Contemplations on Testing Graph Properties

Oded Goldreich
Department of Computer Science
Weizmann Institute of Science
Rehovot, ISRAEL.
oded.goldreich@weizmann.ac.il

August 1, 2005

Abstract

This note documents two programmatic comments regarding testing graph properties, which I made during the Dagstuhl workshop on Sublinear-Time Algorithms (July 2005). The first comment advocates paying more attention to the dependence of the tester's complexity on the proximity parameter. The second comment advocates paying more attention to the question of testing general graphs (rather than dense or bounded-degree ones). In addition, this note includes a suggestion to view property testing within the framework of promise problems.

We assume that the reader is familiar with the basic models underlying testing graph properties (see surveys [14, 25]).

1 Complexity as a function of the proximity parameter

It is indeed an amazing fact that many properties can be tested within (query) complexity that only depends on the proximity parameter (rather than also on the size of the object being tested). This amazing statement seems to put in shadow the question of what is the form of the aforementioned dependence, and blurs the difference between a reasonable dependence (e.g., a polynomial relation) and prohibiting one (e.g., a tower-function relation). We claim that, as in the context of standard approximation problems (cf. [22]), the dependence of the complexity on the approximation (or proximity) parameter is a key issue.

For the sake of simplicity we focus on the *query complexity* of testers, and assume that it only depends on the proximity parameter ϵ. We highlight the difference between the following types of dependencies, where ϵ-testing refers to distinguishing objects having the property from objects that are ϵ-far from having the property:

1. The query complexity is linearly related to the proximity parameter; that is, ϵ-testing can be achieved by using $O(1/\epsilon)$ queries.

   We note that, for any non-trivial graph property, the query complexity of ϵ-testing in the adjacency matrix model is $\Omega(\sqrt{1/\epsilon})$, and we conjecture that a lower-bound of $\Omega(1/\epsilon)$ actually holds in this case. (A lower-bound of $\Omega(1/\epsilon)$ is easy to establish in the bounded-degree incidence list model.)

2. The query complexity is polynomially related to the proximity parameter; that is, ϵ-testing can be achieved by using $\mathrm{poly}(1/\epsilon)$ queries.

For example, all graph property testers in [16] have query complexity $\mathrm{poly}(1/\epsilon)$. We note, however, that some of these testers (e.g., the one for 3-colorability) have time complexity that is exponential in the proximity parameter ϵ, and this seems unavoidable assuming that $\mathcal{NP}$ does not have sub-exponential time algorithms. We wish to praise [3, 9] for further studying the query complexity of testing k-colorability, and in particular for determining the query complexity of non-adaptively testing bipartiteness (up to a polylogarithmic factor).[1]

A natural problem that is ϵ-testable within query complexity that only depends on ϵ, but requires a super-polynomial dependency on $1/\epsilon$ was pointed out by Alon [1]. He proved that ϵ-testing triangle-freeness requires at least $(1/\epsilon)^{\Omega(\log(1/\epsilon))}$ queries. We comment that this is quite far from the known upper-bound (mentioned in Item 4).

3. The query complexity is exponentially related to the proximity parameter; that is, ϵ-testing can be achieved by using $\exp(1/\epsilon)$ queries.

   We are not aware of any natural testing problem having this query complexity.

4. The query complexity is related to the proximity parameter via a function that grows tremendously fast. A notorious example is the tower function $\mathtt{tf}$ defined inductively by $\mathtt{tf}(n) = \exp(\mathtt{tf}(n-1))$ with $\mathtt{tf}(1) = 2$. (Indeed, $\mathtt{tf}$ is the inverse of the $\log^*$ function.)

   Starting in [2], many positive results regarding testing graph properties in the adjacency matrix model establish such a query complexity; that is, they establish ϵ-testers of query complexity $\mathtt{tf}(\mathrm{poly}(1/\epsilon))$ (and sometimes even $\mathtt{tf}(\mathtt{tf}(\mathrm{poly}(1/\epsilon)))$). In particular, ϵ-testing triangle freeness is known only when using $\mathtt{tf}(\mathrm{poly}(1/\epsilon))$ queries. This dependence is an artifact of these results' application of the Regularity Lemma (or stronger variants of it).

We wish to stress that we do value the impressive results of [2, 4, 5, 13], which refer to graph testers having query complexity that is independent of the graph size but depend prohibitingly on the proximity parameter. We view such results as an impressive first step, which called for further investigation directed at determining the actual dependency of the complexity on the proximity parameter.

2 Models of testing graph properties

The bulk of algorithmic research regarding graphs refers to general graphs. Of special interest are graphs that are neither very dense nor have a bounded-degree. In contrast, research in testing properties of graphs started (in [16]) with the study of dense graphs, next (starting in [17]) bounded-degree graphs were considered, and general graphs were considered only in [24, 23]. This evolution has historical reasons to be reviewed first.

