil i'/t' \rceil$. ■

Conclusion. Having established all four bounds, Theorem 5.1 follows.

6 Proof of Theorem 1.7

As stated in Section 1.4, we prove Theorem 1.7 in two steps. In the first step, presented in Section 6.1, we use graph blow-ups to derive a version of Theorem 1.4 from Theorem 1.3, where in this version the $O(\log n)$ factor (in the upper bounds) is replaced by an $O(\log f(n))$ factor. In the second step, presented in Section 6.2, we establish Theorem 1.7 by applying the methodology of [7] to the result of the first step. Each of the two main proofs starts with an overview, which gives an idea of the techniques involved.

6.1 Revising Theorem 1.4

In the following version of Theorem 1.4 the $O(\log n)$ factor in the upper bounds is replaced by an $O(\log f(n))$ factor. (On the other hand, the upper bound on adaptive testing increases by a factor of $1/\epsilon$.)

Theorem 6.1 (a version of Theorem 1.4):[34] *For every functions $f, g : \mathbb{N} \to \mathbb{N}$ such that $f(n) \leq \sqrt{n}$ and $g(m) \leq m$, there exists a graph property Π that satisfies the following two conditions, for $\epsilon \geq 2/f(n)^2$:*

1. *There exists a general* (i.e., adaptive) *tester of Π that makes $O(\epsilon^{-2} \cdot f(n) \cdot \log f(n))$ queries, and any such tester must make $\Omega(f(n))$ queries.*

2. *Any non-adaptive tester of Π must make $\Omega(g(f(n)) \cdot f(n))$ queries, and there exists such a tester that makes $O(\epsilon^{-2} \cdot g(f(n)) \cdot f(n) \cdot \log f(n))$ queries.*

[33] See also the relevant part of the proof of Theorem 4.2.

[34] The adaptive upper bound presumes $f(n)/\epsilon^2 = o(n)$; otherwise we incur another $O(\log f(n))$ factor. See Footnote 45.

The adaptive upper bound holds also for $\epsilon < 2/f(n)^2$.

Proof Sketch: We start with a rough outline of the proof, which is based on graph blow-ups (i.e., replacing vertices by clouds and connecting clouds that correspond to adjacent vertices by complete bipartite graphs). For $n' = f(n)^2$, starting with a property of n'-vertex graphs as provided by Theorem 1.3 and denoted Π', we apply n/n'-factor blow-up to these graphs, obtaining a property of n-vertex graphs, denoted Π. We prove that testing whether an n-vertex graph is in Π is closely related to testing whether an n'-vertex graph is in Π', where the query complexity of (adaptively) testing Π' is $\widetilde{O}(\sqrt{n'}/\epsilon))$. The foregoing strategy follows [11], but implementing it in the current context is more complex (for reasons to be discussed shortly).

We first show that Π is not easier to test than Π' by reducing testing Π' to testing Π. This reduction is identical to the one in [11] and its validity relies on the fact that (balanced) graph blow-up preserve the relative distance between graphs. The latter result is highly non-trivial, but it was proved in [17, Sec. 4] (see also [11, Lem. 4.3], which establishes a special case that suffices for us).

The more challenging part is showing that Π is not harder to test than Π'. In [11], this was proved by testing that the input graph is an n/n'-factor blow-up of some n'-vertex graph, while using $O(n')^2$ queries, and testing that the n'-vertex graph is in Π' by merely retreiving it. This could be afforded in [11], since there the query complexity of Π' was $\Omega(n')^2$. Here, however, we use a property Π' of query complexity $O(\sqrt{n'}/\epsilon)$ and so we cannot afford such expensive tests. Nevertheless, we show that Π is not harder to test than Π' by using the same high-level strategy with a totally different implementation. We stress that the latter claim (i.e., Π is not harder to test than Π') may not be true in general, but it does hold for the special case of Π' that we shall use (i.e., the properties presented in the proof of Theorem 1.3).[35]

In particular, in the special case that we use (i.e., Π' of Theorem 1.3), we can reduce testing Π to testing Π' by using the fact that the graphs in Π' are locally self-ordered (in a "uniform" sense (i.e., essentially, the same local self-ordering procedure works for all graphs in Π')).[36] The key observation is that using this local self-ordering procedure (for an unknown graph in Π') we can emulate oracle access to $G' \in \Pi'$ by making queries to an n/n'-factor blow-up of G', and if the emulation fails then we can definitely reject.

The foregoing suffices *if it is guaranteed that the tested graph is an n/n'-factor blow-up of some n'-vertex graph*, but we need to test Π without the foregoing guarantee. Towards this end, we design a tester that first checks that the tested graph is "similar" to an n/n'-factor blow-up of an n'-vertex graph for which the foregoing local self-ordering procedure of Π' does not fail. Specifically, we test the distribution induced by selecting uniformly a vertex in the tested graph and "locating" it in the n'-vertex graph (equiv., determining the index of its cloud), accepting if this distribution is uniform over $[n']$ and rejecting if it is far from the uniform distribution over $[n']$. (Indeed, we combine the local self-ordering procedure with a test of the uniform distribution over $[n']$, which has complexity $O(\sqrt{n'}/\epsilon^2)$.) Although this does not establish that the tested graph is close to being an n/n'-factor of an n'-vertex graph (e.g., the density of edges between "clouds" is not necessarily either 0 or 1)[37],

[35]In contrast, if Π' is the set of all n'-vertex graphs, then testing Π' is trivial but testing Π is not.

[36]Recall that Π' consists of $9k$-vertex graphs that are isomorphic to graphs of the form $G_{A,A}$ for some k-by-k matrix A. The foregoing "uniformity" condition means that the same local self-ordering procedure works for all $G_{A,A}$'s (and, in fact, even for all $G_{A,B}$'s). This is because this specific procedure actually ignores the adjacencies that depend on A (and B).

[37]Likewise, the clouds are not necessarily independent sets. Although we could have checked both conditions using $O(\sqrt{n'}/\epsilon^2)$ adaptive queries, it is not clear how to perform such a checking using less than $\Omega(n')$ queries, which we

the foregoing similarity does suffice for emulating the *specific tester* of the specific Π' that we use. We stress that the latter claim relies both on the structure of Π' and on the structure of the tester for Π'.

The actual proof. Our starting point is Theorem 1.3, and our construction of Π includes all n-vertex graphs that result from an $n/f(n)^2$-factor blow-up of some $f(n)^2$-vertex graph in the property of Theorem 1.3. Specifically, letting $n' = f(n)^2$, we first invoke Theorem 1.3 and obtain a property Π' of n'-vertex graphs having a $g(\sqrt{n'})$ factor gap between its adaptive and non-adaptive query complexities (which are $\widetilde{\Theta}(\sqrt{n'})$ and $\Theta(g(\sqrt{n'}) \cdot \sqrt{n'})$, respectively, when ignoring factors that depends on $1/\epsilon$). Next, we consider an n/n'-factor blow-up of each $G' \in \Pi'$, resulting in an n-vertex graph G, which means that the vertex set of G can be partitioned into n' equal-sized sets, called clouds, such that the edges between these clouds represent the edges of G'; that is, if $\{i, j\}$ is an edge in G', then there is complete bipartite between the i^{th} cloud and the j^{th} cloud, and otherwise there are no edges between this pair of clouds. (In particular, the clouds are independent sets.) Hence, Π is the set of graphs obtained by a n/n'-factor blow-up of the graphs in Π'.

The lower bounds on the (adaptive and non-adaptive) query complexity of Π are derived by reducing testing Π' to testing Π. That is, we construct a tester T' for Π' by invoking the tester T for Π and emulating a n/n'-factor blow-up of the tested graph G' (which is given to T' as an oracle). Specifically, the query $(\langle u', i\rangle, \langle v', j\rangle)$ regarding the n/n'-factor blow-up of G', denoted G, is answered by querying G' about (u', v'), where indeed $\langle u', i\rangle$ is adjacent to $\langle v', j\rangle$ in G if and only if u' is adjacent to v' in G'. Clearly, if $G' \in \Pi'$, then $G \in \Pi$. Using [17, Sec. 4] it follows that *if G' is ϵ-far from Π', then G is $\Omega(\epsilon)$-far from* Π. (Alternatively we can use [11, Lem. 4.3] and the fact that vertices in each $G' \in \Pi'$ have significantly different neighborhoods.)[38]

The upper bounds on the (adaptive and non-adaptive) query complexity of Π are derived by reducing testing Π to testing Π'. This reduction is far more complicated. In particular, we cannot afford testing that the input graph G is an n/n'-factor blow-up of some n'-vertex graph, because, unlike in [11], the (general) query complexity we aim at is $\text{poly}(1/\epsilon) \cdot \widetilde{O}(\sqrt{n'})$ (rather than $\text{poly}(1/\epsilon) \cdot O(n')^2)$.[39] Likewise, we cannot afford a trivial test for Π', again because we aim at query complexity $\text{poly}(1/\epsilon) \cdot \widetilde{O}(\sqrt{n'})$ (rather than $\text{poly}(1/\epsilon) \cdot O(n')^2)$. Nevertheless, we employ the same strategy as in [11]: we first test whether the input graph G resembles (albeit in a weaker sense) an n/n'-factor blow-up of some graph G', and then test whether G' is in Π' by running a *specific* tester for Π' and emulating G' (using queries to G). However, each of these steps is performed differently than in [11]. In particular, we make crucial use of various features of Π', including its having (uniform) local self-ordering procedures.

