Average-Case Complexity

Average-Case Complexity

Andrej Bogdanov

DIMACS – Rutgers University
adib@dimacs.rutgers.edu

Luca Trevisan

UC Berkeley
luca@eecs.berkeley.edu

now
the essence of knowledge

Boston – Delft

Foundations and Trends® in Theoretical Computer Science

Published, sold and distributed by:
now Publishers Inc.
PO Box 1024
Hanover, MA 02339
USA
Tel. +1-781-985-4510
www.nowpublishers.com
sales@nowpublishers.com

Outside North America:
now Publishers Inc.
PO Box 179
2600 AD Delft
The Netherlands
Tel. +31-6-51115274

Library of Congress Control Number: 2006935853

The preferred citation for this publication is A. Bogdanov and L. Trevisan, Average-Case Complexity, Foundation and Trends® in Theoretical Computer Science, vol 2, no 1, pp 1–106, 2006

Printed on acid-free paper

ISBN: 1-933019-49-2

Foundations and Trends® in Theoretical Computer Science
Volume 2 Issue 1, 2006

Editorial Board

Editor-in-Chief:
Madhu Sudan
Department of CS and EE
MIT, Stata Center, Room G640
32 Vassar Street, Cambridge
Massachusetts 02139,
USA
madhu@mit.edu

Editorial Scope

Foundations and Trends® in Theoretical Computer Science will publish survey and tutorial articles in the following topics:

- Algorithmic game theory
- Computational algebra
- Computational aspects of combinatorics and graph theory
- Computational aspects of communication
- Computational biology
- Computational complexity
- Computational geometry
- Computational learning
- Computational Models and Complexity
- Computational Number Theory
- Cryptography and information security
- Data structures
- Database theory
- Design and analysis of algorithms
- Distributed computing
- Information retrieval
- Operations Research
- Parallel algorithms
- Quantum Computation
- Randomness in Computation

Information for Librarians
Foundations and Trends® in Theoretical Computer Science, 2006, Volume 2, 4 issues. ISSN paper version 1551-305X. ISSN online version 1551-3068. Also available as a combined paper and online subscription.

Foundations and Trends® in
Theoretical Computer Science
Vol. 2, No 1 (2006) 1–106

DOI: 10.1561/0400000004

now
the essence of knowledge

Average-Case Complexity

Andrej Bogdanov[1] and Luca Trevisan[2]

[1] *DIMACS – Rutgers University, adib@dimacs.rutgers.edu*
[2] *UC Berkeley, luca@eecs.berkeley.edu*

Abstract

We survey the average-case complexity of problems in NP.

We discuss various notions of good-on-average algorithms, and present completeness results due to Impagliazzo and Levin. Such completeness results establish the fact that if a certain specific (but somewhat artificial) NP problem is easy-on-average with respect to the uniform distribution, then all problems in NP are easy-on-average with respect to all samplable distributions. Applying the theory to natural distributional problems remain an outstanding open question. We review some natural distributional problems whose average-case complexity is of particular interest and that do not yet fit into this theory.

A major open question is whether the existence of hard-on-average problems in NP can be based on the $\mathrm{P} \neq \mathrm{NP}$ assumption or on related worst-case assumptions. We review negative results showing that certain proof techniques cannot prove such a result. While the relation between worst-case and average-case complexity for general NP problems remains open, there has been progress in understanding the relation between different "degrees" of average-case complexity. We discuss some of these "hardness amplification" results.

Contents

1

Introduction

The study of the average-case complexity of intractable problems began in the 1970s motivated by two distinct applications: the development of the foundations of cryptography and the search for methods to "cope" with the intractability of NP-hard problems.

