On the query complexity of testing local graph properties in the bounded-degree graph model

Oded Goldreich[*]

March 8, 2024

Abstract

We consider the query complexity of testing local graph properties in the bounded-degree graph model. A local property is defined in terms of forbidden subgraphs that are augmented by degree information, where the latter account also for neighbors that are not in the subgraph. Indeed, this formulation yields a generalized notion of subgraph-freeness, which extends the standard notions of induced and non-induced subgraph freeness.

While it is tempting to conjecture that every local graph property has a constant-query proximity-oblivious tester (in the bounded-degree graph testing model), this conjecture was refuted by Adler, Kohler and Peng (*32nd SODA* and *36th CCC*, 2021). In fact, they showed that there exist local graph properties that cannot be tested (in this model) within a number of queries that does not depend on the size of the graph. However, their proof gives no explicit lower bound on the dependence of the query complexity on the size of the graph.

In this paper, we provide such an explicit bound. This is done by studying the query complexity of the specific local graph property presented by Adler *et. al.*. In a natural (but not standard) model, in which the tester is not given the size of the graph but rather only a rough approximation of this size, this property has logarithmic (in the graph's size) query complexity. In the standard model, we only obtain a quadruple-logarithmic lower bound.

Keywords: Property Testing, Graph Properties, Expander graphs, Zig-Zag product.

[*]Faculty of Mathematics and Computer Science, Weizmann Institute of Science, Rehovot, Israel. Email: `oded.goldreich@weizmann.ac.il`. This research was partially supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No. 819702).

Contents

1 Introduction

The study of testing graph properties is one of the main sub-areas within property testing (see textbook [4]). Within this subarea, the bounded-degree graph model (introduced by Goldreich and Ron [8] and reviewed in [4, Chap. 9]) is one of the two main models (the other being the dense graph model, introduced by Goldreich, Goldwasser and Ron [7] and reviewed in [4, Chap. 8]).

Loosely speaking, in the bounded-degree graph model, for a fixed degree bound $b \in \mathbb{N}$, we represent n-vertex graphs by an incidence function of the form $g : [n] \times [b] \to [n] \cup \{\perp\}$ such that $g(v, i)$ is the i^{th} neighbor of v (and $g(v, i) = \perp$ if v has less than i neighbors). Distance between n-vertex graphs is defined as the fraction of entries on which their (mutually closest) incidence functions differ. The tester (for a fixed graph property) is given oracle access to such an incidence function and has to distinguish between the case that the graph has the property and the case that it is ϵ-far from the property, where $\epsilon > 0$ is called the proximity parameter.

Of special interest are testers that make a number of queries that does not depends on the size of the graph, but rather only on the proximity parameter. Such testers are said to have size-oblivious query complexity. A special case of testers of size-oblivious query complexity is that of (one-sided error) testers that repeat a basic constant-query test for a number of times that depends on the proximity parameter (and accept if and only if all invocations of the basic test accept). These basic tests are called proximity oblivious testers, and the probability that they reject a graph (a.k.a their "detection probability"') is a function of the distance of the graph from the property.

The study of proximity-oblivious testers for the bounded-degree graph model leads naturally to the notion of a *local graph property*. This notion was formulated by Goldreich and Ron [9] in terms of forbidden subgraphs that are augmented by degree information, where the latter account also for neighbors that are not in the subgraph. Indeed, this formulation yields a generalized notion of subgraph-freeness, which extends the standard notions of induced and non-induced subgraph freeness. Loosely speaking, for a finite set $\mathcal{F}$ marked graphs, where vertices are marked `full` or `undetermined`, we say that a graph G is $\mathcal{F}$-free if no graph $F \in \mathcal{F}$ can be embedded in G such that F is isomorphic to the induced subgraph of G and `full` vertices of F are mapped to vertices of G that have no neighbors in G outside this induced subgraph.

It was shown in [9] that only local graph properties have a proximity-oblivious tester (in the bounded-degree graph testing model), and it is tempting to conjecture that every local graph property has a proximity-oblivious tester. However, this conjecture was refuted by Adler, Kohler and Peng [1]. In fact, they showed that there exist local graph properties that cannot be tested (in the bounded-degree graph model) within a number of queries that does not depend on the size of the graph. However, their proof gives no explicit lower bound on the dependence of the query complexity on the size of the graph. This deficiency of Adler *et. al.* [1] is inherent, because they rely on an non-explicit result of [3].

In this paper, we provide such an explicit bound. This is done by studying the query complexity of the specific local graph property presented by Adler *et. al.* [1]. Actually, the property used by Adler *et. al.* [1] contains at most one unlabeled n-vertex graph for each $n \in \mathbb{N}$ (equiv., all n-vertex graphs that are in the property are isomorphic to one another). Specifically, these graphs are expanders (whereas a result of [3] implies that a property consisting solely of expanders cannot be tested within size-oblivious query complexity).

1.1 The starting point and the question we study

As stated above, our starting point is the construction of locally-characterizable expander graphs provided by Adler *et. al.* [1].

Theorem 1.1 (a locally-characterizable sequence of expander graphs [1]): *There exists a finite collection of marked graphs, denoted $\mathcal{F}$, such that the set of $\mathcal{F}$-free graphs is an infinite set of* (bounded-degree) *expander graphs. Furthermore, all n-vertex graphs that are $\mathcal{F}$-free are isomorphic to one another, and for some $d \in \mathbb{N}$ and each $m \in \mathbb{N}$ there exists an $\Theta(d^{4m})$-vertex $\mathcal{F}$-free graph.*

The fact that the set of $\mathcal{F}$-free n-vertex graphs are isomorphic to one another is quite striking, let alone the fact that they are expanders. In contrast, by a prior result of Fichtenberger, Peng, and Sohler [3], every infinite property of graphs that has a tester of size-oblivious query complexity must contain an infinite hyperfinite subproperty. The point is that hyperfinite graphs are the "extreme opposite" of expander graphs; hence, the property asserted in Theorem 1.1 does not have a tester of size-oblivious complexity.

We stress that the foregoing argument cannot possibly yield an explicit lower bound on the query complexity of a tester for the graph property asserted in Theorem 1.1. Obtaining such an explicit lower bound requires specifying a graph property that satisfies Theorem 1.1 and analyzing it with reference to its details (rather than only relying on the fact that it is not hyperfinite).

The graph constructed in [1]. The proof of Theorem 1.1 is pivoted at the Zig-Zag construction of Reingold, Vadhan, and Wigderson [10]. Recall that they presented a sequence of graphs, $(G_i)_{i\in\mathbb{N}}$, such that $G_1 = H^2$ for some constant-size expander H (of degree d and second-eigenvalue $d/4$) and $G_i = G_{i-1}^2 ⓩ H$, where ⓩ denotes the Zig-Zag product. Furthermore, each vertex in G_{i-1} is replaced in G_i by a cloud of vertices of the same size as H, and edges of G_{i-1}^2 yield "connections" between the corresponding clouds in G_i (where the edges of the Zig-Zag product correspond to three-step walks on a graph that combine these connections with copies of H that are "placed" on each cloud).

Loosely speaking, for each $m \in \mathbb{N}$, the proof of Theorem 1.1 (provided in [1]) identifies a graph that consists of the graphs $G_1, ..., G_m$ along with edges that connect each vertex of G_{i-1} to each vertex in the corresponding cloud of G_i. In fact, the construction (as surveyed in [6])[1] is first presented in terms of directed edge-colored graphs, and gadgets are later used to yield undirected graphs (with no colors). Let us sketch this (directed edge-colored) version in more detail.

Constuction 1.2 (the construction of [1], directed edge-colored version (as surveyed in [6])): *For $d \in \mathbb{N}$, let H be a d-regular d^4-vertex graph. For every $m \in \mathbb{N}$, letting $n = \sum_{i=0}^{m} d^{4i}$, we consider the following n-vertex directed graph, denoted $\overline{G}_m$, that is obtained by superimposing a full d^4-ary directed tree of height m and a copy of each of the G_i's such that a copy of G_i is placed on the vertices at level i* (i.e., the vertices at distance i from the root of the tree).

1. *For each $i \in [m]$ and each vertex v of level $i-1$, there are d^4 directed edges, colored $1, ..., d^4$, going from v to d^4 distinct vertices of level i. These edges are called the* tree edges, *and v is called the* parent *of these d^4 vertices.*

[1]The main presentation in [1] is in terms of first-order definable structures (of finite model theory).

2. *The vertices of level 1 are ordered according to the colors of their incoming edges, and a copy of H is superimposed on them. That is, the j^{th} vertex of level 1 is connected by a pair of anti-parallel edges to the k^{th} vertex of level 1 if and only if $\{j,k\}$ is an edge in H.* (These anti-parallel edges are colored according to the ports used for edge $\{j,k\}$ in H).[2]

3. *For every $i \in [m-1]$, the vertices of level $i+1$ are ordered according to the colors of their incoming tree edges and the order of their parent in the tree, and a copy of G_{i+1} is placed between them. Specifically, for each vertex v at level i, its children in level $i+1$ are denoted $(v,1), ..., (v,d^4)$, and a pair of anti-parallel edges connects vertices of this level* (i.e., (v,j) and (w,k)) *if and only if there is an edge between them in G_{i+1}.* (These anti-parallel edges are colored according to the ports used for this edge in G_{i+1}).