The blow-up test. When testing whether G resembles an n/n'-factor of some n-vertex graph, we take advantage of the fact that graphs in Π' have locally self-ordered procedures; furthermore, with very high probability, a random set of logarithmic size is (essentially) a reliable locator. Actually, the locating procedure ignores the edges that connects the two parts of $G' \in \Pi'$ (which are the only edges on which graphs in Π' differ); that is, it first determines the part in which the vertex resides,

cannot afford $g(m) = o(m)$.

[38]In [11], this feature was called *disperseness*. That is, Π' is α-dispersed if every two vertices in any $G' \in \Pi'$ differ on at least $\alpha \cdot n'$ neighbors. Note that the property Π' defined in the proof of Theorem 1.3 is $\Omega(1)$-dispersed, since the graphs in Π' result from combining two robustly self-ordered graphs such that vertices in the two graphs have vastly different degrees. (Now, by [11, Lem. 4.3], if Π' is α-dispersed and G' is ϵ-far from Π', then G (i.e., the blow-up of G') is $\Omega(\alpha\epsilon)$-far from Π.)

[39]The property Π' used in [11] is of extremely high query complexity (i.e., query complexity $\Theta(n')^2)$.

and then locates this vertex within its part (using the corresponding reliable locator).[40]

Specifically, we select a random set of $O(\log n')$ vertices of G, denoted S', and use it to locate (in G') a sample of $O(\sqrt{n'}/\epsilon^2)$ random vertices (of G), which means that we make $O(\epsilon^{-2}\sqrt{n'}\log n')$ queries to G. We stress that both samples are of vertices of the input n-vertex graph G, not of the n'-vertex graph G' that we envision. Nevertheless, when G is a n/n'-factor blow-up of $G' \in \Pi'$, the locating procedure behaves as if the vertices reside in an isomorphic copy of G'. (We may assume that the vertices of S' reside in different clouds, whereas when analyzing an invocation of the locating procedure we consider only the adjacencies among S' and the input vertex (to be located).) Hence, when we invoke the locating procedure on a vertex of G, we either get a location in G' or else we can safely reject. Consequently, we may assume, that we always get a location in G'. Next, we view these locations as a sample of $O(\sqrt{n'}/\epsilon^2)$ vertices in G', and employ a *distribution tester for uniformity* (see, e.g., [6, Sec. 11.2.1]); that is, we test whether the sample looks as being drawn (with repetitions) from the uniform distribution over the set of vertices of G'.

If $G \in \Pi$ (and S' is a reliable locator), then the distribution of locations in G' assigned to uniformly selected vertices of G is uniform over the vertices of G', and the uniformity test accepts with high probability. On the other hand, for any $\epsilon' = \Omega(\epsilon)$ of our choice, if the distribution of locations in G' assigned to uniformly selected vertices of G is ϵ'-far from the uniform distribution (over the vertices of some $G' \in \Pi'$), then the uniformity test rejects with high probability. Hence, we may assume that the distribution of locations in G' is ϵ'-close to the uniform distribution. In this case, the corresponding partition $(S_1, ..., S_{n'})$ of the vertices of G (according to their label in G') is ϵ'-close to an n'-way partition to equal parts. That is, the distribution that results from selecting uniformly $i \in [n']$ and $v \in S_i$ is ϵ'-close to the uniform distribution over $\bigcup_{i\in[n']} S_i$.

A crucial fact is that the foregoing partition $\overline{S} = (S_1, ..., S_{n'})$ induces a mapping of the vertices of the tested graph G to the vertices of a generic n'-vertex graph. Furthermore, if there are no edges inside the S_i's and the density of edges between each pair (S_i, S_j) is either 0 or 1, then G is a (general) blow-up of an n'-vertex graph. Specifically, for an n'-way partition $\overline{S} = (S_1, ..., S_{n'})$ of $[n]$, we say that the graph $G = ([n], E)$ is an $\overline{S}$-blow-up of a graph $G' = ([n'], E')$ if G is obtained by replacing each vertex $i \in [n']$ with a cloud S_i and connecting the i^{th} and j^{th} clouds by a complete bipartite graph if and only if i is connected to j in G'. (Indeed, if $\overline{S}$ is an equipartitions, then G is an n/n'-factor blow-up of G'.)

Loosely speaking, in the following second step, for $\overline{S}$ as determined in the blow-up test, we test whether G is an $\overline{S}$-blow-up of some graph in Π'; actually, we test whether G is an $\overline{S}$-blow-up of a graph of the form $G_{A,A}$ as in Construction 3.1.1 (modulo the modification in Section 4.1). Specifically, let Φ' denote the set of all graphs of this type (i.e., the said $G_{A,A}$'s), and $\Phi_{\overline{S}}$ denote the set of all graphs obtained by $\overline{S}$-blow-up of some graph in Φ'. Note that Φ' (resp., $\Phi_{\overline{S}}$) is not closed under isomorphism[41], and that Π' (resp., Π) is the set of all graphs that are isomorpophic to some graph in Φ' (resp., in $\Phi_{\overline{S}}$). The following (second) step actually tests membership in the set of graphs obtained by an $\overline{S}$-blow-up of graphs in Φ'; that is, it tests membership in $\Phi_{\overline{S}}$. Note that if $\overline{S}$ is an equipartitions, then any graph in $\Phi_{\overline{S}}$ is isomorophic to an n/n'-factor blow-up of some graph in Π' (equiv., to a graph in Π).

The second test. Let us first assume, for simplicity, that $\overline{S} = (S_1, ..., S_{n'})$ is an n'-way partition

[40]Recall that $G' \in \Pi'$ is isomorphic to a $9k$-vertex graph of the form $G_{A,A}$ which consists of two parts that are connected according to the entries of the k-by-k matrix A. These two parts have $2k$ and $7k$ vertices, respectively, and in each part almost every $O(\log k)$-set of vertices constitutes a reliable locator for this part.

[41]In fact, each graph in Φ' is (robustly) self-ordered.

of the vertices of G to equal parts, and consider what happens when we select at random a representative in each S_i, and test the corresponding induced subgraph, denoted R, for membership in Φ'. Clearly, if $G \in \Phi_{\overline{S}}$, then any such choice of representatives would result in a graph in Φ' (i.e., R is always in Φ'), and the tester accepts with high probability. But if G is ϵ-far from $\Phi_{\overline{S}}$, then (as shown in Claim 6.1.1) the expected distance between R and Φ' is at least $\Omega(\epsilon)$, and the tester rejects with high probability. We stress that the foregoing implication is not a generic fact; its validity relies on a feature of the specific property Π' (and it does not hold regardless of Π').[42]

Claim 6.1.1 (the distance of R from Φ' reflects the distance of G from $\Phi_{\overline{S}}$, special case): *Suppose that $|S_i| = n/n'$ for every $i \in [n']$, and let R be defined as above* (i.e., as the subgraph of G induced by selecting a random representative in each S_i). *Then, the expected distance of R to Φ' is at least half the distance of G to $\Phi_{\overline{S}}$ minus $1/n'$.*

Recalling that $\frac{1}{n'} \leq \epsilon/2$, it follows that if G is ϵ-far from Π, then the expected distance of R to Φ' is at least $\epsilon/4$.

Proof: As noted above, the current claim, which is not generic, relies on a feature of the specific property Φ'. Specifically, we rely on the fact that Φ' is chracterized by two conditions:

1. The adjacency of some vertex-pairs is uniquely determined accross Φ'; that is, for some $(u', v') \in [n']^2$ it holds that either u' is adjacent to v' in all graphs in Φ' or u' is not adjacent to v' in all graphs in Φ'.

2. All other vertex-pairs are matched such that, in each graph in Φ', the adjacency of one vertex-pair equals the adjacency of the vertex-pair matched to it. That is, if the vertex-pair (u', v') is matched with the vertex-pair (u'', v''), then in each graph G' in Φ' it holds that u' is adjacent to v' in G' if and only if u'' is adjacent to v'' in G'.

