Electronic Colloquium on Computational Complexity, Report No. 85 (2005)

Homomorphisms in Graph Property Testing - A Survey

Dedicated to Jaroslav Nešetřil on the occasion of his 60^{th} birthday

Noga Alon * Asaf Shapira †

Abstract

Property-testers are fast randomized algorithms for distinguishing between graphs (and other combinatorial structures) satisfying a certain property, from those that are far from satisfying it. In many cases one can design property-testers whose running time is in fact *independent* of the size of the input. In this paper we survey some recent results on testing graph properties. A common thread in all the results surveyed is that they rely heavily on the simple yet useful notion of graph homomorphism. We mainly focus on the combinatorial aspects of property-testing.

1 Introduction

1.1 Property-testing background

The meta problem in the area of property testing is the following: Design a randomized algorithm, which given a combinatorial structure S, can distinguish with high probability between the case that S satisfies some property $\mathcal{P}$ from the case that S is ϵ-far from satisfying $\mathcal{P}$. Here S is said to be ϵ-far from satisfying $\mathcal{P}$ if an ϵ-fraction of its representation should be modified in order to make S satisfy $\mathcal{P}$. The main goal is to design randomized algorithms, which look at a very small portion of the input, and using this information distinguish with high probability between the above two cases. Such algorithms are called *property testers* or simply *testers* for the property $\mathcal{P}$. Preferably, a tester should look at a portion of the input whose size is a function of ϵ only. Blum, Luby and Rubinfeld [19] were the first to formulate a question of this type, and the general notion of property testing was first formulated by Rubinfeld and Sudan [59], who were motivated in studying various algebraic properties such as linearity of functions.

The main focus of the present survey is in testing properties of graphs. In this case a graph G is said to be ϵ-far from satisfying a property $\mathcal{P}$, if one needs to add/delete at least ϵn^2 edges to G in order to turn it into a graph satisfying $\mathcal{P}$. Here we assume that the tester can query an oracle, whether a pair of vertices, i and j, are adjacent in the input graph G. The study of the notion

*Schools of Mathematics and Computer Science, Raymond and Beverly Sackler Faculty of Exact Sciences, Tel Aviv University, Tel Aviv, Israel. Email: nogaa@tau.ac.il. Research supported in part by a USA Israeli BSF grant, by a grant from the Israel Science Foundation, and by the Hermann Minkowski Minerva Center for Geometry at Tel Aviv University.

†School of Computer Science, Raymond and Beverly Sackler Faculty of Exact Sciences, Tel Aviv University, Tel Aviv, Israel. Email: asafico@tau.ac.il. Research supported in part by a Charles Clore Foundation Fellowship and an IBM Ph.D Fellowship.

ISSN 1433-8092

of testability for combinatorial structures, and mainly for labelled graphs, was introduced in the seminal paper of Goldreich, Goldwasser and Ron [34]. In this paper it was shown that many natural graph properties such as k-colorability, having a large clique and having a large cut, have a tester, whose *query complexity* (that is, the number of oracle queries of type "does (i,j) belong to $E(G)$") can be upper bounded by a function of ϵ that is independent of the size of the input. In the present paper we will say that properties having such efficient testers, that is, testers whose query complexity is a function of ϵ only, are simply *testable*. Note, that if the query complexity of a tester can be upper bounded by a function of ϵ only, then so is its running time. Thus, if a property $\mathcal{P}$ is testable then there is a *constant* time randomized algorithm, for distinguishing between graphs satisfying $\mathcal{P}$ from those that are ϵ-far from satisfying it. In general, a property tester has a small probability of accepting graphs that are ϵ-far from satisfying the tested property, as well as a small probability of rejecting graphs satisfying the property. In this case the tester is said to have *two-sided* error. If the tester accepts graphs satisfying the property with probability 1, then the tester is said to have *one-sided* error.

For further reading and pointers on testing properties of graphs and other combinatorial structures the reader is referred to the surveys [28], [33], [57] and their references.

1.2 Background on graph homomorphism

As the title of this survey alludes to, its main focus is applications of homomorphisms in graph property-testing. We start with defining homomorphism between graphs.

Definition 1.1 (Homomorphism) *A homomorphism from a graph F to a graph K is mapping $\varphi : V(F) \mapsto V(K)$, which maps edges to edges, namely $(v,u) \in E(F)$ implies $(\varphi(v), \varphi(u)) \in E(K)$.*

Throughout this survey, $F \mapsto K$ will denote that there is a homomorphism from F to K, and $F \not\mapsto K$ will denote that no such homomorphism exists. Practicing definitions, note that if $F \mapsto K$ then $\chi(F) \leq \chi(K)$, and that F has a homomorphism into the complete graph on k vertices if and only if F is k-colorable.

Though innocent looking, the notion of graph homomorphism comes in handy in many applications in graph theory and theoretical computer science. Note, that a homomorphisms between two graphs is not as informative as an isomorphism between them, and this lack of perfect information is useful in many situations.

It is useful to extend the standard notion of homomorphism to directed graphs as well. Thus, a homomorphism between two directed graphs is a function mapping vertices to vertices so that directed edges are mapped to directed edges (oriented in the same way). The following notion plays a crucial role in the results described in Section 8

Definition 1.2 (Core) *The core of a graph (or digraph) H, is the smallest (in terms of edges) subgraph K, of H, for which there is a homomorphisms from H to K.*

We refer the reader to [17] and [42] for more background and references on digraph homomorphisms, and to [41] for more information and references on cores of graphs. For additional reading on graph homomorphisms the reader is referred to the recent comprehensive book of Hell and Nešetřil [40] and its numerous references.

1.3 The main results surveyed

In this survey we will mainly focus on the *combinatorial* aspects of property-testing rather than on its *algorithmic* aspects. To do so we will show that for the problems considered here, the property-testing tasks, which are algorithmic in nature, can be expressed in terms of (essentially equivalent) extremal graph-theoretic problems. Furthermore, all the properties surveyed here, will turn out to have testers whose query complexity is a function of ϵ only. This will make the property-testers we will consider extremely simple. In fact, all the algorithms for testing some property $\mathcal{P}$, will have the following meta-structure: Given a graph G and an error parameter ϵ: Sample a random set of vertices S, of size $f_{\mathcal{P}}(\epsilon)$ out of $V(G)$, and declare that G satisfies $\mathcal{P}$ if and only if the graph induced by G on S satisfies $\mathcal{P}$ (or is close to satisfying it in some cases). Of course, the main difficultly lies in showing that for a given property $\mathcal{P}$ there exists an $f_{\mathcal{P}}(\epsilon)$ for which such an algorithm distinguishes with high probability between graphs satisfying $\mathcal{P}$ and those that are ϵ-far from satisfying it.

This survey is organized as follows: In Section 2 we describe a result of [35] about the graph partition problems that are testable with one-sided error. Section 3 presents the regularity-lemma of Szemerédi [60] and its applications, which are used in some of the other results surveyed in this paper. In Section 4 we discuss the main result of [10], showing that any monotone graph property is testable. Section 5 is devoted to the main result of [13], which gives approximation algorithms and hardness results for edge-deletion problems. In Section 7 we introduce the notion of colored-homomorphism, which was used in [12] in order to prove that any hereditary graph property is testable, and in order to characterize the "natural" graph properties that are testable with one-sided error. In Section 6 we discuss the main result of [11] about uniform and non-uniform property testing. In Section 8 we discuss the main results of [1] and [7] about testing H-freeness for a fixed directed or undirected graph H. Most of the sections contain some interesting open problems for future research.

2 Testing Partition Problems with One-Sided Error

We start this section by introducing the notion of graph partitioning problems that was studied in [34]: For some integer k, let S be a set of $k + k + \binom{k}{2}$ pairs of reals which we denote by (α_i, β_i) for every $1 \leq i \leq k$ and $(\gamma_{i,j}, \delta_{i,j})$ for every $1 \leq i \leq j \leq k$. Then, a graph G on n vertices is said to satisfy the partition property $\mathcal{P}(S)$, if $V(G)$ can be partitioned into k sets of vertices $V_1, \ldots, V_k$ such that the following holds: (i) for every $1 \leq i \leq k$ we have $\alpha_i \leq |V_i|/n \leq \beta_i$. (ii) for every $1 \leq i \leq j \leq k$ we have $\gamma_{i,j} \leq |E(V_i, V_j)|/n^2 \leq \delta_{i,j}$. In what follows, we will call k the *order* of S. One of the main results of [34] states the following:

Theorem 2.1 ([34]) *For any S of order k, $\mathcal{P}(S)$ can be tested with query complexity $(1/\epsilon)^{c(k)}$.*

Note, that the properties of being k-colorable, having a large cut (of size at least, say, $\frac{1}{8}n^2$), having a large clique and having a large bisection, can all be expressed as partition problems by an appropriate set of constants S. Thus, by the above theorem they are all testable. The testers designed in [34] for general partition problems may have two-sided error. For some special cases, however, one can design one-sided error tester. One example is the k-colorability, which was shown to be testable with one-sided error in [34]. One of the (three) results of [35], is a characterization of the partition problems, which can be tested with one-sided error and query complexity independent

of the size of the input[1].

Theorem 2.2 ([35]) *A (non-trivial) partition problem $\mathcal{P}(S)$, is testable with one-sided error and query complexity bounded by a function of ϵ if and only if one of the following holds:*

1. *There is a graph H, such that satisfying $\mathcal{P}(S)$ is equivalent to having a homomorphism to H.*
2. *$\mathcal{P}(S)$ is equivalent to being a complete graph.*

It is easy to see that the property of being a complete graph is testable with one sided error and query-complexity $O(1/\epsilon)$. Given the techniques of [34] and [5] it can be shown that for any fixed H, the property of having a homomorphism to H is testable with one sided error and query complexity $O(1/\epsilon^2)$. In fact, this result follows as a special case of the main result of [6]. The main difficulty in the proof of Theorem 2.2, thus lies in proving that any other partition problem cannot be tested with one-sided error. The details appear in [35]. It is worth noting that if one only considers "oblivious" testers, as defined in Section 7, then one can derive the negative side of Theorem 2.2 from Theorem 7.9. The reason is that it is not difficult to show that a partition problem defines a semi-hereditary graph property (see Section 7) if and only if it is either equivalent to the clique-property or is equivalent to having a homomorphism to a fixed graph H.

The most well-studied partition problem is probably k-colorability. For the case of $k = 2$ it is known that in order to test the property of being bipartite a sample of $\tilde{\Theta}(1/\epsilon)$ vertices is both sufficient and necessary, see [5]. However, for $k \geq 3$ it is only known that in order to test k-colorability one has to use a sample of size $\Omega(1/\epsilon)$ vertices, and one of size $O(1/\epsilon^2)$ suffices. It seems interesting to close this gap by resolving the following problem.

Open Problem 2.3 *Prove tight bounds for the query complexity of k-colorability for $k \geq 3$.*

3 Regularity Lemmas: Definitions, Statements and Applications

Several of the results discussed in this survey are obtained using the regularity-lemma of Szemerédi [60]. In this section we give the necessary background on this powerful tool, which is needed in order to present these results. For a comprehensive survey on the regularity-lemma and its applications, the interested reader is referred to [45]. We start with some basic definitions. For every pair of nonempty disjoint vertex sets A and B of a graph G, we define $e(A, B)$ to be the number of edges of G between A and B. The *edge density* of the pair is defined by $d(A, B) = e(A, B)/|A||B|$.

