Electronic Colloquium on Computational Complexity, Report No. 12 (2012)

On the Effect of the Proximity Parameter on Property Testers

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

February 9, 2012

Abstract

This note refers to the effect of the proximity parameter on the operation of (standard) property testers. Its bottom-line is that, except in pathological cases, the effect of the proximity parameter is restricted to determining the query complexity of the tester. The point is that, in non-pathological cases, the mapping of the proximity parameter to the query complexity can be reversed in an adequate manner.

1 Introduction

Property Testing is the study of super-fast (randomized) algorithms for approximate decision making. These algorithms are given direct access to items of a huge data set, and determine whether this data set has some predetermined (global) property or is far from having this property. Remarkably, this approximate decision is made by accessing a small portion of the data set. Thus, property testing is a relaxation of decision problems and it focuses on algorithms, called *testers*, that can only read parts of the input.

A basic consequence of the foregoing description is that the testers should be modeled as oracle machines and the input should be modeled as a function to which the tester has oracle access. This modeling convention is explicit in almost all studies of property testing, but what is sometimes not explicit is that the tester also gets ordinary inputs (i.e., inputs that are given as strings and are read for free by the tester). These inputs include (1) the proximity parameter, denoted ϵ, and (2) parameters that describe the domain of the function (at the very least the size of the domain is given as input).[1] Note that the description of the domain must be provided so to allow the tester to make adequate queries.[2] The proximity parameter must also be provided, for reasons detailed next.

Recall that the standard definition of a tester (see Section 2)[3] requires that it accepts (with probability at least 2/3) any function that has some predetermined property but rejects (with

[1]For example, if the domain is a finite field, then one may need to provide its representation (and not merely its size), especially when no standard representation can be assumed (e.g., as in the case that the field has 2^n elements). Another example refers to the bounded-degree graph model (cf. [3]), where one should also provide the degree bound rather than just its product with the number of vertices; actually, one typically provides the degree bound and the number of vertices (and does not provide their multiple).

[2]This crucial fact was overlooked in [5], as pointed out in [1, 6].

[3]We refer to the standard definition (as in, e.g., [2, 7]), and not to the definition of a proximity-oblivious tester (cf. [4]).

ISSN 1433-8092

probability at least 2/3) any function that is ϵ-far from the set of functions having the property, where distances between functions are defined as the fraction of the domain on which the functions disagree. Note that, except in degenerated cases, one may avoid querying the function on its entire domain D only if $\epsilon > 1/|D|$. Thus, the tester must know that this is the case (i.e., that $\epsilon > 1/|D|$), if it is to make less than $|D|$ queries. In general, the query complexity of the tester typically depends on ϵ, and so the tester must obtain ϵ in order to determine its query complexity. The question addressed in this note is whether or not ϵ is needed for any other purpose.

The foregoing natural question has also a concrete motivation. Various studies of property testing seem to assume that the tester only uses the proximity parameter to determine its query complexity (see, e.g., [5, 6]). We show that this assumption is essentially justified: See Section 2.

2 Technical Treatment

An asymptotic analysis is enabled by considering an infinite (indexed) sequence of domains, functions, and properties. That is, for any $s \in \mathbb{N}$, we consider functions from D_s to R_s. Indeed, s may be thought of as a description of the domain D_s, and typically it is related to $|D_s|$ (e.g., $s = |D_s|$).

Definition 1 *Let* $\Pi = \bigcup_{s\in\mathbb{N}} \Pi_s$*, where* Π_s *contains functions defined over the domain* D_s *and range* R_s*. A* tester for a property Π *is a probabilistic oracle machine* T *that satisfies the following two conditions:*

1. *The tester accepts each* $f \in \Pi$ *with probability at least* 2/3*; that is, for every* $s \in \mathbb{N}$ *and* $f \in \Pi_s$ (and every $\epsilon > 0$)*, it holds that* $\Pr[T^f(s,\epsilon)=1] \geq 2/3$.
2. *Given* $\epsilon > 0$ *and oracle access to any* f *that is* ϵ*-far from* Π*, the tester rejects with probability at least* 2/3*; that is, for every* $\epsilon > 0$ *and* $s \in \mathbb{N}$*, if* $f : D_s \to R_s$ *is* ϵ*-far from* Π_s*, then* $\Pr[T^f(s,\epsilon)=0] \geq 2/3$*, where* f *is* ϵ-far from Π_s *if, for every* $g \in \Pi_s$*, it holds that* $|\{e \in D_s : f(e) \neq g(e)\}| \geq \epsilon \cdot |D_s|$.