For each vertex-pair of Type 1, the expected distance (in R) to the determined value equals the fraction of vertex-pairs in G whose adjacency should be modified to fit any graph in $\Phi_{\overline{S}}$ (equiv., fit the n/n'-factor blow-up of any graph in Φ'). We next show that, for each pair of vertex-pairs of Type 2, the expected distance (in R) to equality (between the adjacencies) is at least half the distance to equality in G. Fixing any such pair (u', v') and (u'', v''), let ρ' (resp., ρ'') denote the fraction of adjacent vertex-pairs among the corresponding clouds (in G).[43] Then, the probability that this pair of vertex-pairs should be modified in R is $\rho'(1-\rho'') + (1-\rho')\rho''$, where the first (resp., second) term corresponds to the probability that the first pair is adjacent and the second is non-adjacent (resp., the first pair is non-adjacent and the second is adjacent). On the other hand, the fraction of adjacencies that must be modified in G in order to get the same adjacencies between the two pairs of clouds is $\min((1-\rho') + (1-\rho''), \rho' + \rho'')$, where the first (resp., second) term corresponds to making the corresponding vertex-pairs adjacent (resp., non-adjacent). We note that $\rho'(1-\rho'') + (1-\rho')\rho'' \geq \min((1-\rho') + (1-\rho''), \rho' + \rho'')/2$.[44] Lastly, we note that the number of edges in G that reside within S_i's is upper-bounded by $n' \cdot \binom{n/n'}{2} < \frac{1}{n'} \cdot \binom{n}{2}$. ■

[42]For example, this fact does not hold if Π' is the set of all n'-vertex graphs. Consider a graph G that is obtained by taking an n/n'-factor blow-up of an n'-vertex clique and omitting at random half of the edges in each $K_{\frac{n}{n'},\frac{n}{n'}}$. Then, G is far from any graph in Φ, although any subgraph R selected as above is in Π'.

[43]That is, ρ' (resp., ρ'') is the fraction of vertex-pairs in the clouds of u' and v' (resp., in the clouds of u'' and v'') that are adjacent in G.

[44]For example, assuming, without loss of generality, that $\rho' + \rho'' \leq 1$, we note that $\rho + \rho'' - 2\rho'\rho'' \geq (\rho' + \rho'')/2$, since the function $f(x,y) = x + y - 4xy$ is non-negative in the region $\{(x,y) \in [0,1]^2 : x + y \leq 1\}$.

The foregoing discussion refers to a mental experiment of testing the random induced subgraph R for membership in Φ', but in reality we only have oracle access to the graph G. Hence, we test R for Φ' by invoking the tester for Π' and emulating the answers to the queries made to R by referring them to G. Specifically, recall that we may assume, without loss of generality, that the tester for Π' samples s random vertices and queries some pairs among these s vertices. (If the tester is adaptive, then the queried pairs are determine by previous answers; otherwise, they are determined beforehand.) Hence, we may sample s vertices in G, provide them to the tester of Π', and answer the queries regarding G' by querying the corresponding pairs in G.

The latter outline ignores the fact that the number of vertices sampled by the tester for the relevant Π' is $\Theta(\sqrt{n'}/\epsilon)$, and so the s vertices sampled in G are unlikely to reside in different clouds. Instead, we sample $2s$ vertices in G, check the clouds in which they reside by using the localization procedure (which was already used in the first step), identify s vertices that reside in different clouds, and use them for the emulation of the tester.[45] Note that this means that we make $O(s \log n')$ queries just for this purpose, but this is fine because the query complexity of the specific tester that we use is $\Omega(\sqrt{n'} \log n')$ anyhow. (Note that there is no need to map the vertices of G to vertices of G' via the localization procedure, which maps every vertex in S_i to i, since the subgraph of G that we explore is isomorphic to the corresponding subgraph of G'.)

Recall that the above description relied on the simplified assumption that all S_i's are of equal size. But the success of the first test only allows us to assume that $\overline{S} = (S_1, ..., S_{n'})$ is ϵ'-close to an n'-way partition of the vertices of $G = ([n], E)$ to equal parts; that is, there exists an n'-equipartition $\overline{S}' = (S'_1, ..., S'_{n'})$ of $[n]$ such that for all but at most $\epsilon' n$ of the vertices $v \in [n]$ it holds that $v \in S_i$ if and only if $v \in S'_i$. Nevertheless, we shall proceed essentially as in the simplified case. Specifically, we test that the corresponding n'-partite subgraph of G is an $\overline{S}$-blow-up of a graph in Π' by selecting at random a representative in each S_i, and testing the corresponding induced subgraph. That is, the graph R is defined exactly as in the simplified case, although $\overline{S}$ is not necessarily an equipartition; again, for every $i, j \in [n']$, the ith vertex of R is connected to the jth vertex with probability that equals the fraction of edges between S_i and S_j.

Claim 6.1.2 (the distance of R from Φ' reflects the distance of G from Φ, general case): *Suppose that $\overline{S}$ is ϵ'-close to an n'-equipartition of $[n]$, denoted $\overline{S}'$. Then, the expected distance of R to Φ' is at least half the distance of G to $\Phi_{\overline{S}}$ minus $\frac{1}{n'} + O(\epsilon')$.*

Proof: Let R' be the induced subgraph of G obtained by selecting a random representative in each S'_i; that is, the ith vertex of R' is connected to the jth vertex with probability that equals the fraction of edges between S'_i and S'_j (in G). The key observation is that, for every n'-vertex graph G', the expected distance between R and G' and the expected distance between R' and G' differ by at most $8\epsilon'$. To see this, let $E(A, B)$ denote the set of edges (in G) that connect the disjoint vertex sets A and B, and note that the difference between the foregoing expectations is upper-bounded by

$$\frac{1}{\binom{n'}{2}} \cdot \sum_{\{i,j\} \in \binom{[n']}{2}} \left| \frac{|E(S_i, S_j)|}{|S_i| \cdot |S_j|} - \frac{|E(S'_i, S'_j)|}{|S'_i| \cdot |S'_j|} \right|$$

$$\leq \frac{1}{n'(n'-1)} \cdot \sum_{i,j \in [n']} \left| \frac{|E(S_i, S_j)|}{|S'_i| \cdot |S'_j|} - \frac{|E(S'_i, S'_j)|}{|S'_i| \cdot |S'_j|} \right| + \frac{1}{n'(n'-1)} \cdot \sum_{i,j \in [n']} \left| \frac{|E(S_i, S_j)|}{|S'_i| \cdot |S'_j|} - \frac{|E(S_i, S_j)|}{|S_i| \cdot |S_j|} \right|$$

[45] This presupposes that $s = o(n')$; otherwise we just find all vertices of G' using $O(n' \log n')$ attempts.

$$\begin{aligned}
&= \frac{1+o(1)}{n^2} \cdot \sum_{i,j\in[n']} \big||E(S_i,S_j)| - |E(S'_i,S'_j)|\big| \\
&\quad + \frac{1+o(1)}{n^2} \cdot \sum_{i,j\in[n']} \frac{|E(S_i,S_j)|}{|S_i|\cdot|S_j|} \cdot \big||S_i|\cdot|S_j| - |S'_i|\cdot|S'_j|\big| \\
&\le \frac{1+o(1)}{n^2} \cdot \sum_{i,j\in[n']} \texttt{SymDiff}(S_i\times S_j, S'_i\times S'_j) \\
&\quad + \frac{1+o(1)}{n^2} \cdot \sum_{i,j\in[n']} \big||S_i|\cdot|S_j| - |S'_i|\cdot|S'_j|\big| \\
&\le \frac{2+o(1)}{n^2} \cdot \sum_{i,j\in[n']} \big(\texttt{SymDiff}(S_i,S'_i)\cdot(|S_j|+|S'_j|) + \texttt{SymDiff}(S_j,S'_j)\cdot(|S_i|+|S'_i|)\big)\,.
\end{aligned}$$

Observing that the symmetric difference between S_k and S'_k consists of either $S_k \setminus S'_k$ or $S'_k \setminus S_k$, we get an upper bound of

$$\frac{2+o(1)}{n^2} \cdot 2 \cdot \sum_{i,j\in[n']} \big||S_i| - |S'_i|\big| \cdot (|S_j|+|S'_j|) \;=\; \frac{4+o(1)}{n^2} \cdot 2n \cdot \sum_{i\in[n']} \big||S_i| - |S'_i|\big|\,,$$

which is upper-bounded by $17\epsilon'$. Next, applying Claim 6.1.1 we lower-bound the expected distance between R' and Φ' by half the distance of G to $\Phi_{\overline{S}'}$ minus $1/n'$. Lastly, we note that the distance between $\Phi_{\overline{S}}$ and $\Phi_{\overline{S}'}$ is at most ϵ', since an $\overline{S}$-blow-up of any $G' \in \Phi'$ is ϵ'-close to an $\overline{S}'$-blow-up of $G' \in \Phi'$ (and vice versa). The claim follows. ■

With these preliminaries is place, recall that we test R for property Φ' by invoking the corresponding tester of Π' and emulating the answers to the queries made to R by referring them to G. Specifically, if the emulated tester asks to sample s vertices of G', we sample $2s$ vertices in G, indentify s vertices in different clouds, provide them to the tester of Π', and answer the queries regarding G' by querying the corresponding pairs in G.