All definitions of security for cryptographic problems require that any efficient algorithm that tries to "break" the protocol "succeeds" only with a very small probability. The formalizations of *breaking* and *succeeding* depend on the specific application, but it has been known since the 1980s that there is a unifying concept: no cryptographic task (e.g., electronic signature or data encryption) is possible unless *one-way functions* exist.[1] Informally, a one-way function is an efficiently computable function $f : \{0,1\}^* \to \{0,1\}^*$ that maps $\{0,1\}^n$ to $\{0,1\}^n$ and such that, if we are given $f(x)$ for a random $x \in \{0,1\}^n$, it is intractable (in time polynomial in n) to find a pre-image x' such that $f(x') = f(x)$. In particular, the existence of one-way functions implies that there is a search problem in NP (given $y \in \{0,1\}^n$, find $x \in \{0,1\}^n$ such that $f(x) = y$) that is intractable to solve on random inputs sampled from

[1] The realizability of many cryptographic tasks, in fact, is *equivalent* to the assumption that one-way functions exist.

a simple distribution (the distribution $f(x)$, where x is chosen randomly from $\{0,1\}^n$). The fact that all of cryptography is predicated on the existence of average-case intractable problems in NP is a main motivation for the study of the theory we describe in this study.

In particular, a long-standing open question is whether it is possible to *base the existence of one-way functions on the* $\mathrm{P} \neq \mathrm{NP}$ *assumption*, or related ones (such as NP-complete problems not allowing polynomial size circuits).

The second motivation for the study of the average-case complexity of problems in NP comes from the analysis of heuristic algorithms. Unless $\mathrm{P} = \mathrm{NP}$, we cannot hope for efficient algorithms that solve NP-complete problems exactly on all inputs. We may hope, however, for algorithms that are "typically efficient" on inputs sampled from distributions that occur in practice. In order to understand the limitations of such an approach, it would be desirable to have an "average-case analog" of the theory of NP-completeness. Such a theory would enable us to prove that for certain problems, with respect to certain distributions, it is impossible to have algorithms that perform well on "typical" inputs, unless an entire class of presumably intractable problems can be efficiently solved.

The basic foundations of such a theory have been laid out. Surprisingly, subtle difficulties arise even when just developing the analogs of trivial elements of the theory of NP-completeness, such as the definitions of *computational problem, efficient algorithm, reduction*, and *completeness*, and the equivalent complexity of *decision versus search* for NP-complete problems. In this study we will discuss these difficulties and show how they were resolved. We will see a number of results, insights, and proof techniques the usefulness of which goes beyond the study of average-case complexity.

The right techniques to apply such a theory to natural problems and distributions have not been discovered yet. From this point of view, the current state of the theory of average-case complexity in NP is similar to the state of the theory of inapproximability of NP optimization problems before the PCP Theorem.

Finding ways of *applying this theory to natural problems* is another outstanding open question in this area.

1.1 Roadmap

In this section we give an overview of the content of this survey.

1.1.1 Definitions of tractability

The first difficulty in developing a theory of average-case intractability is to come up with a formal definition of what it means for a problem to be "intractable on average" or, equivalently, what it means to be "average-case tractable." A natural definition would be to consider an algorithm efficient-on-average if it runs in *expected polynomial time*. Such a definition has various shortcomings (related to the fact that it is too restrictive). For example, if an algorithm A runs in time $t(x)$, and its simulation B (in a different model of computation) runs in time $t^2(x)$, it is natural that we would like our definition to be such that A is efficient-on-average if and only if B is. Suppose, however, that our inputs come from the uniform distribution, and that A runs in time n^2 on all inputs of length n, except on one input on which A takes time 2^n. Then the expected running time of A is polynomial but the expected running time of B is exponential. Looking at the *median* running time of an algorithm gives us a more robust measure of complexity, but still a very unsatisfactory one: if an algorithm runs in polynomial time on 70% of the inputs, and in exponential time on 30% of the inputs, it seems absurd to consider it an efficient-on-average algorithm. The right way to capture the notion of "efficient on typical instances" should be that it is fine for an algorithm to take a large amount of time on certain inputs, provided that such inputs do not occur with high probability: that is, inputs requiring larger and larger running times should have proportionally smaller and smaller probability. This is the idea of Levin's definition of average-case complexity. In (an equivalent formulation of) Levin's definition [53], an algorithm is polynomial-time-on-average if there is a constant $c > 0$ such that the probability, over inputs of length n, that the algorithm takes more than time T is at most $\text{poly}(n)/T^c$. As is usual with complexity theory, various choices can be made in the definition: we may look at deterministic algorithms, randomized algorithms, or non-uniform families of circuits. An additional choice is whether we require our algorithm to always be

correct, but possibly run in superpolynomial time on some inputs, versus requiring the algorithm to always run in polynomial time, but to give an incorrect answer to some inputs. This will lead to several possible definitions, each meaningful in some applications. (See Chapter 2.) The important thing will be that almost all the results we discuss in this study are based on reductions that preserve tractability under all of these definitions. Hence, the treatment of completeness, reductions, families of distributions, and decision versus search is independent of the specific notion of tractability that one is interested in.

