Introduction to Property Testing

Property testing is concerned with the design of superfast algorithms for the structural analysis of large quantities of data. The aim is to unveil global features of the data, such as determining whether the data have a particular property or estimating global parameters. Remarkably, it is possible to achieve this aim by accessing only a small portion of the data. Property testing focuses on properties and parameters that go beyond simple statistics.

This book provides an extensive and authoritative introduction to property testing. It provides a wide range of algorithmic techniques for the design and analysis of tests for algebraic properties, properties of Boolean functions, graph properties, and properties of distributions.

Oded Goldreich is Professor of Computer Science at the Weizmann Institute of Science and holds the Meyer W. Weisgal Professorial Chair. He has made numerous contributions to property testing and to the theory of computation at large, and was awarded the 2017 Knuth Prize for this work. He is the author of several books, including the two-volume work *Foundations of Cryptography* and the book *Computational Complexity: A Conceptual Perspective*. He is an associate editor of the journal *Computational Complexity*, former editor of the *Journal of Cryptology* and *SIAM Journal on Computing*, and has been invited to speak at numerous conferences including the 1994 International Congress of Mathematicians.

Introduction to Property Testing

Oded Goldreich

Weizmann Institute of Science, Israel

CAMBRIDGE
UNIVERSITY PRESS

Shaftesbury Road, Cambridge CB2 8EA, United Kingdom

One Liberty Plaza, 20th Floor, New York, NY 10006, USA

477 Williamstown Road, Port Melbourne, VIC 3207, Australia

314–321, 3rd Floor, Plot 3, Splendor Forum, Jasola District Centre, New Delhi – 110025, India

103 Penang Road, #05–06/07, Visioncrest Commercial, Singapore 238467

Cambridge University Press is part of Cambridge University Press & Assessment,
a department of the University of Cambridge.

We share the University's mission to contribute to society through the pursuit of
education, learning and research at the highest international levels of excellence.

www.cambridge.org
Information on this title: www.cambridge.org/9781107194052

DOI: 10.1017/9781108135252

First published 2017

A catalogue record for this publication is available from the British Library

Library of Congress Cataloging-in-Publication data
Names: Goldreich, Oded, author.
Title: Introduction to property testing / Oded Goldreich, Weizmann Institute of Science, Israel.
Description: Cambridge, United Kingdom ; New York, NY, USA : Cambridge University Press, 2018. |
Includes bibliographical references and index.
Identifiers: LCCN 2017023051 | ISBN 9781107194052 (hardback : alk. paper)
Subjects: LCSH: Computer algorithms. | Structural analysis (Engineering)
Classification: LCC QA76.9.A43 G655 2018 | DDC 005.1 – dc23
LC record available at https://lccn.loc.gov/2017023051

ISBN 978-1-107-19405-2 Hardback

Contents

Preface *page* xi
Organization and Chapter Summaries xv
Notation xxiii
Standard Notation xxiii
Specific Notation Used Extensively xxiv
Acknowledgments xxv

1 The Main Themes: Approximate Decision and Sublinear Complexity 1
1.1 Introduction 1
1.1.1 Property Testing at a Glance 2
1.1.2 On the Potential Benefits of Property Testers 3
1.1.3 On the Flavor of Property Testing Research 4
1.1.4 Organization and Some Notations 5
1.2 Approximate Decisions 6
1.2.1 A Detour: Approximate Search Problems 6
1.2.2 Property Testing: Approximate Decision Problems 6
1.2.3 Property Testing: Sublinear Complexity 7
1.2.4 Symmetries and Invariants 10
1.2.5 Objects and Representation 13
1.3 Notions, Definitions, Goals, and Basic Observations 14
1.3.1 Basics 15
1.3.2 Ramifications 18
1.3.3 Proximity-Oblivious Testers (POTs) 19
1.3.4 The Algebra of Property Testing 22
1.3.5 Testing via Learning 26
1.4 Historical Notes 28
1.5 Suggested Reading and Exercises 30
Basic Exercises 31
Additional Exercises 36
1.6 Digest: The Most Important Points 38

2 Testing Linearity (Group Homomorphism) 40
2.1 Preliminaries 40
2.2 The Tester 40
2.3 Chapter Notes 45

3 Low-Degree Tests 49
3.1 A Brief Introduction 49
3.2 A Kind of Intuition (which may be skipped) 50
3.2.1 The Univariate Case 50
3.2.2 The Multivariate Case 51
3.2.3 Linking the Above Intuition to the Actual Proof 52
3.3 Background 53
3.4 The Tester 57
3.4.1 Analysis of the Tester 58
3.4.2 Digest (or an Abstraction) 64
3.5 Chapter Notes 65
Exercises 67

4 Testing Monotonicity 69
4.1 Introduction 69
4.2 Boolean Functions on the Boolean Hypercube 70
4.2.1 The Edge Test 70
4.2.2 Path Tests 75
4.3 Multivalued Functions on the Discrete Line 77
4.3.1 A Tester Based on Binary Search 77
4.3.2 Other Testers 81
4.4 Multivalued Functions on the Hypergrid 82
4.4.1 Dimension Reduction (Proof of Lemma 4.13) 84
4.4.2 Range Reduction (Overview of the Proof of Lemma 4.14) 85
4.5 Chapter Notes 87
4.5.1 History and Credits 87
4.5.2 Related Problems 88
4.5.3 Exercises 89