Definition 3.1 (γ-regular pair) *A pair (A, B) is* γ-regular*, if for any two subsets $A' \subseteq A$ and $B' \subseteq B$, satisfying $|A'| \geq \gamma|A|$ and $|B'| \geq \gamma|B|$, the inequality $|d(A', B') - d(A, B)| \leq \gamma$ holds.*

Note, that a sufficiently large random bipartite graph, where each edge is chosen independently with probability d, is very likely to be a γ-regular pair with density roughly d, for any $\gamma > 0$. Thus, in some sense, the smaller γ is, the closer a γ-regular pair is to looking like a random bipartite graph. For this reason, the reader who is unfamiliar with the regularity lemma and its applications, should try and compare the statements given in this section to analogous statements about random graphs.

[1]The Characterization in [35] also considers degenerate cases, such as partition problems that are satisfied by only finitely many graphs. We focus here on the non-degenerate cases, which we call non-trivial.

Let F be a graph on f vertices and K a graph on k vertices, and suppose $F \mapsto K$. Let G be a graph obtained by taking a copy of K, replacing every vertex with a sufficiently large independent set, and every edge with a random bipartite graph of edge density d. It is easy to show that with high probability G contains many copies of F. The following lemma shows that in order to infer that G contains many copies of F, it is enough to replace every edge with a "regular enough" pair. Intuitively, the larger f and k are, and the sparser the regular pairs are, the more regular we need each pair to be, because we need the graph to be "closer" to a random graph. This is formulated in Lemma 3.2 below. Several versions of this lemma were previously proved in papers using the regularity lemma. See, e.g., [45]. The reader should think of the mapping φ in the statement of the lemma as defining the homomorphism from F to the (implicit) graph K.

Lemma 3.2 (The Embedding Lemma) *For every real $0 < \eta < 1$, and integers $k, f \geq 1$ there exist $\gamma = \gamma_{3.2}(\eta, k, f)$, $\delta = \delta_{3.2}(\eta, k, f)$ and $M = M_{3.2}(\eta, k, f)$ with the following property. Let F be any graph on f vertices, and let $U_1, \ldots, U_k$ be k pairwise disjoint sets of vertices, where $|U_1| = \ldots = |U_k| = m \geq M$. Suppose there is a mapping $\varphi : V(F) \mapsto \{1, \ldots, k\}$ such that the following holds: If (i, j) is an edge of F then $(U_{\varphi(i)}, U_{\varphi(j)})$ is γ-regular with density at least η. Then, the sets $U_1, \ldots, U_k$ span at least δm^f copies of F.*

A partition $\mathcal{A} = \{V_i \mid 1 \leq i \leq k\}$ of the vertex set of a graph is called an *equipartition* if $|V_i|$ and $|V_j|$ differ by no more than 1 for all $1 \leq i < j \leq k$ (so in particular each V_i has one of two possible sizes). The Regularity Lemma of Szemerédi can be formulated as follows.

Lemma 3.3 (Szemerédi's Regularity Lemma [60]) *For every m and $\gamma > 0$ there exists a number $T = T_{3.3}(m, \gamma)$ with the following property: Any graph G on $n \geq T$ vertices, has an equipartition $\mathcal{A} = \{V_i \mid 1 \leq i \leq k\}$ of $V(G)$ with $m \leq k \leq T$, for which all pairs (V_i, V_j), but at most $\gamma\binom{k}{2}$ of them, are γ-regular.*

In most of the applications of Lemma 3.3 one removes from G the following three types of edges: (i) edges spanned by the sets $V_1, \ldots, V_k$ (ii) edges connecting pairs (V_i, V_j), whose density $d(V_i, V_j)$ is small, say, smaller than γ (iii) edges connecting pairs (V_i, V_j), which are not γ-regular. The main reason for disregarding these edges (at least in applications similar in nature to the ones considered here) is that we would want to apply Lemma 3.2 on some of the sets $V_1, \ldots, V_k$ and to use this lemma we need pairs that are both regular and dense. A simple, yet useful observation, is that if we apply Lemma 3.2 with $m > 1/\gamma$, then the total number of edges removed in (i),(ii) and (ii) above is smaller than $2\gamma n^2$. Thus we can restate the regularity lemma in the following more convenient way:

Lemma 3.4 *For every $\gamma > 0$ and $m > 1/\gamma$ there exists a number $T = T_{3.4}(m, \gamma)$ with the following property: from any graph G on $n > T$ vertices, one can remove at most $2\gamma n^2$ edges to obtain a k-partite graph, with partition classes $V_1, \ldots, V_k$, such that the following holds:*

1. *$m \leq k \leq T$.*
2. *Any pair (V_i, V_j) is γ-regular[2] and satisfies either $d(V_i, V_j) = 0$ or $d(V_i, V_j) \geq \gamma$.*

[2]Note that if $d(V_i, V_j) = 0$ then (V_i, V_j) is trivially γ-regular.

For a (possibly infinite) family of graphs $\mathcal{F}$, a graph G is said to be $\mathcal{F}$-free if it contains no $F \in \mathcal{F}$ as a (not necessarily induced) subgraph. We need the following simple lemma, which reduces the task of testing the property of being $\mathcal{F}$-free to a purely combinatorial problem.

Lemma 3.5 *Let $\mathcal{F}$ be a (possibly infinite) family of graphs, and suppose there are functions $f_{\mathcal{F}}(\epsilon)$ and $\delta_{\mathcal{F}}(\epsilon)$ such that the following holds for every $\epsilon > 0$: Every graph G on n vertices, which is ϵ-far from begin $\mathcal{F}$-free contains at least $\delta_{\mathcal{F}}(\epsilon)n^f$ copies of a graph $F \in \mathcal{F}$ of size $f \leq f_{\mathcal{F}}(\epsilon)$. Then, being $\mathcal{F}$-free is testable with one-sided error.*

Proof: Suppose $\mathcal{F}$ is a family for which the functions $f_{\mathcal{P}}(\epsilon)$ and $\delta_{\mathcal{P}}(\epsilon)$ exist. For any $\epsilon > 0$ put $f = f_{\mathcal{P}}(\epsilon)$ and $\delta = \delta_{\mathcal{P}}(\epsilon)$. Given a graph the tester picks a set of vertices S, of size $2f/\delta$ and accepts G if and only if the subgraph of G spanned by S is $\mathcal{F}$-free. Clearly, if G is $\mathcal{F}$-free the algorithm accepts G with probability 1. Assume now that G is ϵ-far from being $\mathcal{F}$-free. In this case a randomly chosen set of f vertices spans a copy of a graph $F \in \mathcal{F}$ with probability at least δ. Thus, a sample of size $2f/\delta$ spans a copy of F, with probability at least $2/3$. ■

A standard application of Lemmas 3.2 and 3.4 gives the following:

Lemma 3.6 *For any* **finite** *set of graphs $\mathcal{F}$, the property of being $\mathcal{F}$-free is testable.*

Proof (sketch): The strategy is quite simple: we would like to show that if a graph G is ϵ-far from being $\mathcal{F}$-free then there is a graph $F \in \mathcal{F}$ and t subsets of $V(G)$, denoted $V_1, \ldots, V_t$, such that Lemma 3.2 can be applied on these sets with respect to F. By Lemma 3.5 this will be sufficient for inferring that $\mathcal{F}$-freeness is testable.

To carry out the above strategy we first need to know how regular should the sets $V_1, \ldots, V_t$ be in order to let us use Lemma 3.2 on them. As $\mathcal{F}$ is finite, there is an upper bound, say f_{max}, on the size of the largest graph in $\mathcal{F}$. Thus, we would not need to apply Lemma 3.2 with $k > f_{max}$ or with $f > f_{max}$. Therefore, if the sets $V_1, \ldots, V_t$ are γ'-regular, where we set $\gamma' = \gamma_{3.2}(\epsilon, f_{max}, f_{max})$ (it will next become clear why we use ϵ here), then we will be able to apply Lemma 3.2 on the sets $V_1, \ldots, V_t$ to find many copies of *any* $F \in \mathcal{F}$. But, to do so we still have to find the t sets $U_1, \ldots, U_t$ and argue why the densities between them correspond to the edge set of some graph $F \in \mathcal{F}$. To this end, we apply Lemma 3.4 with $\gamma = \frac{1}{2}\min(\epsilon, \gamma')$ and $m = 1/\gamma$. By Lemma 3.4, we can remove from G at most ϵn^2 edges and get a k-partite graph, denoted G', as in the statement of the lemma. As G is by assumption ϵ-far from being $\mathcal{F}$-free G' still spans a copy of some $F \in \mathcal{F}$. Suppose F is spanned by the sets $V_1, \ldots, V_t$ and recall that we must have $|V(F)| \leq f_{max}$ and $t \leq f_{max}$. Furthermore, by the definition of G' we have the following important property: if $x, y \in V(F)$ are connected in F, vertex x belongs to V_i and vertex y belongs to V_j, then (V_i, V_j) must be ϵ-regular and satisfy $d(V_i, V_j) \geq \epsilon$. As we chose $\gamma' = \gamma_{3.2}(\epsilon, f_{max}, f_{max})$, we can apply Lemma 3.2 on the sets $V_1, \ldots, V_t$. As each of these sets is of size at least $n/T_{3.4}(1/\gamma, \gamma)$, we conclude from Lemma 3.2 that $V_1, \ldots, V_t$ span at least

$$\delta_{3.2}(\epsilon, f_{max}, f_{max}) \left(\frac{n}{T_{3.4}(1/\gamma, \gamma)} \right)^{|V(F)|}$$

copies of F. As γ was defined in terms of γ', which is defined in terms ϵ we get from the above expression that G' spans some $\delta_{\mathcal{F}}(\epsilon)n^{|V(F)|}$. As G' is a subgraph of G, G contains at least this many copies of F. ■

Let us try to apply the argument given in the above proof when $\mathcal{F}$ is an infinite set. The main problem is that there is no upper-bound on the size of the graphs in $\mathcal{F}$. Thus, we do not know a priori the size of the member of $\mathcal{F}$ that we will eventually find in the equipartition that Lemma 3.4 returns. After finding $F \in \mathcal{F}$ in an equipartition, we may find out that F is too large for Lemma 3.2 to be applied, because Lemma 3.4 was not used with a small enough γ. One may then try to find a new equipartition based on the size of F. However, that requires using a smaller γ, and thus the new equipartition may be larger (that is, contain more partition classes), and thus contain only larger members of $\mathcal{F}$. Hence, even the new γ is not good enough in order to apply Lemma 3.2. This leads to a circular definition of constants, which seems unbreakable. The main tool in the proof of Theorem 4.3 is Lemma 3.7 below, proved in [4] with a different motivation, which enables one to break this circular chain of definitions. This lemma can be considered a variant of the standard regularity lemma, where one can use a *function* that defines γ as a function of the size of the partition, rather then having to use a *fixed* γ as in Lemma 3.4.

Lemma 3.7 ([4]) *For every integer m and monotone non-increasing function $\mathcal{E}(r) : \mathbb{N} \mapsto (0,1)$ there is an integer $S = S_{3.7}(m, \mathcal{E})$ satisfying the following: For any graph G on $n \geq S$ vertices, there exist an equipartition $\mathcal{A} = \{V_i \mid 1 \leq i \leq k\}$ of $V(G)$ and an induced subgraph U of G, with an equipartition $\mathcal{B} = \{U_i \mid 1 \leq i \leq k\}$ of the vertices of U, that satisfy:*

1. *$m \leq k \leq S$.*
2. *$U_i \subseteq V_i$ for all $i \geq 1$, and $|U_i| \geq n/S$.*
3. *In the equipartition $\mathcal{B}$,* **all** *pairs are $\mathcal{E}(k)$-regular.*
4. *All but at most $\mathcal{E}(0)\binom{k}{2}$ of the pairs $1 \leq i < j \leq k$ are such that $|d(V_i, V_j) - d(U_i, U_j)| < \mathcal{E}(0)$.*

The dependency of the function $T_{3.3}(m, \gamma)$ on γ is a tower of exponents of height polynomial in $1/\gamma$ (see the proof in [45]). As one can infer from the details in [4], this implies that even for moderate functions $\mathcal{E}$ the integer $S(m, \mathcal{E})$ grows as fast as a tower of towers of exponents (see the proof in [4]).