   (In other words, the vertices of $\overline{G}_m$ are associated with sequences of length at most m over $[d^4]$, such that the vertex $(\sigma_1, ..., \sigma_i, \sigma_{i+1})$ is the $\sigma_{i+1}^{\mathrm{th}}$ child of $(\sigma_1, ..., \sigma_i)$; that is, a tree-edge colored σ_{i+1} goes from $(\sigma_1, ..., \sigma_i)$ to $(\sigma_1, ..., \sigma_i, \sigma_{i+1})$).

Let $\mathcal{G}$ denote the set of all graphs that are isomorphic to one of the $\overline{G}_m$'s; that is, for every $n = \sum_{i=0}^{m} d^{4i}$, an n-vertex graph G is in $\mathcal{G}$ if and only if G is isomorphic to $\overline{G}_m$.

Indeed, for every $i \in [m-1]$, the tree edges identify individual vertices at level i (i.e., a vertex of G_i) with the d^4-vertex clouds associated with them at level $i+1$ (i.e., a cloud in G_{i+1}), and the vertices of that cloud are ordered according to the colors of the relevant tree edges. This ordering combined with the colors of the edges in the i^{th} level (which represent the ports of the corresponding edges of G_i) determines the edges in the next level (as well as their color). (This determination is based on the fact that $G_{i+1} = G_i^2 \textcircled{z} H$.)

The fact that the graph property $\mathcal{G}$ is locally characterizable is a key result of [1]; that is, it is shown that there exists a finite set of marked graphs $\mathcal{F}$ such that the set of $\mathcal{F}$-free graphs equals $\mathcal{G}$. The proof is based on the tree structure of the $\overline{G}_m$'s as well as on the connectivity of the G_i's. The fact that the graphs in $\mathcal{G}$ are expanders follows from the fact that the G_i's are expanders and the d^4-to-one connections of vertices in level $i+1$ to vertices in level i.

As mentioned above, the fact that $\mathcal{G}$ is a set of expander graphs implies that $\mathcal{G}$ does not have a tester of size-oblivious complexity. Recall that this is due to the fact, proved by Fichtenberger, Peng, and Sohler [3], that every infinite property of graphs that has a tester of size-oblivious query complexity must contain an infinite hyperfinite subproperty. But this argument does not provide an explicit lower bound on the query complexity of testing $\mathcal{G}$. Indeed, the main question addressed in the current paper is

What is the query complexity of testing $\mathcal{G}$?

We note that $\mathcal{G}$ can be tested within logarithmic query complexity: Essentially, using a logarithmic number of queries, we can reach any desired vertex in the graph (provided that the graph is in $\mathcal{G}$) and check whether this vertex satisfies all local conditions. (See Section 4.1 for details.) In contrast, it seems that locating a random vertex in a graph that is in $\mathcal{G}$ (e.g., even determining its distance to the root) requires a logarithmic number of queries, and that this observation should yield a lower bound on the query complexity of testing $\mathcal{G}$. We clarify this strategy next.

[2]The port of an edge $\{u,v\}$ at vertex u is the index of v in the incidence list of u. When representing the graph by the incidence function g, we say that the j^{th} port of vertex u is used to connect to v if $g(u,j) = v$ holds.

An important note: Our entire presentation is carried out in terms of directed edge-colored graphs, where the number of colors is a constant. To obtain analogous results for undirected graphs (with no edge-colors), we replace the directed colored edges by constant-size gadgets such that a different gadget is used for each different type of edges (i.e., a color of directed edges). In addition, we make sure that the vertices of these gadgets are easily distinguishable from the original edges (e.g., they can be made to have a different range of degrees). Hence, algorithms in one model can be emulated by algorithms in the other model with a constant factor overhead.

1.2 The basic approach

We take the king's road for proving query complexity lower bounds, which means following the "indistinguishability technique" (see [4, Sec. 7.2]). Specifically, we wish to define two distributions that must be distinguished by a tester for $\mathcal{G}$, but cannot be distinguished by an algorithm that makes fewer queries than the desired lower bound. In our case, for each $n \in \{\sum_{i=0}^{m} d^{4i} : m \in \mathbb{N}\}$, a natural choice of two distributions is the followings.

1. A random n-vertex graph (with vertex set $[n]$) in $\mathcal{G}$; that is, a random isomorphic copy of $\overline{G}_m$, where $n = \sum_{i=0}^{m} d^{4i}$.
2. An n-vertex graph consisting of an isolated vertex and d^4 graphs such each of these graphs is a random $\frac{n-1}{d^4}$-vertex graph in $\mathcal{G}$ (equiv., d^4 random isomorphic copies of $\overline{G}_{m-1}$, where $n = \sum_{i=0}^{m} d^{4i}$).[3]

Note that every graph G in the support of the second distribution is far from being an n-vertex expander, because G consists of connected components that are each smaller than $n/2$. It also seems hard to distinguish these two distributions when making a sub-logarithmic number of queries (because we are unlikely to reach a root of a tree). Indeed, this is the case if the distinguisher is not given the size of the graph (i.e., n) but is rather given only a sampling access to the vertices of the graph (on top of being given oracle access to its incidence function). The point is that, in that case, a random isomorphic copy of $\overline{G}_{m-1}$ is indistinguishable from d^4 random isomorphic copies of $\overline{G}_{m-1}$. However, in the standard model of testing graph properties, the tester is given the size of the graph as an explicit input. In contrast, a more flexible model, put forward in [5], does support a natural version that can be used here.

We start with describing the relevant flexible model that supports the aforementioned logarithmic lower bound (see Section 1.3). We then adapt the argument to the standard model, but derive a much weaker lower bound (see Section 1.4).

1.3 In the flexible model

Recall that in the standard model, the tester is given the size of the graph, denoted n, as an explicit input, and it is assumed that the vertex set equals $[n]$. This allows the tester to sample vertices in the input graph. Treating graphs with different vertex-sets requires either providing the tester with the vertex-set as explicit input or providing it with a "vertex-sampling device" (that returns a uniformly distributed vertex in each invocation).

In [5], the latter option is taken as a starting point for the introduction of a general framework in which the tester is given access to such a vertex-sampling device along with partial information

[3]Indeed, for $n = \sum_{i=0}^{m} d^{4i}$, it holds that $\frac{n-1}{d^4} = \sum_{i=1}^{m} d^{4(i-1)} = \sum_{i=0}^{m-1} d^{4i}$.

on the vertex-set. This partial information may take various forms, and three natural choices were outlined in [5]:

***Exact size version*:** As mentioned above, in this case the tester is explicitly given the exact number of vertices in the input graph. Recalling that, in all cases, the tester is also given a vertex-sampling device, this model is essentially equivalent to the standard model [5, Obs. 2.2].

***No information version*:** This version is not adequate in case the query complexity depends on the size of the graph. Hence, we don't discuss it here (and the interested reader is referred to [5]).

***Approximate size version*:** In this intermediate version, the tester is given an adequate approximation on the number of vertices in the graph. It seems natural to allow this approximation to suffice for determining the query complexity of the tester (as well as reject in case the property contains no graph with a number of vertices that equals the one in the input graph). In our case, when given an n-bit input graph, it is natural to provide the tester with $\Theta(\log n)$ (i.e., any number in the interval $[\Omega(\log n), O(\log n)]$) as well as with a bit indicating whether or not $n \in \{\sum_{i=0}^{m} d^{4i} : m \in \mathbb{N}\}$.

Our logarithmic lower bound applies to the approximate version. Specifically, we rely on the fact that an arbitrary value in the interval $[\log_{d^4} n \pm O(1)]$ does not allow to distinguish m from $m-1$. It follows that a sublogarithnmic-query tester cannot distinguish the second distribution (i.e., essentially, an n-vertex graph consisting of d^4 isomorphic copies of $\overline{G}_{m-1}$) from a random isomorphic copy of $\overline{G}_{m-1}$, which has $n' \stackrel{\rm def}{=} (n-1)/d^4$ vertices. The point is that the approximated size of the graph, given to the tester, does not separate n from n' (i.e., $\log_{d^4} n' \in [\log_{d^4} n \pm 2]$). Hence, we prove

Theorem 1.3 (a lower bound on the query complexity of testing $\mathcal{G}$ in the flexible model): *Consider the task of testing whether an n-vertex graph is in $\mathcal{G}$ in the flexible bounded-degree model when the tester obtains a* poly(d)*-factor approximation of n, and is guaranteed that $n \in \{\sum_{i=0}^{m} d^{4i} : m \in \mathbb{N}\}$. Then, the tester must make $\Omega(\log n)$ queries.*

Note that the $O(\log n)$-query tester laconically outlined in Section 1.1 can operate also in the flexible model, because it needs only a constant-factor approximation of $\log_2 n$. See Section 4.1 for details.

1.4 In the standard model

In contrast, when given the exact size of the graph, the tester is not asked to distinguish between the second distribution (i.e., essentially d^4 random isomorphic copies of $\overline{G}_{m-1}$) and a random isomorphic copy of $\overline{G}_{m-1}$, but rather between the second distribution and a random isomorphic copy of $\overline{G}_m$. Furthermore, here, the vertex set of $\overline{G}_m$ equals $[n]$, where $n = \sum_{i=0}^{m} d^{4i}$, and a random isomorphic copy of $\overline{G}_m$ is obtained by applying a random permutation to the vertex names.

We stress that in Section 1.3 we avoided the question of whether a sublogarithmic-query algorithm can distinguish the two distributions presented in Section 1.2; that is, rather than asking whether the second distribution (i.e., essentially, an n-vertex graph consisting of d^4 isomorphic copies of $\overline{G}_{m-1}$) is distinguishable from a random isomorphic copy of $\overline{G}_m$, we asked whether it is distinguishable from a random isomorphic copy of $\overline{G}_{m-1}$.