We warn that there is a potential problem with the foregoing outline. The problem is that we sample the vertices of R according to a distribution that is ϵ'-close to the uniform one, whereas the tester for Π' presumes that the vertices are sampled uniformly. This problem is addressed by considering the specific (adaptive) tester presented in the proof of Claim 3.1.3 (and modified in Section 4.1), and observing that this specific tester is not significantly affected by a small change in the distribution of the sampled vertices.[46] Actually, we need to slightly modify this tester, as follows:

- In the modified Step 6, the tester rejects if the number of matrix-collisions exceeds $O(1/\epsilon^2)$ and checks all matrix-collisions otherwise. (Recall that the original Step 6 does not check the number of matrix-collisions, but rather selects uniformly $O(1/\epsilon)$ *disjoint* matrix-collisions and checks them.)

  Note that if the vertices are sampled uniformly, then, with high probability, the number of matrix-collisions is $O(1/\epsilon^2)$, since the number of row-collisions (resp., column-collisions) is

[46]Again, this feature does not hold in general. The point is that a sample of $\omega(1/\epsilon')$ elements drawn according to a distribution that is ϵ'-close to distribution X may be easy to distinguish from a sample drawn from X. Hence, the claim depends on what the tester does with the sampled vertices.

$O(1/\epsilon)$. This modification does not increase the complexity of the tester, which is dominated by other steps. (The reason for this modification is discussed at the end of the proof of the following claim.)

- We also slighly modify Step 2: While the original tester find reliable locator of size ℓ by selecting these sets at random among the relevant parts of the sample S (which are each of size $O(\ell)$), we find reliable locators of size $\ell - \epsilon_2\ell$ by searching all possible candidates subsets in S.

Note that these modifications have negligible effect when the vertices are sampled from the uniform distribution.

Claim 6.1.3 (robustness of the tester of Claim 3.1.3): *For a sufficiently small $\epsilon' = \Omega(\epsilon)$, the algorithm presented in the proof of Claim 3.1.3, modulo the foregoing modification, rejects with high probability graphs that are ϵ-far from Π' even if the vertices are sampled from a distribution that is ϵ'-close to the uniform distribution on the vertices of the tested graph. The same holds for the modified tester presented in Section 4.1.*

We stress that the modified algorithms accepts graph in Π', with high probability, when the vertices are sampled from the uniform distribution.

Proof: Recall that the sampled vertices are used to provide degree estimates (in Step 1), derive reliable locators (in Step 2), test isomorphism to a fixed graph via a fixed mapping (in Steps 4 and 5), and test equality betweeen matrices (in Step 6).[47]

The analysis of Step 1 (resp., Step 2) refers to the probability that a set of logarithmically many vertices, selected uniformly and independently in the vertex set, is a good degree estimator (resp., reliable detector). The probability of failure is $1/\text{poly}(n')$, which means that in both cases, for $\ell = O(\log n')$, the fraction of failed ℓ-tuples is $1/\text{poly}(n')$. Denoting each of these two failure sets by $B \subset [n']^\ell$, we consider the probability that ℓ vertices drawn from a distribution that is ϵ'-close to uniform yields an ℓ-tuple in B. We shall use two features that hold for these two specific sets (which capture failed events in Steps 1 and 2, respectively).[48]

1. The set B is closed under permutations of the sampled vertices; that is, for every $(v_1, ..., v_\ell) \in [n']^\ell$ and every permutation $\pi : [\ell] \to [\ell]$ it holds that $(v_1, ..., v_\ell) \in B$ if and only if $(v_{\pi(1)}, ..., v_{\pi(\ell)}) \in B$.

2. For some $\rho = \exp(-\Omega(\ell))$ and a sufficiently small constant $\epsilon_2 > 0$, the set B is "locally ρ-sparse" in the sense that for every $(v_1, ..., v_{\epsilon_2\ell}) \in [n']^{\epsilon_2\ell}$ it holds that
$$|\{(v_{\epsilon_2\ell+1}, ..., v_\ell) \in [n']^{\ell-\epsilon_2\ell} : (v_1, ..., v_\ell) \in B\}| \leq \rho \cdot (n')^{\ell-\epsilon_2\ell}.$$

[47] In the original description, two samples of different sizes are taken in Steps 1 and 3.

[48] Actually, in Step 1 degrees are estimated based on 9ℓ random vertices. More importantly, the estimates are all good enough (w.v.h.p.), even if ϵ_2 fraction of the vertices are selected adversarially (and only the rest are uniformly distributed). Likewise, for Step 2, we consider the event in which the set of low degree vertices contains an $(\ell - \epsilon_2\ell)$-set that is a reliable locator.

Using these two features[49], we prove that, for sufficiently small constant $\epsilon_0 > 0$, it holds that ℓ samples drawn from any distribution X that is ϵ_0-close to uniform yields an ℓ-tuple in B with probability at most $\exp(-\Omega(\ell))$. Towards this end, we fix some constant $\epsilon_1 > 0$ and call $x \in [n']$ heavy if $\Pr[X = x] > (1+\epsilon_1)/n'$. Then, X is heavy with probability at most ϵ_0/ϵ_1, and assuming that $\epsilon_0/\epsilon_1 = \epsilon_2/t$ it follows that the probability that ℓ samples of X contain more than $\epsilon_2\ell$ heavy samples is at most

$$\binom{\ell}{\epsilon_2\ell+1} \cdot (\epsilon_0/\epsilon_1)^{\epsilon_2\ell+1} \;=\; 2^{H_2(\epsilon_2+o(1))\ell} \cdot (\epsilon_2/t)^{\epsilon_2\ell+1} \;<\; O(1/t)^{\Omega(\epsilon_2\ell)}.$$

where H_2 is the binary entropy function (and so $H_2(\epsilon) < \epsilon\log_2(1/\epsilon) + O(\epsilon)$). On the other hand, if this bad event does not occur, then the ℓ-tuple resides in B with probability at most

$$\binom{\ell}{\epsilon_2\ell} \cdot (1+\epsilon_1)^{\ell-\epsilon_2\ell} \cdot \rho \;=\; \exp(H_2(\epsilon_2)\ell + \epsilon_1\ell - \Omega(\ell)),$$

where the first factor represents a choice of the heavy samples, the second factor represents the probability of selecting a specific $(\ell-\epsilon_2\ell)$-tuple of non-heavy vertices, and the third factor represents the density of B. Hence, for a sufficiently small $\epsilon_1, \epsilon_2 > 0$ and a sufficiently large constant t, the claim follows (i.e., an ℓ-sample drawn from X hits B with probability at most $\exp(-\Omega(\ell))$), where this requires setting $\epsilon_0 = \epsilon_1\epsilon_2/t > 0$ (and using $\epsilon' \le \epsilon_0$).

For the other tests (performed in Steps 4–6), the issue at hand is hitting sets of vertex-pairs (resp., pairs of vertex-pairs) that have density $\Omega(\epsilon)$ (resp., density $\Omega(\epsilon/n')$). These tests are analyzed based on laws of large numbers, which presumes that the events in question are pairwise independent. Hence, the analysis actually refers to 4-tuples (resp., 8-tuples) of vertices, and so a deviation of ϵ' in the sampling of single vertices translates to a deviation of $O(\epsilon')$ in the distribution of the relevant tuples. Actually, the analysis of Step 6 requites more care, and is outlined next.

Recall that the claim refers to a modification of the algorithm presented in of Claim 3.1.3. Specifically, in Step 6, rather than checking $O(1/\epsilon)$ random matrix-collisions, we check all of them, provided that their number does not exceed $O(1/\epsilon^2)$. The need for this modification arises from the fact that, in the current setting, the matrix-collisions are not necessarily distributed uniformly among the matrix entries. This means that the distribution of collisions may be strongly slanted towards vertices that appear with higher probability. Still, at least $1 - 2\epsilon'$ of the distribution is assigned to vertices that appear with probability approximately $1/n'$, and such vertices are likely to form $\Omega(1/\epsilon)$ matrix-collisions that are almost uniformly distributed among the matrix's entries. Hence, by checking all matrix-collisions, we benefit from the matrix-collisions that are almost uniformly distributed (even in the case that there are more matrix-collisions that are far from being uniformly distributed). ■

Combining Claims 6.1.2 and 6.1.3, we conclude that the (two-step adaptive) tester rejects graphs that are ϵ-far from Π with high probability. Recall that the modified Step 6 is unlikely to reject graphs in Π (since it rejects if the number of matrix-collisions exceeds $O(1/\epsilon^2)$), and it follows that the (two-step adaptive) tester accepts graphs in Π with high probability. The same considerations apply to the non-adaptive version of the foregoing tester, where in this version Step 6 is modified

[49]In contrast, the claim does not hold when the second feature does not hold. Consider for example a set B that consist of all ℓ-tuples that contain at least c occurances of the value 1, and a distribution X that assigns 1 probability ϵ' and is uniform over $\{2, ..., n'\}$ otherwise. Then, $|B| < \binom{\ell}{c} \cdot (n')^{\ell-c} < |[n']|^{\ell}/\text{poly}(n')$, but (for any constants $\epsilon' > 0$ and c) with very high probability ℓ samples of X form an ℓ-tuple in B.

anyhow (see the end of the proof of Theorem 4.1). Hence, the claimed upper bounds follow, where the crucial fact is that the query complexity of local self-ordering is $O(\log n')$ rather than $O(\log n)$, whereas the complexity of testing whether R' is in Π' refers to the size of R (which is an n'-vertex graph). Specifically, note that the first test (i.e., the distribution test) makes $O(\epsilon^{-2}\sqrt{n'} \cdot \log n')$ queries, whereas the complexity of the second test is as stated in Theorem 1.3 (with n replaced by n'). ■

Digest. The proof of Theorem 6.1 is based on using an n/n'-factor blow-up of the n'-vertex graphs used in Theorem 1.3 (with n replaced by n'). The analysis of the resulting property is based on the (highly non-trivial) fact that *the blow-up operation preseves distances* (see [17, Sec. 4] and [11, Lem. 4.3]) and on *specific features of the property used in Theorem 1.3*. Specifically, we used both features of the property itself and of the testers used to establish the complexity upper bounds.