4 Testing Monotone Graph Properties

The problem of characterizing the testable graph properties is widely considered to be the most important open problem in the area of property testing. A natural step in this direction is in showing that large and natural families of graph properties are testable. In this section we consider the family of monotone graph properties. A graph property is *monotone* if it is closed under removal of vertices and edges, or equivalently if it is closed under taking (not necessarily induced) subgraphs. Various monotone graph properties were extensively studied in graph theory. As examples of monotone properties one can consider the property of having a homomorphism to a fixed graph H (which includes as a special case the property of being k-colorable, see Definition 1.1), and the property of not containing a (not necessarily induced) copy of some fixed graph H. Another monotone property is being $(k, \mathcal{H})$-*Ramsey*: For a (possibly infinite) family of graphs $\mathcal{H}$, a graph is said to be $(k, \mathcal{H})$-Ramsey if one can color its edges using k colors, such that no color class contains a copy of a graph $H \in \mathcal{H}$. This property is the main focus of Ramsey-Theory, see [38] and its references. As another

example, one can consider the property of being $(k, \mathcal{H}, f)$-Multicolorable; For a (possibly infinite) family of graphs $\mathcal{H}$ and a function f from $\mathcal{H}$ to the positive integers, a graph is said to be $(k, \mathcal{H}, f)$-Multicolorable if one can color its edges using k colors, such that every copy of a graph $H \in \mathcal{H}$ receives at least $f(H)$ colors. See [27], [22] and their references for a discussion of some special cases. Another set of well studied monotone properties are those defined by having a bounded *fractional chromatic number*, bounded *vector chromatic number* (see [43]) or bounded *Lovász theta function* (see [48]). The abstract family of monotone graph properties has also been extensively studied in graph theory. See [30], [15], [14] and their references. The main focus of this section is in presenting the main ideas behind the proof of the following result:

Theorem 4.1 ([10]) *Every monotone graph property is testable with one-sided error.*

It is important to note that the main result of [10] is slightly weaker than the statement given in Theorem 4.1 as the monotone property has to satisfy some technical conditions (which cannot be avoided). Roughly speaking, the main problem is that for some properties it may be impossible (that is, non-recursive) given ϵ, to compute the number of queries the tester should perform in order to test the monotone property. See Section 6 for the precise statement and discussion of this issue. In this section we will overlook this issue and focus on the combinatorial part of the problem. Another important note is that in [35], Goldreich and Trevisan define a monotone graph property to be one that is closed under removal of edges, and not necessarily under removal of vertices. They show that there are such properties that are not testable even with two sided error. In fact, their result is stronger as the property they define belongs to NP and requires query complexity $\Omega(n^2)$. This means that Theorem 4.1 cannot be extended, in a strong sense, to properties that are only closed under removal of edges. As we show in Section 7, Theorem 4.1 can be extended to properties, which are closed only under removal of vertices, namely, hereditary properties.

We next introduce a convenient way of handling a monotone graph property.

Definition 4.2 (Forbidden Subgraphs) *For a monotone graph property $\mathcal{P}$, define $\mathcal{F} = \mathcal{F}_{\mathcal{P}}$ to be the set of graphs which are minimal with respect to not satisfying property $\mathcal{P}$. In other words, a graph F belongs to $\mathcal{F}$ if it does not satisfy $\mathcal{P}$, but any graph obtained from F by removing an edge or a vertex, satisfies $\mathcal{P}$.*

As an example of a family of forbidden subgraphs, observe that if $\mathcal{P}$ is the property of being 2-colorable, then $\mathcal{F}_{\mathcal{P}}$ is the set of all odd-cycles. Clearly, a graph satisfies $\mathcal{P}$ if and only it contains no member of $\mathcal{F}_{\mathcal{P}}$ as a (not necessarily induced) subgraph. We say that a graph is $\mathcal{F}$-free if it contains no (not necessarily induced) subgraph $F \in \mathcal{F}$. Clearly, for any family $\mathcal{F}$, being $\mathcal{F}$-free is a monotone property. Thus, the monotone properties are precisely the graph properties of being $\mathcal{F}$-free for some family $\mathcal{F}$. In order to simplify the notation of this section, it will be simpler to talk about properties of type $\mathcal{F}$-free rather than monotone properties. The reader should keep in mind that we allow $\mathcal{F}$ to be an infinite set. By Lemma 3.5 and the above discussion we can prove Theorem 4.1 by proving the following:

Theorem 4.3 *For any (possibly infinite) family of graphs $\mathcal{F}$ there are functions $f_{\mathcal{F}}(\epsilon)$ and $\delta_{\mathcal{F}}(\epsilon)$ satisfying the conditions of Lemma 3.5.*

We remind the reader that as was shown in Section 3, the standard regularity-lemma can be applied in order to prove Theorem 4.3 for any *finite* family $\mathcal{F}$. In order to prove Theorem 4.3 for infinite families of graphs we apply the strengthening of the regularity-lemma, which is stated in Lemma 3.7. A sketch of the proof of Theorem 4.3 is given in Subsection 4.1 below. Before getting into the details we discuss some interesting applications of Theorem 4.3. First, observe that an immediate corollary of this theorem is the following:

Corollary 4.4 ([10]) *For every monotone graph property $\mathcal{P}$, there is a function $W_{\mathcal{P}}(\epsilon)$ with the following property: If G is ϵ-far from satisfying $\mathcal{P}$, then G contains a subgraph of size at most $W_{\mathcal{P}}(\epsilon)$, which does not satisfy $\mathcal{P}$.*

The above corollary significantly extends a result of Rödl and Duke [55], conjectured by Erdős, which asserts that the above statement holds for the k-colorability property. Corollary 4.4 extends this result to any monotone property and in particular to the monotone properties discussed at the beginning of this section.

As the details of the proof of Theorem 4.3 reveal the functions $f_{\mathcal{F}}(\epsilon)$ and $\delta_{\mathcal{F}}(\epsilon)$ depend on the family of graphs $\mathcal{F}$. This means, that for any monotone graph property $\mathcal{P}$ the upper bounds for testing $\mathcal{P}$, which Theorem 4.1 guarantees depend on the specific property being tested. A natural question one may ask, is if the dependency on the specific property being tested can be removed. One can rule out this possibility by proving the following.

Theorem 4.5 ([10]) *For any function $Q : (0,1) \mapsto \mathbb{N}$, there is a monotone graph property $\mathcal{P}$, which cannot be tested with one-sided error and query complexity $o(Q(\epsilon))$.*

Prior to [10], the best lower bound proved for testing a testable graph property with one-sided error was obtained in [1], where it is shown that for every non-bipartite graph H, the query complexity of testing whether a graph does not contain a copy of H is at least $(1/\epsilon)^{\Omega(\log 1/\epsilon)}$ (see Section 8). The fact that for every H this property is testable with one-sided error, follows from [3] and [4], and also as a special case of Theorem 4.1. As by Theorem 4.1 every monotone graph property is testable with one-sided error, Theorem 4.5 establishes that the one-sided error query complexity of testing testable graph properties, even those that are testable with one-sided error, may be *arbitrarily large*. We finally note that the proof of Theorem 4.5 implies a similar statement with respect to Corollary 4.4.

Another application of Theorem 4.3 can be considered a compactness-type result in property testing. Suppose $\mathcal{P}_1, \ldots, \mathcal{P}_k$ are k graph properties that are closed under removal of edges. It is clear that if a graph G is ϵ-far from satisfying these k properties then it is at least ϵ/k-far from satisfying at least one of them. However, it is not clear that there is a fixed $\delta > 0$ such that even if $k \to \infty$, G must be δ-far from satisfying one of these properties. By using Theorem 4.3 one can prove that if these properties are monotone then such an δ exists. It can also be shown that in general, no such δ exists. This is stated in the following theorem, see [10].

Theorem 4.6 ([10]) *For any (possibly infinite) set of monotone graph properties $\mathcal{P} = \{\mathcal{P}_1, \mathcal{P}_2, \ldots\}$, there is a function $\delta_{\mathcal{P}} : (0,1) \mapsto (0,1)$ with the following property: If a graph G is ϵ-far from satisfying all the properties of $\mathcal{P}$, then for some i, the graph G is $\delta_{\mathcal{P}}(\epsilon)$-far from satisfying $\mathcal{P}_i$. Furthermore, there are properties $\mathcal{P} = \{\mathcal{P}_1, \mathcal{P}_2, \ldots\}$, which are closed under removal of edges for which no such $\delta_{\mathcal{P}}$ exists.*

We turn to some interesting open problems, which the above results suggest. As was mentioned above, a result of Goldreich and Trevisan [35] rules out the possibility of extending Theorem 4.1 to graph properties that are only closed under removal of edges. It seems interesting to bridge the gap between their result and Theorem 4.1 by resolving the following problem.

Open Problem 4.7 *Characterize the testable graph properties that are closed under edge removal.*

It also seems interesting to see if the new powerful hypergraph versions of the regularity lemma (see [39], [51] and [56]) can be used to obtain hypergraph versions of Lemma 3.7, and if in that case, one can obtain hypergraph versions of Theorem 4.1 and Corollary 4.4.

Open Problem 4.8 *Prove hypergraph versions of Theorem 4.1 and Corollary 4.4.*

It may also be interesting to extend Theorem 4.5 by showing the following:

Open Problem 4.9 *Prove that for any function $Q : (0,1) \mapsto \mathbb{N}$, there is a monotone graph property $\mathcal{P}$, which cannot be tested (either with one-sided or two-sided error) with query complexity $o(Q(\epsilon))$.*

4.1 Sketch of the proof of Theorem 4.3

In this subsection we give an overview of the proof of Theorem 4.3. The reader is referred to [10] for the full proof. Though this result may seem to have nothing to do with homomorphisms, the key idea in the proof is in using a certain graph functional that involves graph homomorphisms.

We first need to introduce another standard notion frequently used in applications of the regularity-lemma. For an equipartition of a graph G, let the *regularity graph* of G, denoted $R = R(G)$, be the following graph: We first use Lemma 3.4 in order to obtain the equipartition satisfying the assertions of the lemma. Let k be the size of the equipartition. Then, R is a graph on k vertices, where vertices i and j are connected if and only if (V_i, V_j) is γ-regular and $d(V_i, V_j) \geq \gamma$. In some sense, the regularity graph is an approximation of the original graph, up to γn^2 modifications. One of the main (implicit) implications of the regularity lemma is the following: Suppose we consider two graphs to be *similar* if their regularity graphs are isomorphic. It thus follows from Lemma 3.4 that for every $\gamma > 0$, the number of graphs that are pairwise non-similar is bounded by a function of γ, which is roughly $2^{\binom{T}{2}}$, where $T = T_{3.4}(1/\gamma, \gamma)$. In other words, up to γn^2 modifications, all the graphs can be approximated using a set of equipartitions of size bounded by a function of γ only. One (easy) application of this observation is that there are $2^{(\frac{1}{4}+o(1))n^2}$ triangle-free graphs on n vertices. The reader is referred to [26] where this interpretation of the regularity lemma is further (implicitly) used. This leads us to the key definitions of the proof of Theorem 4.3. The reader should think of the graphs R considered below as the set of regularity graphs discussed above, and the parameter r as representing the size of R.