5 Testing Dictatorships, Juntas, and Monomials 92
5.1 Introduction 92
5.2 Testing Dictatorship via Self-correction 93
5.2.1 The Tester 94
5.2.2 Testing Monomials 97
5.2.3 The Self-correction Paradigm: An Abstraction 101
5.3 Testing Juntas 105
5.4 Chapter Notes 112
Basic Exercises 112
Additional Exercises 115

6 Testing by Implicit Sampling 120
6.1 Introduction 120
6.2 Testing Subsets of k-Juntas 121
6.3 Extension to Properties Approximated by Subsets of k-Juntas 128
6.4 Chapter Notes 131
On Testing Problems Associated with Sets of Boolean Functions 131
Exercises 132

7 Lower Bounds Techniques 134
7.1 Introduction 134
7.2 Indistinguishability of Distributions 135
7.2.1 The Actual Method 137
7.2.2 Illustrating the Application of the Method 140
7.2.3 Further Reflections 141
7.3 Reduction from Communication Complexity 142
7.3.1 Communication Complexity 143
7.3.2 The Methodology 144
7.3.3 Illustrating the Application of the Methodology 146
7.4 Reduction among Testing Problems 149
7.5 Lower Bounds for Restricted Testers 153
7.5.1 One-Sided Error Testers 153
7.5.2 Nonadaptive Testers 154
7.6 Chapter Notes 157
Exercises 157

8 Testing Graph Properties in the Dense Graph Model 162
8.1 The General Context: Introduction to Testing Graph Properties 163
8.1.1 Basic Background 163
8.1.2 Three Models of Testing Graph Properties 164
8.2 The Dense Graph Model: Some Basics 167
8.2.1 The Actual Definition 167
8.2.2 Abuses of the Model: Trivial and Sparse Properties 169
8.2.3 Testing Degree Regularity 170
8.2.4 Digest: Levin's Economical Work Investment Strategy 175
8.3 Graph Partition Problems 176
8.3.1 Testing Bipartiteness 180
8.3.2 The Actual Definition and the General Result 184
8.4 Connection to Szemeŕedi's Regularity Lemma 187
8.4.1 The Regularity Lemma 188
8.4.2 Subgraph Freeness 189
8.4.3 The Structure of Properties That Have Size-Oblivious Testers 196
8.5 A Taxonomy of the Known Results 197
8.5.1 Testability in $q(\epsilon)$ Queries, for any Function q 199
8.5.2 Testability in $\mathrm{poly}(1/\epsilon)$ Queries 200
8.5.3 Testability in $\widetilde{O}(1/\epsilon)$ Queries 201
8.5.4 Additional Issues 202
8.6 Chapter Notes 204
8.6.1 Historical Perspective and Credits 204
8.6.2 Testing versus Other Forms of Approximation 206
8.6.3 A Contrast with Recognizing Graph Properties 207
8.6.4 Exercises 208

9 Testing Graph Properties in the Bounded-Degree Graph Model 213
9.1 The Bounded-Degree Model: Definitions and Issues 214
9.2 Testing by a Local Search 218
9.2.1 Testing Subgraph Freeness 218

9.2.2 Testing Degree Regularity 219
9.2.3 Testing Connectivity 221
9.2.4 Testing t-Connectivity (Overview and One Detail) 223
9.2.5 Testing Cycle-Freeness (with Two-Sided Error) 227
9.3 Lower Bounds 230
9.3.1 Bipartiteness 230
9.3.2 Applications to Other Properties 232
9.3.3 Linear Lower Bounds 235
9.4 Testing by Random Walks 236
9.4.1 Testing `Bipartiteness` 236
9.4.2 One-Sided Error Tester for `Cycle-Freeness` 243
9.5 Testing by Implementing and Utilizing Partition Oracles 248
9.5.1 The Simpler Implementation 252
9.5.2 The Better Implementation 257
9.6 A Taxonomy of the Known Results 259
9.6.1 Testability in $q(\epsilon)$ Queries, for Any Function q 259
9.6.2 Testability in $\widetilde{O}(k^{1/2}) \cdot \mathrm{poly}(1/\epsilon)$ Queries 260
9.6.3 Additional Issues 261
9.7 Chapter Notes 262
9.7.1 Historical Perspective and Credits 262
9.7.2 Directed Graphs 263
9.7.3 Exercises 264

10 Testing Graph Properties in the General Graph Model 271
10.1 The General Graph Model: Definitions and Issues 272
10.1.1 Perspective: Comparison to the Two Previous Models 272
10.1.2 The Actual Definition 273
10.2 On Obtaining Testers for the Current Model 275
10.2.1 An Explicit Adaptation: The Case of Connectivity 276
10.2.2 Using a Reduction: The Case of Bipartiteness 277
10.3 Estimating the Average Degree and Selecting Random Edges 281
10.3.1 Lower Bounds 282
10.3.2 Algorithms 283
10.4 Using Adjacency Queries: The Case of Bipartiteness 293
10.5 Chapter Notes 296
10.5.1 Gaps between the General Graph Model and the Bounded-Degree Model 296
10.5.2 History and Credits 298
10.5.3 Reflections 298
10.5.4 Exercises 300

11 Testing Properties of Distributions 304
11.1 The Model 304
11.1.1 Testing Properties of Single Distributions 305
11.1.2 Testing Properties of Pairs of Distributions 307
11.1.3 Label-invariant Properties 308
11.1.4 Organization 309

11.2 Testing Equality to a Fixed Distribution 309
11.2.1 The Collision Probability Tester and Its Analysis 310
11.2.2 The General Case (Treated by a Reduction to Testing Uniformity) 313
11.2.3 A Lower Bound 319
11.3 Testing Equality between Two Unknown Distributions 320
11.3.1 Detour: Poisson Distributions 321
11.3.2 The Actual Algorithm and Its Analysis 324
11.3.3 Applications: Reduction to the Case of Small Norms 330
11.4 On the Complexity of Testing Properties of Distributions 335
11.5 Chapter Notes 340
11.5.1 History and Credits 340
11.5.2 Exercises 341