We highlight the fact that the proof of Claim 6.1.3 contains a proof of the following result: *If X is close to the uniform distribution on $[m]$ and $B \subseteq [m]^\ell$ has density at most* $\exp(-\Omega(\ell))$ *and satisfies some additional conditions, then the probability that ℓ samples drawn from X form an ℓ-tuple in B is* $\exp(-\Omega(\ell))$. We stress that the claim does not hold in general, even when closeness means variation distance $o(1)$, but it does hold under some natural conditions. Specifically, a sufficient condition is that, *for some constant $\epsilon_2 > 0$, the residual set obtained from B by fixing $\epsilon_2\ell$ of the coordinates has density* $\exp(-\Omega(\ell))$.

6.2 Translation to complexities that also depend on the proximity parameter

Here we use the methodology of [7], which transforms existential results regarding graph properties of almost arbitrary size-dependent query complexity to results that support almost any query complexity; that is, the latter complexities that may depend both on the size and on the proximity parameter. One interesting aspect in this transformation is that it yields results also for complexities that depend only on the proximity parameter. Using this methodology, we transform Theorem 6.1 to Theorem 1.7, which asserts rather tight complexity bounds that *may also depend on the proximity parameter.*

Theorem 6.2 (Theorem 1.7, restated)[50] *For every functions $f, g : \mathbb{N} \times (0,1] \to \mathbb{N}$ that are monotonically non-decreasing in the first parameter and monotonically non-increasing in the second parameter and every $n \in \mathbb{N}$ and $\epsilon > 0$ such that $f(n,\epsilon) \in [O(1), \sqrt{\epsilon n}]$ and $g(m,\epsilon) \le m$, there exists a graph property Π that satisfies the following two conditions:*

1. *There exists a general* (i.e., adaptive) *tester of* Π *that makes $O(\epsilon^{-2} \cdot f(n,\Omega(\epsilon)) \cdot \log f(n,\Omega(\epsilon)))$ queries, and any such tester must make $\Omega(f(n,O(\epsilon)))$ queries.*
2. *Any non-adaptive tester must make $\Omega(g(f(n,O(\epsilon))) \cdot f(n,O(\epsilon)))$ queries, and there exists such a tester that makes $O(\epsilon^{-2} \cdot g(f(n,\Omega(\epsilon)),\Omega(\epsilon)) \cdot f(n,\Omega(\epsilon)) \cdot \log f(n,\Omega(\epsilon)))$ queries.*

The proof adapts the strategy used in [7] to the current context. We apply this strategy to different properties, and care both about the complexity of general and non-adaptive testing, but no new ideas are used here. Still, for the sake of self-containment as well as greater clarity, we provide the following proof sketch.

[50] As in Footnote 34, the adaptive upper bound presumes $f(n,\epsilon)/\epsilon^2 = o(n)$; otherwise we incur another $O(\log f(n,\epsilon))$ factor.

Proof Sketch: The issue at hand is the difference between results regarding testing with a fixed proximity parameter $\epsilon_0 = \Omega(1)$, hereafter referred to as $\Omega(1)$-testing, and testing with varying proximity parameter ϵ, hereafter referred to as ϵ-testing. The *lower bounds* of Theorem 6.1 refer only to testing with a fixed proximity parameter, although the upper bounds extend to ϵ-testing. Our goal is to provide relatively tight lower and upper bounds on the (general and non-adaptive) query complexities of ϵ-testing, where the complexities may be arbitrary function os ϵ (as well as of the size of the graph). We start with a rough outline of our proof strategy.[51]

Our starting point is Theorem 6.1, which we re-interpret in concrete (rather than functional) terms. Essentially, for some universal constant $c > 5$, and for every fixed natural numbers q', q'', n' such that $\sqrt{2c} < q' < q'' < q'^2 < n'$, Theorem 6.1 asserts the existence of a property $\Pi'_{n',q',q''}$ of n'-vertex graphs for which the query complexity of general $1/c$-testing is between q' and $\widetilde{O}(q')$, and the analogous non-adaptive complexity is between q'' and $\widetilde{O}(q'')$. Furthermore, the upper bound extends to $O(1/\epsilon^2) \cdot \widetilde{O}(q')$ (resp., $O(1/\epsilon^2) \cdot \widetilde{O}(q'')$), when testing with proximity parameter ϵ.

Now, for any $\epsilon > 0$, suppose that we want to present a property of n-vertex graphs such that ϵ-testing it has (general) complexity between q' and $\text{poly}(1/\epsilon) \cdot \widetilde{O}(q')$ and non-adaptive complexity between q'' and $\text{poly}(1/\epsilon) \cdot \widetilde{O}(q'')$. Then, we set $n' = \sqrt{c\epsilon} \cdot n$ and define a property of n-vertex graphs, denoted Π, such that each graph in Π consist of a "base" graph from $\Pi'_{n',q',q''}$ and a clique of size $n - n'$. The $(n - n')$-vertex clique is viewed as a padding of the base graph, and the vertex pairs in the base graph occupies an $(n'/n)^2 = c \cdot \epsilon$ fraction of $[n]^2$.

The lower bounds on the complexities of ϵ-testing Π are derived by inference from the lower bounds on $1/c$-testing $\Pi'_{n',q',q''}$. That is, we reduce $1/c$-testing $\Pi'_{n',q',q''}$ to ϵ-testing Π, which establishes the desired lower bounds (of q' and q'', resp). The upper bounds on the complexities of ϵ-testing Π are obtained by testing that the base graph has size n' (and that it is padded by an $(n - n')$-vertex clique), and $1/c$-testing that the base graph is in $\Pi'_{n',q',q''}$, (using the adequate $1/c$-tester for $\Pi'_{n',q',q''}$). In particular, sampling a vertex in the base graph is emulated by sampling $O(n/n') = O(1/\sqrt{\epsilon})$ vertices of the tested graph.

The foregoing construction is tailored for a fixed value of the proximity parameter (i.e., Π depends on ϵ (via the definition of $n' = \sqrt{c\epsilon} \cdot n$)), whereas we seek properties that exhibit the designated complexity for any value of the proximity parameter. This is achieved by creating properties that are the union of properties defined as above for a geometric sequence of values of the proximity parameter. Specifically, when seeking to establish general and non-adaptive query complexity $q : \mathbb{N} \times (0, 1] \to \mathbb{N}$ and $Q : \mathbb{N} \times (0, 1] \to \mathbb{N}$, we use the base properties $\Pi'_{c^{-i}n,q(n,c^{-2i-1}),Q(n,c^{-2i-1})}$, for $i = 1, ..., \log_c n$. That is, we take the union of these properties after padding each of them to size n.

Establishing the lower and upper bounds in this case requires more care. Specifically, for the lower bounds on the complexities of ϵ-testing, we show that the properties introduced for handling the other values of the proximity parameter (i.e., the c^{-i}'s for all $i \in [\log_c n] \setminus \{0.5 \cdot \log_c(1/\epsilon)\}$) do not interfere with the argument that refers to the value of the proximity parameter that is of interest to us (i.e., ϵ). For the upper bounds on the complexities of ϵ-testing, we present testers that first determine the size of the base graph that seems to underlie the tested graph (rejecting if none fits). Next, our testers emulate testing the base graph (with suitable proximity), using the size parameter n' determined in the first step. (The fact that we only obtain a (rough) approximation of the size of the base graph raises some difficulties, but they are resolved just as in [7, Sec. 4]. In particular, we rely on the fact that the testers used in the proof of Theorem 6.1 use the size parameter only in order to sample vertices in the tested graph.)

[51]The following outline is adapted (with small changes) from [7, Sec. 1.2].

The actual proof. The ballpark of the general and non-adaptive query complexities are set to $q(n,\epsilon) = f(n,\epsilon)$ and $Q(n,\epsilon) = g(f(n,\epsilon),\epsilon) \cdot f(n,\epsilon)$, respectively. Following the foregoing outline, we let $\Pi_n = \bigcup_{i\in[\log_c n]} \Pi_n^{(i)}$ such that an n-vertex graph is in $\Pi_n^{(i)}$ if it consists of a $(n-n_i)$-vertex clique, where $n_i = c^{-i} \cdot n$, and a graph in $\Pi'_{n_i,q(n,c^{-2i-1}),Q(n,c^{-2i-1})}$, which is called the base graph. Note that each graph in $\Pi_n^{(i)}$ is c^{-2i}-close to a graph that consists of an $(1-c^{-i}) \cdot n$-vertex clique and $c^{-i} \cdot n$ isolated vertices.