Returning to the original question (which refers to the two distributions on n-vertex graphs presented in Section 1.2), we proceed in several steps. First, we follow [9] in considering a *canonical tester* that selects uniformly q start vertices (in the n-vertex graph) and explores all vertices that are at distance at most q from these start vertices. This canonical tester, which has query complexity $Q \overset{\text{def}}{=} q \cdot O(d^4)^q = \exp(O(q))$, can emulate any q-query tester, but its queries are oblivious of the size of the graph (i.e., n). The subgraph seen in each of the q explorations is called the q-neighborhoods of the corresponding start vertex.

Next, we observe that the canonical tester can distinguish between a random isomorphic copy of $\overline{G}_m$ and a graph consisting of d^4 random isomorphic copies of $\overline{G}_{m-1}$ (and an isolated vertex) based either on a difference in the *probabilities that two random explorations collide* (where these explorations are started at two different vertices) or on a difference between the *distributions of q-neighborhoods of random vertices in* (random isomorphic copies of) $\overline{G}_m$ *and* $\overline{G}_{m-1}$.

Assuming that $q = o(m)$, the first event (i.e., collision of random explorations) occurs with negligible probability, and so the real issue is the possible difference between random q-neighborhoods (in the two distributions of graphs). Fixing q, let X_i denote the distribution of a random q-neighborhood in (a random isomorphic copy of) $\overline{G}_i$ (i.e., the distribution of the q-neighborhood of a random vertex in $\overline{G}_i$). Then, the problem we face is that the distribution of X_m *may* (i.e., unless proved differently) be very different from the distribution of X_{m-1}. Of course, it is possible that X_m and X_{m-1} are statistically close due to their arising from a similar iterative process (i.e., the definition of $\overline{G}_i$), but we don't know if that is the case. Instead, we show that there exist $m' < m''$ in $[0.5m, m]$ such that $X_{m'}$ and $X_{m''}$ are sufficiently close.

The key observation is that X_i ranges over at most $s \overset{\text{def}}{=} Q^{O(d^4) \cdot Q} = \exp(\widetilde{O}(Q)) = \exp(\exp(O(q)))$ values. Seeking $m' < m''$ that are both in $[0.5m, m]$ such that the total variation distance between $X_{m'}$ and $X_{m''}$ is at most $\delta \overset{\text{def}}{=} o(1/q)$, we observe that such a choice exists whenever $(s/\delta)^s < m/2$ (i.e., $\exp(\widetilde{O}(s)) = o(m)$).[4] Hence, for a given $n = \Theta(d^{4m})$, we may set q such that

$$\exp(\widetilde{O}(\exp(\exp(O(q))))) = o(\log n) \tag{1}$$

and obtain $m' < m''$ in $[0.5m, m]$ such that the total variation distance between $X_{m'}$ and $X_{m''}$ is at most $o(1/q)$. Before spelling out how this yields the desired indistinguishability, we note that Eq. (1) is satified by $q = \Omega(\log\log\log\log n)$.

The actual argument. For each $m \in \mathbb{N}$, letting $q = \Omega(\log\log\log m)$, we determine $m' < m''$ in $[0.5m, m]$ such that the total variation distance between $X_{m'}$ and $X_{m''}$ is at most $o(1/q)$, set $n = \sum_{i=0}^{m''} d^{4i}$, and consider the following two distributions.

1. A random n-vertex graph in $\mathcal{G}$; that is, a random isomorphic copy of $\overline{G}_{m''}$.
2. An n-vertex graph consisting of $\sum_{i=0}^{m''-m'-1} d^{4i} < d^{4(m''-m')}$ isolated vertices and $d^{4(m''-m')}$ graphs such each of these graphs is a random isomorphic copy of $\overline{G}_{m'}$.

   (Recall that each graph in this distribution is far from $\mathcal{G}$.)

(In Section 1.2 we used $m'' = m$ and $m' = m'' - 1$.)

[4]This observation can be proved by approximating each distribution over $[s]$ by a distribution in which all probabilities are integer multiples of δ/s.

As stated above, the execution of an arbitrary q-query tester can be emulated by a canonical tester that selects uniformly at random q start vertices and explores their q-neighborhood. On input drawn from the first (resp., second) distribution, this algorithm will get q samples of $X_{m''}$ (resp., $X_{m'}$), whereas (by our choice of $m', m'' \in [m/2, m]$) these two samples are statistically close. It follows that the tester cannot distinguish the two distribution of n-vertex graphs unless $q = \Omega(\log\log\log m)$, whereas a valid test must distinguish these distributions (because the first distribution is supported by $\mathcal{G}$ and the second distribution is supported by graphs that are far from $\mathcal{G}$). Observing that $m > 0.2 \cdot \log_d n$, we obtain –

Theorem 1.4 (a lower bound on the query complexity of testing $\mathcal{G}$ in the standard model): *Testing $\mathcal{G}$ in the standard bounded-degree model has query complexity $\Omega(\log\log\log\log n)$, where n is the number of vertices in the input graph.*

Note that the argument does not use the fact that the various X_i's are related via a relatively simple and local operation; the argument is totally generic and uses only an upper bound on the support size of the X_i's (akin the argument in [2]).

1.5 Discussion

Our focus on the graph property $\mathcal{G}$ is rooted in the fact that this is (essentially) the only local property that is known to have no testers of size-oblivious query complexity.[5] While we established a tight logarithmic lower bounds on the query complexity of testing $\mathcal{G}$ in the flexible model, our lower bound in the standard model is significantly weaker. We leave improving it as an open problem, conjecturing that a logarithmic lower bound holds also in this model.

Open Problem 1.5 (improving the lower bound on the query complexity of testing $\mathcal{G}$ in the standard model): *For starters, prove that the query complexity of testing $\mathcal{G}$ in the standard bounded-degree model is $\Omega(\log\log\log n)$, where n is the number of vertices in the input graph. Ultimately, is the query complexity of this task $\Omega(\log n)$?*

Of course, one may be more adventurous and ask whether there exist local graph properties with significantly higher query complexity. This question can be asked also for the flexible model where one is given an approximation of the size of the graph.

Open Problem 1.6 (a lower bound on the query complexity of testing local properties): *Is there a local graph property such that testing it requires $q(n) = \omega(\log n)$ queries, where n denotes the number of vertices in the input graph? For starters, consider the flexible bounded-degree graph model when the tester obtains an adequate approximation of n, and is guaranteed that the property contains an n-vertex graph.*[6] *Ultimately, what about query complexity $q(n) = n^{\Omega(1)}$?*

1.6 Organization

In Section 2 we review the relevant background, which includes the flexible model outlined in Section 1.3 and the notion of generalized subgraph freeness (which underlies Theorem 1.1). In Section 3 we provide a more detailed discussion of the graph property $\mathcal{G}$. Specifically, we provide

[5] Variations on Construction 1.2 are presented in [6, Sec. 4].

[6] Tentatively, fixing a query complexity bound q, we consider $\widetilde{n}$ an adequate approximation of n if $q(\widetilde{n}) = \Theta(q(n))$.

some hints as to why $\mathcal{G}$ is locally characterizable. This discussion (i.e., Section 3) is not essential for reading the rest of this paper.

The core of the current paper is presented in Sections 4 and 5, which detail the overviews provided in Sections 1.3 and 1.4, respectively. Specifically, Section 4 refers to the flexible model, whereas Section 5 refers to the standard model.

2 Preliminaries

Throughout this text, we focus on bounded-degree graphs. In Section 2.1 we review the flexible framework of testing graph properties in the bounded-degree model. This framework was introduced in [5], and extends the standard model (introduced in [8] and reviewed in [4, Chap. 9]), which is viewed here as a special case. In Section 2.2, we review the notion of *generalized subgraph freeness* (introduced in [9]). In both sections, we include generalizations for directed graphs with edge coloring.

2.1 Testing in the Bounded-Degree Graph Model

The standard model was introduced in [8] and is reviewed in [4, Chap. 9]. Here we present the more flexible framework of [5]. For sake of convenience, we denote the degree bound by b (rather than by d as is more common).[7] Recall that graph properties are sets of graphs that are closed under isomorphism.

The bounded-degree graph model refers to a fixed (constant) degree bound, denoted $b \geq 2$. In this model, a graph $G = (V, E)$ of maximum degree b is represented by the incidence function $g : V \times [b] \to V \cup \{\perp\}$ such that $g(v, j) = u \in V$ if u is the j^{th} neighbor of v and $g(v, j) = \perp \notin V$ if v has less than j neighbors.[8] Distance between graphs is measured in terms of their foregoing representation; that is, as the fraction of (the number of) different array entries (over $b \cdot |V|$).

The tester is given oracle access to the representation of the input graph (i.e., to the incidence function g) as well as to a device that returns uniformly distributed elements in the graph's vertex-set, which is viewed as a set of strings (i.e., a finite subset of $\{0,1\}^*$). As usual, the tester is also given the proximity parameter $\epsilon \in (0, 1]$. In addition, the tester gets some partial information about the vertex-set (i.e., V) as auxiliary input, where this partial information is an element of a set of possibilities, denoted $p(V)$. Two extreme possibilities are $p(V) = \{V\}$, which is closely related to the standard formulation (see [5, Obs. 2.2]) and $p(V) = \{\lambda\}$, but one can also consider natural cases such as $p(V) = \{|V|, |V| + 1, ..., O(|V|)\}$.