12 Ramifications and Related Topics 348

12.1 Tolerant Testing and Distance Approximation 348
12.2 Additional Promises on the Input 351
12.3 Sample-Based Testers 352
12.4 Testing with Respect to Other Distance Measures 353
12.5 Local Computation Algorithms 355
12.5.1 Definitions 356
12.5.2 Finding Huge Structures 358
12.5.3 Local Reconstruction 360
12.6 Noninteractive Proofs of Proximity (MAPs) 361
12.7 Chapter Notes 365
12.7.1 Historical Notes 365
12.7.2 Massively Parameterized Properties 366
12.7.3 Exercises 366

13 Locally Testable Codes and Proofs 370

13.1 Introduction 371
13.2 Definitions 372
13.2.1 Codeword Testers 372
13.2.2 Proof Testers 374
13.2.3 Ramifications and Relation to Property Testing 376
13.2.4 On Relating Locally Testable Codes and Proofs 382
13.3 Results and Ideas 383
13.3.1 The Mere Existence of Locally Testable Codes and Proofs 384
13.3.2 Locally Testable Codes and Proofs of Polynomial Length 387
13.3.3 Locally Testable Codes and Proofs of Nearly Linear Length 397
13.4 Chapter Notes 403
13.4.1 Historical Notes 403
13.4.2 On Obtaining Superfast testers 404
13.4.3 The Alternative Regime: LTCs of Linear Length 405
13.4.4 Locally Decodable Codes 407
13.4.5 Exercises 408

Appendix A: Probabilistic Preliminaries 411
A.1 Notational Conventions 411
A.2 Some Basic Notions and Facts 412
A.3 Basic Facts Regarding Expectation and Variance 413
A.4 Three Inequalities 415
A.4.1 Markov's Inequality 415
A.4.2 Chebyshev's Inequality 416
A.4.3 Chernoff Bound 419
A.4.4 Pairwise Independent versus Totally Independent Sampling 422

Appendix B: A Mini-Compendium of General Results 423

Appendix C: An Index of Specific Results 427

References 429
Index 443

Preface

Property testing is concerned with the design of superfast algorithms for structural analysis of huge amounts of data, where by structural analysis we mean an analysis aimed at unveiling global features of the data. Examples include determining whether the data as a whole have some property or estimating some global parameter of the data. The focus is on properties and parameters that go beyond simple statistics of the type that refers to the frequency of occurrence of various local patterns. The 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 decision is made by accessing only a small portion of the data set.

In other words, property testing is concerned with the design of *superfast algorithms for approximate decision making*, where the decision refers to properties or parameters of huge objects. In particular, we seek algorithms that inspect only relatively small portions of the huge object. Such algorithms must be randomized and can provide only approximate answers. Indeed, two salient aspects of property testing are that (1) it studies algorithms that can read only parts of the input, and (2) it focuses on algorithms that solve "approximate decision" problems. Both aspects are quite puzzling: What can one do without even reading the entire input? What does approximate decision mean?

The answer is that these two aspects are indeed linked: *Approximate decision* means distinguishing objects that have some predetermined property (i.e., reside in some predetermined set) from objects that are "far" from having the property (i.e., are far from any object having the property), where the notion of distance employed here is the relative number of different symbols in the descriptions of the objects. Such approximate decisions may be valuable in settings in which an exact decision is infeasible or very expensive or just considerably more expensive than obtaining an approximate decision.

The point is that, in many cases, an approximate decision can be achieved by means of *superfast randomized algorithms*. One well-known example is the common practice of estimating various statistics by sampling, which can be cast as a small collection of approximate decision problems (with respect to some threshold values). Research in property testing aims to extend this useful practice to properties that cannot be cast as statistics of values (which are associated with individual members of a large population). Examples in which this goal was achieved include testing properties of functions such as being a low-degree polynomial, being monotone, and depending on a specified number of attributes; testing properties of graphs such as being bipartite and being triangle-free, and testing properties of geometric objects and visual images such as being well clustered and being a convex body.

Objects as Functions and Their Exploration. Viewing the input object as a function is natural in the context of algorithms that do not read their entire input. Such algorithms must probe the input at locations of their choice, and such probing can be thought of as querying a function that represents the input. The key point here is that the number of probes (or queries) is smaller than the size of the input, and so decisions are taken after seeing only a small part of the input. However, the inspected positions are not fixed but rather are chosen at random by the algorithm, possibly based on answers obtained to prior queries. Thus, in general, these algorithms may "explore" the input, rather than merely obtain its value at a uniformly selected sample of locations. Such exploration is most appealing when the tested input is a graph, which may be represented by a function (e.g., by its adjacency predicate), but the notion of exploration applies also in other cases.

Wider Perspective and Connections. Research in property testing may be both algorithmic and complexity theoretic. This is reflected both in its goals, which may be either the design of better algorithms or the presentation of lower bounds on their complexity, and in its tools and techniques. Such research is related to several areas of computer science and mathematics including combinatorics, statistics, computational learning theory, computational geometry, and coding theory. Historically, property testing was closely associated with the study of Probabilistically Checkable Proofs (PCPs), and some connections do exist between the two, but property testing is not confined to PCPs (and/or to the study of "locally testable codes").

