Property Testing of Massively Parametrized problems - A survey*

Ilan Newman†

April 21, 2010

Abstract

We survey here property testing results for the so called 'massively parametrized' model (or problems). This paper is based on a survey talk gave at the workshop on property testing, Beijing, Jan 2010.

1 Introduction

Property Testing of *massively parametrized* problems falls formally into the combinatorial property testing definitions of [15]. However, due to some initial bulk of otherwise sporadic results, and to some shifted focus, as will be explained in what follows, I find it natural to consider such problems together and survey what seems now a collection of results in property testing that have much in common.

I will assume in this survey, a knowledge of the standard definition of property testing. For details on this, see [15, 8, 21].

To better understand the different focus between this 'model' and the standard property testing model, let us outline the main features of a problem in the standard model: To this end, each such testing 'problem' contains three main elements:

- A *fixed structure:* For each problem size n there is a fixed structure that determines all inputs.
- *Inputs:* A set of vectors associated with the structure, e.g, coloring of its points, (or sometimes viewed as a function from the points of the structure to a certain range).
- The *Property:* A subset of all possible input vectors.

For example, consider the graph property of being bipartite, in the 'dense graph' model. Given the problem size n, the *fixed structure* is K_n, the complete graph on n vertices. (alternatively, the fixed structure can be viewed also as an $n \times n$ array). The collection of *inputs* is the set of all subgraphs of K_n which are just all possible graphs on n vertices. Alternatively, the inputs are all $0/1$ coloring of

*This work is based on a survey talk gave at the workshop on property testing, Beijing, Jan 2010.

†Department of Computer Science, University of Haifa, Haifa, Israel. E-mail: ilan@cs.haifa.ac.il. Research supported in part by an Israel Science Foundation grant number 1011/06.

the $n \times n$ array, viewed as the adjacency matrix of graphs on n vertices. The *property* is the set of all bipartite graphs on n vertices, or all labeling of the corresponding array.

Any other graph property induces a similar example. Thus all graph properties of graphs on n vertices share the same fixed structure (that is K_n) and the same set of inputs.

For digraph properties, the fixed structure is again an array as above, and the set of inputs contains all the Boolean labeling of it, with the corresponding interpretation of an input as a digraph.

When properties of Boolean functions are being considered, the fixed structure is the Boolean cube (of a given dimension). The set of inputs are all $0/1$ coloring of vertices of the cube, where each coloring is viewed as Boolean function. A property is then, any subset of functions, e.g., the linear functions, small-degree polynomials, monotone functions etc.

Lets us now examine a typical situation in the 'massively-parametrized' model. For a given input length, the structure *is not* fixed in advance. E.g., for the property of being bipartite, the *structure* could be any graph $G = (V, E)$ with m edges. The *inputs* would be all $0/1$ coloring of the edges of G, each viewed as defining a subgraph of G. The property is then all those subgraphs $G' \subseteq G$ that are bipartite. Likewise, any graph property can be considered, e.g., Eulerianity, being connected, being H-free for some fixed graph H etc.

Properties of digraphs can be considered in a similar way. The structure is any given digraph, the inputs: all subgraphs defined by $0/1$ coloring of its edges, and the property - any collection of subgraphs. One can also consider properties of *induced* subgraphs, rather than just subgraphs. Namely, taking the same structure (either for the undirected or directed case), with the set of inputs being the set of all Boolean coloring of the vertices, each viewed as an encoding of an induced (di)subgraph.

Examples of a different flavor are the following. Let the structure be a given Boolean circuit, formula, or any other computational mechanism. The set of inputs will be the set of all assignments to the Boolean variables (or appropriate input words to the computation - e.g., words over some alphabet for, say, a given grammar/ automata, etc.). The property that is considered in this setting will be fixed: that is, the set of all inputs accepted by the computation. We call this property the *satisfiability* (or sometimes 'membership) property of the corresponding structure.

There are several points to be emphasized.

1. The *distance* - In all the examples above, the set of inputs can be viewed as all possible Boolean vectors of a certain dimension that is defined by the structure. (E.g., the set of Boolean coloring of the edges for the example of subgraph properties, or the set of assignments of all Boolean variables for a given formula). Thus, the distance between inputs is just the hamming distance, the same as it is in the standard model of property-testing (as per the definitions of [15]). In this respect, all the examples considered above, as well as these that are discussed in what follows, fall directly into the standard framework of property-testing.

2. By fixing the property, rather then the structure, the testability (or non-testability) of it becomes a property of the structure, which becomes the focus of study. In particular, for the satisfiability properties described above, the property is unique and fixed, while the structure varies.

3. Note that a test here may be viewed as being composed of two parts: a *preprocessing* part which depends on the structure, and the 'standard testing' part (usually some sampling of the input). As the structure varies, one would expect that there is much to be changed in the test, which, in turn, shifts somewhat the focus towards the input-independent algorithmic preprocessing of the structure. Indeed in all positive examples, the tests exhibit significantly complicated and interesting preprocessing.

In the rest of this survey we outline some of the results that may be viewed as falling into the framework of massively parametrized property testing.

2 General Notations

In what follows a graph $G = (V, E)$ is always undirected, unless explicitly said otherwise. Directed graphs will be denoted similarly but, we will explicitly say that $G = (V, E)$ is digraph (directed graph) when applies.

For a graph $G = (V, E)$ n will usually denote $n = |V|$, and will be considered as a parameter when asymptotic results are stated. Namely, when $O(), \Omega()$, etc. notations are used, one should interpret the results as hold for the set of all graphs, and with the corresponding dependence on n.