Definition 2.1 (property testing in the bounded-degree graph model, flexible version): *For a fixed $b \in \mathbb{N}$, let Π be a property of graphs of degree at most b, and $p : 2^{\{0,1\}^*} \to 2^{\{0,1\}^*}$. A* tester for the graph property Π (in the bounded-degree graph model) with partial information p *is a probabilistic oracle machine T that is given access to two oracles, an incidence function $g : V \times [d] \to V \cup \{\perp\}$ and a device denoted* $\texttt{Samp}(V)$ *that samples uniformly in V, and outputs a binary verdict that satisfies the following two conditions:*

[7] This is done in order to allow using d for the degree of the basis graph in the Zig-Zag construction.

[8] For simplicity, we adopt the standard convention by which the neighbors of v appear in arbitrary order in the sequence $(g(v, 1), ..., g(v, \deg(v)))$, where $\deg(v) \stackrel{\text{def}}{=} |\{j \in [b] : g(v, j) \neq \perp\}|$.

1. *The tester accepts each graph $G = (V, E) \in \Pi$ with probability at least $2/3$; that is, for every $g : V \times [b] \to V \cup \{\perp\}$ representing a graph in Π and every $\iota \in p(V)$* (and $\epsilon > 0$)*, it holds that* $\Pr[T^{g,\mathtt{Samp}(V)}(\iota, \epsilon) = 1] \geq 2/3$.

2. *Given $\epsilon > 0$ and oracle access to any graph G that is ϵ-far from Π, the tester rejects with probability at least $2/3$; that is, for every $\epsilon > 0$ and $g : V \times [b] \to V \cup \{\perp\}$ that represents a graph that is ϵ-far from Π and $\iota \in p(V)$, it holds that* $\Pr[T^{g,\mathtt{Samp}(V)}(\iota, \epsilon) = 0] \geq 2/3$*, where the graph represented by $g : V \times [b] \to V \cup \{\perp\}$ is* ϵ-far from Π *if for every $g' : V \times [b] \to V \cup \{\perp\}$ that represents a graph in Π it holds that $|\{(v, j) \in V \times [b] : g(v, j) \neq g'(v, j)\}| > \epsilon \cdot b \cdot |V|$.*

The tester is said to have one-sided error probability *if it always accepts graphs in Π; that is, for every $g : V \times [b] \to V \cup \{\perp\}$ representing a graph in Π* (and every $\iota \in p(V)$ and $\epsilon > 0$)*, it holds that* $\Pr[T^{g,\mathtt{Samp}(V)}(\iota, \epsilon) = 1] = 1$.

The query complexity of a tester for Π is a function (of the parameters b, n and ϵ) that represents the number of queries made by the tester on the worst-case n-vertex graph of maximum degree b, when given the proximity parameter ϵ. Fixing b, we typically ignore its effect on the complexity (equiv., treat b as a hidden constant).

Generalization for directed graphs with edge coloring. Directed graphs with edge colors are represented by two incidence functions, one representing outgoing edges (and their colors) and the other representing incoming edges (and their colors). The degree bound, b, refers to both outgoing and incoming edges. Thus, a directed graph $G = (V, E)$ (of maximum degree b) with edge coloring $\chi : E \to C$ is represented by the incidence functions $g_{\mathtt{in}}, g_{\mathtt{out}} : V \times [b] \to (V \times C) \cup \{\perp\}$ such that $g_{\mathtt{out}}(v, j) = (u, c) \in V \times C$ (resp., $g_{\mathtt{in}}(v, j) = (u, c) \in V \times C$) if the i^{th} outgoing (resp., incoming) edge of v is colored c and is an incoming (resp., outgoing) edge of vertex u; if v has less than j outgoing (resp., incoming) edges, then $g_{\mathtt{out}}(v, j) = \perp$ (resp., $g_{\mathtt{in}}(v, j) = \perp$). Needless to say, that is, if $g_{\mathtt{out}}(v, j) = (u, c)$, then there exists $k \in [b]$ such that $g_{\mathtt{in}}(u, k) = (v, c)$.

Properties of directed edge-colored graphs are defined as sets of such graphs that are preserved under relabeling of vertices (while leaving the edge directions and colors intact). Hence, saying that the directed edge-colored graph represented by $g_{\mathtt{in}}, g_{\mathtt{out}} : V \times [b] \to (V \times C) \cup \{\perp\}$ is ϵ-far from a property Π of such graphs means that for every $g'_{\mathtt{in}}, g'_{\mathtt{out}} : V \times [b] \to (V \times C) \cup \{\perp\}$ that represents a graph in Π it holds that $|\{(v, j) \in V \times [b] : g_{\mathtt{out}}(v, j) \neq g'_{\mathtt{out}}(v, j)\}| > \epsilon \cdot b \cdot |V|$. (Needless to say, in this case $|\{(v, j) \in V \times [b] : g_{\mathtt{in}}(v, j) \neq g'_{\mathtt{in}}(v, j)\}| > \epsilon \cdot b \cdot |V|$ holds too.)

2.2 Generalized Subgraph Freeness Properties

The notion of a *generalized subgraph-freeness*, which extends the standard notions of induced and non-induced subgraph freeness, was introduced in [9]. It is aimed to capture what one can see by exploring a constant-radius neighborhood of a vertex in a graph that has some predetermined graph property. The issue is that some vertices are fully explored (i.e., the explorer sees all their neighbors), whereas for other vertices (at the boundary of the exploration) the explorer may only encounter them but not all their neighbors (since it has not traversed their incident edges).

We shall actually consider the set of subgraphs that the explorer cannot encounter when exploring a graph that has the property, where these forbidden subgraphs are represented by *marked graphs*, which are graphs in which each vertex is marked either `full` or `semi-full` or `partial`. Intuitively, the marking `full` represent a vertex that is not at the boundary of the exploration, which means that all its incident edges were traversed. In contrast, vertices at the boundary are marked as `partial`, whereas the marking `semi-full` is inessential (and is included for sake of greater flexibility (see Footnote 9)).

Definition 2.2 (marked graphs, embedding, and generalized subgraph freeness): *A* marked graph *is a pair consisting of a graph and a marking of its vertices such that each vertex is marked either* `full` *or* `semi-full` *or* `partial`*. We say that a marked graph $F = ([h], A)$ can be* embedded *in a graph $G = ([N], E)$ if there exists a 1-1 mapping $\phi : [h] \to [N]$ such that for every $i \in [h]$ the following three conditions hold:*

1. *If i is marked* `full`*, then ϕ yields a bijection between the set of neighbors of i in F and the set of neighbors of $\phi(i)$ in G; that is, $\Gamma_G(\phi(i)) = \phi(\Gamma_F(i))$, where $\Gamma_X(v)$ denotes the set of neighbors of v in the graph X, and $\phi(S) = \{\phi(v) : v \in S\}$.*
2. *If i is marked* `semi-full`*, then ϕ yields a bijection between the set of neighbors of i in F and the set of neighbors of $\phi(i)$ in the* subgraph *of G induced by $\phi([h])$; that is, denoting this induced subgraph by $G|_{\phi([h])}$, it holds that $\Gamma_{G|_{\phi([h])}}(\phi(i)) = \phi(\Gamma_F(i))$* (equiv., $\Gamma_G(\phi(i)) \cap \phi([h]) = \phi(\Gamma_F(i))$).
3. *If i is marked* `partial`*, then ϕ yields an* injection *of the set of neighbors of i in F to the set of neighbors of $\phi(i)$ in G; that is, $\Gamma_G(\phi(i)) \supseteq \phi(\Gamma_F(i))$.*

*The graph G is called F-*free *if F cannot be embedded in G* (i.e., there is no embedding of F in G that satisfies the foregoing conditions)*. For a set of marked graphs $\mathcal{F}$, a graph G is called $\mathcal{F}$-*free *if for every $F \in \mathcal{F}$ the graph G is F-free.*

Indeed, the standard notion of (non-induced) subgraph freeness is a special case of generalized subgraph freeness, obtained by considering the corresponding marked graph in which all vertices are marked `partial`. Similarly, the notion of induced subgraph freeness is a special case of generalized subgraph freeness, obtained by considering the corresponding marked graph in which all vertices are marked `semi-full`.[9]

Marking vertices as `full` introduces a new type of constraint; specifically, this constraint mandates the non-existence of neighbors that are outside the subgraph that constitutes the image of the embedding (of a marked graph). For example, using vertices that are marked `full`, it is possible to disallow certain degrees in the graph (see Example 2.3). Thus, the generalized notion of subgraph freeness includes properties that are not hereditary (e.g., regular graphs), whereas induced and non-induced subgraph freeness are hereditary.

Example 2.3 (disallowing certian degrees via generalized subgraph freeness): *For every $d \in \{0, 1, ..., b\}$, we can disallow vertices of degree d by using a $(d+1)$-vertex star in which the center is marked* `full` *and the d leaves are marked* `partial`*.*

[9]Indeed, the `semi-full` marking (resp., the `partial` marking) can be avoided by emulating marked graphs by sets of mark graphs that use only `full` and `partial` (resp., `semi-full`) marking. Emulating the `partial` marking by `semi-full` marking is analogous to the emulation of non-induced subgraph freeness by induced subgraph freeness. As for emulating the `semi-full` marking, here we replace each marked graph F by a set of marked graphs $\mathcal{F}'$ such that each $F' \in \mathcal{F}'$ consists of a copy of F in which all `semi-full`-marked vertices are replaced by `full`-marked vertices and are connected to some auxiliary vertices, which are all marked `partial`. We stress that $\mathcal{F}'$ reflects all possible ways of connecting the newly `full`-marked vertices with the auxiliary vertices.