This Book. The current book aims to provide an introduction to property testing, by presenting some of the main themes, results, and techniques that characterize and are used in the area. As usual in such cases, *the choice of material reflects a judgment of what is most adequate for presentation in the context of such an introductory text*, and this selection does not reflect lack of appreciation of the omitted material but rather an opinion that it is less suitable for the intended purpose of the text.

In addition to the selection of material for this book, several choices were made regarding the organization of the material and the amount of interdependencies among its parts.

Organizational Choices. We chose to organize the material by the type of objects and the properties being tested. By the "type of object" we refer to the natural perception of the object; for example, whether it is most naturally perceived as a function or as a graph. Within the world of functions, the types correspond to the structure of the domain on which the function is defined (e.g., a group, a vector space, a Boolean hypercube, or a hypergrid). The structure of the domain is often reflected in the invariances that are satisfied by the properties that we consider (e.g., affine invariance, closure under graph isomorphism, etc.). Hence, our organization may be viewed as structurally oriented. (Possible alternatives to our organization include an organization by techniques (as in Ron [242]) or by complexity levels, for example, whether the complexity of testing is independent of the size of the object, is mildly dependent on it, is barely sublinear, or somewhere in between.)[1]

[1] Denoting the size of the object by n, one may distinguish bounds that are independent of n from bounds that are polylogarithmic in n, bounds that are expressed by a constant power of n (i.e., n^c for $c \in (0, 1)$), or are barely sublinear such as $n/\mathrm{poly}(\log n)$.

We chose to present the material with as few links between chapters as possible. Of course, all chapters depend on the core notions that are introduced in the first chapter, but additional interdependencies are rare and never heavily relied upon. Hence, the ordering of the other chapters is not very important, although we preferred a specific one (for reasons outlined in the Organization and Chapter Summaries section).

More Specific Choices. We chose to use (one-sided error) proximity-oblivious testers (POTs) *whenever possible*. This reflects our belief that when a tester (implicitly or explicitly) consists of repeating a POT for a number of times that depends on the proximity parameter, one should focus on the POT itself and rely on the generic transformation from POTs to standard testers.

For the sake of uniformity, n always denotes the size of the object in its natural representation (which is not grossly redundant). Hence, objects are typically viewed as functions $f : [n] \to R_n$. Consequently, Boolean functions are presented as $f : \{0,1\}^\ell \to \{0,1\}$, where $n = 2^\ell$ (rather than as having domain $\{0,1\}^n$).

We made the choice of defining ϵ-far as the negation of ϵ-close. That is, a string x is ϵ-far from S if its relative distance from S is *strictly greater* than ϵ, and it is ϵ-close otherwise. We warn that in various sources different conventions are used regarding this minor issue.

The Use of Footnotes. We use footnotes quite heavily, and definitely much more often than is the norm in textbooks. Although this style is often criticized, it does offer the advantage of focusing on a main thread while deferring relevant elaborations of some related issues to an easy to locate place. We trust the reader to make the choice of whether to continue with the main thread or go for some elaborations of a point. Typical uses of such footnotes fall into two distinct categories. The first use is for the elaboration of technical details, which many readers may be willing to take on faith (and some may even figure out by themselves) but other readers may want to see fully justified before proceeding. The second use of footnotes is for advanced comments, which provide a somewhat wider perspective or refer to sources where such perspectives can be found.

Technical Style. At times, justifications for a sequence of (in)equalities appear after the sequence. This is typically done when we believe that these justifications are implicit in the text that preceded the sequence.

> **Teaching Note:** The book contains several teaching notes, set as boxed text such as this.

Required Preliminaries. There are no required preliminaries for this text, but basic familiarity with some notions and results of the theory of computation and probability theory will be useful. These include

1. The notions of decision, search, and promise problems (see, e.g., [131, Sec. 1.2]);
2. Probabilistic algorithms (see, e.g., [131, Sec. 6.1] or [212]); and
3. Basic notions and facts regarding discrete probability distributions, including probabilistic inequalities such as the Union Bound and Chernoff Bound (see Appendix A,

although this material is covered in many textbooks, including [131, Apdx. D.1] and [212]).

Website for Notices Regarding This Book. We intend to maintain a website for this book, listing corrections and updates of various types. The site is located at

http://www.wisdom.weizmann.ac.il/~oded/pt-intro.html

Organization and Chapter Summaries

All chapters rely on a few core notions that are introduced in Sections 1.3.1 and 1.3.3. Although these parts of Section 1.3 provide a sufficient basis for reading any of the subsequent chapters, we strongly recommend reading the entire first chapter before proceeding to any other one.

In contrast to the central role of Chapter 1, no other chapter is essential for reading the other chapters. In particular, interdependencies between the other chapters are rare and never heavily relied on. The main dependencies are depicted in Figure 1, where thematic dependencies are marked by solid lines and technical dependencies by dashed lines.

Although the ordering of the chapters that follow Chapter 1 is not very important, a choice had to be made. We chose to start with simple properties of functions such as group homomorphism, low-degree polynomials, monotonicity (with respect to various partial orders), and depending on few variables (i.e., juntas). In all these cases, the correspondence between the object and its representation is transparent: *the function is the object*. In contrast, when moving to graph properties, the question of representation arises in an acute manner, and three different chapters are devoted to three different representations that correspond to three different testing models. Hence, from the perspective of property testing per se, it seems to make sense to start with functions and then move to graphs.