Finally, the hamming distance between two vectors of $u, v \in \{0, 1\}^n$ is $dist(u, v) = |\{i|\ u_i \neq v_i\}|$. For $0 < \epsilon < 1$, and two such n dimensional vectors, u, v, we say that u is ϵ-far from v is $dist(u, v) \geq \epsilon \cdot n$, and ϵ-close otherwise.

3 Old results

As made clear above, massively-parametrized testing falls strictly into the area of combinatorial property testing as defined formally in [15]. In particular, some older results may be viewed in retrospect as results on massively-parametrized problems, although they have not been considered or stated as such at the time. We review some of these results.

3.1 Testing membership in read-once branching programs:

In [20] the following problem is considered: Given a read-once deterministic oblivious branching program P on n Boolean variables[1], the set of inputs is the set of all Boolean assignments to the variables. It is proved in [20] that the satisfiability (or membership) problem can be tested (non-adaptively, 1-sided error test) using $O(exp(2^w/\epsilon))$ queries.

The preprocessing is rather heavy, but can be done in time $poly(n)$.

We note that this result is a generalization of a result of [2] stating that testing membership in regular languages is testable. The result of [2] can already be viewed as fits the massively-parametrized framework. There however, the preprocessing is 'light' as the same fixed (hence constant size) automaton serves as the defining mechanism for every input length word.

3.2 Testing monotonicity in general posets:

Monotonicity testing in general posets generalizes monotonicity-testing of Boolean functions. Here a poset $P = (V, \leq)$ is the given structure and the set of inputs is the collection of functions C^V where C is typically a total ordered set (e.g., $C = [n]$ with the natural order, and in particular the Boolean case $n = 2$) but could be taken as any poset. The *fixed* property here is the set of all functions that are monotone, namely all such f for which $f(x) >_C f(y)$ implies that $x >_P y$.

The main results in [9] are lower bounds for testing monotonicity for Boolean labeling of general posets, and in particular, a construction of some posets for which monotonicity requires relatively large

[1]For readers that are not familiar with this concept, this can be replaced with, say, a depth $d = O(1)$ Boolean formula where each variable appears only once

query complexity (previous lower bounds where only for large range [11]). In addition it is shown in [9] that monotonicity is testable when the poset is the transitive closure of an orientation of a tree (or a forest) and C is Boolean. It is also shown that there is a test with $O(\log |V|/\epsilon)$ queries when P is the transitive closure of a directed acyclic graph (DAG) that as an undirected graph is of bounded tree-width[2].

There are some recent interesting generalization of the positive results above, as well as some improvements (E.g., for posets induced by planar graphs) in [3], using 2TC-spanners. This will not be further described here.

4 Contemporary results

Here we survey some of the more contemporary results.

As already discussed in the previous sections, one model for graph related properties is the following. The given but variable underlying structure is an *undirected* graph $G = (V, E)$. The set of inputs contains all Boolean assignments to the *edges*, namely $\{0,1\}^E$. The interpretation of an input $\alpha \in \{0,1\}^E$ is either the subgraph of G that is defined by (V, E_1) where $E_1 = \{e \in E |\ \alpha(e) = 1\}$, or as an orientation of the edges relative to a fixed orientation. E.g., one can interpret α as the digraph $G_\alpha = (V, E_\alpha)$ where

$$E_\alpha = \{(i \to j) |\ i < j, \alpha(i,j) = 1\} \cup \{(i \to j) |\ i > j, \alpha(i,j) = 0\}.$$

Most work in the model was done w.r.t. the later interpretation above, which was often referred to as the *orientation* model.

4.1 Main results in the orientation model

For an undirected graph $G = (V, E)$ and a property P of digraphs, one can consider the property P_G containing these orientations of G that have P. E.g., Eulerianity, being strongly connected, not containing a forbidden subgraph H, etc. We refer to the set of all orientation of G as the G-orientations, and denote such an orientation by $\vec{G}$.

4.1.1 Testing H-freeness

Let $H = (V_H, E_H)$ be a fixed directed graph. The digraph property of *H-freeness* contains all digraphs that do not have a subgraph isomorphic to H, and is denoted by $F_H(G)$. A sink in a digraph is a vertex v for which $d_{out}(v) = 0$, namely, has no outgoing edges. Similarly v is a source if $d_{in}(v) = 0$. A graph is source-free (analogously sink-free) if it contains no source.

The following Theorem is proved in [16].

Theorem 1 *[16] Let H be a fixed digraph that is either source-free or sink-free. Then, for any $O(1)$-bounded degree graph G and $\epsilon > 0$, there is a 1-sided error ϵ-test for $F_H(G)$ making $poly(1/\epsilon)$ queries.*

[2]The test here has the same complexity even if the range is any total ordered set. There are other positive results in [9] for special types of posets.

Proof. **(Sketch.)** Assume w.l.o.g that H is source-free. To make a G-orientation $\vec{G}$, H-free, one can pick a vertex v from each copy of H in $\vec{G}$ and reorient all edges so to make v a source. Obviously this should be done sequentially picking no two adjacent vertices. This implies that if $\vec{G}$ is ϵ-far from being H-free, every subset $C \subseteq V$ that hits all copies of H in $\vec{G}$ must have $|C| = \Omega(n)$. Hence, sampling a random vertex and checking its neighborhood of size $|H|$ to discover if it belongs to a H-copy (and reject if it does) is a test for H-freeness. ∎

Note that the test relies on the fact that G is of bounded degree and that H is either source free or sink free. One may wonder whether these conditions are truly necessary. We first address the issue of being either source or sink free.