1.1.2 Reductions between distributional problems

Let L be a decision problem and $\mathcal{D}$ be a distribution over inputs[2]; we call the pair $(L, \mathcal{D})$ a *distributional problem*. All the definitions of average-case tractability have a characteristic in common: an algorithm A is efficient for $(L, \mathcal{D})$ if a certain set of "bad" inputs has low probability under $\mathcal{D}$. (The bad inputs could be the ones where the algorithm A takes a very long time, or those on which A outputs an incorrect answer.) This motivates the following definition of reduction [53]: we say that $(L, \mathcal{D})$ reduces to $(L', \mathcal{D}')$ if there is a polynomial time computable function f such that $x \in L$ if and only if $f(x) \in L'$ and, in addition, for every input y, the probability of generating y by picking x at random according to $\mathcal{D}$ and then computing $f(x)$ is at most $\text{poly}(|x|)$ larger than the probability of sampling y at random from $\mathcal{D}'$.[3] The motivation for this definition is the following. Suppose that A' is a good algorithm for $(L', \mathcal{D}')$, so that the set B' of inputs that are bad for A' has a small probability according to $\mathcal{D}'$. Consider the following algorithm for $(L, \mathcal{D})$: on input x, output $A'(f(x))$. Now, the bad inputs for this algorithm are the inputs x such that $f(x) \in B'$. The probability of sampling such an x, according to $\mathcal{D}$, however, is upper bounded by $\text{poly}(|x|)$ times the probability of sampling an element of B' according to $\mathcal{D}'$, which we had assumed to be small. Hence, we have a good algorithm for $(L, \mathcal{D})$, and the definition of reduction preserves average-case tractability. Note that, in this argument, we used nothing about the

[2] Additional difficulties arise in defining how to specify $\mathcal{D}$.
[3] When the second condition holds, we say that $\mathcal{D}'$ *dominates* $\mathcal{D}$.

definition of tractability except the notion of "bad" input. (See also Chapter 3.)

1.1.3 A completeness result

Having given the definition of computational problem and of reduction, we will present a *completeness result* [53]. We consider the *bounded halting* problem BH, where on input $(M, x, 1^t)$ we have to determine whether the non-deterministic Turing machine M accepts input x within t steps. This problem is readily seen to be NP-complete. We show that for every distributional problem $(L, \mathcal{D})$, where L is in NP and $\mathcal{D}$ is a *polynomial-time computable* distribution there is a reduction from $(L, \mathcal{D})$ to $(\mathrm{BH}, \mathcal{U}^{\mathrm{BH}})$, where $\mathcal{U}^{\mathrm{BH}}$ is a reasonable formalization of the notion of a "uniformly chosen" random input for BH. Informally, the reduction maps an input x into the triple $(M', C(x), 1^t)$, where C is a (carefully chosen) injective polynomial-time computable encoding function; M' is a non-deterministic machine that first recovers x from $C(x)$ and then simulates the non-deterministic polynomial time Turing machine that decides whether $x \in L$ (recall that L is in NP); and t is a polynomial upper bound to the running time of M'. The main claim in the analysis of the reduction is that, for x selected from $\mathcal{D}$, $C(x)$ is "approximately" uniformly distributed. Technically, we show that the distribution of $C(x)$ is dominated by the uniform distribution. This will follow from a choice of C as an information-theoretically optimal compression scheme.

The completeness result implies that if $(\mathrm{BH}, \mathcal{U}^{\mathrm{BH}})$ has a good-on-average algorithm (according to one of the possible definitions), then all problems $(L, \mathcal{D})$, where L is in NP and $\mathcal{D}$ is polynomial-time computable, also have good-on-average algorithms.