In accordance with the foregoing, the first cluster of chapters (Chapters 2–6) deals with testing properties of functions, whereas a second cluster (Chapters 8–10) deals with testing properties of graphs. A chapter on lower bound techniques (i.e., Chapter 7) is located in between these two clusters, since lower bounds are hardly mentioned in the first cluster, whereas they appear quite prominently in Chapters 9 and 10. The reason for this phenomenon is that these lower bounds are used to justify the significantly higher complexity of some testers that are presented in Chapters 9 and 10. Indeed, in the context of this book, we view lower bounds mainly as a justification for algorithms that may be considered to have a higher than expected complexity; the lower bounds assert that this impression is actually false, and that one cannot do significantly better.

Chapters 11–13 form a third cluster, which is actually a cluster of outliers with respect to the rest of this book. These chapters are indeed related to the previous chapters and yet they have a different flavor: Chapter 11 deals with testing properties of distributions, Chapter 12 explores a few variants of the basic setting (some of which were mentioned in Section 1.3.2), and Chapter 13 reviews locally testable codes and proofs. We stress that in Chapter 11 the tested objects are fundamentally different from those considered in all the other chapters, whereas in Chapter 13 we consider objects that are artificially designed so to offer superfast testing.

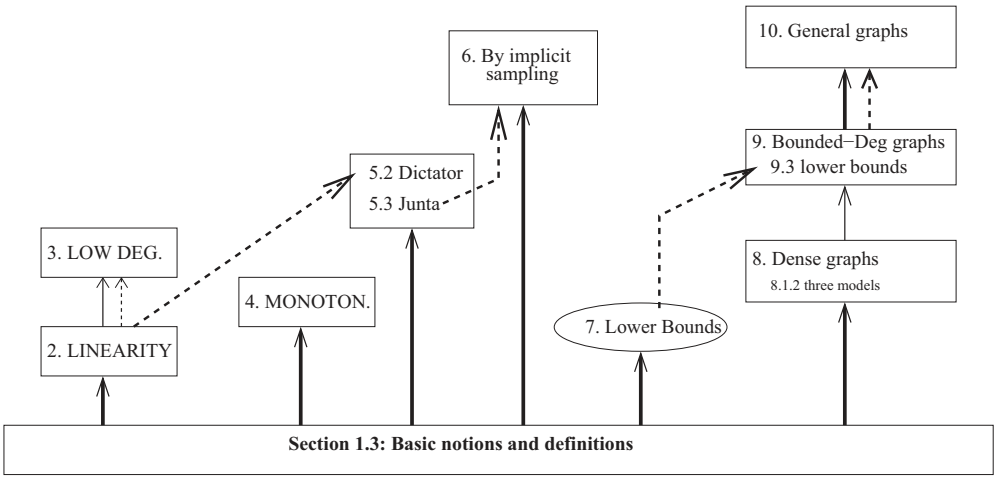

Figure 1: Dependencies among Chapters 1–10. Thicker lines represent greater dependency.

Chapter 1: The Main Themes (Approximate Decision and Sublinear Complexity). This chapter introduces and illustrates the basic notions of property testing, emphasizing the themes of *approximate decision* and *sublinear complexity*. The chapter starts with a discussion of the potential benefits of property testing and culminates with a presentation of the definitions of (standard) testers and of proximity-oblivious testers (POTs). These definitions (and the relationship between them) are used extensively throughout the book. In addition, Chapter 1 discusses the key role of representation, points out the focus on properties that are not fully symmetric, and presents several general observations regarding POTs, testing, and learning.

> **Teaching Note:** The conceptual framework put forward in Chapter 1 is pivotal for the rest of the book. It is essential that the main definitions (presented in Sections 1.3.1 and 1.3.3) and the rationales underlying them be internalized.

The following five chapters deal with properties of objects that are most naturally viewed as functions: Chapters 2 and 3 deal with algebraic properties, whereas Chapters 4–6 deal mostly with properties of Boolean functions. This distinction is quite fuzzy, and is reflected in the difference between the invariants that these properties satisfy: Algebraic properties are invariant under general affine transformations of the corresponding vector spaces, whereas the properties of Boolean functions that we consider are invariant only under transformations that permute the basis vectors.

Chapter 2: Testing Linearity (Group Homomorphism). This chapter presents an analysis of a linearity tester that, on input a description of two groups G, H and oracle access to a function $f : G \to H$, queries the function at three points and satisfies the following conditions:

1. If f is a homomorphism from G to H, then the tester accepts with probability 1.
2. If f is δ-far from the set of all homomorphisms from G to H, then the tester rejects with probability at least $\min(0.5\delta, 0.1666)$.

The three queries are $x, y, x+y$, where x and y are selected uniformly at random in G. The archetypical case is that H is a finite field and G is a vector space over this field.

Chapter 3: Low-Degree Tests. For a finite field of prime cardinality $\mathcal{F}$, a degree bound $d < |\mathcal{F}|/2$, and number $m \in \mathbb{N}$, we consider the problem of testing whether a function $f : \mathcal{F}^m \to \mathcal{F}$ is a polynomial of total degree at most d. We present and analyze a low-degree tester that, given oracle access to $f : \mathcal{F}^m \to \mathcal{F}$, queries it at $d+2$ points and satisfies the following conditions:

1. If f is an m-variate polynomial of (total) degree d, then the tester accepts with probability 1.
2. If f is δ-far from the set of m-variate polynomials of (total) degree d, then the tester rejects with probability at least $\min(0.5\delta, \Omega(d^{-2}))$.

The sequence of queries is generated by selecting at random $\bar{x}$ and $\bar{h}$ uniformly in $\mathcal{F}^m$, and using $\bar{x} + i\bar{h}$ as the i^{th} query.