The simplest interesting digraph H that has both a source and a sink is P_2, the path of length 2 on 3 vertices (every digraph with non-empty edge set is not P_1-free). Testing P_2-freeness is easy since a G-orientation is P_2-free if and only if G is bipartite $G = (X, Y; E)$ and all its edges are oriented, say, from X to Y. Thus sampling a random edge is a good test. Going however one step further, it is shown in [16] that testing P_3-freeness requires a linear number of queries for some bounded degree graphs. The reduction is from testing whether a 3-coloring of a 3-colorable graph is indeed a proper coloring, which is shown to require a linear number of queries (implicit in [4]).

Regarding the degree restriction, we don't know if this restriction is absolutely required, however, [16] observed:

Let C_6 be the directed cycle on 6 vertices.

Observation 4.1 *[16] There exists an infinite family of graphs, each with $O(1)$-average degree, for which every 1-sided error ϵ-test for the G-orientation property of C_6-freeness makes $(1/\epsilon)^{\Omega(\log(1/\epsilon))}$ queries.*

The proof is immediate: testing triangle-freeness in the dense graph model is reducible to testing C_6-freeness in the orientation model where the underlying graph is a subdivided K_n. The result of [1] completes the proof.

4.1.2 Testing strong connectivity

Strong connectivity is a very natural and basic property of digraphs. We are not aware of any universal result for this property of G-orientation (namely, testability for every underlying graph). [16] presents some initial positive results which we describe below.

We start with a few observations. Obviously, one may assume that the underlying graph $G = (V, E)$ is 2 edge-connected as otherwise no orientation is strongly connected. We can also assume that $|E| = O(|V|)$ as otherwise every G-orientation is ϵ-close to be strongly connected (by simply orienting a minimal 2-edge connected subgraph of G to be strongly connected). Thus it is enough to consider 2-edge connected sparse graphs.

Let D be a directed graph. We denote by $SC(D)$ the DAG that is defined on the strongly connected components of D in the standard way. A *source component* of D is a strongly connected component C of D that corresponds to a source vertex in $SC(D)$ (in other words, every edge between a vertex in C and a vertex in $V(D) \setminus C$ is directed away from C). A *sink component* of D is defined similarly. The following is an observation (used in the context of testing directed graphs properties) from [5].

Observation 4.2 *[5] Let $G = (V, E)$ be a graph on n vertices, and $\vec{G}$ a G-orientation. If $\vec{G}$ has at least $\Omega(n)$ sources or sinks components then a source or sink component of $\vec{G}$ can be found in $O(1)$ queries.* ∎

The only known test for strong connectivity of G-orientations is based on Observation 4.2. The following property identifies a class of underlying graphs on n vertices for which any orientation that is far from being strongly connected has $\Omega(n)$ sources or sinks.

Definition 1 *A family of undirected 2-edge connected graphs $\{G = (V, E)\}$ is called efficiently-Steiner-connected if for every $\delta < 1$ and graph G in the family, for every $S \subseteq V$ such that $|S| \leq \delta^2|V|$, there is a* connected *subgraph $T = (V', E')$ with $S \subseteq V'$ and $|E'| \leq 10\delta|V|$.*

We note that the constant 10 and the function types of δ in the definition are somewhat arbitrary.

Theorem 2 *[16] If G is efficiently-Steiner-connected then the G-orientations property of being strongly connected is testable by a 1-sided error test.*

We note that any 'slightly expanding' graph is strongly-Steiner-connected, while for example, the cycle is not. A simple application of Theorem 2 is given by the following two theorems.

Theorem 3 *[16] For any G that is a linear expander graph as well for the $\sqrt{n} \times \sqrt{n}$ two dimensional grid, the G-orientations property of being strongly connected is testable by a 1-sided error test.*

4.2 Testing $s - t$ Connectivity

Let $G = (V, E)$ be an underlying graph, and $s, t \in V(G)$. The G-orientations property of being $s - t$ connected is another natural property of digraphs. It is shown in [6] that this property is testable for any underlying graph G. The test cannot easily be described. It is composed of a series of non trivial reduction steps that finally reduces the problem to that of testing membership in a bounded width branching program. The preprocessing, however, is in polynomial time. It seems that the dependence on ϵ might be unnecessarily high, although we don't know of a better algorithm or any non-trivial lower bound.

We state the main theorem of [6]. Let P_G^{st} be the G-orientation property of being $s - t$-connected.

Theorem 4 *[6] For any undirected graph G, two vertices $s, t \in V(G)$ and every $\epsilon > 0$, there is a 1-sided error ϵ-test for the property P_G^{st} with query complexity q such that,*

$$q = (2/\epsilon)^{2^{O((1/\epsilon)^{(1/\epsilon)})}}$$

4.3 Testing Eulerianity

A digraph $G = (V, E)$ is Euler if for every $v \in V$, $d_{out}(v) = d_{in}(v)$. It is shown in [12] that the orientation problem of being Eulerian is not testable in general, even by 2-sided error tests. However, the upper and lower bounds are still quite far apart. We state some of the results in [12].