The proof uses the fact that all *polynomial-time computable* distributions $\mathcal{D}$ allow polynomial-time computable optimal compression schemes. Many natural distributions are polynomial-time computable, but there are a number of important exceptions. The output of a pseudorandom generator, for example, defines a distribution that is not optimally compressible in polynomial time and, hence, is not polynomial-time computable.

1.1.4 Decision versus search

The second result that we present, due to Ben-David *et al.* [12], shows that if $(\mathrm{BH},\mathcal{U}^{\mathrm{BH}})$ has a good-on-average algorithm, then for all NP relations R and all polynomial-time computable distributions $\mathcal{D}$, there is an efficient algorithm that, given x sampled from $\mathcal{D}$, almost always finds a y such that $R(x,y)$, provided that such a y exists. This shows that the question of whether there are intractable-on-average search problems in NP (with respect to polynomial-time computable distributions) is equivalent to the question of whether there are intractable-on-average decision problems in NP (with respect to such distributions). Both questions are equivalent to the specific decision problem $(\mathrm{BH},\mathcal{U}^{\mathrm{BH}})$ being intractable.

1.1.5 Computable, samplable, and arbitrary distributions

The restriction of the completeness result to samplable distributions is quite undesirable because it rules out reasonably natural distributions that can occur in certain applications. Ideally, it would be desirable that the theory put no restriction whatsoever on the distributions, and that we could prove results of the form "if there is a good-on-average algorithm for $(\mathrm{BH},\mathcal{U}^{\mathrm{BH}})$, then for every L in NP and every distribution $\mathcal{D}$ there is a good-on-average algorithm for $(L,\mathcal{D})$." The conclusion, however, is equivalent to $\mathrm{P} = \mathrm{NP}$.[4] More specifically, there is a distribution $\mathcal{D}$ such that, for every language L in NP, if there is a good-on-average algorithm for $(L,\mathcal{D})$ then there is an efficient worst-case algorithm for L. As we discuss below, there are difficulties in relating the worst-case complexity to the average-case complexity of all problems in NP, and so it seems unlikely that the theory can be generalized to handle completely arbitrary distributions. An important intermediate case between polynomial-time computable distributions and arbitrary distributions is the class of *polynomial-time samplable distributions*. This class includes some natural distributions that are not polynomial-time computable (e.g., the output of a pseudorandom generator), and an

[4] This was first proved by Levin. In Section 2.5 we present a later proof by Li and Vitányi [55].

argument can be made that any distribution that occurs "in nature" should be samplable. Impagliazzo and Levin [42] show that the completeness result can be extended to all samplable distributions. That is, if (BH,$\mathcal{U}^{\mathrm{BH}}$) admits a good-on-average algorithm, then for every problem L in NP and every samplable distribution $\mathcal{D}$, the problem $(L,\mathcal{D})$ has a good-on-average algorithm. In Sections 5.1 and 5.2, we present two proofs of this result. A simpler one, appearing in the article of Impagliazzo and Levin, which applies only to some (but not all) definitions of "good-on-average," and a second proof, also due to Impagliazzo and Levin, but unpublished, that is more complex but that applies to all definitions. The first proof is similar to the proof of the completeness result for polynomial-time computable distributions, but using a randomized encoding scheme. An input x for L is mapped into an input $(M',(r,C(r,x)),1^t)$ for BH, where r is randomly chosen. The desired properties of the randomized encoding C are (i) over the choices of r, the encoding $x \to (r,C(x,r))$ is "approximately injective," and (ii) the distribution $(r,C(x,r))$ is "approximately uniform" when r is uniformly chosen and x is sampled from $\mathcal{D}$. Some additional difficulties arise: in order to compute the randomized encoding one needs some extra information about x, and the reduction just "guesses" all possible values for this extra information, and, for technical reasons, this forces us to work with the *search* rather than the *decision* version of L. This is done without loss of generality given the reduction of Ben-David *et al.* [12]. The idea for the second proof is that, if S is the sampling algorithm for L, and L is hard-on-average over the outputs of S, then the problem "on input r, is it true that $S(r) \in L$?" should be hard-on-average with respect to the uniform distribution. This intuition is quite difficult to translate into a proof, especially in the case in which the computation of the sampler S is a one-way function.