> **Teaching Note:** The analysis of the low-degree test is quite similar to the analysis of the linearity test, alas it is more complex (let alone that it depends on elementary preliminaries that are presented in Section 3.3). Hence, if short on time, then do consider skipping Chapter 3.

Chapter 4: Testing Monotonicity. For each n, we consider functions from a partially ordered set D_n to a totally ordered set R_n. Such a function $f : D_n \to R_n$ is called monotone if for every $x < y$ in D_n it holds that $f(x) \leq f(y)$, where $<$ denotes the partial order of D_n and $\leq$ refers to the total order in R_n. Two special cases of interest are

1. Boolean functions on the Boolean hypercube: In this case, D_n is the ℓ-dimensional Boolean hypercube (with the natural partial order), where $\ell = \log_2 n$, and $R_n = \{0,1\}$. According to this partial order, $x_1 \cdots x_\ell \leq y_1 \cdots y_\ell$ if and only if $x_i \leq y_i$ for every $i \in [\ell]$.
2. Real functions on the discrete line: In this case, $D_n = [n]$ and $R_n = \mathbb{R}$, both with the natural total order.

Combining these two extremes, we also consider the case of the hypergrid domain $D_n = [m]^\ell$, for any $m, \ell \in \mathbb{N}$ such that $m^\ell = n$, and general ranges R_n. In all these cases, we present property testers of complexity $\text{poly}(\epsilon^{-1} \log n)$. In addition, we briefly survey relatively recent developments regarding the first case as well as known results regarding testing convexity, submodularity, and the Lipschitz property of functions from $[m]^\ell$ to $\mathbb{R}$.

> **Teaching Note:** Only parts of Chapter 4 (i.e., Sections 4.2.1 and 4.3.1) are recommended for teaching, and the rest is better left for optional independent reading.

Chapter 5: Testing Dictatorships, Juntas, and Monomials. We consider testing three basic properties of Boolean functions of the form $f : \{0,1\}^\ell \to \{0,1\}$:

1. Dictatorship: The case where the value of f depends on a single Boolean variable (i.e., $f(x) = x_i \oplus \sigma$ for some $i \in [\ell]$ and $\sigma \in \{0,1\}$)

2. Junta (of size k): The case where the value of f depends on at most k Boolean variables (i.e., $f(x) = f'(x_I)$ for some k-subset $I \subset [\ell]$ and $f' : \{0,1\}^k \to \{0,1\}$)
3. Monomial (of size k): The case where the value of f is the conjunction of exactly k Boolean literals (i.e., $f(x) = \wedge_{i \in I}(x_i \oplus \sigma_i)$ for some k-subset $I \subseteq [\ell]$ and $\sigma_1, ..., \sigma_\ell \in \{0,1\}$)

We present two different testers for dictatorship, where one generalizes to testing k-juntas and the other generalizes to testing k-monomials. (The presentation starts with the latter tester for dictatorship, sketches its generalization to testing k-monomials, and ends with the tester for k-juntas.)

> **Teaching Note:** We suggest leaving the overview section that discusses testing monomials (i.e., Section 5.2.2) for advanced independent reading.

Chapter 6: Testing by Implicit Sampling. Building on the junta tester, we present a general methodology for constructing testers for properties of Boolean functions (of the form $f : \{0,1\}^\ell \to \{0,1\}$) that can be approximated by small juntas. This methodology yields testers of low query complexity for many natural properties, which contain functions that depend on relatively few relevant variables; specifically, the query complexity is related to the size of the junta and is independent of the length of the input to the function (i.e., ℓ).

Chapter 7: Lower Bounds Techniques. We present and illustrate three techniques for proving lower bounds on the query complexity of property testers.

1. Showing a distribution on instances that have the property and a distribution on instances that are far from the property such that an oracle machine of low query complexity cannot distinguish these two distributions
2. Showing a reduction from communication complexity; that is, showing that a communication complexity problem of high complexity can be solved within communication complexity that is related to the query complexity of the property testing task that we are interested in
3. Showing a reduction from another testing problem; that is, showing a "local" reduction of a hard testing problem to the testing problem that we are interested in

We also present simplifications of these techniques for the cases of one-sided error probability testers and nonadaptive testers.

> **Teaching Note:** The first method (i.e., the method of "indistinguishability of distributions") is used much more often than the other two methods, and studying it should be at the highest priority.

The following three chapters deal with properties of objects that are most naturally viewed as graphs: These chapters consider three models that differ in the way that graphs are represented (and by the definition of relative distance between graphs).

Teaching Note: Chapters 8–10 contain material that may occupy half the duration of a course that is based on the current book. Hence, painful choices will have to be made, unless a decision is made to spend this amount of time on studying the topic of testing graph properties, which is not an unreasonable decision in light of the ubiquitous presence of graphs in computer science. Our own (painful) choices regarding the material to be taught appear in the introductory sections of these chapters.

Chapter 8: Testing Graph Properties in the Dense Graph Model. Following a general introduction to testing graph properties, this chapter focuses on the dense graph model, where graphs are represented by their adjacency matrix (predicate). The highlights of this chapter include

1. A presentation of a natural class of graph properties that can each be tested within query complexity that is polynomial in the reciprocal of the proximity parameter. This class, called general graph partition problems, contains properties such as t-`colorability` (for any $t \geq 2$) and properties that refer to the density of the max-clique and to the density of the max-cut in a graph.
2. An exposition of the connection of testing (in this model) to Szemeŕedi's Regularity Lemma. The starting point and pivot of this exposition is the existence of constant-query (one-sided error) proximity-oblivious testers for all subgraph freeness properties.