Theorem 5 *[12] There exists an infinite family of graphs for which every 1-sided error test for being Eulerian must make $\Omega(|E|)$ queries.*

The proof is based on the following observation: every 1-sided error test on far inputs must discover a witness for non-Eulerianity. Clearly if a digraph is Eulerian then for every cut $C = (S, \bar{S})$, the number of edges oriented from S to $\bar{S}$ equals the number of edges that are oriented in the opposite direction. Such a cut will be called a *balanced cut*. Thus for a cut C, any orientation of at most half the edges of

C does not exclude the possibility that C is balanced. Indeed, it is shown in [12] that any witness for non-Eulerianity must contain at least half the edges of some unbalanced cut.

The lower bound follows by constructing an underlying graph and set of orientations that are ϵ-far from being Eulerian, and such that each unbalanced cut is of size $\Omega(|E|)$.

The lower bound for 2-sided error test is much weaker, however, it holds for bounded degree graphs as well.

Theorem 6 *[12] For every $0 < \epsilon \leq 1/64$, every non-adaptive (2-sided) ϵ-test for Eulerian orientations of bounded degree graphs must use $\Omega\left(\sqrt{\frac{\log n}{\log\log n}}\right)$ queries. Consequently, every adaptive test requires $\Omega(\log\log n)$ queries.*

We note that [12] also show a stronger lower bound for 1-sided error tests for bounded degree graphs, which will not be quoted here.

As for positive results, [12] include several (sublinear) upper bounds for testing Eulerianity. Their complexity depends on the maximum and average degree of the underlying graph. There is, however, a large gap between the upper bounds and lower bounds in this case.

4.4 Some lower bounds - an on-going work

Proving lower bounds for the subgraph model has been so far with limited success. In what follows we present a fairly general lower bound for 1-sided error *non-adaptive* tests. We don't know the corresponding right bound for adaptive tests.

It will be easier now to switch to the interpretation of an input as a subgraph of the underlying graph, rather than an orientation of it. Namely, let $G = (V, E)$ be an underlying graph, an input $g \in \{0,1\}^E$ is identified with a subgraph G_g of G containing the edges that are assigned 1 by g. A property will be interpreted as a collection of subgraphs of G.

Consider the property of being bipartite. Any 1-sided error test must find an odd cycle in every far input. However, the graph G could be taken to be a bounded degree Ramanujan expander, hence, with no short cycles. Thus an obvious lower bound for a 1-sided error test is the length of the shortest cycle which would be $\Omega(\log n)$ in that case. Can one do better ?

[13] prove a fairly general lower bound for any graph property in which every 0-witness contains a cycle (e.g., bipartiteness). The lower bound holds only for *non-adaptive* tests and gives a result of the type $\Omega(n^\alpha)$ for some $0 < \alpha < 1$.

We present here the results of [13]. For this we need to define the following distribution, analog to the Erdös-Rényi random graph model $\mathcal{G}(n,p)$ for arbitrary underlying graph rather than for the complete graph.

Definition 2 *Let $0 < \delta < 1$ and $G = (V, E)$. We define the distribution $D(\delta)$ on subgraphs of G as follows. We select each edge of E independently, with probability δ, and set E' to include all the edges that were selected. $G' = (V, E')$ is the resulting subgraph.*

Theorem 7 *[13] Let $G = (V, E)$ be an underlying fixed graph of girth $\geq g$. Let $\mathcal{P}$ a property of subgraphs of G such that every 0-witness for $\mathcal{P}$ contains a cycle. If $\delta < 1$ is such that $Prob_{D(\delta)}[dist(G', P) > \epsilon|E|] \geq 0.8$ then every 1-sided error ϵ-test for P requires at least $\Omega(\delta^{-g/3})$ queries.*

We note that the constant 8 in the theorem is somewhat arbitrary.

Proof. (Sketch.) Since any witness for ϵ-far inputs contains a cycle, a 1-sided error test must find a cycle on any far input. Let D be any distribution on ϵ-far inputs, and $q > 0$. Suppose that for every fixed set of queries Q of size q, the probability (over D) that Q contains a cycle is bounded by, say $1/3$ then, using Yao's argument, every non-adaptive algorithm fails to find a cycle with probability $2/3$ on ϵ-far inputs, implying that any 1-sided error ϵ-test for $\mathcal{P}$ must make at least q queries.

Now let $D = D(\delta)$, with δ such that $D(\delta)$ is essentially concentrated on ϵ-far inputs, as stated by the assumption of the theorem. For any fixed set $Q \subseteq E(G)$, Q contains at most $2q$ end-points. Let v be an end-point of an edge in Q. The probability that v is in a cycle $C \subseteq Q$ is bounded by $2q^2 \cdot \delta^g$, since there are only $2q^2$ possible end-points for the two disjoint path of length $g/2$ that start at v and are part of a cycle of length at least g. Hence, by the union bound (on the $2q$ different v's), we conclude that the probability for finding a witness is bounded by $4q^3\delta^g$, implying that $q = \Omega(\delta^{-g/3})$ to guarantee constant probability of finding a witness. We note that although D is not strictly concentrated on far inputs but rather has most of its mass there, the proof does follow e.g, by formally using $D^* = D|Far$ where Far is the event that the graph chosen according to D is indeed ϵ-far from P. It is standard to show that D well approximates D^*. ∎

Corollary 4.1 *Let $G = (V, E)$ be an underlying fixed graph of girth $\geq g$. Let P be a property of subgraphs of G for which every 0-witness for P contains a cycle. Assume further that $dist(G, P) > (1-\delta)|E|$ for some fixed $0 < \delta < 1/2$. Then every 1-sided error δ-test for P requires at least $\Omega(\delta)^{-g/3})$ queries.*

Proof. (Sketch.) Let G as above and δ for which the assumption of the corollary holds. By Chernoff bound $D(3\delta)$ will be essentially concentrated on subgraphs of G that have at least $2\delta|E|$ edges. Hence by the assumption are δ-far from P. Thus the corollary is implied by Theorem 7. ∎

Here are some applications of Corollary 4.1.