Definition 4.10 (The family $\mathcal{F}_r$) *For any (possibly infinite) family of graphs $\mathcal{F}$, and any integer r let $\mathcal{F}_r$ be the following set of graphs: A graph R belongs to $\mathcal{F}_r$ if it has at most r vertices and there is at least one $F \in \mathcal{F}$ such that $F \mapsto R$.*

Practicing definitions, observe that if $\mathcal{F}$ is the family of odd cycles, then $\mathcal{F}_r$ is precisely the family of non-bipartite graphs of size at most r. In the proof of Theorem 4.3, the set $\mathcal{F}_r$ will represent a

subset of the regularity graphs of size at most r. Namely, those R for which there is at least one $F \in \mathcal{F}$ such that $F \mapsto R$. As r will be a function of ϵ only, and thus finite, we can take the maximum over all the graphs $R \in \mathcal{F}_r$, of the size of the smallest $F \in \mathcal{F}$ such that $F \mapsto R$. We thus define

Definition 4.11 (The function $\Psi_{\mathcal{F}}$) *For a family of graphs $\mathcal{F}$ and an integer r for which $\mathcal{F}_r \neq \emptyset$, let*

$$\Psi_{\mathcal{F}}(r) = \max_{R \in \mathcal{F}_r} \min_{\{F \in \mathcal{F}: F \mapsto R\}} |V(F)|. \tag{1}$$

Define $\Psi_{\mathcal{F}}(r) = 0$ if $\mathcal{F}_r = \emptyset$. Therefore, $\Psi_{\mathcal{F}}(r)$ is monotone non-decreasing in r.

Practicing definitions again, note that if $\mathcal{F}$ is the family of odd cycles, then $\Psi_{\mathcal{F}}(r) = r$ when r is odd, and $\Psi_{\mathcal{F}}(r) = r-1$ when r is even. The "right" way to think of the function $\Psi_{\mathcal{F}}$ is the following: Let R be a graph of size at most r and suppose we are guaranteed that there is a graph $F' \in \mathcal{F}$ such that $F' \mapsto R$ (thus $R \in \mathcal{F}_r$). Then, by this information only and *without* having to know the structure of R itself, the definition of $\Psi_{\mathcal{F}}$ implies that there is a graph $F \in \mathcal{F}$ of size at most $\Psi_{\mathcal{F}}(r)$, such that $F \mapsto R$.

The function $\Psi_{\mathcal{F}}$ has a critical role in the proof of Theorem 4.3. The first usage of this function is that as by Lemma 3.7 we can upper bound the size of the regularity graph R, we can also upper bound the size of the smallest graph $F \in \mathcal{F}$ for which $F \mapsto R$. A second important property of $\Psi_{\mathcal{F}}$ is discussed in Section 6. As we have mentioned in the previous section, the main difficulty that prevents one from proving Theorem 4.3 using Lemma 3.2 is that one does not know a priori the size of the graph that one may expect to find in the equipartition. This leads us to define the following function where $0 < \epsilon < 1$ is an arbitrary real.

$$\mathcal{E}(r) = \begin{cases} \epsilon, & r = 0 \\ \gamma_{3.2}(\epsilon,\ r,\ \Psi_{\mathcal{F}}(r)), & r \geq 1 \end{cases} \tag{2}$$

In simple words, given $r > 0$, which will represent the size of the equipartition and thus also the size of the regularity graph which it defines, $\mathcal{E}(r)$ returns "how regular" this equipartition should be in order to allow one to find many copies of the *largest* graph one may possibly have to work with. Note, that we obtain the upper bound on the size of this largest possible graph, by invoking $\Psi_{\mathcal{F}}(r)$. As for different families of graphs $\mathcal{F}$, the function $\Psi_{\mathcal{F}}(r)$ may behave differently, $\mathcal{E}(r)$ may also behave differently for different families $\mathcal{F}$, as it is defined in terms of $\Psi_{\mathcal{F}}(r)$. However, and this is one of the key points of the proof, as we are fixing the family of graphs $\mathcal{F}$, the function $\mathcal{E}(r)$ depends only on r and implicitly on ϵ.

Given the above definitions we apply Lemma 3.7 with $\mathcal{E}(r)$ in order to obtain an equipartition of G. We then throw away edges that reside inside the sets V_i and between (V_i, V_j) whose edge density is either small or differs significantly from that of (U_i, U_j) (similar, but not identical, to what we did in Lemma 3.4). We then argue that we thus throw away less than ϵn^2 edges. As G is by assumption ϵ-far from not containing a member of $\mathcal{F}$, the new graph still contains a copy of $F \in \mathcal{F}$. By the definition of the new graph, it thus means that there is a (natural) homomorphism from F to the regularity graph of G. We then arrive at the main step of the proof, where we use the key property of Lemma 3.7, item (3), and the definition of $\mathcal{E}(r)$ to get that the sets U_i are regular enough to let us use Lemma 3.2 on them and to infer that they span many copies of F.

5 Approximation Algorithms for Edge-Deletion Problems

The main topic of this section is graph modification problems, namely problems of the type: "given a graph G, find the smallest number of modifications that are needed in order to turn G into a graph satisfying property $\mathcal{P}$". The main two types of such problems are the following; In *node modification* problems, one tries to find the smallest set of vertices, whose removal turns G into a graph satisfying $\mathcal{P}$, while in *edge modification* problems, one tries to find the smallest number of edge deletions/additions, which turn G into a graph satisfying $\mathcal{P}$. In this section we will focus on edge-modification problems. Note, that when trying to turn a graph into one satisfying a monotone property we will only use edge deletions. Therefore, in these cases the problem is sometimes called *edge-deletion* problem. Before continuing, we need to introduce some notations. For a graph property $\mathcal{P}$, let $\mathcal{P}_n$ denote the set of graphs on n vertices, which satisfy $\mathcal{P}$. Given two graphs on n vertices, G and G', we denote by $\Delta(G, G')$ the edit distance between G and G', namely the smallest number of edge additions and/or deletions that are needed in order to turn G into G'. For a given property $\mathcal{P}$, we want to denote how far a graph G is from satisfying $\mathcal{P}$. For notational reasons it will be more convenient to normalize this measure so that it is always in the interval $[0, 1]$ (actually $[0, \frac{1}{2}]$). We thus define

Definition 5.1 ($E_{\mathcal{P}}(G)$) *For a graph property $\mathcal{P}$ and a graph G on n vertices, let*

$$E_{\mathcal{P}}(G) = \min_{G' \in \mathcal{P}_n} \frac{\Delta(G, G')}{n^2} .$$

In words, $E_{\mathcal{P}}(G)$ is the minimum edit distance of G to a graph satisfying $\mathcal{P}$ after normalizing it by a factor of n^2. The main result discussed in the previous section, Theorem 4.1, can be rephrased as follows: For any monotone property $\mathcal{P}$, one can distinguish between graphs satisfying $E_{\mathcal{P}}(G) = 0$, from those satisfying $E_{\mathcal{P}}(G) > \epsilon$. In this section we consider the natural extension of this problem, which asks for actually computing $E_{\mathcal{P}}(G)$, or to at least approximate it.

Graph modification problems have been studied extensively. Already in 1979, Garey and Johnson [31] mentioned 18 types of vertex and edge modification problems. Graph modification problems were extensively studied as these problems have applications in several fields, including Molecular Biology and Numerical Algebra. In these applications a graph is used to model experimental data, where edge modifications correspond to correcting errors in the data: Adding an edge means correcting a false negative, while deleting an edge means correcting a false positive. Computing $E_{\mathcal{P}}(G)$ for appropriately defined properties $\mathcal{P}$ have important applications in physical mapping of DNA (see [21], [32] and [37]). Computing $E_{\mathcal{P}}(G)$ for other properties arises when optimizing the running time of performing Gaussian elimination on a sparse symmetric positive-definite matrix (see [58]). Other modification problems arise as subroutines for heuristic algorithms for computing the largest clique in a graph (see [62]). Some edge modification problems also arise naturally in optimization of circuit design [23]. We briefly mention that there are also many results about *vertex* modification problems, notably that of Lewis and Yannakakis [47], who proved that for any nontrivial hereditary property $\mathcal{P}$, it is NP-hard to compute the smallest number of vertex deletions, which turn a graph into one satisfying $\mathcal{P}$.

As in Section 4 it will be simpler to consider a monotone property via its family of forbidden subgraphs $\mathcal{F}$ (see Definition 4.2). Therefore, $E_{\mathcal{F}}(G)$ will denote $E_{\mathcal{P}}(G)$, where $\mathcal{P}$ is the property of being $\mathcal{F}$-free. The main idea behind the algorithm for approximating $E_{\mathcal{F}}(G)$ is very simple: Given

G and ϵ we would like to construct a small weighted (complete) graph W, of size depending on ϵ only, such that $E_{\mathcal{F}}(G)$ is close to some natural function of W. Surprisingly, again, this function is related to homomorphisms. We need the following definitions:

Definition 5.2 ($\mathcal{F}$-homomorphism-free) *For a family of graphs $\mathcal{F}$, a graph W is called $\mathcal{F}$-homomorphism-free if $F \not\mapsto W$ for any $F \in \mathcal{F}$.*

Definition 5.3 ($\mathcal{H}_{\mathcal{F}}(W)$) *For a family of graphs $\mathcal{F}$ and a weighted complete graph W of size k, let $\mathcal{H}'_{\mathcal{F}}(W)$ denote the minimum total weight of a set of edges, whose removal from W turns it into an $\mathcal{F}$-homomorphism-free graph. Let $\mathcal{H}_{\mathcal{F}}(W) = \frac{1}{k^2}\mathcal{H}'_{\mathcal{F}}(W)$.*

Note, that in Definition 5.2 the graph W is an unweighed not necessarily complete graph. The following is one of the main (implicit) results of [13]:

Theorem 5.4 ([13]) *For any family of graphs $\mathcal{F}$ and any $\epsilon > 0$ there is a deterministic $O(|V|+|E|)$ time algorithm with the following property: Given any graph $G = (V, E)$, the algorithm constructs a complete weighted graph W, of size $c = c(\epsilon, \mathcal{F})$ such that*

$$|E_{\mathcal{F}}(G) - \mathcal{H}_{\mathcal{F}}(W)| \leq \epsilon \ .$$

After obtaining the graph W using the above theorem, we can use exhaustive search in order to precisely compute $\mathcal{H}_{\mathcal{F}}(W)$. Note, that the time needed to precisely compute $\mathcal{H}_{\mathcal{F}}(W)$ is only a function of ϵ, because $|V(W)|$ is independent of the size of the input graph. This implies the following:

Corollary 5.5 ([13]) *For any fixed $\epsilon > 0$ and any monotone property $\mathcal{P}$ there is a* **deterministic** *algorithm that given a graph $G = (V, E)$ computes in time $O(|V| + |E|)$ a real E satisfying $|E - E_{\mathcal{P}}(G)| \leq \epsilon$.*

Observe, that the running time of the algorithm on an n vertex dense graph is of type $f(\epsilon)n^2$. We note that Corollary 5.5 was not known for many monotone properties. In particular, such an approximation algorithm was not even known for the property of being triangle-free (and more generally for the property of being H-free for any non-bipartite H).

Theorem 5.4 is proved in [13] via a novel structural graph theoretic technique, whose proof is not included here. We just mention that it uses some of the ideas used to prove Theorem 4.1 along with several additional ideas. It is also the first result to make a non-trivial application of the algorithmic version of the regularity lemma stated in Lemma 3.7. An additional interesting application of the structural result of [13] is the following concentration-type result regarding the edit distance of small subgraphs of a graph.