The proofs of the following two claims are obtained by modification of corresponding proofs that appear in [7], and additional details can be found there.[52]

Claim 6.2.1 (upper bounds): *The property Π_n is ϵ-testable in $O(\epsilon^{-2} \cdot q(n,\epsilon/c) \cdot \log q(n,\epsilon/c))$ queries. It is also ϵ-testable by $O(\epsilon^{-2} \cdot Q(n,\epsilon/c) \cdot \log Q(n,\epsilon/c))$ non-adaptive queries.*

Proof: We first approximate the number of "low degree" vertices (i.e., vertices of degree at most n/c) up to an additive deviation of $0.1\epsilon \cdot n$. If the number of these vertices is approximately $c^{-i^*} \cdot n$, then we invoke a $(c^{2i^*} \cdot \epsilon/2)$-tester for $\bigcup_{n'\in\mathbb{N}} \Pi'_{n',q(n,c^{-2i^*}),Q(n,c^{-2i^*})}$ on the subgraph induced by the "low degree" vertices. Here we use the fact that the tester for $\Pi'_{n_i,q(n,c^{-2i^*-1}),Q(n,c^{-2i^*-1})}$ actually works for $\bigcup_{n'\in\mathbb{N}} \Pi'_{n',q(n,c^{-2i^*-1}),Q(n,c^{-2i i^*-1})}$. That is, we test the property by pretending that the tested graph has n_i vertices, whereas in reality it may have $n_i \pm 0.1\epsilon n$ vertices. For sake of clarity, we spell out the derived tester (as operating on input an n-vertex graph G and proximity parameter ϵ, while letting $\ell = \lceil 0.5 \log_c(1/\epsilon) \rceil$).

1. *Determining i^**: The tester finds a vertex, denoted s, that seems to have degree at least $n/2$, by sampling $O(1)$ vertices and roughly approximating their degree (by considering their adjacency relation to $O(1)$ random vertices).

   Letting B denote the set of vertices that do not neighbor s in G, the tester obtains a very rough estimate of the size of B; specifically, using a sample of $O(1/\epsilon^2)$ vertices, with high probability, the deviation error is smaller than $0.1\epsilon \cdot n$. If the estimated size of B is smaller than $1.1 \cdot c^{-(\ell+1)} \cdot n$, then the tester sets $i^* = \ell + 1$, which corresponds to the case that the base graph has less than $\frac{\epsilon}{2} \cdot n^2$ vertex pairs. If, for some $i^* \in [\ell]$, the number of low-degree vertices is $(1 \pm 0.1) \cdot c^{-i^*} \cdot n$, then i^* is set accordingly, which corresponds to a base graph of size $c^{-i^*} \cdot n$; otherwise (i.e., if i^* was not set), the tester *rejects*.

2. *Testing whether the subgraph induced by $[n] \setminus B$ is a clique*: The tester selects $O(1/\epsilon)$ random vertices in G and checks that the vertices that neighbor s are adjacent to one another. If two vertices that neighbor s were found not to be adjacent, then the tester *rejects*.

3. *Testing the subgraph induced by B*: If $i^* = \ell + 1$, then the tester *accepts*. Otherwise, the tester invokes the $(c^{2i^*} \cdot \epsilon/2)$-tester for $\bigcup_{n'\in\mathbb{N}} \Pi'_{n',q(n,c^{-2i^*-1}),Q(n,c^{-2i^*-1})}$ on the subgraph of G induced by B, and *outputs the verdict of this tester*. In particular, we invoke the general tester

[52]Specifically, the general strategy is first presented in the context of testing Boolean functions [7, Sec. 2], and then adapted to testing graph properties, first in the bounded degree graph model and then in the dense graph model (see [7, Sec. 3] and [7, Sec. 4], respectively). This is somewhat unfortunate for us, since our focus is on the dense graph model. Furthermore, the main presentation (i.e., [7, Sec. 2–4]) refers to complexity bounds that only depend on the proximity parameter, and it is extended to general complexity bounds in [7, Sec. 5]. Here, we incorporate all the relevant modifications. Furthermore, the original presentation refers to general testers, but we extend it also for non-adaptive testers.

of the proof of Theorem 6.1 and emulate each vertex selection (for the induced subgraph) by sampling $O(c^{i^*})$ vertices of G (in expectation).

The complexity of this emulation is

$$c^{i^*} \cdot O((c^{2i^*} \cdot \epsilon/2)^{-2} \cdot q(n, c^{-2i^*-1}) \cdot \log q(n, c^{-2i^*-1})) = O(\epsilon^{-2} \cdot q(n, \epsilon/c) \cdot \log q(n, \epsilon/c)).$$

The query complexity of the purported ϵ-tester is upper-bounded by $O(1/\epsilon^2)+O(\epsilon^{-2}q(n,\epsilon/c)\log q(n,\epsilon/c))$. A corresponding non-adaptive tester is obtained by observing that Steps 1 and 2 can be made non-adaptive by performing them for all $O(1)$ sampled vertices (rather than for s only), and Step 3 can be made non-adaptive by performing it for all possible $i^* \in [\ell]$ while using the corresponding non-adaptive tester of Theorem 6.1.

In analyzing the foregoing testers, we may assume that s has degree at least $n/2$, and that we obtained a good approximation of the size of B. Hence, if $G \in \Pi_n$, then Step 2 will never reject, and Step 3 will accept with high probability. If G is ϵ-far from Π and Steps 1-2 do not reject, then G is close to consisting of an $(n - n_{i^*})$-vertex clique and some n_{i^*}-vertex graph that is $c^{2i^*} \cdot \epsilon/2$-far from $\Pi'_{n_{i^*},q(n,c^{-2i^*-1}),Q(n,c^{-2i^*-1})}$. In this case, Step 3 rejects with high probability. ■

Claim 6.2.2 (lower bounds): *The property Π_n is not ϵ-testable in $o(q(n, c^2 \cdot \epsilon))$ queries. Ditto for $o(Q(n, c^2 \cdot \epsilon))$ non-adaptive queries.*

Proof: For any $\epsilon > 0$, letting $i = \lfloor 0.5 \log_c(1/c\epsilon) \rfloor$ and $n' = c^{-i}n$, we reduce $1/c$-testing the property $\Pi'_{n',q(n,c^{-2i-1}),Q(n,c^{-2i-1})}$ to ϵ-testing Π_n by emulating answers to the queries issued by the latter tester, which means that we emulate an n-vertex graph using queries to an n'-vertex graph. Specifically, when $1/c$-testing the n'-vertex graph G' we invoke the ϵ-tester on an imaginary n-vertex graph G that consists of G' and an $(n-n')$-vertex clique. Indeed, we can place the vertices of G' at fixed locations of our choice (e.g., we may just assign these vertices labels in $[c^{-i} \cdot n]$).

The crucial fact is that the vertices of the base graph have low degree, whereas the vertices of the clique have high degree. This fact is used to prove the key observation that asserts that, for any $j \in [1 + \log_c n] \setminus \{i\}$, graphs in $\Pi_n^{(i)}$ are relatively far from graphs in $\Pi_n^{(j)}$; that is, the relative distance is at least $|c^{-2i} - c^{-2j}|/2 > \epsilon$. The latter fact implies that if G' is $1/c$-far from $\Pi'_{n',q(n,c^{-2i-1}),Q(n,c^{-2i-1})}$, then G is ϵ-far from Π_n rather than only being ϵ-far from $\Pi_n^{(i)}$. On the other hand, $G' \in \Pi'_{n',q(n,c^{-2i-1}),Q(n,c^{-2i-1})}$ implies $G \in \Pi_n$. ■

Conclusion. Combining Claims 6.2.1 and 6.2.2, and using $q(n, e) = f(n, \epsilon)$ and $Q(n, \epsilon) = g(f(n, \epsilon), \epsilon) \cdot f(n, \epsilon)$, we infer that the query complexity of ϵ-testing Π_n is between $\Omega(f(n, c^2 \cdot \epsilon))$ and $O(\epsilon^{-2} \cdot f(n, \epsilon/c) \cdot \log f(n, \epsilon/c))$, whereas the corresponding non-adaptive complexity is between $\Omega(g(n, f(n, c^2 \cdot \epsilon)) \cdot f(n, c^2 \cdot \epsilon))$ and $O(\epsilon^2 \cdot g(n, f(n, \epsilon/c)) \cdot f(n, \epsilon/c) \cdot \log f(n, \epsilon/c))$, where we use $g(m, \cdot) \leq m$. The theorem follows. ■

7 Open Problems

While this paper provides a nearly-tight understanding of the query gap between adaptive and non-adaprtive testing o f dense graph properties, it does leave some open problems. In particular:

1. Our separation results assert the mere *existence* of graph properties that exhibit gaps in their testing complexity. This raises the question of whether such gaps can be proved for properties that are efficiently recognizable. Furthermore, one may seek efficient testers for such properties; that is, testers whose time complexity is closely related to their query complexity.