Let H be a fixed graph. A graph G contains H as a minor if after deletion and contraction of some edges in G one arrives at a graph isomorphic to H. The graph property of not containing specific minors is extensively studied in graph theory, e.g., extending planarity.

Theorem 8 *Let $H = (V_H, E_H)$ be a fixed graph containing a cycle. Then for some fixed $\epsilon, \alpha > 0$, there is an infinite sequence of graphs $\{G_n\}$, where G_n is an n vertex graph, such that any non-adaptive 1-sided error test for the G_n-subgraphs property of not having H as a minor must make $\Omega(n^\alpha)$ queries.*

We sketch the proof for $H = K_r$ the complete graph on $r \geq 3$ vertices. The same proof holds for any such H.

Proof. (Sketch.) In view of Corollary 4.1 it is enough to show a high girth underlying graph that is sufficiently far from being K_r-free. Lemma 4.1 asserts this for $\delta = 1/3$.

Lemma 4.1 *Let r be fixed, then there is a graph on n vertices G that has girth $\Omega(\log n)$, and is $(2/3)$-far from being K_r minor free.*

Proof. Let G be a random graph from the standard Erdös-Rényi model of random graphs, $\mathcal{G}(n, c/n)$ (namely, each edges is chosen with probability c/n and independent of any other edge), for $c = c(r)$ to be defined later.

It is standard to show that after deleting $o(n)$ edges such a random graph will have (with very high probability) girth $g = \Omega(\log n)$, and at least $cn/3$ edges (or average degree at least $2c/3$).

A theorem of Kostochka, Thomason [18, 22], asserts that every graph of average degree d contains a K_ℓ for $\ell = \ell(d) = \frac{\beta d}{\sqrt{\log d}}$ as a minor, for some fixed universal constant β. This means that if we fix $c = \frac{a}{\beta} r\sqrt{\log r}$, for large enough constant a, we are guaranteed that $\ell(2c/9) > r$. Hence, not only that G has a K_r minor but also every subgraph of it containing $1/3$ of the edges has such minor. We conclude that G is $2/3$-far from being K_r minor free. ∎

As every 0-witness for not being H-free has a cycle, corollary 4.1 ends the proof. ∎

For an underlying graph $G = (V, E)$, let $\mathrm{Bi(G)}$ be the property of G-subgraphs containing these subgraphs of G that are bipartite. Another application is for testing $\mathrm{Bi(G)}$.

Theorem 9 *For some fixed $\epsilon, \alpha > 0$, threre is an infinite sequence of graphs $\{G_n\}$, where G_n is an n vertex graph, such that any non-adaptive 1-sided error test for $\mathrm{Bi(G_n)}$ must query $\Omega(n^\alpha)$ queries.*

Proof. (Sketch.) As any witness for non-bipartiteness must contain an odd cycle, it is enough, by corollary 4.1 to show a high girth underlying graph that is sufficiently far from being bipartite. To construct such graph, again we choose one from the random graph model $\mathcal{G}(n, c/n)$. It is not hard to see that for large enough constant c, one can ensure that such a graph is $2/3$-far from being bipartite, simply since each partition will contain many edges with both endpoints inside the parts. To ensure logarithmic girth, as before, one may delete $o(n)$ edges to get rid of all small cycles. ∎

A similar application with a similar proof is the following: For $G = (V, E)$ and $\phi : E \to \{0, 1\}$, we now interpret ϕ as defining an orientation rather then a subgraph. The property $Ac(G)$ contains all $\vec{G}$-orientations that are acyclic.

Theorem 10 *For some fixed $\epsilon, \alpha > 0$, there is an infinite sequence of graphs $\{G_n\}$, where G_n is an n vertex graph, such that any non-adaptive 1-sided error test for $Ac(G_n)$ must query $\Omega(n^\alpha)$ queries.*

4.5 Testing membership in Boolean formulae

Testing membership in Boolean formula is a natural property and is extensively studied in TCS. We have seen some corresponding testability results for the more general situation of membership in different models of language representation in Section 1. Here we concentrate on CNF formulae, and formulae that are the conjunction of typically many small subformlae.

4.5.1 Constraint graph formulae

We start with a model for formulae that is quite related to the orientation model discussed in Section 4.1, and that was formulated in [17].

A constraint-graph is a labeled multi-graph (a graph where loops and parallel edges are allowed), where each edge is labeled by a distinct Boolean variable, and every vertex is associate with a Boolean function over the variables that label its adjacent edges[3]. A Boolean assignment to the variables satisfies the constraint graph if it satisfies every vertex function[4]. We associate with a constraint-graph G the property of all assignments satisfying G, denoted $SAT(G)$.

[3]One should not confuse this definition of 'constraint-graphs' with the definition used by Dinur in [7]. There, each vertex is associated with a *distinct* Boolean variable and the edges are labeled by constraints.

[4]Thus formulae are conjunction of the individual vertex formulae. One may similarly consider other types of top gates to connect the vertex formulae, e.g, threshold gates, etc.