1.1.6 Worst case versus average case

In order to unify the theory of average-case complexity with the rest of complexity theory, it would be highly desirable to prove a theorem of the form, "if P $\neq$ NP then there is a hard-on-average problem $(L,\mathcal{D})$, where L is in NP and $\mathcal{D}$ is samplable." In order to prove such a result

via a reduction, we would need to find an oracle algorithm R (the reduction) such that if A is a good-on-average algorithm for $(L,\mathcal{D})$, then R^A is a worst-case efficient algorithm for, say, 3SAT. Feigenbaum and Fortnow [27] show that (under standard assumptions) such a result cannot be proved via a *non-adaptive random reduction*, that is, via an algorithm R that makes non-adaptive queries and such that each query has the distribution $\mathcal{D}$ (regardless of the input of R). Bogdanov and Trevisan [15] show that the same impossibility result holds even if R is allowed to make arbitrary non-adaptive queries, provided that R works for arbitrary oracles. It remains possible that a worst-case-to-average-case reduction in NP exists which makes *adaptive* access to the oracle, or that uses the *code* of the algorithm A (and, hence, does not work for arbitrary oracles). Guttfreund and Ta-Shma [37] make some progress in the latter direction. An even more ambitious goal is to show, via reductions, that "if P $\neq$ NP then one-way functions exist." The result of Bogdanov and Trevisan rules out the possibility of proving such a result via oracle non-adaptive reductions; Akavia *et al.* [9] present a simpler proof in the setting of one-way functions (which, unlike the Bogdanov-Trevisan proof, works also in the uniform setting) and are also able, for a restricted class of one-way functions, to rule out non-adaptive reductions.

1.1.7 Degrees of average-case intractability

If a problem L is worst-case intractable, then every efficient algorithm makes an infinite number of mistakes; if a problem $(L,\mathcal{D})$ is average-case intractable, then every efficient algorithm makes mistakes[5] on a set of inputs that has noticeably large probability according to $\mathcal{D}$. Given the difficulties in relating these two settings, it is interesting to ask what happens if we consider different quantitative formulations of "noticeably large." O'Donnell [61] shows that any quantification between $1/2 - 1/n^{.33}$ and $1/\mathrm{poly}(n)$ leads essentially to an equivalent intractability assumption. O'Donnell's argument, presented in Chapter 6, gives a far-reaching generalization of Yao's XOR Lemma [76].

[5] Or *fails*, depending on the definition of average-case tractability that we are using.

1.1.8 Specific problems

Eventually, we would like the theory to talk about the complexity of specific natural problems with specific natural distributions. It follows from Cook's reduction that if there is a hard-on-average problem $(L, \mathcal{D})$, where L is in NP and $\mathcal{D}$ is samplable, then every NP-hard problem is hard on average with respect to some samplable distribution, albeit a very unnatural one. On the other hand, Levin's completeness result shows (under the same assumption) that there are hard-on-average problems $(L, \mathcal{D})$, where $\mathcal{D}$ is uniform, but L is quite artificial. Yet, the theory of average-case completeness has little to say about specific cases of interest where both L and $\mathcal{D}$ are natural: for instance, the hardness of 3SAT or maximum independent set with respect to natural distributions on inputs.

A specific problem whose average-case behavior has been widely investigated is random kSAT with respect to the following distribution of instances: Choose at random $m_k(n)$ out of the $2^k\binom{n}{k}$ possible clauses of kSAT independently. The tractability of this problem appears to depend heavily on the number of clauses $m_k(n)$. While it is believed that random kSAT is hard for certain choices of $m_k(n)$, no hardness result supporting this intuition is known. However, Feige [23] shows the following surprising connection between hardness of random 3SAT and hardness of approximation: Assuming that random 3SAT is hard for certain values of $m_3(n)$, it is *worst-case hard* to approximate certain problems in NP (e.g., maximum bipartite clique within $n^{-\varepsilon}$ for some $\varepsilon > 0$.)

For certain *lattice problems* we know an equivalence between worst-case and average-case complexity [5, 57, 59, 64]. If such equivalences could be proved for NP-complete lattice problems, we would have a positive solution to the question of whether the existence of hard-on-average problems in NP can be based on the worst-case hardness of NP-complete problems.