The foregoing example as well as the next one are actually used in the proof of Theorem 1.1. The following example refer to the case that we want to mandate that if the graph contains some fixed subgraph H' then it actually contains additional edges (i.e., $H \setminus H'$) on the same vertices.

Example 2.4 (mandating some subgraph via generalized subgraph freeness): *Let $H' = ([h], A')$ be a subgraph of $H = ([h], A)$, and suppose that we want to enforce that every induced subgraph of G that contains H' also contains H. This can be obtained by requiring G to be $\mathcal{F}$-free, where $\mathcal{F}$ is the set of all marked h-vertex graphs that are consistent with H' but not with H. Specifically, F is in $\mathcal{F}$ if F is embedded in every h-vertex graph that contains H' but not H.*

For sake of completeness, we present the following definition, which we actually use only in informal discussions and headings.

Definition 2.5 (locally characterizable properties): *A graph property Π is called* locally characterizable *if there exists a finite set of marked graphs $\mathcal{F}$ such that Π equals the set of $\mathcal{F}$-free graphs.*

We stress that Definition 2.5 is more restricted than [9, Def. 5.2]. The latter definition allows a different set of marked graphs to be used for each graph size (as long as there is a uniform bound on the size of all marked graphs used).

Generalization for directed graphs with edge coloring. The notions of marked graphs, embedding, and generalized subgraph freeness extend naturally to the context of directed edge-colored graphs. Specifically, we only need to redefine the function Γ_X such that $\Gamma_X(v)$ describes the set of (out-)neighbors of v in the graph X *along with the corresponding edges*; that is, $(u, \sigma, c) \in \Gamma_X(v)$ if and only if there exists an edge colored c that goes from v to u. Plugging this revised definition in Definition 2.2, we obtained the desire definition of $\mathcal{F}$-freeness, where $\mathcal{F}$ is a set of marked directed graphs with edge colors.

3 On the graph property $\mathcal{G}$

(The current section, which is not essential for the results of this paper, is reproduced from [6].)

Recall that the property $\mathcal{G}$ is defined by superimposing the graphs obtained in the Zig-Zag construction with a directed tree of fixed arity. The local characterization of $\mathcal{G}$ is define by constraints that capture the relation between G_i and G_{i+1}, where $G_1, G_2, ...$ is the sequence defined by the Zig-Zag construction (i.e., $G_1 = H^2$ and $G_{i+1} = G_i^2 \textcircled{z} H$), which in turn is based on the Zig-Zag product. Hence, we start with a description of the Zig-Zag product, which was introduced and first studied in [10]. With these preliminaries in place, we provide some hints regarding the proof that $\mathcal{G}$ is locally characterizable (see Section 3.2).

3.1 Preliminaries: The Zig-Zag Product

Given a (big) D-regular graph $G = (V, E)$, and a (small) d-regular graph $H = ([D], F)$, their Zig-Zag product, denoted $G \textcircled{z} H$, consists of the vertex set $V \times [D]$, which is partitioned to D-vertex clouds such that the cloud that corresponds to vertex $v \in V$ is the set of vertices $C_v = \{(v, i) : i \in [D]\}$, and edges that correspond to certain 3-step walks (as detailed next).

Actually, it is instructive to first consider the graph, denoted $G \textcircled{r} H$, in which copies of H are placed on the clouds (i.e., for every $v \in V$ and $\{i,j\} \in F$ we place the intra-cloud edge $\{(v,i),(v,j)\}$), and edges of G connect the corresponding clouds by using corresponding edges; that is, if $\{u,v\} \in E$ is the i^{th} (resp., j^{th}) edge incident at u (resp., at v), then we place the inter-cloud edge $\{(u,i),(v,j)\}$. Note that each vertex in $G \textcircled{r} H$ has d intra-cloud edges and a single inter-cloud edge. Now, the edges of $G \textcircled{z} H$ correspond to 3-step walks in $G \textcircled{r} H$ that start with an intra-cloud edge, then take the (only available) inter-cloud edge, and lastly take some intra-cloud edge; that is, such a generic walk has the form $(v,i) \to (v,j) \to (w,k) \to (w,\ell)$, where $\{i,j\},\{k,\ell\} \in F$ and $\{(v,j),(w,k)\}$ is an inter-cloud edge in $G \textcircled{r} H$ (i.e., $\{v,w\} \in E$ is the j^{th} edge incident at v and the k^{th} edge incident at w).

We shall assume that both G and H are connected and are not bipartite. In that case, it is clear that the graph $G \textcircled{r} H$ is also connected and non-bipartite, and it can be shown that also $G \textcircled{z} H$ has these properties. The main technical result of [10] asserts that the convergence rate of a random walk on $G \textcircled{z} H$ (a.k.a the relative second eigenvalue of the graph) can be upper-bounded in terms of the convergence rates of random walks on G and on H. A simple form of their bound asserts that $\lambda(G \textcircled{z} H) \leq \lambda(G) + \lambda(H)$, where $\lambda(X)$ denotes the convergence rate of a random walk on the graph X. Using $\lambda(H) \leq 1/4$, it follows that if $\lambda(G) \leq 1/2$, then $\lambda(G^2 \textcircled{z} H) \leq 1/2$.

Note: For sake of simplicity, we assume that the edges of H can be colored using d colors. This assumption can be met (since we are quite free in our choice of H).

3.2 On the local characterization of $\mathcal{G}$

As stated in Section 1.1, Construction 1.2 describes a property of directed graphs with edge-colors such that the property is local and the corresponding underlying graphs are expanders. The construction will be presented in terms of local conditions that the edges of the graph are required to satisfy, where the local conditions are enforced by forbidden neighborhoods of constant size (akin those in Examples 2.3 and 2.4). Indeed, the forbidden neighborhoods correspond to directed and edge-colored versions of marked graphs, which are defined analogously to the Definition 2.2.

Recall that, for each $m \in \mathbb{N}$, Construction 1.2 identifies a graph that consists of the graphs $G_1, ..., G_m$ (of the Zig-Zag construction) along with edges that connect each vertex of G_{i-1} to each vertex in the corresponding cloud of G_i. Hence, the construction consists of two parts: (1) edges that represent the edges of $G_1, ..., G_m$, and (2) edges that form a d^4-ary tree in which each vertex in G_{i-1} is connected to all vertices of the corresponding cloud of G_i. Below, we show how this structure is enforced by postulates that can be expressed by local conditions.

The main part of the construction will be directed edges that represents the edge-rotation functions of the graphs $G_1, ..., G_m$. Recall that the edge-rotation function of an undirected graph extend its incidence function such that the pair (u,α) is mapped to the pair (v,β) if the α^{th} edge of u equals the β^{th} edge of v (equiv., the α^{th} port of vertex u is connected to the β^{th} port of vertex v). In such a case, we shall color the directed edge (u,v) with the color (α,β).

Recall that H is a d-regular d^4-vertex graph and that $G_1 = H^2$ and $G_i = G_{i-1}^2 \textcircled{z} H$ are d^2-regular d^{4i}-vertex graphs. The edges of these d^2-regular graphs (i.e., $G_1, ..., G_m$) will be represented by d^4 edge sets such that each edge set represents edges between a pair of indices of possible ports. Specifically, for every $\alpha, \beta \in [d^2]$, we consider the edge-set $E_{\alpha,\beta}$ such that $(u,v) \in E_{\alpha,\beta}$ if for some i there exists an edge in G_i that connects the α^{th} port of vertex u to the β^{th} port of vertex v. Indeed, $E_{\alpha,\beta}$ is viewed as a set of directed edges that are colored (α,β), and we *postulate that*

$(u,v) \in E_{\alpha,\beta}$ *if and only if* $(v,u) \in E_{\beta,\alpha}$. Letting $E \stackrel{\rm def}{=} \bigcup_{\alpha,\beta\in[d^2]} E_{\alpha,\beta}$, we refer to $(u,v) \in E$ as an E-edge (and to $(u,v) \in E_{\alpha,\beta}$ as an $E_{\alpha,\beta}$-edge). The E-edges represent the edges of $G_1, ..., G_m$ without distinguishing the different G_i's.

We stress that the foregoing anti-parallel postulate is a very minimal one; far more substantial conditions will be postulated about the E-edges by referring also to other edges that will induce a layered directed acyclic graph win which G_i is identified with the $i^{\rm th}$ layer. Indeed, the actual structure of the graphs $G_1, ..., G_m$ will be enforced by relating each G_i to G_{i-1}.

As a *warm-up*, suppose that we want to augment the graph with auxiliary (colored) edges that will capture 2-step walks on the original graph. In such a case, we introduce, for every $\alpha, \beta, \gamma, \delta \in [d^2]$, an edge-set $E'_{(\alpha,\gamma),(\delta,\beta)}$ such that $(u,w) \in E'_{(\alpha,\gamma),(\delta,\beta)}$ if and only if there exists v such that $(u,v) \in E_{\alpha,\beta}$ and $(v,w) \in E_{\gamma,\delta}$. (We stress that the latter condition is a local condition about the edge-sets $E_{\alpha,\beta}$, $E_{\gamma,\delta}$ and $E'_{(\alpha,\gamma),(\delta,\beta)}$; actually, we will use E' only as a shorthand.)