Thus, e.g., the property of subgraphs (or of G-orientations) of being Eulerian can be expressed in an obvious way. Similarly, the property of G-orientations of not having a source vertex can be expressed, as well as the property of edge bi-coloring so that each vertex sees both colors.

Obviously, formulae represented by constraint graphs are general enough to represent any Boolean formula (as one can take a two vertex graph, with many parallel edges between the two vertices, and any complex formula on one of the vertices). Hence, the interesting case is when further restrictions are put on the complexity of the vertex formulae.

The main result of [17] is a test for the following class of Boolean formulae. For a vertex $v \in V(G)$ let $E_v = \{(v,w) \in E(G)\}$, namely all edges adjacent to v. For an assignment $\sigma \in \{0,1\}^E$ and $F \subseteq E$ we denote by $\sigma(F)$ the restriction of the assignment σ on F.

Definition 3 *Let $\mathcal{LD}_i$ be the set of constraint graph formulae obtained from an underlying graph G such that for every two assignments σ_1, σ_2 and every vertex v with $deg_G(v) \geq 3$, if σ_1, σ_2 do not satisfy f_v then either $\sigma_1(E_v) = \sigma_2(E_v)$ or they differ on at least i variables.*

Theorem 11 *[17] For every constraint-graph $G \in \mathcal{LD}_3$ there exists a 1-sided error, non-adaptive $(\epsilon, 2^{\tilde{O}(1/\epsilon)})$-test for $SAT(G)$.*

The test is too complicated to be described here. We further mention that it is a generalization of a test for the property of G-orientations of not containing a source. The preprocessing involved in constructing the test, given G, is in polynomial time. We also note that this property, as well as the property of edge bi-coloring in which each vertex sees edges of both colors are in the family $\mathcal{LD}_3$ (while the property of being Eulerian is not).

An interesting application of Theorem 11 is to better understand the testability 'boundary' of the membership problem for restricted CNF formulae. Obviously general CNF formulae are hard to test. In [4] the authors show that there exists family of 3-CNF formulae that are highly non-testable (every test requires linear number of queries). Hence some restriction on the formulae has to be put in order to allow testability.

A Read-k-times formula is a formula in which every variable appears at most k times. By standard arguments the result of [4] can also be extended to read-three times formulae (and any constant $k \geq 3$ size clauses).

What about 2CNF ? It turns out that this does not yet allow testability either. A corollary from the lower bound of [9] for testing monotonicity over general posets is that even *monotone* 2CNF are hard to test (although the lower bound is far from being linear). This is so as in [9] a two leveled poset is constructed for which monotonicity is hard to test. Thus this hard to test property can be expressed as 2CNF. Moreover, by renaming the variables it is, in fact, a monotone 2CNF[5].

Theorem 11 implies that restricting the CNF to be Read-2-times (a.k.a 'read-twice') already guarantee testability, disregarding the clause size. In view of the discussion above, this seems best possible in this respect.

Corollary 4.2 *Every read-twice CNF formulae is testable by a 1-sided error test.*

Proof. A read-twice formula can obviously be represented by a constraint graph. Further, as per the definition of $\mathcal{L}D_3$, clauses of size smaller than 3 pose no restrictions. Since every clause as only one

[5]The 2CNF can also be transformed into a read-3-times hard to test 2CNF using the standard reduction to bounded number of occurrences of a variable.

0-assignment, every two non identical 0-assignments that falsify a clause of size 3 or more differ on at least 3 variables. Thus Theorem 11 applies, and implies the claim. ∎

One may wonder whether $\mathcal{LD}_3$ is coincidental, namely are $\mathcal{LD}_2$ formulae testable. Indeed [10] (see more details in [17]) noted that this is not the case: For every Boolean formula θ there is a formulae $\phi \in \mathcal{LD}_2$ so that for any q, there is an ϵ-test for θ-membership with q queries if and only if there is an ϵ-test for ϕ-membership using q queries. To see this let θ be a Boolean formula over a set variables $X = \{x_1, \ldots, x_n\}$. Let G be the constraint-graph on two vertices $\{v, t\}$, that has $n+1$ parallel edges between v and t, one that is associated with the variable y and the rest n edges are labeled each with a distinct variable in X. Let $f_v = y \oplus (\bigoplus_{i=1}^{n} x_i)$ and $f_t = \theta(x_1, \ldots, x_n) \vee (y = \bigoplus_{i=1}^{n} x_i)$. Obviously the resulting graph G is in $\mathcal{LD}_2$. Given a test for one of the properties it is straightforward to build a test for the other property.

4.5.2 Testing general Read-Once-formulae

Here we discuss testing membership in other restricted formulae that are not the results of [20] or [17].

Read-Once formulae are Boolean formulae with $\wedge$ (AND) and $\vee$ (OR) gates in which every variable appears at most once. In addition, in what follows, we will assume that each variable appear unnegated, thus such formulae represent monotone Boolean functions.[6] Read-Once formulae have been studied extensively in the past, but surprisingly, not with respect to whether their corresponding membership problem is testable. We note that bounded depth read-once formulae are represented by constant width branching program, thus their testability follows from [20]. We present here a result of Fischer, Lachish and Nimbhorkar that is much stronger.

Theorem 12 *[14] For every read-once formula F, the membership problem for F is 1-sided error testable.*

We present a relatively detailed proof as the idea here is somewhat different from the standard ideas one meets in property testing. The analysis we bring here implies an ϵ-test that has query complexity $(1/\epsilon)^{O(1/\epsilon)}$, and that the test can be constructed (that is, the preprocessing) in polynomial time. A variation of this idea and better analysis [14] implies a much better dependence on ϵ.