We conclude this chapter with a taxonomy of known testers, organized according to their query complexity.

Chapter 9: Testing Graph Properties in the Bounded-Degree Graph Model. This chapter is devoted to testing graph properties in the bounded-degree graph model, where graphs are represented by their incidence lists (lumped together in an incidence function). The highlights of this chapter include

1. Upper and lower bounds on the complexity of testing `Bipartiteness`; specifically, we present a $\mathrm{poly}(1/\epsilon) \cdot \widetilde{O}(\sqrt{k})$-time tester, and an $\Omega\sqrt{k})$ lower bound on the query complexity of any tester for `Bipartiteness`.
2. A quasi-$\mathrm{poly}(1/\epsilon)$-time tester for `Planarity`. The result extends to testing any minor-closed property (i.e., a graph property that is preserved under the omission of edges and vertices and under edge contraction).

We conclude this chapter with a taxonomy of known testers, organized according to their query complexity.

Chapter 10: Testing Graph Properties in the General Graph Model. This chapter is devoted to testing graph properties in the general graph model, where graphs are inspected via incidence and adjacency queries, and distances between graphs are normalized by their actual size (i.e., actual number of edges). The highlights of this chapter include

1. Demonstrating the derivation of testers for this model from testers for the bounded-degree graph model

2. Studying the tasks of estimating the number of edges in a graph and sampling edges uniformly at random

We conclude this chapter with some reflections regarding the three models of testing graph properties.

> **Teaching Note:** Although it is possible to study Chapter 10 without first studying Chapter 9, we strongly recommend not doing so. A basic familiarity with the bounded-degree graph model and some of the results regarding it will greatly facilitate the study of the general graph model. See further comments at the beginning of Chapter 10.

The last three chapters explore topics that are related to but significantly different from the topics studied in the previous chapters. Chapter 11 is most different in flavor, since it refers to a totally different type of objects and to a very different model of testing such objects. Chapter 13 seems more in line with the previous chapters, but it differs from them in considering objects that are artificially designed so to offer superfast testing. The topics explored in Chapter 12 are closest in spirit to those explored in previous chapters (and, indeed, some of these topics were mentioned in Section 1.3.2).

> **Teaching Note:** Chapter 11 can be read without reading any prior chapter (i.e., not even Sections 1.3.1 and 1.3.3), but some perspectives will be lost when doing so. Given the different flavor of this chapter, we recommend placing it at the end of a course based on the current book.

Chapter 11: Testing Properties of Distributions. This chapter provides an introduction to the study of testing properties of distributions, where the tester obtains samples of an unknown distribution (resp., samples from several unknown distributions) and is required to determine whether the distribution (resp., the tuple of distributions) has some predetermined property. We focus on the problems of testing whether an unknown distribution equals a fixed distribution and of testing equality between two unknown distributions. Our presentation is based on reductions from the general cases to some seemingly easier special cases. In addition, we also provide a brief survey of general results.

> **Teaching Note:** Chapters 12 and 13 are intended for optional independent reading. They both have more of the flavor of a survey than of a textbook. Chapter 12 follows up on topics that were mentioned briefly in prior chapters (including in Section 1.3.2). Chapter 13 focuses on topics that are somewhat related to property testing, while building on results presented in Chapters 2 and 3 (but doing so in a self-contained manner). Indeed, Chapter 13 can be read independently of the other chapters.

Chapter 12: Ramifications and Related Topics. We briefly review a few ramifications of the notion of property testers as well as related topics. The list includes tolerant testing and distance approximation; testing in the presence of additional promises on the input; sample-based testers; testing with respect to other distance measures; local computation algorithms; and noninteractive proofs of proximity (MAPs). The different sections of this chapter can be read independently of one another.

Chapter 13: Locally Testable Codes and Proofs. We survey known results regarding locally testable codes and locally testable proofs (known as PCPs). Local testability refers to approximately testing large objects based on a very small number of probes, each retrieving a single bit in the representation of the object. This yields superfast approximate testing of the corresponding property (i.e., being a codeword or being a valid proof). In terms of property testing, locally testable codes are error-correcting codes such that the property of being a codeword can be tested within low query complexity. As for locally testable proofs (PCPs), these can be viewed as massively parameterized properties that are testable within low query complexity such that the parameterized property is nonempty if and only if the corresponding parameter is in a predetermined set (of "valid statements"). Our first priority is minimizing the number of probes, and we focus on the case that this number is a constant. In this case (of a constant number of probes), we aim at minimizing the length of the constructs. That is, we seek locally testable codes and proofs of short length.

Appendix A: Probabilistic Preliminaries. This appendix presents background from probability theory, which is used extensively throughout the book. This background and preliminaries include conventions regarding random variables, basic notions and facts, and three useful probabilistic inequalities (i.e., Markov's Inequality, Chebyshev's Inequality, and Chernoff Bound).

Appendix B: A Mini-Compendium of General Results. This appendix restates several general results that were presented in this book, including deriving standard testers from POTs; positive results on the algebra of property testing; reducing testing to learning; the randomness complexity of testers; archetypical application of self-correction; and the effect of local reductions.

Appendix C: An Index of Specific Results. This appendix provides an index of all results regarding specific properties that were presented in this book. For each property, we provide only references to the sections (or statements) in which relevant results can be found.