Theorem 5.6 ([13]) *For every $\epsilon > 0$ and any monotone property $\mathcal{P}$ there is $r = r(\epsilon, \mathcal{P})$ with the following property: Let G be any graph, and suppose we randomly pick a subset R, of r vertices from $V(G)$. Denote by G' the graph induced by G on R. Then,*

$$Prob[\ |E_{\mathcal{P}}(G') - E_{\mathcal{P}}(G)| > \epsilon] < \epsilon \ .$$

Again, once we have G', whose size depends only on ϵ, we can use exhaustive search in order to precisely compute $E_{\mathcal{P}}(G')$. Therefore, an immediate implication of the above theorem is the following extension of Theorem 4.1:

Corollary 5.7 ([13]) *For every fixed $\epsilon > 0$ and any monotone property $\mathcal{P}$ there is a* **randomized** *algorithm, which given a graph G computes in time $O(1)$ a real E satisfying $|E - E_{\mathcal{P}}(G)| \leq \epsilon$ with high probability.*

In standard Property-Testing one wants to distinguish between the graphs G that satisfy a certain graph property $\mathcal{P}$, or equivalently those G for which $E_{\mathcal{P}}(G) = 0$, from those that satisfy $E_{\mathcal{P}}(G) > \epsilon$. Parnas, Ron and Rubinfeld [53] introduced the notion of Tolerant Property-Testing, where one wants to distinguish between the graphs G that satisfy $E_{\mathcal{P}}(G) < \delta$ from those that satisfy $E_{\mathcal{P}}(G) > \epsilon$, where $0 \leq \delta < \epsilon \leq 1$ are some constants. Recently, there have been several results in this line of work. Specifically, Fischer and Newman [29] have recently shown that if a graph property is testable with number of queries depending on ϵ only, then it is also tolerantly testable for any $0 \leq \delta < \epsilon \leq 1$ and with query complexity depending on $|\epsilon - \delta|$. Combining this with Theorem 4.1 implies that any monotone property is tolerantly testable for any $0 \leq \delta < \epsilon \leq 1$ and with query complexity depending on $|\epsilon - \delta|$. Note, that Corollary 5.7 implicitly states the same. In fact, the algorithm implied by Corollary 5.7 is the "natural" one, where one picks a random subset of vertices S, and approximates $E_{\mathcal{P}}(G)$ by computing $E_{\mathcal{P}}$ on the graph induced by S. The algorithm of [29] is far more complicated. Furthermore, due to the nature of our algorithm if the input graph satisfies a monotone property $\mathcal{P}$, namely if $E_{\mathcal{P}}(G) = 0$, we will always detect that this is the case. The algorithm of [29] may declare that $E_{\mathcal{P}}(G) > 0$ even if $E_{\mathcal{P}}(G) = 0$.

Theorem 5.4 implies that it is possible to efficiently approximate the distance of an n vertex graph from any monotone graph property $\mathcal{P}$, to within an error of ϵn^2 for any $\epsilon > 0$. A natural question one can ask is for which monotone properties it is possible to improve the additive error to $n^{2-\delta}$ for some fixed $\delta > 0$. In the terminology of Definition 5.1, this means to approximate $E_{\mathcal{P}}$ to within an additive error of $n^{-\delta}$ for some $\delta > 0$ in polynomial time. In [13], an essentially precise characterization of the monotone graph properties for which such a $\delta > 0$ exists, was given.

Theorem 5.8 ([13]) *Let $\mathcal{P}$ be a monotone graph property. Then,*

1. *If there is a bipartite graph that does not satisfy $\mathcal{P}$, then there is a fixed $\delta > 0$ for which it is possible to approximate $E_{\mathcal{P}}$ to within an additive error of $n^{-\delta}$ in polynomial time.*
2. *On the other hand, if all bipartite graphs satisfy $\mathcal{P}$, then for any fixed $\delta > 0$ it is NP-hard to approximate $E_{\mathcal{P}}$ to within an additive error of $n^{-\delta}$.*

While the first part of the above theorem follows easily from the known results about the Turán numbers of bipartite graphs (see, e.g., [61]), the proof of the second item involves various combinatorial tools including Szemerédi's Regularity Lemma, a new result in Extremal Graph Theory that extends the main result of [20] and [16], and the basic approach of [2]. The proof of Theorem 5.8 is not included here, and the reader is referred to [13] for the details.

For a fixed graph H, let $\mathcal{P}_H$ denote the property of being H-free. Note, that the above theorem implies that for any non-bipartite H, computing $E_{\mathcal{P}_H}$ is NP-hard. Observe, that this does not hold for the family of bipartite graphs H, as for some bipartite graphs such as the single edge, computing $E_{\mathcal{P}_H}$ can be done in polynomial time. It may thus be interesting to consider the following problem:

Open Problem 5.9 *Characterize the bipartite graphs H, for which computing $E_{\mathcal{P}_H}$ is NP-hard.*

6 Uniform versus Non-Uniform Property Testing

One of the fundamental problems of complexity theory is in understanding the relations between various models of computation. In particular, one would like to know if two models are equivalent or if there are problems, which can be solved in one model but not in the other. Regretfully, in many interesting cases, though it seems intuitively obvious that two models of computation are not equivalent, the current techniques are far from enabling one to formally prove that. In this section we discuss two models of property-testing, which were introduced in [11]. Surprisingly, in this case, though it seems at first that these models are equivalent, the main result of [11] shows that it is possible to formally prove that they are in fact distinct (without making any hardness-type assumptions). The proof of this result relies heavily on the notion of graph homomorphism.

In defining a tester in Section 1 we did not mention, whether the error parameter ϵ is given as part of the input of the tester, or whether the tester is designed to distinguish between graphs that satisfy $\mathcal{P}$ from those that are ϵ-far from satisfying it, when ϵ is a known fixed constant. Prior to [11] the literature about property testing was not clear about this issue as in some papers ϵ was assumed to be a part of the input while in others it is not. In [11], the following two notions of property-testing were introduced:

Definition 6.1 (Uniformly testable) *A graph property $\mathcal{P}$ is* uniformly testable *if there is a tester that given ϵ, can distinguish with probability 2/3 between graphs that satisfy $\mathcal{P}$ and those that are ϵ-far from satisfying it, and whose query complexity is a function of ϵ only*[3]*. $\mathcal{P}$ is uniformly testable with one-sided error if the above tester accepts with probability 1 any graph satisfying $\mathcal{P}$.*

Definition 6.2 (Non-uniformly testable) *A graph property $\mathcal{P}$ is* non-uniformly testable *if for every fixed ϵ there is a tester that can distinguish with probability 2/3 between graphs that satisfy $\mathcal{P}$ and those that are ϵ-far from satisfying it, and whose query complexity is a constant (that is, a function of ϵ only). $\mathcal{P}$ is non-uniformly testable with one-sided error if each of the above testers accepts with probability 1 any graph satisfying $\mathcal{P}$.*

Note, that in the above two definitions the testers are not given the size of the input. This type of oblivious testers were defined and studied in [12], where it was observed that all natural properties can be tested by testers that do not accept the size of the input. Therefore, for natural properties it seems that these testers are as powerful as testers accepting the size of the graph as part of the input. See Section 7 for more details on these slightly restricted testers.

It may seem, at least at first glance, that uniformly and non-uniformly property-testing are identical notions. The reader should note that the difference between being uniformly testable and non-uniformly testable, is not as sharp as, say, the difference between P and $P/Poly$ (see [52]). The reason is that in $P/Poly$ the non-uniformity is with respect to the *inputs*, while in our case the non-uniformity is over the *error parameter*. In particular, a non-uniform tester for some fixed ϵ should be able to handle *any* input graph. The main result of [11] shows that the above two notions of

[3]Note, that here and in the following definition we require that the query complexity of the tester will not only be *upper bounded* by a function of ϵ (as defined in Section 1), but actually be a function of ϵ only.

property testing are in fact distinct. Moreover, these notions are shown to be distinct while confining to graph properties, which are natural with respect to both their combinatorial structure and their computational difficulty. This is stated in the following result:

Theorem 6.3 ([11]) *There is a graph property $\mathcal{P}$ with the following properties:*

1. *$\mathcal{P}$ can be non-uniformly tested with one-sided error.*
2. *$\mathcal{P}$ cannot be uniformly tested, even with two-sided error.*

Moreover, satisfying $\mathcal{P}$ belongs to coNP and can be expressed in terms of forbidden subgraphs.

The property $\mathcal{P}$, which is constructed in [11] in order to prove Theorem 6.3, is simply the property of being $\mathcal{F}$-free for some carefully defined family of graphs $\mathcal{F}$. We next give an overview of the proof of Theorem 6.3. The proof of this theorem heavily relies on the main result about testing monotone graph properties discussed in Section 4. As we have mentioned in Section 4 the actual result obtained in [10] is slightly weaker than what was stated in Theorem 4.1. We are now ready to state the precise result of [10]. In the statement we refer to the function $\Psi_{\mathcal{F}}$ defined in Definition 4.11. We also call a function *recursive* if there is an algorithm for computing it in finite time (see [52]).

Theorem 6.4 ([10]) *For every (possibly infinite) family of graphs $\mathcal{F}$, the property of being $\mathcal{F}$-free is non-uniformly testable with one-sided error. Moreover, if $\Psi_{\mathcal{F}}$ is recursive then being $\mathcal{F}$-free is also uniformly testable with one-sided error.*

The proof of Theorem 6.3 consists of two steps. The first establishes the somewhat surprising fact, that the property of $\Psi_{\mathcal{F}}(k)$ being recursive is not only sufficient for inferring that being $\mathcal{F}$-free is uniformly testable (as is stated in Theorem 6.4), but this condition is also necessary. This is formulated in the following Theorem.

Theorem 6.5 ([10]) *Suppose $\mathcal{F}$ is a family of graphs for which the function $\Psi_{\mathcal{F}}$ is not recursive. Then, the property of being $\mathcal{F}$-free cannot be uniformly tested with one-sided error.*

Note, that in Definition 6.1 the tester is defined as one that may have arbitrarily large query complexity, as long as it is a function of ϵ only. Hence, if $\Psi_{\mathcal{F}}$ is not recursive Theorem 6.5 rules out the possibility of designing a tester that receives ϵ as part of the input, no matter how large its query complexity is, as long as it is a function of ϵ only.

The main idea behind the proof of Theorem 6.5 is that by "inspecting" the behavior of a property tester for the property of being $\mathcal{F}$-free one can compute the function $\Psi_{\mathcal{F}}$. The main combinatorial tool in the proof of Theorem 6.5 is a Theorem of Erdős [25] in extremal graph theory, which can be considered as a hypergraph version of Zarankiewicz problem [46]. As an immediate corollary of Theorems 4.1 and 6.5 we obtain the following result, which *precisely* characterizes the families of graphs $\mathcal{F}$, for which the property of being $\mathcal{F}$-free can be tested uniformly (recall that by Theorem 4.1, for *any* family $\mathcal{F}$, being $\mathcal{F}$-free is non-uniformly testable).

Corollary 6.6 *For every family of graphs $\mathcal{F}$, the property of being $\mathcal{F}$-free is uniformly testable with one-sided error if and only if the function $\Psi_{\mathcal{F}}$ is recursive.*

As for any monotone graph property $\mathcal{P}$, there is a family of graphs $\mathcal{F}$, for which $\mathcal{P}$ is equivalent to being $\mathcal{F}$-free (see Section 4) the above result also gives A *precisely* characterization of the monotone graph properties that can be tested uniformly.