As stated above, the structure of the graphs $G_1, ..., G_m$ is enforced by relating each G_i to G_{i-1}, where we define G_0 to be the graph consisting of a single vertex. The first step in enforcing this relation is the association of vertices in G_{i-1} with clouds of vertices in G_i such that each cloud contains d^4 vertices that are identified (equiv., ordered) within the cloud; that is, the d^4 ports of each vertex in G_{i-1}^2 are associated with distinct vertices of the corresponding cloud. This association is enforced by using edges that are directed from each vertex of G_{i-1} to the corresponding cloud of G_i such that these d^4 edges are assigned different colors. Specifically, for each $\sigma \in [d^4]$, we introduce a set of directed edges, denoted P_σ, and *postulate that each vertex has at most one outgoing P_σ-edge and at most one incoming P-edge*, where $P \stackrel{\rm def}{=} \bigcup_{\sigma\in[d^4]} P_\sigma$. Indeed, $(u,v) \in P_\sigma$ implies that v is the $\sigma^{\rm th}$ vertex in the cloud associated with u, where u is the "parent" of v in the directed tree induced by P. Additional postulates are added to identify the vertices of G_0 and G_m; specifically:

1. Intuitively, we postulate that there exists a single vertex with no incoming P-edges; the graph G_0 will consist of this vertex.

   Actually, *we postulate that there exists at most one vertex with no incoming P-edges*; the existence of such a vertex will follows from the tree structure of the P-edges (see below).

2. We postulate that the P-outdegree of each vertex is either 0 or d^4 (equiv., *each vertex either has no outgoing P-edges or has at least d^4 outgoing P-edges*).

3. Intuitively, we postulate that all vertices that have no outgoing P-edges belong to the same G_i, and that $i = m$.

   Actually, we *postulate that vertices that are connected by E-edges have the same number of outgoing P-edges* (equiv., the same number of outgoing P_σ-edges, for every $\sigma \in [d^4]$). The fact that vertices with no outgoing P-edges are in G_m follows from the first item (i.e., for $i \geq 1$, the graph G_i cannot contain vertices with no incoming P-edges).

Combining the foregoing postulates with additional postulates that refer to E-edges, this implies that the P-edges form a directed d^4-ary tree such that all leaves are at the same distance from the root. We warn that establishing this tree structure is the most complex part of the proof of [1, Thm. 3.1].

Next, we *postulate that the E-edges between the d^4 vertices that neighbor the single vertex of P-indegree 0* (i.e., the vertex of G_0) *form a copy of H^2*. Specifically, recalling that these vertices are identified by their incoming P-edges, we postulate that $(u,v) \in E$ if and only if there exist

$\sigma, \tau \in [d^4]$ such that u (resp., v) has an incoming P_σ-edge (resp., P_τ-edge) from G_0 and $\{\sigma, \tau\}$ is an edge in H^2. Furthermore, in this case $(u, v) \in E_{\alpha,\beta}$ if and only if the foregoing edge in H^2 uses the α^{th} port of u and the β^{th} port of v (for some $\alpha, \beta \in [d^2]$).

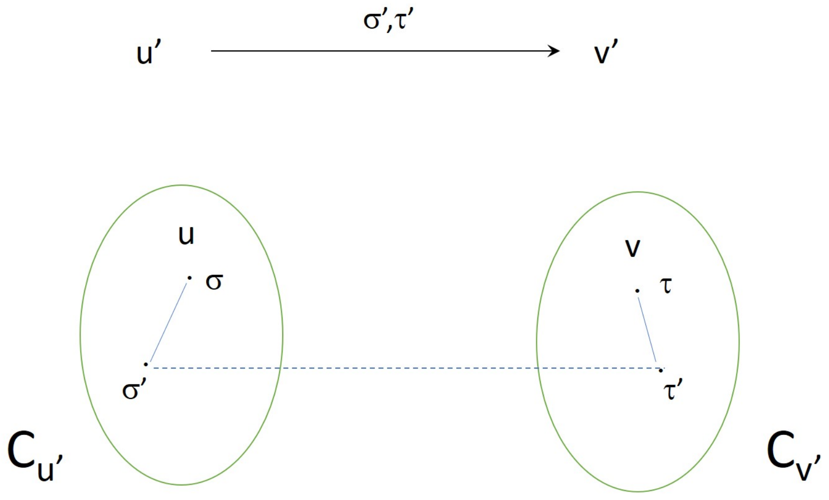

Figure 1: Vertex u (resp., v) is the σ^{th} (resp., τ^{th}) vertex in the cloud $C_{u'}$ (resp., $C_{v'}$) that replaces u' (resp., v'); these clouds are connected by an edge colored σ', τ'.

The main issue is relating the E-edges of $G_i = G_{i-1}^2 \textcircled{z} H$ to those of G_{i-1}, for $i > 1$. We stress that i itself cannot and is not referred to in this enforcement. Instead, we refer to any $(x, y) \in E$ such that x and y have outgoing P-edges and introduce conditions on the opposite endpoints of these P-edges; that is, we mandate E-edges among d of the P-neighbors of x (which reside in the cloud that replaces x) and d vertices of the P-neighbors of y (which reside in the cloud that replaces y). Specifically (as depicted in Figure 1 (while ignoring c_1, c_2)), for $(c_1, c_2) \in [d]^2 \equiv [d^2]$, we postulate that $(u, v) \in E_{(c_1,c_2),(c_2,c_1)}$ if and only if there exist $\sigma, \tau, \sigma', \tau' \in [d^4]$ and $(u', v') \in E'_{\sigma',\tau'}$ (i.e., u' and v' are connected by a 2-path colored (σ', τ') (see warm-up)) such that

1. $\{\sigma, \sigma'\}$ is an edge colored c_1 in H.[10]

2. $\{\tau, \tau'\}$ is an edge colored c_2 in H.

3. $(u', u) \in P_\sigma$ and $(v', v) \in P_\tau$.

Intuitively, these conditions imply that, for some i, the vertices u' and v' are connected in G_{i-1}^2, whereas u and v are vertices in the corresponding clouds of G_i. Furthermore, u (resp., v) is associated with the σ^{th} (resp., τ^{th}) vertex of H, which in turn neighbor vertex σ' (resp., τ') of H (see Figure 1). Moreover, for $\sigma \equiv (\alpha, \gamma) \in [d^2]^2$, vertices u' and v' are connected in G_{i-1} by a 2-path that uses the port $\alpha \in [d^2]$ of u' and the port $\gamma \in [d^2]$ of the intermediate vertex (whereas $\tau' = (\delta, \beta)$ such that δ and β are the ports used in walking this 2-path in the opposite direction). Indeed, the 2-paths referred to here are the edges of $E' \overset{\text{def}}{=} \bigcup_{\sigma',\tau' \in [d^4]} E'_{\sigma',\tau'}$, which were defined in the warm-up.

[10]Recall that we assumed, for sake of simplicity, that the edges of H can be colored using d colors.

The foregoing description suggests that the P-edges of a graph that satisfies the listed postulates form a d^4-ary directed tree such that all leaves are at the same distance from the root, and that the subgraph (of E-edges) induced by the set of vertices that are at distance i from the root equals G_i. This is indeed the case, but proving the former fact (which refers to the P-edges) requires using also the postulates that refer to the E-edges.[11] This is core of the analysis provided in [1, Sec. 3.1]. Hence, we have

Lemma 3.1 (the postulated conditions determine a unique unlabeled directed n-vertex graph): *For $n = \sum_{i=0}^{m} d^{4i}$, an n-vertex* (unlabeled edge-colored) *directed graph satisfies the foregoing conditions*[12] *if and only if it consists of the graphs $G_0, G_1, ..., G_m$* (such that $G_1 = H^2$ and $G_i = G_{i-1}^2 \textcircled{z} H$) *that are connected by P-edges as outlined above* (i.e., each vertex in G_{i-1} is connected by an P_σ-edge to the σ^{th} vertex in the corresponding cloud of G_i). *If $n > 1$ is not of the foregoing form, then no n-vertex graph satisfies these conditions.*

We note that the foregoing conditions can be enforced by forbidden neighborhoods of constant distance (akin those in Examples 2.3 and 2.4). Indeed, the forbidden neighborhoods correspond to directed and edge-colored versions of marked graphs, which are defined analogously to the Definition 2.2.

The foregoing n-vertex graph (consisting of $G_0, G_1, ..., G_m$ and the P-edges) has constant degree. We also observe that the corresponding undirected graph is an expander, by using the combinatorial notion of expansion. This is the case because each of the G_i's is an expander, whereas each vertex in G_{i-1} is connected to d^4 different vertices in G_i. Hence, for every set of vertices S and each i, letting S_i denote the vertices of S that reside in the d^{4i}-vertex graph G_i, we observe that if $|S_{i-1}| \leq d^{4(i-1)}/2$ then S_{i-1} contributes to the expansion inside G_{i-1} and otherwise S_{i-1} neighbors $d^4 \cdot |S_{i-1}| > d^{4i}/2$ vertices in G_i.

[11] Specifically, the postulates that refer to P-edges only enforce that the corresponding graph consists of at most one directed tree and a collection of directed cycles such that each vertex on each cycle is a root of a directed tree. However, the postulates that refer to E-edges imply that no such cycles exist, and so the graph consists of a single directed tree.

[12] Following is a compilation of all the conditions.