Testing graph properties was initially conceived (in [16]) as a special case of the framework of testing properties of functions (cf. [26]). Thus, graphs had to be represented by functions, and two standard representations of graphs seemed most fitting in this context:

1. The *adjacency matrix representation* [16]: That is, a graph $G = (V, E)$ is represented by a function $g : V \times V \to \{0, 1\}$ such that $g(u, v) = 1$ if and only if $\{u, v\} \in E$. This representation corresponds to the so-called *adjacency queries*, and suggests that the (relative) distance

[1] Alon and Krivelevich [3] presented an ϵ-tester that inspects the subgraph induced by $\widetilde{O}(1/\epsilon)$ randomly chosen vertices (thus making $\widetilde{O}(1/\epsilon^2)$ non-adaptive queries), whereas Bogdanov and Trevisan [9] prove a lower-bound of $\Omega(1/\epsilon^2)$ non-adaptive queries (and $\Omega(1/\epsilon^{3/2})$ adaptive queries).

between graphs be measured as the fraction of vertex-pairs on which the corresponding adjacency matrices differ.

Needless to say, when considering ϵ-testing, this model is interesting mostly for ϵ-dense graphs (i.e., graphs $G = (V, E)$ such that $2|E| > \epsilon|V|^2$).

(A partial list of the works done in this model includes [1, 2, 3, 4, 5, 6, 9, 13, 16, 21].)

2. The (bounded-degree) *incidence list representation* [17]: Specifically, for a fixed integer d, the graph $G = (V, E)$ is represented by a function $g : V \times [d] \rightarrow V \cup \{\perp\}$ such that $g(u, i) = v$ if v is the i^{th} neighbor of u and $g(u, i) = \perp$ if u has less than i neighbors. This representation corresponds to the so-called *neighbor queries*, and suggests that the (relative) distance between graphs be measured as the fraction of vertex-index pairs on which the corresponding incidence lists differ.

   Needless to say, this model can only be applied to graphs of *maximum* degree d. Indeed, we may take d to be arbitrary large, but in this case the model is interesting mostly for ϵ-testing graphs than having *average degree* at least ϵd.

   (Work done in this model includes [8, 17, 18, 19].)

The reader may note that both models were formulated in a way that identifies the graphs with a specific functional representation, which in turn defines the type of queries allowed to the tester as well as the notion of fractional distance (which underlies the performance guarantee).

The identification of graphs with any specific functional representation was abandoned by Parnas and Ron [24] who developed a more general model by decoupling the type of queries allowed to the tester from the distance measure: Whatever is the mechanism of accessing the graph, the distance between graphs is defined as the number of edges in their symmetric difference (rather than the number of different entries with respect to some specific functional representation). Furthermore, the relative distance may be defined as the size of the symmetric difference divided by the actual (total) number of edges in both graphs (rather than divided by some (possibly non-tight) upper-bound on the latter quantity). As advocated by Kaufman *et. al.* [23], it may be reasonable to allow the tester to perform both adjacency and neighbor queries (and indeed each type of query may be useful in a different range of edge densities). Needless to say, this model seems adequate for the study of testing properties of arbitrary graphs, and it strictly generalizes the positive aspects of the two prior models (i.e., the models based on the adjacency matrix and bounded-degree incidence list representations).

We wish to advocate further study of the latter model. We believe that this model, which allows for a meaningful treatment of property testing of general graphs, is the one that is most relevant to computer science applications. Furthermore, it seems that designing testers in this model requires the development of algorithmic techniques that may be applicable also in other areas of algorithmic research. As an example, we mention that techniques in [23] that underly the average degree approximation of [20]. (Likewise techniques of [17] underly the minimum spanning tree weight approximation of [10]; indeed, as noted next, the bonuded-degree incidence list model is also more algorithmic oriented than the adjacency matrix model.)

Let us focus on the algorithmic contents of property testing of graphs. We first note that, ignoring a quadratic blow-up in the query complexity, property testing in the adjacency matrix representation reduces to sheer combinatorics: To ϵ-test if G has property $\mathtt{P}$, it suffices to check whether a random induced graph (of adequate size) of G has some "related" property $\mathtt{P}'$ (see [21])[2].

[2] We note that the transformation of [21] may increase the query complexity in a quadratic manner. It is conceivable

In contrast, property testing in the incidence list representation employs some non-trivial algorithmic techniques such local search (cf. [17]) and random walks (cf. [18]). Testers in the general ("flexible") graph models seem to require even more algorithmic ideas (cf. [23]).

To summarize, we advocate further study of the model of [24, 23] for two reasons. The first reason is that we believe in the greater relevance of this model to computer science applications. The second reason is that we believe in the greater potential of this model to have cross fertilization with other branches of algorithmic research.