An immediate consequence of Theorems 4.1 and 6.5 is that in order to separate uniform testing with one-sided error from non-uniform testing with one-sided error, and thus (almost) prove Theorem 6.3, it is enough to construct a family of graphs $\mathcal{F}$ with the following two properties: (i) There is an algorithm for deciding whether a graph F belongs to $\mathcal{F}$ (recall that we confine ourselves to decidable graph properties). (ii) The function $\Psi_{\mathcal{F}}$ is non-recursive. The main combinatorial ingredient in the construction of $\mathcal{F}$ is the fundamental theorem of Erdős [24] in extremal graph theory, which guarantees the existence of graphs with arbitrarily large girth and chromatic number. As we want to prove Theorem 6.3 with a graph property, which is not only decidable, but even belongs to $coNP$, we need explicit constructions of such graphs. To this end, we use explicit constructions of expanders, which are given in [49]. For the construction we also apply some ideas from the theory of recursive functions. Finally, in order to obtain that being $\mathcal{F}$-free cannot be tested even with two-sided error, we use a result of the first author ([35] Appendix D) about testing hereditary graph properties. The full details appear in [11].

A final remark with regards to Theorem 6.3 is that as noted above, the testers considered in this theorem are not given the size of the input graph. It can be shown that if an oblivious and non-oblivious testers are given the size of the input then the notions of uniform and non-uniform testers coincide, namely, a graph property is uniformly testable if and only if it is non-uniformly testable. This observation combined with Theorem 6.3 can be considered as showing that a tester can use the size of the input in order to perform natural tasks, which it cannot do if it is not given this information.

7 Testing Hereditary Graph Properties

In this section we consider a family of graph properties, which significantly extends the family of monotone graph properties considered in Section 4. Besides including many additional interesting graph properties, this family can be proved to contain all the "natural" graph properties that can be tested with one-sided error. A sketch of the details follows.

A graph property is *hereditary* if it is closed under removal of vertices (and not necessarily under removal of edges). Clearly, every monotone graph property is also hereditary, but there are also many well-studied hereditary properties, which are not monotone. Examples are being Perfect, Chordal, Interval, Comparability, Permutation and more. See [36], [50] and [54] for definitions and results about these as well as about other hereditary properties. The main results discussed in the previous sections deal with special cases of hereditary properties, namely, monotone properties. All the results about monotone properties heavily rely on the fact that removing an edge from a graph cannot increase its distance from satisfying a monotone property (see, e.g., Lemma 3.4). As it turns out, handling hereditary non-monotone graph properties, such as being Perfect or not containing an induced cycle of length 4, is significantly more involved than handling monotone properties. In Section 3 we have demonstrated that it is easy to show that for any *finite* $\mathcal{F}$ the property of being $\mathcal{F}$-free is testable. It is far more complicated to show that even for any single graph H, the property of being *induced* H-free is testable. This was only proved in [4] by applying Lemma 3.7.

For a (possibly infinite) family of graphs $\mathcal{F}$, a graph G is said to be *induced* $\mathcal{F}$-free if it contains

no $F \in \mathcal{F}$ as an induced subgraph. As in the case of monotone properties, one can use essentially the same argument used in Lemma 3.5 in order to reduce the task of testing the property of being induced $\mathcal{F}$-free to a purely combinatorial problem.

Lemma 7.1 *Let $\mathcal{F}$ be a (possibly infinite) family of graphs, and suppose there are functions $f_{\mathcal{F}}(\epsilon)$ and $\delta_{\mathcal{F}}(\epsilon)$ such that the following holds for every $\epsilon > 0$: Every graph G on n vertices, which is ϵ-far from being* **induced** *$\mathcal{F}$-free contains at least $\delta_{\mathcal{F}}(\epsilon)n^f$* **induced** *copies of a graph $F \in \mathcal{F}$ of size $f \leq f_{\mathcal{F}}(\epsilon)$. Then, being induced $\mathcal{F}$-free is testable with one-sided error.*

The main result we discuss in this section is the following:

Theorem 7.2 ([12]) *For any (possibly infinite) family of graph $\mathcal{F}$ there are functions $f_{\mathcal{F}}(\epsilon)$ and $\delta_{\mathcal{F}}(\epsilon)$ satisfying the conditions of Lemma 7.1.*

As we have done for monotone properties, one can define for any hereditary property $\mathcal{P}$, a (possibly infinite) family of graphs $\mathcal{F}_{\mathcal{P}}$ such that satisfying $\mathcal{P}$ is equivalent to being *induced* $\mathcal{F}_{\mathcal{P}}$-free. We simply put a graph F in $\mathcal{F}_{\mathcal{P}}$ if and only if F does not satisfy $\mathcal{P}$ but any graph obtained from F by removing a vertex satisfies $\mathcal{P}$. It thus follows that Theorem 7.2, combined with Lemma 7.1, implies the following

Theorem 7.3 ([12]) *Every hereditary graph property is testable with one-sided error.*

It should be noted that besides certain partition properties such as having a large cut and having a large clique, which are testable with two-sided error by Theorem 2.1, essentially any graph property that was studied in the literature is hereditary. Thus, Theorem 7.3 combined with the graph partition problems of [34] imply the testability of (nearly) any natural graph property.

The upper bounds for testing a given hereditary property, which Theorem 7.2 guarantees, have an enormous dependency on $1/\epsilon$. Moreover, the proof of the general result is quite involved. It may thus be interesting to address the following problem.

Open Problem 7.4 *Find efficient and simple to analyze testers for specific hereditary properties such as being Perfect, Chordal and Interval.*

One can infer from Theorem 7.2 the following analogue of Corollary 4.4:

Corollary 7.5 ([12]) *For every hereditary graph property $\mathcal{P}$, there is a function $W_{\mathcal{P}}(\epsilon)$ with the following property: If G is ϵ-far from satisfying $\mathcal{P}$, then G contains an induced subgraph of size at most $W_{\mathcal{P}}(\epsilon)$, which does not satisfy $\mathcal{P}$.*

The above corollary implies, for example, that for every ϵ there is $c = c(\epsilon)$, such that if a graph G is ϵ-far from being Chordal then G contains an **induced** cycle of length at most c, and that similar results hold for any other hereditary property. This is non-trivial as it is not clear a priori that there is no graph that is, say, $\frac{1}{100}$-far from being Chordal and yet contains only induced cycles of length at least, say, $\Omega(\log n)$.

Another interesting application of Theorem 7.2 is an analogue of Theorem 4.6:

Theorem 7.6 ([12]) *For any (possibly infinite) set of hereditary graph properties $\mathcal{P} = \{\mathcal{P}_1, \mathcal{P}_2, \ldots\}$, there is a function $\delta_{\mathcal{P}} : (0,1) \mapsto (0,1)$ with the following property: If a graph G is ϵ-far from satisfying all the properties of $\mathcal{P}$, then for some i, the graph G is $\delta_{\mathcal{P}}(\epsilon)$-far from satisfying $\mathcal{P}_i$.*

Observe, that when considering hereditary non-monotone properties, it is not even clear that if a graph is ϵ-far from satisfying *two* such properties $\mathcal{P} = \{\mathcal{P}_1, \mathcal{P}_2\}$, then it is $\delta_{\mathcal{P}}(\epsilon)$-far from satisfying one of them. (Recall that if the two properties are monotone then this is trivially true, as we can take $\delta_{\mathcal{P}}(\epsilon) = \epsilon/2$ for any such $\mathcal{P}$.) The above theorem asserts that this statement is true even for an infinite family of properties.

We now turn to discuss the most interesting application of Theorem 7.3. The problem of characterizing the graph properties, which are testable (either with one-sided or two-sided error) is considered by many to be the most important open problem in the area of property-testing. An interesting special case of this open problem is to characterize the graph properties, which are testable with one-sided error. This problem should be somewhat easier to resolve as numerous previous works, many of which are presented in this paper, demonstrated that testing with one-sided error is intimately related to various well-studied combinatorial problems, which can be handled using combinatorial tools. It seems, though, that even this seemingly easier problem is still very challenging. Using Theorem 7.3 one can obtain a characterization of the "natural" graph properties, which are testable with one-sided error.

As stated in Lemma 8.3 (see Section 8), by a result of [4] and [35], it is possible to assume that a property tester works by making its queries non-adaptively. In other words, the tester first picks a random subset of vertices S, and then continues without making additional queries. The following more restricted type of tester was introduced in [12]:

Definition 7.7 (Oblivious Tester) *A tester (one-sided or two-sided) is said to be* oblivious *if it works as follows: given ϵ the tester computes an integer $Q = Q(\epsilon)$ and asks an oracle for a subgraph induced by a set of vertices S of size Q, where the oracle chooses S randomly and uniformly from the vertices of the input graph. If Q is larger than the size of the input graph then the oracle returns the entire graph. The tester then accepts or rejects according to the graph induced by S.*

In some sense, oblivious testers capture the essence of property testing as essentially all the testers that have been analyzed in the literature were in fact oblivious, or could trivially be turned into oblivious testers. Even the testers for properties such as having an independent set of size $\frac{1}{2}n$ or a cut with $\frac{1}{8}n^2$ edges (see [34]), whose definition involves the size of the graph, have oblivious testers. The reason is simply that these properties can easily be expressed without using the size of the graph. Clearly, some properties cannot have oblivious testers, however, these properties are not natural. One example out of many, is the property of not containing an induced cycle of length 4 if the number of vertices is even, and not containing an induced cycle of length 5 if the number of vertices is odd. Using Theorem 7.3 it can be shown that if one considers only oblivious testers, then it is possible to precisely characterize the graph properties, which are testable with one-sided error. To state this characterization we need the following definition:

Definition 7.8 (Semi-Hereditary) *A graph property $\mathcal{P}$ is semi-hereditary if there exists a hereditary graph property $\mathcal{H}$ such that the following holds:*

1. *Any graph satisfying $\mathcal{P}$ also satisfies $\mathcal{H}$.*

2. *For any $\epsilon >$ there is an $M(\epsilon)$, such that any graph of size at least $M(\epsilon)$, which is ϵ-far from satisfying $\mathcal{P}$, contains an induced subgraph, which does not satisfy $\mathcal{H}$.*

Clearly, any hereditary graph property $\mathcal{P}$ is also semi-hereditary because we can take $\mathcal{H}$ in the above definition to be $\mathcal{P}$ itself. In simple words, a semi-hereditary $\mathcal{P}$ is obtained by taking a hereditary graph property $\mathcal{H}$, and removing from it a (possibly infinite, carefully chosen) set of graphs. This means that the first item in Definition 7.8 is satisfied. As there are graphs not satisfying $\mathcal{P}$ that do satisfy $\mathcal{H}$ these graphs do not contain any induced subgraph that does not satisfy $\mathcal{H}$ (because $\mathcal{H}$ is hereditary). The only restriction, which is needed in order to get item 2 in Definition 7.8, is that $\mathcal{P}$ will be such that for any $\epsilon > 0$ there will be only finitely many graphs that are ϵ-far from satisfying it, and yet contain no induced subgraph that does not satisfy $\mathcal{H}$. We are now ready to state the characterization.