- *The edge-rotation* (i.e., anti-parallel) *condition*: $(u, v) \in E_{\alpha,\beta}$ if and only if $(v, u) \in E_{\beta,\alpha}$
- *The parental* (i.e., P-edges) *conditions*:
  - For every $\sigma \in [d^4]$ and every u, there exists at most one v such that $(u, v) \in P_\sigma$.
  - For every v, there exists at most one u such that $(u, v) \in P$.
  - There exists at most one v such that for every u it holds that $(u, v) \notin P$.
  - For every u, the set $\{v : (u, v) \in P\}$ is either empty or has size d^4, where in the latter case it holds that, for every $\sigma \in [d^4]$, there exists a unique v such that $(u, v) \in P_\sigma$.
  - For every $(u, u') \in E$ it holds that $|\{v : (u, v) \in P\}| = |\{v' : (u', v') \in P\}|$.
- *The base graph* ("level one") *condition*: Let r denote the vertex that has no incoming P-edges, and v_σ be such that $(r, v_\sigma) \in P_\sigma$ (for $\sigma \in [d^4]$). Then, $(v_\sigma, v_\tau) \in E_{(c_1,c_2),(c_2,c_1)}$ if and only if there exists $\rho \in [d^4]$ such that $\{\sigma, \rho\}$ is an edge colored c_1 in H and $\{\rho, \tau\}$ is an edge colored c_2 in H.
- *The Zig-Zag condition*: For $(c_1, c_2) \in [d]^2 \equiv [d^2]$, we postulate that $(u, v) \in E_{(c_1,c_2),(c_2,c_1)}$ if and only if there exist $\sigma, \tau, \sigma', \tau' \in [d^4]$ and $(u', v') \in E'_{\sigma',\tau'}$ (i.e., u' and v' are connected by a 2-path colored (σ', τ') (see warm-up)) such that $\{\sigma, \sigma'\}$ is an edge colored c_1 in H, $\{\tau, \tau'\}$ is an edge colored c_2 in H, $(u', u) \in P_\sigma$, and $(v', v) \in P_\tau$.

Lastly, we observe that the foregoing construction can be converted to the context of (simple) undirected graphs (with no edge-colors). This is done by replacing each color class and edge-direction by a different gadget such that the gadgets are non-isomorphic and their vertices can be distinguished from the original ones. In particular, we may use gadgets that contain vertices of higher degree than the degree of the original vertices. The same transformation is applied to the (directed and edge-colored) forbidden neighborhoods that enforce the conditions imposed on the (directed and edge-colored) construction. This yields a corresponding finite set of marked graphs, denoted $\mathcal{F}$, that satisfies the following –

Proposition 3.2 (a locally-characterizable property of expander graphs): *The set of $\mathcal{F}$-free graphs is an infinite set of expander graphs. Furthermore, this set contains a single unlabeled $\Theta(d^{4m})$-vertex graph for every $m \in \mathbb{N}$.*

We mention that, by using additional constraints, one can force these expanders to be regular graphs. In fact, this is done in [1].

4 In the Flexible Model

We shall consider a flexible version of the bounded-degree graph model in which the tester is provided with an $O(b)$-factor approximation of n (i.e., the number of vertices in the input graph) along with an indication of whether or not n is in $\{\sum_{i=0}^{m} d^{4i} : m \in \mathbb{N}\}$. Actually, we shall only consider inputs that are n-vertex graphs such that $n \in \{\sum_{i=0}^{m} d^{4i} : m \in \mathbb{N}\}$. In addition, as usual, the tester is provided with uniformly and independently distributed samples of the vertex-set and oracle access to the incidence function of the graph.

4.1 The tester

Following the laconic outline in Section 1.1, we first present a tester for $\mathcal{G}$. We stress that this tester uses the approximation to the number of vertices in the input graph only in order to determine a bound on the number of levels in the tree (which exists in the input graph in case the input graph is in $\mathcal{G}$). Using the colors of edges, it is easy to determine which edges are tree edges, and to reach the root of the tree by making a logarithmic number of steps. This also determines a candidate for the number of levels in the tree, denoted m. Furthermore, the colors of the tree edges allow to identify each vertex in each of the G_i's that reside in the levels of any graph that is isomorphic to $\overline{G}_m$.

Algorithm 4.1 (the tester for the directed edge-colored version): *On input a proximity parameter $\epsilon > 0$ and an approximation $\widetilde{n}$ to $n \in \{\sum_{i=0}^{m} d^{4i} : m \in \mathbb{N}\}$, when given oracle access to an n-vertex directed edge-colored graph G, the algorithm sets $\ell = O(\log \widetilde{n})$ and proceeds as follows.*

1. *It selects uniformly $O(1/\epsilon)$ random vertices, and checks whether the same vertex is reached from each of these vertices when taking at most ℓ steps against the direction of the tree edges* (i.e., at each step, the algorithm traverses the incoming tree edge, and if this edge is not unique, then the algorithm rejects).

   If not all walks reached the same vertex, the algorithm reject. Otherwise, the reached vertex is declared ti be the tentative root of the tree, and m is defined to be the maximal number of steps taken in one of these walks (till reaching the root).

2. *For each $i \in [m]$, the algorithm repeats the following check for $t_i \stackrel{\rm def}{=} \lfloor O(2^{-(m-i)}/\epsilon) \rfloor$ times.*
    (a) *The algorithm takes a random walk of length i from the root towards the leaves, where an i-step random walk is determined by selecting a sequence of i colors of tree edges* (i.e., a random element of $[d^4]^i$) *and traversing the corresponding edges in the forward direction.*
    (b) *For each vertex reached, the algorithm checks whether its neighborhood* (at the same level) *matches the neighbors of the corresponding vertex in G_i.*
    Recall that the vertex in the input graph G is identified by the sequence of the colors of the tree edges leading to it, and that the vertices in G_i are identified analogously (via the iterative process $G_j = G_{j-1}^2 \textcircled{z} H$, where $G_1 = H^2$).[13] *By the neighborhood of a vertex we mean the set of its neighbors along with the color of each of the corresponding edges.*
3. *The algorithm takes $O(1/\epsilon)$ random walks of length m from the root towards the leaves and checks that the vertices reached have no outgoing tree edges.*

If any of the checks performed in Steps 2 and 3 fails, then the algorithm rejects. Also, if during any of the walks the algorithm encounters a vertex with more than a single incoming tree edge or two or more outgoing tree edges that have the same color, then it rejects. Otherwise, the algorithm accepts.

Algorithm 4.1 makes $\sum_{i\in[m]} t_i = O(\epsilon^{-1} \cdot \log \widetilde{n})$ queries, which means that the query complexity is $O(\epsilon^{-1} \cdot \log n)$, provided that $\widetilde{n} \le \text{poly}(n)$. Note that the foregoing description presumes that $n \in \{\sum_{i=0}^m d^{4i} : m \in \mathbb{N}\}$. Recall that in the model that we consider, the tester also gets an indication of whether or not n is in $\{\sum_{i=0}^m d^{4i} : m \in \mathbb{N}\}$. Indeed, in the case of a negative indication, the tester may reject without even looking at the input graph.

Claim 4.2 (Algorithm 4.1 accepts inputs in $\mathcal{G}$): *If $G = \overline{G}_m$ and $\widetilde{n} \ge n \stackrel{\rm def}{=} \sum_{i=0}^m d^{4i}$, then Algorithm 4.1 accepts with probability at least* $2/3$.

The two-sided error of Algorithm 4.1 is due to Step 3, and arises from the fact that this algorithm does not get n. Hence, m is determined based on the empirical evidence (rather than derived from n), and may be smaller than the actual value (as determined by n).[14]

Proof: We first observe that Step 1 never rejects, and that, with very high probability, at least one of the vertices selected in Step 1 is at level m of $G = \overline{G}_m$. In this case, Step 1 determines m correctly. Consequently, none of the checks performed in Steps 2 and 3 fails, and the algorithm accepts. ■

Lemma 4.3 (Algorithm 4.1 rejects inputs that are far from $\mathcal{G}$): *If G is ϵ-far from $\mathcal{G}$, then Algorithm 4.1 rejects with probability at least* $2/3$.

Proof: We actually prove the contrapositive; that is, assuming that G is accepted with probability greater than $1/3$, we shall prove that G is ϵ-close to $\overline{G}_m$ such that $n = \sum_{i=0}^m d^{4i}$. Recall that n denotes the actual number of vertices in G, and let m be such that $n = \sum_{i=0}^m d^{4i}$ holds.

[13]For more details, see Construction 1.2 and Section 3.2.

[14]Indeed, an analogous algorithm for the standard model, where the tester gets the exact value of n, has one-sided error.

We first observe that if Step 1 determines a wrong value of m, denoted m', then (with high probability) the algorithm rejects. Specifically, if $m' > m$, then (with high probability) a random m'-step walk towards the leaves of the tree gets stuck, because the tentative root of G along with the tree edges determine a tree with at most d^{4m} leaves. On the other hand, $m' < m$ is unlikely because if most vertices reside at distance at most m' from some root, then Step 1 rejects (with high probability), and otherwise Step 3 rejects. (In the first case at most $\sum_{i=0}^{m'} d^{4i} < n/d^4$ vertices reside in the sub-tree that is rooted in any specific vertex.)