If the tester accepts every function in Π *with probability 1, then we say that it has* one-sided error*; that is,* T *has one-sided error if for every* $f \in \Pi_s$ *and every* $\epsilon > 0$*, it holds that* $\Pr[T^f(s,\epsilon)=1] = 1$*. A tester is called* non-adaptive *if it determines all its queries based solely on its internal coin tosses* (and the parameters n and ϵ)*; otherwise it is called* adaptive.

The query complexity of T, viewed as a function of $|D_s|$ and ϵ, is an upper bound (which holds for all $f : D_s \to R_s$) on the number of queries that T makes on (explicit) input (s,ϵ).

For the sake of simplicity, we assume that $s = |D_s|$, which means that the function domain is fully specified by its size. The following proposition holds also when s is an arbitrary specification of the function's domain, which determines the domain's size (or just allows to determine $q(|D_s|,\epsilon)$ for any given ϵ).

Proposition 2 (on the use of the proximity parameter): *Let* $q : \mathbb{N} \times (0,1) \to \mathbb{N}$ *be a computable function that is monotonically non-increasing in its second variable. Suppose that property* Π *has a tester* T *of query complexity* q*. Then,* Π *has a tester* T' *of query complexity* q *that only uses the proximity parameter to determine its query complexity* (i.e., on input parameters (s,ϵ), this tester first computes and records $q(s,\epsilon)$, and then continues after omitting ϵ from its record)*. Furthermore, if* T *has one-sided error and/or is non-adaptive, then so is* T'.

A typical case of a function q for which the hypothesis holds is $q(s,\epsilon) \stackrel{\text{def}}{=} \lceil f_s(\epsilon) \rceil$ for some collection of continuous and monotonically decreasing functions $f_s : (0,1) \to \mathbb{N}$ (e.g., $f_s(\epsilon) = 100/\epsilon^2$ or $f_s(\epsilon) = 2^{5/\epsilon} \cdot \log_2 s$).

Proof. Consider the following algorithm T':

1. On input parameters s and ϵ, algorithm T' computes $\rho \leftarrow \lceil s\epsilon \rceil / s$ and $v \leftarrow q(s, \rho)$.

2. Next, T' computes the minimal k such that $q(s, k/s) = v$.

3. Algorithm T' invokes T on input parameters s and k/s.

Note that T' only uses ϵ to determine its query complexity, and that it maintains many features of T (e.g., non-adaptivity and one-sided error probability). In typical cases, the overhead in the time complexity is not significant (since k can be determined by a binary search).

In analyzing T', we first consider any $s \in \mathbb{N}$ and $f \in \Pi_s$. Then, for every ϵ', it holds that $\Pr[T^f(s, \epsilon') = 1] \geq c$, where $c = 1$ if T has one-sided error and $c = 2/3$ otherwise. Clearly, this holds also for $\epsilon' = k/s$, where k is as determined in Step 1, and therefore $\Pr[T'^f(s, \epsilon) = 1] = \Pr[T^f(s, k/s) = 1] \geq c$. Suppose, on the other hand, that $f : D_s \rightarrow R_s$ is ϵ-far from Π. Since $k/s \leq \rho = \lceil s\epsilon \rceil / s$ (and the distance between functions over D_s is a multiple of $1/s$), it follows that f is k/s-far from Π, and therefore $\Pr[T'^f(s, \epsilon) = 0] = \Pr[T^f(s, k/s) = 0] \geq 2/3$. The proposition follows. ■

Comment. An alternative presentation may suggest to invoke T on proximity parameter ϵ' that is chosen as the minimum for which $q(s, \epsilon') = v$ holds. This, seemingly more elegant approach, requires assuming that for every $s, v \in \mathbb{N}$ the set $\{\epsilon \in (0, 1) : q(s, \epsilon) = v\}$ is either empty or has a minimum element. More annoyingly, this minimum may have an infinite binary expansion, and so an actual algorithm will need to use a truncation of it anyhow. Indeed, one may always assume that the value of the proximity parameter is a multiple of $1/s$.

References

[1] N. Alon and A. Shapira. A Characterization of the (natural) Graph Properties Testable with One-Sided. *SIAM Journal on Computing*, Vol. 37 (6), pages 1703–1727, 2008.

[2] 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.

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

[4] O. Goldreich and D. Ron. On Proximity Oblivious Testing. *SIAM Jour. on Comput.*, Vol. 40, No. 2, pages 534–566, 2011.

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

[6] O. Goldreich and L. Trevisan. Errata to [5]. Manuscript, August 2005. Available from `http://www.wisdom.weizmann.ac.il/~oded/p_ttt.html`

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

ECCC ISSN 1433-8092

http://eccc.hpi-web.de