Proof. (**Sketch**), Along the sequel we identify the formula F with the Boolean function that it computes $F : \{0,1\}^n \mapsto \{0,1\}$. We say that F is an $\vee$-formula if its top gate is $\vee$, similarly $\wedge$-formulae are defined. A single variable is considered both a $\vee$-formula and an $\wedge$-formula.

We first note that the formula F may be assumed to be layered so that the entries to each $\vee$-gate are variables or come from $\wedge$-gates, and the entries for $\wedge$-gates are variable or come from $\vee$-gates.

The test will be recursive w.r.t ϵ, that is, an ϵ-test for a $\vee$-formula will call a δ-test for a $\wedge$-subformula, with δ significantly larger than ϵ, while the test for $\wedge$-formula will call a test for $\vee$-formula with nearly the same ϵ. For this to end we first note that for any read-once $\vee$-formula F on at least two variables, any assignment is $1/2$-close to $F^{-1}(1)$.

Let $F = \circ_{i=1}^{k} F_k$ where $\circ$ is either $\vee$ or $\wedge$. For any assignment x to F, x is naturally being partitioned to $x_1, \ldots x_k$, where x_i, $i = 1, \ldots, k$ is an assignment to the variables in F_i. We keep this notation and for each assignment x to F refer to $x_1, \ldots, x_k$ as the corresponding assignments for $F_1, \ldots, F_k$.

We next describe a $\wedge$-test for $\wedge$-formulae and $\vee$-test for $\vee$-formula. In the following n is the number of variables of F.

[6] This is w.l.o.g as the formula is known in this model, and by renaming we arrive to the desired situation.

ϵ-$\wedge$-test: Let $F = \wedge_{i=1}^{k} F_k$ with F_i on s_i variables. If $k = 1$ and F_1 is a single variable test deterministically F_1 by probing the variable.

Otherwise, independently for $9/\epsilon^2$ times pick an $i \in [k]$ with probability s_i/n and δ-$\vee$-test F_i independently twice, with $\delta = \epsilon(1 - \epsilon/3)$. If for the chosen i, both tests answer 1, set $T_i = 1$ and otherwise set $T_i = 0$. If for any of the chosen i's $T_i = 0$ stop and output 0 (that is reject x as not being in $F^{-1}(1)$), and otherwise (if for all i's $T_i = 1$) accept x (as being ϵ-close to $F^{-1}(1)$).

ϵ-$\vee$-test: Let $F = \wedge_{i=1}^{k} F_k$ with F_i on s_i variables.

If $k = 1$ and F_1 is a single variable, deterministically test F_1 using one query. If for some i, $s_i < \epsilon n$, or if $\epsilon \geq 1/2$ stop and accept x (as being ϵ-close to $F^{-1}(1)$).

Otherwise, for every $i \in [k]$ set $\alpha_i = \epsilon \cdot n/s_i$ and perform $\ell = 1 + \log_3 k$ independent times α_i-$\wedge$-test for F_i. For such i set $T_i = 1$ if all the ℓ independent tests return 1 and 0 otherwise. If any of the $T_i,\ i = 1, \ldots, k$ has returned 1 accept x (as being ϵ-close to $F^{-1}(1)$). Otherwise (if for every $i,\ T_i = 0$) reject x as being ϵ-far from $F^{-1}(1)$.

To assert the correctness of the tests, assume inductively that both tests are 1-sided error and have error probability at most $1/3$ on smaller formulae for any ϵ (the reader may check this for the base case of one gate formulae). It is easy to verify that both tests have 1-sided error assuming inductively that they do on subformulae.

To assess the error probability of the $\wedge$-test, assume that x is ϵ-far from a $\wedge$-formula F. Then x_i is ϵ_i-far from being a member of F_i with $\sum \epsilon_i \cdot \frac{s_i}{n} = \epsilon$. From this it follows, by simple average-argument, that there are at most $1-\epsilon^2/3$-fraction of the variables that may appear in F_i for which x_i is $(1-\epsilon/3)\epsilon$-close to F_i. Thus an error will occur on x only if all $9/\epsilon^2$ i's that are sampled resulted in F_i for which x_i is $(1 - \epsilon/3)\epsilon$-close to F_i, or when an i for which x_i is $(1 - \epsilon/3)\epsilon$-far from F_i is sampled, and the $\vee$-test for F_i errs twice. The former event happens with probability smaller than e^{-3} and the later with probability at most $1/9$, resulting a total error bounded by $1/3$.

For the $\vee$-test, note that if x is ϵ-far from being a member of F, namely one needs to change at least ϵn bits of x in order to become in $F^{-1}(1)$, then in each x_i one should change at least ϵn bits to become a member of $F_i^{-1}(1)$. It follows that x_i is at least α_i-far from F_i. Hence, an error on x occurs only if at least one of the T_i's is wrong, meaning that all ℓ tests were wrong for F_i. By the union bound this will happen with probability at most $k \cdot 1/(3k) \leq 1/3$.