A parenthetical comment: We seize the opportunity to call attention also to the study of testing properties of directed graphs, initiated in [7].

3 Property testing as a type of a promise problem

We advocate viewing property testing within the framework of promise problems, a framework introduced in [11] (see recent survey [15]). Formally, a promise problem is a partition of the set of all strings into three subsets:

1. The set of strings representing YES-instances.
2. The set of strings representing NO-instances.
3. The set of disallowed strings (which represent neither YES-instances nor NO-instances).

The algorithm (or process) that is supposed to solve the promise problem is required to distinguish YES-instances from NO-instances, and is allowed arbitrary behavior on inputs that are neither YES-instances nor NO-instances. Intuitively, this algorithm (or rather its designer) is "promised" that the input is either a YES-instance or a NO-instance, and is only required to distinguish these two cases. This generalizes the standard notion of a decision problem, in which each string is either a YES-instances or a NO-instance.

Gap problems constitute a special type of promise problems in which instances are partitioned according to some metric leaving a "gap" between YES-instances and NO-instances. Standard approximation problems refer to one such type of metric in which instances are positioned according to the value of the best corresponding "solution" (with respect to some predetermined objective function). Property testing refer to a second type of metric in which instances are positioned according to their distance from the set of objects that satisfy some predetermined property.

Indeed, property testing is a relaxation of decision problems, where this relaxation leaves a gap between instances that should be accepted (with high probability) and instances that should be rejected (with high probability). The former contain all instances that have the predetermined property, whereas the latter contain all instances that are "far from having the property" (i.e., being at large distance from any instance in the former set). Typically, the distance (or proximity) parameter is given as input to the tester, which makes the positive results stronger and more appealing (especially in light of a separation recently shown in [6]). In contrast, negative results typically refer to a fixed value of the distance parameter.

Thus, for any property P and any *distance function* (e.g., Hamming distance between bit strings), two natural types of promise problems emerge:

that an adaptive tester (which is less dull from an algorithmic perspective) may perform better than the canonical tester of [21] (which merely examines a random induced subgraph).

1. *Testing w.r.t variable distance*: Here instances are pairs (x, δ), where x is a description of an object and δ is a distance parameter. The YES-instances are pairs (x, δ) such that x has property $\mathtt{P}$, whereas (x, δ) is a NO-instance if x is δ-far from any x' that has property $\mathtt{P}$.

2. *Testing w.r.t a fixed distance*: Here we fix the distance parameter δ, and so the instances are merely descriptions of objects, and the partition to YES and NO instances is as above.

For example, for some fixed integer d, consider the following promise problem, denoted $\mathtt{BPG}_d$, regarding bipartiteness of bounded-degree graphs. The YES-instance are pairs (G, δ) such that G is a bipartite graph of maximum degree d, whereas (G, δ) is a NO-instance if G is an N-vertex graph of maximum degree d such that more than $\delta \cdot dN/2$ edges must be omitted from G in order to obtain a bipartite graph. Similarly, for fixed integer d and $\delta > 0$, the promise problem $\mathtt{BPG}_{d,\delta}$ has YES-instances that are bipartite graphs of maximum degree d and NO-instances that are N-vertex graphs of maximum degree d such that more than $\delta \cdot dN/2$ edges must be omitted from the graph in order to obtain a bipartite graph. In [17] it was shown that any tester for $\mathtt{BPG}_{3,0.01}$ must make $\Omega(\sqrt{N})$ neighbor queries. In contrast, for every d and δ, the tester presented in [18] decides $\mathtt{BPG}_{d,\delta}$ in time $\widetilde{O}(\sqrt{N}/\mathrm{poly}(\delta))$. In fact, this algorithm decides $\mathtt{BPG}_d$ in time $\widetilde{O}(\sqrt{N}/\mathrm{poly}(\delta))$, where N and δ are explicitly given parameters.

The formulation typically used in the literature. Indeed, all research on property testing refers to the two aforementioned types of promise problems, where typically positive results refer to the first type and negative results refer to the second type. However, most works do not provide a strictly formal statement of their results (see further discussion below), because the formulation is rather cumbersome and straightforward. Furthermore, in light of the greater focus on positive results (and in accordance with the traditions of algorithmic research), such a formal statement is believed to be unnecessary.[3] Let us consider what is required for a formal statement of property testing results.

- The starting point is a specification of a property and a distance function, *the combination of which yields a promise problem* (of the first type).

  [Needless to say, the starting point is common to all property testing work, but the fact that it constitutes a promise problem is rarely stated.]

- The first step is to postulate that the potential "solvers" (i.e., property testers) are probabilistic oracle machines that are given oracle access to the "primary" input (i.e., the object in the aforementioned problem types).

  [Indeed, this step need to be taken and is taken in all works in the area.]