Theorem 7.9 ([12]) *A graph property $\mathcal{P}$ has an oblivious one-sided tester* **if and only if** *$\mathcal{P}$ is semi-hereditary.*

We briefly discuss the proof of Theorem 7.2, which is far more involved than the proof of Theorem 4.1. Again, the proof heavily relies on a graph functional similar to $\Psi_{\mathcal{F}}$. However, this time $\Psi_{\mathcal{F}}$ is defined with respect to the following new type of homomorphism:

Definition 7.10 (Colored-Homomorphism) *Let K be a complete graph whose vertices are colored black or white, and whose edges are colored black, white or grey (neither the vertex coloring nor the edge coloring is assumed to be proper in the standard sense). A* colored-homomorphism *from a graph F to a graph K is a mapping $\varphi : V(F) \mapsto V(K)$, which satisfies the following:*

1. *If $(u,v) \in E(F)$ then either $\varphi(u) = \varphi(v) = t$ and t is colored black, or $\varphi(u) \neq \varphi(v)$ and $(\varphi(u), \varphi(v))$ is colored black or grey.*
2. *If $(u,v) \notin E(F)$ then either $\varphi(u) = \varphi(v) = t$ and t is colored white, or $\varphi(u) \neq \varphi(v)$ and $(\varphi(u), \varphi(v))$ is colored white or grey.*

We briefly mention that the color black represents edges, the color white represents non-edges and grey edges represent "don't care" namely we allow edges and non-edges. Similarly, a black vertex represents a complete graph, while a white vertex represents an edgeless graph. The reason for using colored-homomorphism instead of standard homomorphism is that the fact that $H \mapsto K$ does not supply enough information about H. In particular, it does not supply any information about the non-edges of H. When dealing with monotone properties this information is not that important as we never increase the distance to satisfying a monotone property if we remove an edge from a graph. For non-monotone properties this is no longer the case, and this is where the notion of colored homomorphism comes in handy. The reader is referred to [12] for the complete proof of Theorem 7.2.

We conclude this section with some open problems. Two graph properties $\mathcal{P}_1$ and $\mathcal{P}_2$ are defined in [4] to be *indistinguishable* if for every $\epsilon > 0$ and large enough n, any graph on n vertices satisfying one property is never ϵ-far from satisfying the other. It is shown in [4] that in this case, $\mathcal{P}_1$ is testable if and only if $\mathcal{P}_2$ is testable. This suggests the following problem.

Open Problem 7.11 *Characterize (either combinatorially, logically or by other means) the graph properties that are indistinguishable from some hereditary graph property.*

Note, that by Theorem 7.3 and the result of [4] mentioned above, any property that is indistinguishable from a hereditary property is testable, possibly with two-sided error.

The general characterization of the testable graph properties seems a very challenging open problem. In fact, even characterizing graph properties that are testable with one-sided error seems a difficult task. As Theorem 7.9 demonstrates, if one considers only oblivious testers, then one can characterize the graph properties, which are testable with one-sided error. The following problem may thus be an interesting step towards a characterization of the testable graph properties.

Open Problem 7.12 *Which graph properties have (possibly two-sided) oblivious testers, whose query complexity is bounded by a function of ϵ only?*

Another intriguing problem is to prove a version of Theorem 5.8 for the family of hereditary graph properties. In fact, even the following special case seems hard to resolve.

Open Problem 7.13 *For which graphs H, can one compute in polynomial time the number of edge modifications needed to make a graph induced H-free?*

8 Testing Subgraphs in Directed and Undirected Graphs

The main results discussed in the previous sections establish the testability of any hereditary graph property. However, checking the details of the proofs of these results reveal that the upper bounds, which these results supply, have an enormous dependency on $1/\epsilon$. Even for simple properties, these bounds are given by the so called WOW function, which is a tower of towers of exponents of height polynomial in $1/\epsilon$. This raises the natural problem of obtaining better upper bounds for specific properties. Specifically, the following is an intriguing open problem:

Open Problem 8.1 *Which hereditary properties $\mathcal{P}$ can be tested with number of queries, which has a polynomial dependency on $1/\epsilon$.*

In fact, even characterizing the monotone graph properties that are testable with $poly(1/\epsilon)$ queries ($poly(1/\epsilon) \equiv$ some polynomial in $1/\epsilon$) seems a challenging open problem. An easy example of a property that can be tested with $poly(1/\epsilon)$ queries is the property of being H-free when H is a single edge: It is easy to see that in this case a sample of $O(1/\epsilon)$ vertices (or pairs) suffices (and is also necessary). A more involved argument shows that the property of being k-colorable can also be tested with $poly(1/\epsilon)$ queries. See [34] and [5]. A natural family of properties for which one may try to characterize the graph properties that are easily testable are the properties of being H-free and induced H-free, for a given fixed graph H. We will focus in this section on properties of being H-free, and only briefly discuss results about properties of being induced H-free at the end of this section. In this section we will denote by $\mathcal{P}_H$ the property of being H-free. We will say that $\mathcal{P}_H$ is *easily testable* with one sided error if it has a one-sided error tester with query complexity $poly(1/\epsilon)$. As we would like to focus on the combinatorial part of the problem, we will restate the problem combinatorially in Corollary 8.5 below, for which we need the following two lemmas:

Lemma 8.2 ([11]) *Let T be a one-sided error tester for testing $\mathcal{P}_H$. Then, T must accept an input graph if the graph induced by the sample of vertices satisfies $\mathcal{P}_H$.*

Lemma 8.3 ([4],[35]) *If there exists an ϵ-tester for a graph property that makes q queries, then there exists such an ϵ-tester that makes its queries by uniformly and randomly choosing a set of $2q$ vertices and querying all their pairs. In particular, it is a non-adaptive ϵ-tester making $\binom{2q}{2}$ queries.*

Lemma 8.4 *Let H be a fixed graph on h vertices. Then, $\mathcal{P}_H$ is testable with one sided error and with query complexity $(q(\epsilon))^{O(1)}$ if and only if any graph on n vertices, which is ϵ-far from being H-free, contains at least $n^h/q^c(\epsilon)$ copies of H, for some $c = c(h)$.*

Proof: Suppose first that any graph on n vertices, which is ϵ-far from being H-free contains at least $n^h/q^c(\epsilon)$ copies of H. The tester for $\mathcal{P}_H$ samples a set of vertices S, of size $h \cdot q^c(\epsilon)$ and accepts the input graph if and only if S spans no copy of H. The tester is clearly one-sided. Also, if the input graph is ϵ-far from being H-free then by assumption, each h-tuple of vertices spans a copy of H with probability at least $1/q^c(\epsilon)$. Thus, a sample of size $3h \cdot q^c(\epsilon)$ spans a copy of H with probability at least $2/3$. Suppose now that for any (small enough) ϵ and any (large enough) n there is a graph G of size n, which is ϵ-far from being H-free, and yet it contains only $n^h/q^h(\epsilon)$ copies of H. Thus, by the union bound, the probability that a random set of $q(\epsilon)$ vertices spans a copy of H is at most $h!\binom{q(\epsilon)}{h}/q^h(\epsilon) < 2/3$. By Lemmas 8.3 and 8.2 this means that $\mathcal{P}_H$ cannot be tested with one-sided error and query-complexity $q(\epsilon)$. ■

As our focus in this section is in testing with number of queries, which is polynomial in $1/\epsilon$, the following implication of the above will be useful.

Corollary 8.5 *$\mathcal{P}_H$ is easily testable with one sided error if and only if any graph on n vertices, which is ϵ-far from being H-free, contains at least $n^h/poly(1/\epsilon)$ copies of H.*

The problem of characterizing the graphs H for which $\mathcal{P}_H$ is easily testable with one-sided error was resolved in [1].

Theorem 8.6 ([1]) *$\mathcal{P}_H$ is easily testable with one-sided error if and only if H is bipartite.*

The first part of the above characterization, that of showing that when H is bipartite, then $poly(1/\epsilon)$ queries suffice for testing $\mathcal{P}_H$, follows by showing that any graph with ϵn^2 edges, contains at least $\epsilon^{h^2/4} n^h$ copies of H. As any graph, which is ϵ-far from being H-free, must contain at least ϵn^2 edges, one gets from Lemma 8.5 that in this case being H-free is testable with $poly(1/\epsilon)$ queries. The proof of the other direction is more complicated. To prove this, one shows that for any non-bipartite graph H there are graphs, which are ϵ-far from being H-free and yet contain only $\epsilon^{c \log 1/\epsilon} n^h$ copies of H. By Lemma 8.5 this is sufficient for obtaining the lower bound. The construction of [1] involves a subtle application of a variant of Behrend's construction [18] of a dense subset of the first n integers containing no 3-term arithmetic progressions. It also applies some properties of cores of graphs defined in subsection 1.2. Theorem 8.6 was extended in [7] to the case of two-sided error testers:

Theorem 8.7 ([7]) *$\mathcal{P}_H$ is easily testable (either with one-sided or two-sided error) if and only if H is bipartite.*

One of the main results of [7], is a characterizing of the *directed* graphs (digraph, for short) H, for which $\mathcal{P}_H$ is easily testable. As it turns out, the case of directed graphs is significantly more involved than the case of undirected graphs. To state the main result of [7] recall the definition of a core of a graph (or a digraph) given in subsection 1.2. The following is the main result of [7]:

Theorem 8.8 ([7]) *Let H be a fixed connected digraph, and let K be its core. Then, $\mathcal{P}_H$ is easily testable if and only if K is either a cycle of length 2 or any oriented tree.*

Observe, that Theorem 8.7 follows as a special case of Theorem 8.8, but with the following (equivalent) formulation: $\mathcal{P}_H$ is easily testable if and only if the core of H is a single edge. As is apparent from the statement of Theorem 8.8, the characterization for digraphs is far more complicated than the characterization for undirected graphs, which is given in Theorem 8.7. The characterization for undirected graphs is also simple in the sense that one can check it in polynomial time. As it turns out, the characterization for digraphs is not complicated by chance as the following holds:

Proposition 8.9 *The following problem is NP-complete; Given a digraph H, decide if $\mathcal{P}_H$ is easily testable.*

The above proposition follows easily by combining Theorem 8.8 with a result of of Hell, Nešetřil, and Zhu [42], who proved that deciding if the core of a digraph is a tree is NP-complete.

Observe, that Theorem 8.8 implies that the property $\mathcal{P}_C$ is easily testable for the oriented cycle C on the vertices $v_1, \ldots, v_{2k}$, that consists of two edge-disjoint directed paths from v_1 to v_{k+1}, as each of the two paths is a core of C. Theorem 8.8 also implies that the property $\mathcal{P}_{C'}$ is *not* easily testable for every oriented cycle C' that is obtained from the above cycle C, by changing the direction of *any* single edge, because in this case the core of C' is the entire digraph. This example shows that the query complexity needed to test $\mathcal{P}_H$ does not depend solely on the structure of H as an undirected graph.

The proof of Theorem 8.8 has three major steps. The simplest involves showing that if the core of H is a 2-cycle then $\mathcal{P}_H$ is easily testable. In this case it is shown that a variant of the argument used in [1] in order to show that testing $\mathcal{P}_H$ is easily testable for any bipartite H, gives an analogous result for directed H, whose core is a 2-cycle. In fact, the argument in [7], when considered for the special case of undirected graphs, improves the upper bound given in [1]. The second step involves the case when the core of H is neither a tree nor a 2-cycle. In this case one has to construct graphs, which are ϵ-far from being H-free and yet contain few copies of H. The construction of such graphs also uses Behrend's construction. The most involved case to handle is in showing that when the core of H is a tree then $\mathcal{P}_H$ is also easily testable.

In this section we have only considered properties of being H-free, because of their relation to homomorphisms. The problem of testing properties of being *induced* H-free have also been considered, but in this case the characterization is much simpler (to state, not to prove) as in this case for any graph or digraph on at least 5 vertices, it can be shown that induced H-freeness cannot be tested with $poly(1/\epsilon)$ queries. See the proof in [8].