   For Theorems 1.2–1.4 and 1.6–1.7, this challenge was meet in a subsequent work [9] (see [9, Cor. 1.9] and a subsequent comment), but it remains open for Theorem 1.5.

2. Regarding Theorem 1.5, which addresses one-sided error testers, it would be nice to have such testers in the parameter regime of Theorems 1.6 and 1.7.

3. Our separation results are not meaningful for properties that can be tested within (general) query complexity $q(\epsilon) \leq \widetilde{O}(1/\epsilon^2)$, since in this case Theorem 1.6 only yields a non-adaptive lower bound of $\Omega((\epsilon^2 \cdot q(\epsilon))^2)$. Hence, the challenge is to establish a non-adaptive lower bound of $\Omega(q(\epsilon)^2)$ for such cases (i.e., for any $q(\epsilon) \in [\widetilde{O}(1/\epsilon), \text{poly}(1/\epsilon)]$).

   Recall that [12, Thm. 1.2] establishes a non-adaptive lower bound of $\Omega(q(\epsilon)^{3/2})$ on testing a (natural) graph property that has a general tester of complexity $q(\epsilon) = \widetilde{O}(1/\epsilon)$.

4. Turning to the high end of the the query complexity, we note that our separation results only apply to properties that can be tested within (general) query complexity $q(n) \leq \widetilde{O}(n^{1/2})$. The challenge here is to establish a non-adaptive lower bound of $\Omega(q(n)^2)$ for any $q(n) \in [\omega(n^{1/2}), n]$.

5. Lastly, we mention the notion of *rounds of adaptivity*, which was studied in [3] when referring to property testers. A k-round tester makes its queries in k rounds such that all queries made in the i^{th} round depend only on the tester's explicit input, its randomness, and the answers it has obtained to the queries made in the prior $i-1$ rounds. Note that non-adaptive testers use a single round, whereas the adaptive testers presented in our proofs use two rounds. The begging question is what is the relation between the complexity of k-round testers versus $(k+1)$-round testers, for any $k > 1$.

Our results rely on graphs that are both robustly self-ordered and have local self-ordering procedures. As we show, random graphs happen to satisfy these two features, but one may ask whether one feature implies the other. This question is the focus of [9], where it is considered both for the dense graph model (as here) and for the bounded-degree graph model.

References

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

[2] 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.

[3] C. Canonne and T. Gur. An Adaptivity Hierarchy Theorem for Property Testing *ECCC*, TR17-029, 2017.

[4] E. Chattopadhyay. Guest Column: A Recipe for Constructing Two-Source Extractors. *SIGACT News*, Vol. 51 (2), pages 38–57, 2020.

[5] E. Fischer. On the Strength of Comparisons in Property Testing. *Information Processing Letters*, Vol. 189 (1), pages 107–116, 2004.

[6] O. Goldreich. *Introduction to Property Testing.* Cambridge University Press, 2017.

[7] O. Goldreich. Hierarchy Theorems for Testing Properties in Size-Oblivious Query Complexity. *Computational Complexity*, Vol. 28 (4), pages 709–747, 2019.

[8] O. Goldreich. On Testing Asymmetry in the Bounded Degree Graph Model. *ECCC*, TR20-118, 2020.

[9] O. Goldreich. Robust Self-Ordering versus Local Self-Ordering. *ECCC*, TR21-034, 2021.

[10] 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. Extended abstract in *37th FOCS*, 1996.

[11] O. Goldreich, M. Krivelevich, I. Newman, and E. Rozenberg. Hierarchy Theorems for Property Testing. *Computational Complexity*, Vol. 21(1), pages 129–192, 2012.

[12] O. Goldreich and D. Ron. Algorithmic Aspects of Property Testing in the Dense Graphs Model. *SIAM Journal on Computing*, Vol. 40, No. 2, pages 376–445, 2011.

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

[14] O. Goldreich and A. Wigderson. Robustly Self-Ordered Graphs: Constructions and Applications to Property Testing. *ECCC*, TR20-149, 2020. (Note: Numbered items refers to the revised version.)

[15] M. Gonen and D. Ron. On the Benefit of Adaptivity in Property Testing of Dense Graphs. *Algorithmica*, Vol. 58 (4), pages 811–830, 2010.

[16] J.H. Kim, B. Sudakov, and V.H. Vu. On the asymmetry of random regular graphs and random graphs. *Random Structures & Algorithms*, Vol. 21 (3-4), pages 216–224, 2002.

[17] O. Pikhurko. An Analytic Approach to Stability. *Discrete Mathematics*, Vol. 310 (21), pages 2951–2964, 2010.

[18] S. Raskhodnikova and A. Smith. A Note on Adaptivity in Testing Properties of Bounded Degree Graphs. *ECCC*, TR06-089, 2006.

[19] R. Shaltiel. An Introduction to Randomness Extractors. In *38th ICALP*, Part II, Lecture Notes in Computer Science (Vol. 6756), pages 21–41, Springer, 2011.

Appendix A: Proof of Theorem 2.3

For each (non-trivial) permutation $\mu : [n] \to [n]$, letting $T \overset{\rm def}{=} \{i \in [n] : \mu(i) \neq i\}$ denote its (non-empty) set of non-fixed-points, we show that, with probability $1 - \exp(-\Omega(n \cdot |T|))$, the size of the symmetric different between a random n-vertex graph $G_n = ([n], E_n)$ and $\mu(G_n)$ is $\Omega(n \cdot |T|)$.

For every $u,v \in [n]$ such that $u < v$, let $\chi_{u,v} = \chi^{\mu}_{u,v}(G_n)$ represent the event that *the pair* $(\mu(u),\mu(v))$ *contributes to the symmetric difference between* G_n *and* $\mu(G_n)$; that is, $\chi_{u,v} = 1$ if exactly one of the edges $\{\mu(u),\mu(v)\}$ and $\{u,v\}$ is in G_n, since $\{u,v\}$ is an edge of G_n if and only if $\{\mu(u),\mu(v)\}$ is an edge of $\mu(G_n)$. We shall prove that

$$\mathrm{Pr}_{G_n}\left[\sum_{u<v\in[n]} \chi^{\mu}_{u,v}(G_n) < \frac{n\cdot|T|}{20}\right] = \exp(-\Omega(n\cdot|T|)). \tag{9}$$

We prove Eq. (9) by using a $\lceil |T|/3\rceil$-subset $I \subseteq T$ such that $I\cap\mu(I)=\emptyset$. Let $T' = T\setminus(I\cup\mu^{-1}(I))$, which implies $T'\cap I=\emptyset$ and $\mu(T')\cap I=\emptyset$. Let $J = ([n]\setminus T)\cup T'$, and note that $|J| = n-|T|+(|T|-2\cdot\lceil|T|/3\rceil) \geq n-(2|T|/3)-2 \geq (n/3)-2$. Observe that, for every $(u,v)\in J\times I$, it holds that $u\neq v$ and $\Pr[\chi_{u,v}=1] = 1/2$, where the equality is due to $\{u,v\}\neq\{\mu(u),\mu(v)\}$, which holds since $(u,v)\in J\times I$ but $\mu(u),\mu(v)\in[n]\setminus I$. Furthermore, the events the correspond to the pairs in $J\times I$ are independent, because the sets $\{\{u,v\}:(u,v)\in J\times I\}$ and $\{\{\mu(u),\mu(v)\}:(u,v)\in J\times I\}$ are disjoint; that is, $(u,v)\in J\times I$ implies $(\mu(u),\mu(v))\in([n]\setminus I)\times([n]\setminus I)$. Hence (using $n\leq 3(|J|+2)$ and $|T|\leq 3|I|$ (as well as $3(|J|+2)\cdot 3|I| < 9.9\cdot|J|\cdot|I|$)), the l.h.s. of Eq. (9) is upper-bounded by

$$\begin{aligned}\mathrm{Pr}_{G_n}\left[\sum_{(u,v)\in J\times I}\chi^{\mu}_{u,v}(G_n) < \frac{3(|J|+2)\cdot 3|I|}{20}\right] &\leq \mathrm{Pr}_{G_n}\left[\sum_{(u,v)\in J\times I}\chi^{\mu}_{u,v}(G_n) < \frac{0.99\cdot|J|\cdot|I|}{2}\right]\\ &= \exp(-\Omega(|J|\cdot|I|))\end{aligned}$$

which is $\exp(-\Omega(n\cdot|T|))$. Having established Eq. (9), the claim follows by a union bound (over all non-trivial permutations $\mu:[n]\to[n]$); specifically, denoting the set of non-trivial permutations by P_n, we upper-bound the probability that G_n is not 0.05-robust by

$$\begin{aligned}&\sum_{\mu\in P_n}\mathrm{Pr}_{G_n}[\mu \text{ violates the condition in Eq. (9)}]\\ &\leq \sum_{t\in[n]}\binom{n}{t}\cdot(t!)\cdot\exp(-\Omega(n\cdot t))\\ &< n\cdot\max_{t\in[n]}\{n^t\cdot\exp(-\Omega(n\cdot t))\}\\ &= \exp(-\Omega(n))\end{aligned}$$

where t represents the size of the set of non-fixed-points (w.r.t μ). ■

Appendix B: Proof of Claim 2.6.1

We start with the main claim. Recall that G_n denotes a uniformly distributed n-vertex graph, S is a fixed ℓ-subset of $[n]$, and our goal is to upper-bound the probability that the subgraph of G_n induced by S is isomorphic to the subgraph induced by some other subset. Hence, we let $S'\neq S$ be an arbitrary ℓ-subset of $[n]$, and upper-bound the probability that the subgraphs of G_n induced by S and S' are isomorphic.