- Secondly, for a formal asymptotic complexity statement, one needs to specify the "secondary" (explicit) inputs, which consist of various problem-dependent parameters (e.g., N and d in the above examples) and the distance parameter δ (in case of $\mathtt{BPG}_d$ and any other problem of the first aforementioned type).

  [This step is rarely done explicitly in the literature. The importance of this step is highlighted in [5, 6], which explicitly distinguish testers that decide obliviously of N from general testers

[3]Needless to say, a higher level of rigor is typically required in negative statements. Indeed, property testing is positioned between algorithmic research and complexity theory, and seems to be more influenced by the mind-frame of algorithmic research. (We comment that the positioning of a discipline is determined both by its contents and by sociology-of-science factors.)

the decision of which may depend on N even in case their query complexity is independent of N.]

- Finally, one should state the complexity of the tester in terms of these explicit inputs.

  [Needless to say, this is always done...]

Thus, the only step that is acutely missing in typical works is a rigorous definition of the relevant explicit inputs (especially, the various problem-dependent parameters). Regardless of whether or not one explicitly uses the promise problem formulation, we suggest to be more careful about specifying the (problem-dependent) explicit inputs given to the tester.

References

[1] N. Alon. Testing subgraphs of large graphs. *Random Structures and Algorithms*, Vol. 21, pages 359–370, 2002.

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

[3] N. Alon and M. Krivelevich. Testing k-Colorability. *SIAM Journal on Disc. Math.*, Vol. 15 (2), pages 211-227, 2002.

[4] N. Alon and A. Shapira. Every Monotone Graph Property is Testable. In *37th STOC*, pages 128–137, 2005.

[5] N. Alon and A. Shapira. A Characterization of the (natural) Graph Properties Testable with One-Sided. In *46th FOCS*, to appear, 2005.

[6] N. Alon and A. Shapira. A Separation Theorem in Property Testing. Unpublished manuscript, 2004.

[7] M. Bender and D. Ron. Testing acyclicity of directed graphs in sublinear time. *Random Structures and Algorithms*, pages 184–205, 2002.

[8] A. Bogdanov, K. Obata, and L. Trevisan. A lower bound for testing 3-colorability in bounded-degree graphs. In *43rd FOCS*, pages 93–102, 2002.

[9] A. Bogdanov and L. Trevisan. Lower Bounds for Testing Bipartiteness in Dense Graphs. In *IEEE Conference on Computational Complexity*, pages 75–81, 2004.

[10] B. Chazelle, R. Rubinfeld, and L. Trevisan. Approximating the minimum spanning tree weight in sublinear time. In *19th ICALP*, pages 190–200, 2001.

[11] S. Even, A.L. Selman, and Y. Yacobi. The Complexity of Promise Problems with Applications to Public-Key Cryptography. *Inform. and Control*, Vol. 61, pages 159–173, 1984.

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

[13] E. Fischer and I. Newman. Testing versus estimation of graph properties. In *37th STOC*, pages 138–146, 2005.

[14] O. Goldreich. Combinatorial Property Testing – A Survey. In *DIMACS Series in Disc. Math. and Theoretical Computer Science*, Vol. 43 (Randomization Methods in Algorithm Design), pages 45–59, 1998.

[15] O. Goldreich. On Promise Problems: In memory of Shimon Even (1935–2004). *ECCC*, TR05-018, January 2005.

[16] O. Goldreich, S. Goldwasser, and D. Ron. Property testing and its connection to learning and approximation. *Journal of the ACM*, pages 653–750, July 1998.

[17] O. Goldreich and D. Ron. Property testing in bounded degree graphs. *Algorithmica*, pages 302–343, 2002.

[18] O. Goldreich and D. Ron. A sublinear bipartite tester for bounded degree graphs. *Combinatorica*, Vol. 19 (3), pages 335–373, 1999.

[19] O. Goldreich and D. Ron. On Testing Expansion in Bounded-Degree Graphs. *ECCC*, TR00-020, March 2000.

[20] O. Goldreich and D. Ron. Approximating Average Parameters of Graphs. *ECCC*, TR05-073, July 2005.

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

[22] D. Hochbaum (ed.). *Approximation Algorithms for NP-Hard Problems.* PWS, 1996.

[23] T. Kaufman, M. Krivelevich, and D. Ron. Tight Bounds for Testing Bipartiteness in General Graphs. *SIAM Journal on Computing*, Vol. 33 (6), pages 1441–1483, 2004.

[24] M. Parnas and D. Ron. Testing the diameter of graphs. *Random Structures and Algorithms*, Vol. 20 (2), pages 165–183, 2002.

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

[26] R. Rubinfeld and M. Sudan. Robust characterization of polynomials with applications to program testing. *SIAM Journal on Computing*, 25(2), pages 252–271, 1996.