Next, based on the checks performed in Step 2, we infer that, for every $i \in [m]$, at least a $(1 - 2^{m-i-2} \cdot \epsilon)$ fraction of the i-step walks from the declared root (towards the leaves) reach vertices with a neighbourhood that matches the neighbourhood of the corresponding vertex in G_i. (Recall that the vertices in G are identified by the colors of the sequence of tree edges leading to them from the root, whereas vertices in G_i are identified via the iterative process $G_j = G_{j-1}^2 ⓩ H$.) It follows that, for every $i \in [m]$, the subgraph of G induced by the set of vertices that are at distance i from the declared root is $2^{m-i-2} \cdot \epsilon$-close to G_i.

Lastly, by Step 3 all but at most $\epsilon d^{4m}/4$ vertices at distance m from the root have outgoing tree edges. Likewise, all but at most $\epsilon d^{4i}/4$ vertices at distance $i \in [m]$ from the root have more than a single incoming tree edge or more that a single outgoing tree edge per each color. Removing the possible excessive tree edges, whose total number is smaller than $\epsilon bn/4$, we obtain a graph that is $\epsilon/2$-close to $\overline{G}_m$. ■

4.2 The lower bound (i.e., proof of Theorem 1.3)

The proof of Theorem 1.3 reduces to proving the following.

Claim 4.4 (indistinguishability claim): *Let A be an arbitrary $(m - \omega(\log m))$-query algorithm that explores an input graph when given the parameter $m \in \mathbb{N}$. Then, for any $m \in \mathbb{N}$, algorithm A cannot distinguish a random isomorphic copy of $\overline{G}_{m-1}$ from a graph consisting of an isolated vertex and d^4 random isomorphic copies of $\overline{G}_{m-1}$; that is, denoting the two distributions by Y and Z, respectively, it holds that*

$$|\Pr[A^Y(m)=1] - \Pr[A^Z(m)=1]| = o(1).$$

Note that a tester for $\mathcal{G}$ (in the current model) may be given $\widetilde{n} \stackrel{\rm def}{=} \sum_{i=0}^{m} d^{4i}$ in both cases (i.e., we may think of it as being given m), since $\widetilde{n}$ is an $O(d^4)$-approximation of the size of the graph in both cases.[15] Recalling that a tester for $\mathcal{G}$ (in the current model) must accept (with probability at least 2/3) any graph in the support of the first distribution and reject (with probability at least 2/3) any graph in the support of the second distribution, the theorem follows.

Proof: Letting $q = m - \omega(\log m)$ denote the query complexity of the algorithm, we observe that the algorithm can distinguish the two distributions only if one of the following event occurs:

1. The algorithm reached a vertex with no incoming tree edges.
2. The algorithm reached the same vertex by two different explorations that started at different sampled vertices (i.e., different vertices provided by the vertex sampling device).

[15] Actually, $\widetilde{n}$ equals the size of the graph in the second case (i.e., the size of Z).

The first event may occur only if the algorithm got a sample that is at distance $\ell \overset{\text{def}}{=} (m-1)-q$ from the leaves, but such an event may occur with probability at most $q \cdot 2 \cdot d^{-4\ell} < m \cdot \exp(-\omega(\log m)) = o(1)$. Turning to the second event, we observe that it occurs with probability at most $\binom{q}{2} \cdot d^{-4(m-1)} = o(1)$. The claim follows. ■

5 In the Standard Model: Proof of Theorem 1.4

Recall that a random isomorphic copy of $\overline{G}_m$ is obtained by relabeling the vertices of the graph (which are elements of $[n]$) by using a uniformly distributed permutation of $[n]$, where $n = \sum_{i=0}^{m} d^{4i}$. In continuation to the outline in Section 1.4, we prove the following claim

Claim 5.1 (indistinguishability claim): *For any $m \in \mathbb{N}$ and some $q = \Theta(\log\log\log m)$, there exists $m' < m'' \in [0.5m, m]$ such that no q-query algorithm, when given $n = \sum_{i=0}^{m''} d^{4i}$, can distinguish the following two distributions over n-vertex graphs.*

1. (Distribution 1): *A random n-vertex graph in $\mathcal{G}$; that is, a random isomorphic copy of $\overline{G}_{m''}$.*
2. (Distribution 2): *A n-vertex graph consisting of $\sum_{i=0}^{m''-m'-1} d^{4i}$ isolated vertices and $d^{4(m''-m')}$ graphs such each of these graphs is a random isomorphic copy of $\overline{G}_{m'}$.*

Observing that $n = O(d^{4m})$ and that the graphs in the second distribution are all far from being in $\mathcal{G}$, it follows that a tester for n-vertx graphs in $\mathcal{G}$ must make more than q queries. Hence, Theorem 1.4 follows.

Proof: Following [9], we consider a canonical tester that, on input n and oracle access to an n-vertex graph G, selects uniformly q start vertices in G and explores their q-neighborhoods (i.e., the subgraphs induced by all vertices that are at distance at most q from each of these start vertices). Such a canonical tester can emulate the execution of any q-query, and its decision depends only on n and on the q samples of q-neighborhoods of G, where a sample of a q-neighborhood is the q-neighborhood of a uniformly distributed vertex in G. Letting Y (resp., Z) denote the q-neighborhood of a random vertex in a graph selected from Distribution 1 (resp., Distribution 2), it follows that the distinguishing gap of the canonical tester equals the total variation distance between q independent samples of Y and q independent samples of Z.

In light of the fact that the probability that different samples of Y collide (i.e., the intersection between q-neighborhoods is not empty) is different from the probability of collision among samples of Z, we first upper-bound the probability of such collisions. Recalling that the number of vertices in each of these q-neighborhoods is $Q \overset{\text{def}}{=} \exp(O(q))$, we upper-bound the probability of collision by $\binom{2}{q} \cdot Q^2 \cdot d^{-4m'} = o(1)$, since $q = o(m')$. Hence, it suffices to show that the total variation distance between a single sample of Y and a single sample of Z is $o(1/q)$.

Fixing q, let X_i denote the distribution of a random q-neighborhood in (a random isomorphic copy of) $\overline{G}_i$ (i.e., the distribution of the q-neighborhood of a random vertex). We show that there exist $m' < m''$ in $[0.5m, m]$ such that the total variation distance between $X_{m'}$ and $X_{m''}$ is $o(1/q)$. The key observation is that X_i ranges over at most $s \overset{\text{def}}{=} Q^{O(d^4)\cdot Q} = \exp(\widetilde{O}(Q)) = \exp(\exp(O(q))$ values. Now, for $\delta = o(1/q)$, let X'_i be a distribution that is δ-close to X_i such that $\Pr[X'_i = x]$ is an integer multiple of δ/s. Then, the number of possible distributions of the latter form (called s/δ-grained in [4, Sec. 11.2.2]) is smaller than $(s/\delta)^s < \exp(\widetilde{O}(s)) = \exp(\exp(\exp(q)))$, which is smaller than $m/2$ by a suitable choice of q. The claim follows ■

A possible avenue towards addressing Problem 1.5. For every $m, q \in \mathbb{N}$ and X_m as defined in the proof of Claim 5.1, it is tempting to conjecture that the total variation distance between X_m and X_{m-1} is $\exp(-(m - O(q)))$. If this is indeed the case, then the query complexity of testing $\mathcal{G}$ (in the standard bounded-degree model) is $\Omega(\log n)$, where n is the number of vertices in the input graph. In general, an upper bound of the form $\exp(-(m - f(q)))$ would yield a lower bound of the form $\Omega(f^{-1}(\log n))$. Hence, we suggest the following open problem.

Open Problem 5.2 (relating X_m and X_{m-1}): *For every $m, q \in \mathbb{N}$, let X_m denote the q-neighborhood of a random vertex in a random isomorphic copy of $\overline{G}_m$. Provide an upper bound on the total variation distance between X_m and X_{m-1}.*

References

[1] Isolde Adler, Noleen Kohler and Pan Peng. On Testability of First-Order Properties in Bounded-Degree Graphs and Connections to Proximity-Oblivious Testing. In `arXiv:2304.03810` [cs.LO]. Combines preliminary versions that appeared in *SODA21* and *CCC21*.

[2] Itai Benjamini, Oded Schramm, and Asaf Shapira. Every Minor-Closed Property of Sparse Graphs is Testable. In *40th STOC*, pages 393–402, 2008.

[3] Hendrik Fichtenberger, Pan Peng, and Christian Sohler. Every testable (infinite) property of bounded-degree graphs contains an infinite hyperfinite subproperty. In *30th SODA*, pages 714–726, 2019.

[4] Oded Goldreich. *Introduction to Property Testing.* Cambridge University Press, 2017.

[5] Oded Goldreich. Flexible Models for Testing Graph Properties. In *Computational Complexity and Property Testing*, Lecture Notes in Computer Science (Vol. 12050), Springer, pages 352–362, 2020.

[6] Oded Goldreich. On locally-characterized expander graphs (a survey). *ECCC*, TR24-013, 2024.

[7] Oded Goldreich, Shafi Goldwasser, and Dana Ron. Property Testing and its connection to Learning and Approximation. *Journal of the ACM*, pages 653–750, July 1998.

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

[9] Oded Goldreich and Dana Ron. On Proximity Oblivious Testing. *SIAM Journal on Computing*, Vol. 40, No. 2, pages 534–566, 2011.

[10] Omer Reingold, Salil Vadhan, and Avi Wigderson. Entropy Waves, the Zig-Zag Graph Product, and New Constant-Degree Expanders and Extractors. *Annals of Mathematics*, Vol. 155 (1), pages 157–187, 2001. Preliminary version in *41st FOCS*, pages 3–13, 2000.