Notation

Standard Notation

Sets and Sequences. We often consider the set $[n] = \{1, ..., n\}$, where n is a natural number. Likewise, we often consider the set

$$\Sigma^\ell = \{\sigma_1 \cdots \sigma_\ell : \sigma_1, ..., \sigma_\ell \in \Sigma\}$$

of all ℓ-long sequences over Σ, where often $\Sigma = \{0, 1\}$. For $x \in \Sigma^\ell$ and $i \in [\ell]$, we let x_i denote the i^{th} symbol of x, and for $I = \{i_1, ..., i_t\} \subseteq [\ell]$ such that $i_1 < \cdots < i_t$, we let $x_I = x_{i_1} \cdots x_{i_t} \in \Sigma^t$.

For a set S and a natural number $t \leq |S|$, we denote by $\binom{S}{t}$ the set of all t-subsets of S; that is, $\binom{S}{t} = \{S' \subseteq S : |S'| = t\}$. Needless to say, the size of $\binom{S}{t}$ equals $\binom{|S|}{t}$. Likewise, the set of all subsets of S is denoted 2^S; that is, $2^S = \bigcup_{t \geq 0} \binom{S}{t}$, where $\binom{S}{0} = \emptyset$.

Graphs. Unless explicitly stated differently, a graph $G = (V, E)$, consists of a pair of finite sets V and $E \subseteq \binom{V}{2}$. The elements of V are called vertices, and the elements of E are called edges. (That is, we consider simple (undirected) graphs with no self-loops and no parallel edges.)[2] Each edge consists of a pair of vertices, called its endpoints.

Integrality Issues. We often ignore integrality issues, treating values such as $\log n$ and $\sqrt{n}$ as if they were integers. In such cases, rounding in an adequate manner will do.

Probabilistic Notation. We denote the probability that $\chi(e)$ holds when e is distributed according to D by $\mathbf{Pr}_{e \sim D}[\chi(e)]$. When D is the uniform distribution over a set S, we shall write $\mathbf{Pr}_{e \in S}[\chi(e)]$ instead of $\mathbf{Pr}_{e \sim D}[\chi(e)]$. Often, when S or D is understood from the context, we just omit it from the notation and write $\mathbf{Pr}_e[\chi(e)]$. For more probabilistic preliminaries, see Appendix A.

Asymptotic Notation. We use standard notation such as O, Ω, Θ, and their tilde versions. By writing $f(n) = O(g(n))$ (resp. $f(n) = \Omega(g(n))$) we mean that there exists a positive constant c such that $f(n) \leq c \cdot g(n)$ (resp., $f(n) \geq c \cdot g(n)$) holds for all $n \in \mathbb{N}$. Likewise, $f(n) = \widetilde{O}(g(n))$ (resp. $f(n) = \widetilde{\Omega}(g(n))$) means that there exists a positive

[2] In contrast, one may consider (nonsimple) graphs in which E is a multiset of edges, and each edge is a multiset of size 2. An edge that consists of two copies of the same vertex is called a self-loop, and identical edges are called parallel.

constant c such that $f(n) \leq c \cdot (\log n)^c \cdot g(n)$ (resp., $f(n) \geq c \cdot g(n)/(\log n)^{1/c}$) holds for all $n \in \mathbb{N}$. We write $f(n) = \Theta(g(n))$ (resp., $f(n) = \widetilde{\Theta}(g(n))$) if both $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$ (resp., $f(n) = \widetilde{O}(g(n))$ and $f(n) = \widetilde{\Omega}(g(n))$) hold.

Common Abbreviations. We often use the following abbreviations:

- e.g. for example
- i.e. that is
- iff if and only if
- l.h.s. left hand side
- r.h.s. right hand side
- s.t. such that
- w.h.p. with high probability
- w.l.o.g. without loss of generality
- w.r.t. with respect to
- w.v.h.p. with very high probability

Typically, w.h.p. means with probability at least $1 - c$ for an arbitrary small constant $c > 0$, and w.v.h.p. means with probability at least $1 - \eta$ for a fastly decreasing function η in a relevant parameter.

Specific Notation Used Extensively

The following notions are redefined several times in this book (see, e.g., Sections 1.2.2 and 1.6).

The Notion of Distance. For $x, y \in \Sigma^n$, we consider their relative Hamming distance, denoted $\delta(x, y) \stackrel{\text{def}}{=} |\{i \in [n] : x_i \neq y_i\}|/n$. For $x \in \Sigma^n$ and $S \subseteq \Sigma^n$, we denote by $\delta_S(x)$ the relative Hamming distance of x from S; that is, $\delta_S(x)$ is the minimum, taken over all $z \in S \cap \{0,1\}^{|x|}$, of $\delta(x, z)$. (If $S = \emptyset$, then $\delta_S(x) = \infty$.) We say that x is ϵ-far from S (resp., ϵ-close to S) if $\delta_S(x) > \epsilon$ (resp., $\delta_S(x) \leq \epsilon$). The same notations are used for functions from $[n]$ to Σ; that is, for $f, g : [n] \to \Sigma$, we let $\delta(f, g) \stackrel{\text{def}}{=} |\{i \in [n] : f(i) \neq g(i)\}|/n$.

Acknowledgments

We wish to thank Noga Alon, Clement Canonne, Constantinos Daskalakis, Ilias Diakonikolas, Zeev Dvir, Shafi Goldwasser, Robi Krauthgamer, Or Meir, Krzysztof Onak, Sofya Raskhodnikova, Dana Ron, Ronitt Rubinfeld, Madhu Sudan, and Theertha Suresh for offering advice regarding specific topics.

We also benefitted from comments on parts of the text offered by Ben Berger, Clement Canonne, Tom Gur, Yael Hitron, Nikolay Karpov, Akash Kumar, Inbal Livni, and Roei Tell. Special thanks to Clement and Roei, who proofread significant portions of prior versions.