A natural extension of the results considered in this section is the case of k-uniform hypergraphs (k-graphs, for short). In this case a k-graph G is ϵ-far from being (induced) H-free if one must add/remove at least ϵn^k edges in order to turn G into a (induced) H-free k-graph. The case of being induced H-free was essentially resolved in [9] where it was shown that for $k \geq 3$, the only H

for which the property of being induced H-free can be tested with $poly(1/\epsilon)$ queries are the single k-edges (which are trivially testable with $O(1/\epsilon)$ queries) as well as a single 3-graph on 4 vertices, H, for which it is still not known if being induced H-free can be tested with $poly(1/\epsilon)$ queries. The properties of being (not necessarily induced) H-free, denoted as above by $\mathcal{P}_H$, seems much more involved. It is easy to show (see [44] and [9]) that for any k-partite k-graph H, the property $\mathcal{P}_H$ is testable with $poly(1/\epsilon)$ queries. [9] also contains a proof of a sufficient condition for inferring that for a given H, $\mathcal{P}_H$ is not testable with $poly(1/\epsilon)$ queries. Regretfully, this condition does not include all non-k-partite k-graphs. It will be interesting to complete the picture by proving (or disproving) the following:

Open Problem 8.10 *Is it the case that for $k \geq 3$, $\mathcal{P}_H$ is testable with $poly(1/\epsilon)$ queries if and only if H is a k-partite k-graph.*

References

[1] N. Alon, Testing subgraphs in large graphs, Proc. 42^{nd} IEEE FOCS, IEEE (2001), 434-441. Also: Random Structures and Algorithms 21 (2002), 359-370.

[2] N. Alon, Ranking tournaments, to appear.

[3] N. Alon, R. A. Duke, H. Lefmann, V. Rödl and R. Yuster, The algorithmic aspects of the Regularity Lemma, Proc. 33^{rd} IEEE FOCS, Pittsburgh, IEEE (1992), 473-481. Also: J. of Algorithms 16 (1994), 80-109.

[4] N. Alon, E. Fischer, M. Krivelevich and M. Szegedy, Efficient testing of large graphs, Proc. of 40^{th} FOCS, New York, NY, IEEE (1999), 656–666. Also: Combinatorica 20 (2000), 451-476.

[5] N. Alon and M. Krivelevich, Testing k-colorability, SIAM J. Discrete Math., 15 (2002), 211-227.

[6] N. Alon and A. Shapira, Testing satisfiability, Proc. 13^{th} Annual ACM-SIAM SODA, ACM Press (2002), 645-654. Also: Journal of Algorithms, 47 (2003), 87-103.

[7] N. Alon and A. Shapira, Testing subgraphs in directed graphs, Proc. of the 35^{th} Annual Symp. on Theory of Computing (STOC), San Diego, California, 2003, 700–709. Also: JCSS 69 (2004), 354-382.

[8] N. Alon and A. Shapira, A characterization of easily testable induced subgraphs, Proc. of the 15^{th} Annual ACM-SIAM SODA, ACM Press (2004), 935-944. Also: Combinatorics, Probability and Computing, to appear.

[9] N. Alon and A. Shapira, Linear equation, arithmetic progressions and hypergraph property testing, Proc. of the 16^{th} Annual ACM-SIAM SODA, ACM Press (2005), 708-717.

[10] N. Alon and A. Shapira, Every monotone graph property is testable, Proc. of the 37^{th} Annual Symp. on Theory of Computing (STOC), Baltimore, Maryland, 2005, 128-137.

[11] N. Alon and A. Shapira, A separation theorem in property-testing, manuscript, 2005.

[12] N. Alon and A. Shapira, A characterization of the (natural) graph properties testable with one-sided error, Proc. of FOCS 2005, to appear.

[13] N. Alon, A. Shapira and B. Sudakov, Additive approximation for edge-deletion problems, Proc. of FOCS 2005, to appear.

[14] N. Alon and J. H. Spencer, **The Probabilistic Method**, Second Edition, Wiley, New York, 2000.

[15] J. Balogh, B. Bollobás and D. Weinreich, Measures on monotone properties of graphs, Discrete Applied Mathematics, to appear.

[16] J. Balogh, P. Keevash and B. Sudakov On the minimal degree implying equality of the largest triangle-free and bipartite subgraphs, submitted.

[17] J. Bang-Jensen and P. Hell, The effect of two cycles on the complexity of colorings by directed graphs, Discrete Applied Math. 26 (1990), 1-23.

[18] F. A. Behrend, On sets of integers which contain no three terms in arithmetic progression, *Proc. National Academy of Sciences USA* 32 (1946), 331–332.

[19] M. Blum, M. Luby and R. Rubinfeld, Self-testing/correcting with applications to numerical problems, JCSS 47 (1993), 549-595.

[20] J. Bondy, J. Shen, S. Thomassé and C. Thomassen, Density conditions for triangles in multipartite graphs, *Combinatorica*, to appear.

[21] K. Cirino, S. Muthukrishnan, N. Narayanaswamy and H. Ramesh, graph editing to bipartite interval graphs: exact and asymptotic bounds, Proc. of 17^{th} FSTTCS (1997), 37-53.

[22] D. Eichhorn and D. Mubayi, Edge-coloring cliques with many colors on subcliques, Combinatorica 20 (2000), 441-444.

[23] E. S. El-Mallah and C. J. Colbourn, The complexity of some edge-deletion problems, IEEE transactions on circuits and systems, 35 (1988), 354-362.

[24] P. Erdős, Graph theory and probability, Canad. J. Mathematics, (11) 1959, 34-38.

[25] P. Erdős, On extremal problems of graphs and generalized

graphs. Israel J. Math. 2 1964 183-190.

[26] P. Erdős, P. Frankl and V. Rödl, The asymptotic number of graphs not containing a fixed subgraph and a problem for hypergraphs having no exponent, Graphs Combin. 2 (1986) 113-121.

[27] P. Erdős and A. Gyárfás, A variant of the classical Ramsey problem, Combinatorica 17 (1997), 459-467.

[28] E. Fischer, The art of uninformed decisions: A primer to property testing, The Computational Complexity Column of The Bulletin of the European Association for Theoretical Computer Science 75 (2001), 97-126.

[29] E. Fischer and I. Newman, Testing versus estimation of graph properties, Proc. of the 37^{th} Annual Symp. on Theory of Computing (STOC), Baltimore, Maryland, 2005, 138-146.

[30] E. Friedgut and G. Kalai, Every monotone graph property has a sharp threshold. Proc. Amer. Math. Soc. 124 (1996), 2993-3002.

[31] M.R. Garey and D.S. Johnson, Computers and Intratability: A guide to the Theory of NP-Completeness, W.H. Freeman and Co., San Francisco, 1979.

[32] P. W. Goldberg, M. C. Golumbic, H. Kaplan and R. Shamir, Four strikes against physical mapping of DNA, Journal of Computational Biology 2 (1995), 139–152.

[33] O. Goldreich, Combinatorial property testing - a survey, In: Randomization Methods in Algorithm Design (P. Pardalos, S. Rajasekaran and J. Rolim eds.), AMS-DIMACS (1998), 45-60.

[34] O. Goldreich, S. Goldwasser and D. Ron, Property testing and its connection to learning and approximation, Proc. of 37^{th} Annual IEEE FOCS, (1996), 339–348. Also: JACM 45(4): 653-750 (1998).

[35] O. Goldreich and L. Trevisan, Three theorems regarding testing graph properties, Proc. 42^{nd} IEEE FOCS, IEEE (2001), 460-469. Also, Random Structures and Algorithms, 23(1):23-57, 2003.

[36] M.C. Golumbic, **Algorithmic Graph Theory and Perfect Graphs**, Academic Press, 1980.

[37] M. C. Golumbic, H. Kaplan and R. Shamir, On the complexity of DNA physical mapping, Advances in Applied Mathematics, 15 (1994), 251-261.

[38] R. L. Graham, B. L. Rothschild and J. H. Spencer, *Ramsey Theory*, Second Edition, Wiley, New York, 1990.

[39] W. T. Gowers, Hypergraph regularity and the multidimensional Szemerédi theorem, manuscript.

[40] P. Hell and J. Nešetřil, **Graphs and Homomorphisms**, Oxford University Press, 2004.

[41] P. Hell and J. Nešetřil, The core of a graph, Discrete Math 109 (1992), 117-126.

[42] P. Hell, J. Nešetřil, and X. Zhu, Duality of graph homomorphisms, in : Combinatorics, Paul Erdős is Eighty, (D. Miklós et. al, eds.), Bolyai Society Mathematical Studies, Vol.2, 1996, pp. 271-282.

[43] D. Karger, R. Motwani and M. Sudan, Approximate graph coloring by semidefinite programming, JACM 45(2), 1998, 246-265.

[44] Y. Kohayakawa, B. Nagle and V. Rödl, Efficient testing of hypergraphs, Proc. of 29^{th} ICALP, (2002), 1017–1028.

[45] J. Komlós and M. Simonovits, Szemerédi's Regularity Lemma and its applications in graph theory. In: *Combinatorics, Paul Erdös is Eighty*, Vol II (D. Miklós, V. T. Sós, T. Szönyi eds.), János Bolyai Math. Soc., Budapest (1996), 295–352.

[46] T. Kővári, V. T. Sós and P. Turán, On a problem of K. Zarankiewicz, Colloquium Math., 3, (1954), 50-57.

[47] J. Lewis and M. Yannakakis, The node deletion problem for hereditary properties is NP-complete, JCSS 20 (1980), 219-230.

[48] L. Lovász, On the shannon capacity of a graph, IEEE Transactions on Information Theory 25(1), 1979, 1-7.

[49] A. Lubotzky, R. Phillips and P. Sarnak, Ramanujan graphs, Combinatorica, 8 (1988), 261-277.

[50] T. A. McKee and F.R. McMorris, **Topics in Intersection Graph Theory**, SIAM, Philadelphia, PA, 1999.

[51] B. Nagle, V. Rödl and M. Schacht, The counting lemma for regular k-uniform hypegraphs, manuscript.

[52] C. Papadimitriou, **Computational Complexity**, Addison Wesley, 1994.

[53] M. Parnas, D. Ron and R. Rubinfeld, Tolerant property testing and distance approximation, manuscrip, 2004.

[54] J. L. Ramírez-Alfonsín, B. A. Reed (Editors), **Perfect Graphs**, Wiley, 2001.

[55] V. Rödl and R. Duke, On graphs with small subgraphs of large chromatic number, *Graphs and Combinatorics* 1 (1985), 91–96.

[56] V. Rödl and J. Skokan, Regularity lemma for k-uniform hypergraphs, Random Structures and Algorithms, 25 (2004), 1-42.

[57] D. Ron, Property testing, in: P. M. Pardalos, S. Rajasekaran, J. Reif and J. D. P. Rolim, editors, *Handbook of Randomized Computing*, Vol. II, Kluwer Academic Publishers, 2001, 597–649.

[58] J. D. Rose, A graph-theoretic study of the numerical solution of sparse positive-definite systems of linear equations, Graph Theory and Computing, R.C. Reed, ed., Academic Press, N.Y., 1972, 183-217.

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

[60] E. Szemerédi, Regular partitions of graphs, In: *Proc. Colloque Inter. CNRS* (J. C. Bermond, J. C. Fournier, M. Las Vergnas and D. Sotteau, eds.), 1978, 399–401.

[61] D. B. West, **Introduction to Graph Theory**, Prentice Hall, 2001.

[62] J. Xue, Edge-maximal triangulated subgraphs and heuristics for the maximum clique problem. Networks 24 (1994), 109-120

ECCC ISSN 1433-8092
http://www.eccc.uni-trier.de/eccc
ftp://ftp.eccc.uni-trier.de/pub/eccc
ftpmail@ftp.eccc.uni-trier.de, subject 'help eccc'