We end by estimating the query complexity. Let $t_\vee(\epsilon), t_\wedge(\epsilon)$ denote the corresponding complexity of the $\vee$ and $\wedge$ tests. Note first that for $\epsilon \geq 1/2$ $t_\vee(\epsilon) = 1$. Also, this is the case if F is $\vee$-formula with $k > 1/\epsilon$, or $s_i < \epsilon n$. Thus we get $t_\vee(\epsilon) \leq \frac{\log(1/\epsilon)}{\epsilon} \cdot t_\wedge(\epsilon/(1-\epsilon))$. (Since $s_i \geq (1-\epsilon)n$ for all i's, which implies that $\alpha_i \leq \frac{\epsilon}{1-\epsilon}$).

Similarly, $\wedge$-test takes 1-query for $k = 1$, and for general ϵ it takes $t_\wedge(\epsilon) \leq \frac{6}{\epsilon^2} \cdot t_\vee(\epsilon(1-\epsilon))$. Solving the recurrence implies the result. ∎

5 Open Problems

There are many open problems that arise in view of the results above. I will mention here but some.

1. There are nearly no testability results for any interesting subgraph-property of a given undirected underlying graph. There are no interesting results at all on the *induced* analog (namely, where the subgraph is defined by vertex labeling).

2. Testing monotonicity is not a main theme here. However, testing monotonicity in general posets does fall into this context. In view of the large gap between the upper and lower bounds in [9]

this gives rise to a interesting open problem. One should mention that even for the Boolean cube, the complexity of testing monotonicity of Boolean functions (2-sided error) is far from being understood.

3. In [20] the dependence in the width is doubly exponential. There is no matching lower bound for this dependence (see also [19] for further related details). As this result is what partially determines the dependence in ϵ in the result for testing $s-t$-connectivity (Section 4.2), this further motivates resolving the question of the exact dependence in ϵ of both testing problems.

4. A general result on testing strong connectivity for G-orientations, or a non-testability result are still missing. It still might be the case that for every underlying graph this problem is testable. On the other hand, we don't have any non-trivial upper bound even of the form n^{α} for $\alpha < 1$. Similarly the variants of $s-t$ strong connectivity and s-connectivity (namely, that s can reach every other vertex) are open.

References

[1] N. Alon. Testing subgraphs in large graphs. *Random Struct. Algorithms*, 21(3-4):359–370, 2002.

[2] N. Alon, M. Krivelevich, I. Newman, and M. Szegedy. Regular languages are testable with a constant number of queries. *SIAM Journal on Computing*, 30:1842–1862, 2001.

[3] A. Bhattacharyya, E. Grigorescu, K. Jung, S. Raskhodnikova, and D. P. Woodruff. Transitive-closure spanners. In *SODA*, pages 932–941, 2009.

[4] E. Ben-Sasson, P. Harsha, and S. Raskhodnikova. Some 3CNF properties are hard to test. *SIAM J. Comput.*, 35(1):1–21, 2005.

[5] M. A. Bender and D. Ron. Testing properties of directed graphs: acyclicity and connectivity. *Random Struct. Algorithms*, 20(2):184–205, 2002.

[6] S. Chakraborty, E. Fischer, O. Lachish, A. Matsliah, and I. Newman. Testing *t* -connectivity. In *APPROX-RANDOM*, pages 380–394, 2007.

[7] I. Dinur. The PCP Theorem by gap amplification. *J. ACM*, 54(3):12, 2007.

[8] E. Fischer. The art of uninformed decisions: A primer to property testing. *BEATCS: Bulletin of the European Association for Theoretical Computer Science*, 75, 2001.

[9] E. Fischer, E. Lehman, I. Newman, S. Raskhodnikova, R. Rubinfeld, and A. Samorodnitsky. Monotonicity testing over general poset domains. *Proceedings of the 34th ACM STOC*, pages 474–483, 2002.

[10] E. Fischer. Personal communication.

[11] E. Fischer. On the strength of comparisons in property testing. *Inf. Comput.*, 189(1):107–116, 2004.

[12] E. Fischer, O. Lachish, A. Matsliah, I. Newmanand and O. Yahalom. On the query complexity of testing orientations for being Eulerian. In *APPROX-RANDOM*, page 1, 2008.

[13] E. Fischer, O. Lachish, I. Newman, and E. Rosenberg. Lower bound technique for properties of underlying graphs.

[14] E. Fischer, O. Lachish and P. Nimbhorkar. Personal communication, 2010.

[15] O. Goldreich, S. Goldwasser and D. Ron. Property testing and its connection to learning and approximation. *J. ACM*, 45(4):653–750, 1998.

[16] S. Halevy, O. Lachish, I. Newman and D. Tsur. Testing orientation properties. *Electronic Colloquium on Computational Complexity (ECCC)*, (153), 2005.

[17] S. Halevy, O. Lachish, I. Newman and D. Tsur. Testing properties of constraint-graphs. In *IEEE Conference on Computational Complexity*, pages 264–277, 2007.

[18] A. Kostochka. The minimum Hadwiger number for graphs with a given mean degree of vertices. *Metody Diskret. Analiz.*, 38:37–58, 1982.

[19] O. Lachish, I. Newman and A. Shapira. Space Complexity vs. Query Complexity. *Computational Complexity*, 17(1): 70–93, 2008.

[20] I. Newman. Testing membership in languages that have small width branching programs. *SIAM J. Comput.*, 31(5):1557–1570, 2002.

[21] D. Ron. *Property testing (A Tutorial). Handbook of Randomized Computing* (S.Rajasekaran, P. M. Pardalos, J. H. Reif and J. D. P. Rolin eds), Kluwer Press (2001).

[22] A. Thomason. An extremal function for contractions of graphs. *Math. Proc. Cambridge Philos. Soc.*, 95:261–265, 1984.