The case of $S'\cap S=\emptyset$ is easy, because in this case the we may fix the subgraph of G_n induced by S', whereas a random ℓ-vertex graph (i.e., the subgraph of G_n induced by S) is isomorphic to this fixed graph with probability at most $(\ell!) \cdot 2^{-\binom{\ell}{2}} \ll \binom{n}{\ell}^{-1}$, where the inequality uses a sufficiently large $\ell = O(\log n)$. Hence, we can afford to take a union bound over all ℓ-subsets that are disjoint of S. However, for sets that are not disjoint of S, the foregoing probability bound does not hold, and a more careful analysis is called for. Nevertheless, the foregoing analysis does provide a good warm-up towards the rest.

Turning to the general case, for each ℓ-set $S' \subset [n]$ such that $S' \neq S$, we shall upper-bound the probability that the subgraphs of G_n induced by S and S' are isomorphic as a function of $|S \cap S'|$. For every bijection $\pi : S \to S'$, let $\texttt{FP}(\pi) \stackrel{\rm def}{=} \{v \in S : \pi(v) = v\}$ denote the set of fixed-points of π, and note that $|\texttt{FP}(\pi)| \leq \ell - 1$ (since $S \neq S'$). Now, let $G = G_n$ denote the random n-vertex graph and G_R denote the subgraph of G induced by R. Then, we claim that the probability that there exists a bijection $\pi : S \to S'$ such that $\pi(G_S) = G_{S'}$ is upper-bounded by

$$\sum_{\pi:S \xrightarrow{1\text{-}1} S'} \min\left(2^{-|\texttt{FP}(\pi)|\cdot(\ell-|\texttt{FP}(\pi)|)/3}, 2^{-\binom{(\ell-|\texttt{FP}(\pi)|)/3}{2}}\right) \tag{10}$$

$$\leq \sum_{f\in\{0,\dots,|S\cap S'|\}} \frac{\ell!}{f!} \cdot 2^{-\max(6\cdot f\cdot(\ell-f),(\ell-f)\cdot(\ell-f-1))/18}$$

$$< \frac{\ell!}{|S\cap S'|!} \cdot 2^{-\Omega((\ell-|S\cap S'|)\cdot\ell)} \tag{11}$$

where f represents the size of $\texttt{FP}(\pi)$.

Justifying Eq. (10). To justify the upper bound claimed in Eq. (10), consider an arbitrary bijection $\pi : S \to S'$, and identify a set $I \subseteq S \setminus \texttt{FP}(\pi)$ such that $\pi(I) \cap I = \emptyset$ and $|I| \geq (\ell - |\texttt{FP}(\pi)|)/3$. Define random variables $\chi_{u,v}(\cdot)$ for $\{u, v\} \in \binom{[n]}{2}$ such that $\chi_{u,v}(G) = 1$ if $\{u, v\}$ is an edge in G and $\chi_{u,v}(G) = 0$ otherwise, and observe that $\pi(G_S) = G_{S'}$ if and only if $\chi_{\pi(u),\pi(v)}(\pi(G)) = \chi_{\pi(u),\pi(v)}(G)$ for every $\{u, v\} \in \binom{S}{2}$. Noting that $\chi_{\pi(u),\pi(v)}(\pi(G)) = \chi_{u,v}(G)$, the first bound in Eq. (10) is justified by

$$\begin{aligned}
&\Pr_G\left[\forall (u,v) \in \binom{S}{2} : \quad \chi_{\pi(u),\pi(v)}(\pi(G)) = \chi_{\pi(u),\pi(v)}(G)\right] \\
&\leq \Pr_G\left[\forall (u,v) \in \texttt{FP}(\pi) \times I : \quad \chi_{u,v}(G) = \chi_{\pi(u),\pi(v)}(G)\right] \\
&= \prod_{(u,v)\in\texttt{FP}(\pi)\times I} \Pr_G\left[\chi_{u,v}(G) = \chi_{u,\pi(v)}(G)\right] \\
&= 2^{-|\texttt{FP}(\pi)|\cdot|I|} \\
&\leq 2^{-|\texttt{FP}(\pi)|\cdot(\ell-|\texttt{FP}(\pi)|)/3}
\end{aligned}$$

where the equalities are due to the disjointness of the sets $\texttt{FP}(\pi) \times I$ and $\texttt{FP}(\pi) \times \pi(I)$ (to the fact that $\pi(u) = u$ for every $u \in \texttt{FP}(\pi)$), and to the fact that the different $\chi_{u,v}(G)$'s are independent and uniformly distributed in $\{0, 1\}$. Similarly, we justify the second bound in Eq. (10) by

$$\begin{aligned}
&\Pr_G\left[\forall \{u,v\} \in \binom{S}{2} : \quad \chi_{\pi(u),\pi(v)}(\pi(G)) = \chi_{\pi(u),\pi(v)}(G)\right] \\
&\leq \Pr_G\left[\forall \{u,v\} \in \binom{I}{2} : \quad \chi_{u,v}(G) = \chi_{\pi(u),\pi(v)}(G)\right]
\end{aligned}$$

$$
\begin{aligned}
&= \prod_{\{u,v\}\in\binom{I}{2}} \mathrm{Pr}_G\left[\chi_{u,v}(G)=\chi_{\pi(u),\pi(v)}(G)\right] \\
&= 2^{-\binom{|I|}{2}} \\
&\le 2^{-\binom{(\ell-|\mathtt{FP}(\pi)|)/3}{2}}
\end{aligned}
$$

where the equalities are due to the disjointness of the sets $\binom{I}{2}$ and $\binom{\pi(I)}{2}$, and to the fact that the different $\chi_{u,v}(G)$'s are independent and uniformly distributed in $\{0,1\}$.

Finishing the proof of the main claim. Combining Eq. (10)&(11) with a union bound over all ℓ-subsets $S'\subset[n]$ that are different from S, we upper-bound the probability that the subgraphs of G induced by S and by some other ℓ-set are isomorphic by

$$
\sum_{S'\in\binom{[n]}{\ell}\setminus\{S\}} \frac{\ell!}{|S\cap S'|!}\cdot 2^{-\Omega((\ell-|S\cap S'|)\cdot\ell)} = \sum_{i\in\{0,\dots,\ell-1\}} \binom{\ell}{i}\cdot\binom{n-i}{\ell-i}\cdot\frac{\ell!}{i!}\cdot 2^{-\Omega((\ell-i)\cdot\ell)} \qquad (12)
$$

where the index i represents the size of the intersection with S. Using a sufficiently large $\ell=O(\log n)$, we have

$$
\begin{aligned}
\sum_{i\in\{0,\dots,\ell-1\}} \binom{\ell}{i}\cdot\binom{n-i}{\ell-i}\cdot\frac{\ell!}{i!}\cdot 2^{-\Omega((\ell-i)\cdot\ell)} &= \sum_{i\in\{0,\dots,\ell-1\}} \binom{\ell}{i}^2\cdot\binom{n-i}{\ell-i}\cdot\frac{(n-i)!}{(n-\ell)!}\cdot 2^{-\Omega((\ell-i)\cdot\ell)} \\
&< \sum_{i\in\{0,\dots,\ell-1\}} n^{\ell-i}\cdot\binom{\ell}{i}^2\cdot 2^{-\Omega((\ell-i)\cdot\ell)} \\
&< \ell\cdot\max_{i\in\{0,\dots,\ell-1\}}\left\{n^{\ell-i}\cdot\binom{\ell}{i}^2\cdot 2^{-\Omega((\ell-i)\cdot\ell)}\right\} \\
&= \ell\cdot\left(n\cdot\ell^2\cdot 2^{-\Omega(\ell)}\right)
\end{aligned}
$$

and the main claim follows.

Proving the furthermore claim. The furthermore claim follows by observing that the probability that the subgraph of G induced by S is self-ordered is upper-bounded by an expression analogous to Eq. (10), where $S'=S$ and the difference is that the identity permutation is excluded from the sum. Hence, $|\mathtt{FP}(\pi)|<\ell$ still holds, and so do the justifications given to Eq. (10). ■