1.2 A historical overview

In this section we review the historical progression toward the results described in the previous section.

1.2.1 One-way functions and cryptography

The average-case performance of algorithms on random inputs has been studied since the beginning of the modern theory of efficient algorithms in the 1950s and 1960s. Such work was often focused on problems for which worst-case polynomial-time algorithms were also known. The third volume of *The art of computer programming* [49] (published in 1973) extensively surveys average-case analyses of algorithms for problems such as sorting and median finding.

The study of the average case of (conjectured) intractable problems began in the 1970s motivated by the development of the foundations of cryptography and by interest in heuristic approaches to NP-complete problems.

When Diffie and Hellman [20] introduced the notion of public key cryptography, they speculated that one could base a trapdoor permutation on the difficulty of an NP-complete problem.[6] Even, Yacobi and Lempel [22,51] devised a public key cryptosystem such that an efficient adversary that breaks the system *for every key* implies an efficient algorithm for an NP-complete problem. An efficient adversary that breaks the system *on almost all keys*, however, is also discussed.

Shamir [68] discusses the difficulty in formulating a definition of intractability for cryptographic applications. Worst-case complexity is immediately seen as inadequate. Furthermore, Shamir emphasizes that a cryptographic system cannot be considered secure if there is an attack that takes expected polynomial time. In fact, Shamir adds, it is not even enough to rule out expected polynomial time attacks. Consider, for example, a system that can be broken by an attacker whose *expected* running time is very large but whose *median* running time is efficient. This is possible if the attacker takes a very long time, say, on one-third of the keys but is efficient otherwise. Even though the expected running time of the adversary is large, such a system cannot be considered secure.

[6] Indeed, Diffie and Hellman give two main justifications for their claim that "we stand on the brink of a revolution in cryptography": the availability of cheap and efficient computers (in the 1970s!) and the development of NP-completeness.

The median running time of an adversary is thus a better complexity measure of the expected running time, Shamir notes, but one needs to go beyond, and consider the running time of, say, the 1% fraction of inputs on which the algorithm is fastest. This short discussion anticipates the formal definition of one-way function and the difficulties in defining a robust notion of "average-case tractability" in Levin's theory of average-case complexity.

The work of Blum, Goldwasser, Micali, and Yao [35,14,76] put cryptography on solid foundational grounds, and introduced the modern definitions of one-way functions, trapdoor permutations, pseudorandom generators, and secure encryption. In their definition, an efficiently computable function f is one-way if there is no polynomial-time algorithm that finds a pre-image of $f(x)$ with more than inverse polynomial probability over the choice of x. This means that if f is a one-way function, then the computational problem "given $y = f(x)$ find a pre-image of y," has no algorithm of expected polynomial time, no algorithm of median polynomial time, no algorithm that runs in polynomial time on the easiest 1% fraction of inputs, and so on.

1.2.2 Levin's theory of average-case intractability

The development of the theory of NP-completeness gave evidence that a large number of important computational problems do not admit worst-case efficient algorithms and motivated the design of good-on-average algorithms as a way to "cope" with intractability.

Following this approach, the goal is to analyze worst-case superpolynomial-time algorithms for NP-complete problems and to show that on "typical" instances they are efficient. A celebrated example is Karp's algorithm for TSP in the plane [46]. An annotated bibliography by Karp *et al.* [47] written in 1985 reports several results on average-case tractability of NP-complete problems on natural distributions.

The initial success in the design of good-on-average algorithms led to the question of the limitations of such an approach. Are there NP-complete problems that, with respect to natural distributions, do not even have good-on-average algorithms? Are there general techniques,

analogous to the theory of NP-completeness, to prove average-case intractability?[7]

Levin [53] laid the foundations for a theory of the average-case tractability of problems in NP. He introduced the definition of average-case tractability and of reduction outlined above and proved the first completeness result, for the class $(\mathrm{NP}, \mathrm{PComp})$ of problems $(L, \mathcal{D})$ such that L is in NP and $\mathcal{D}$ is polynomial-time computable.

Levin's article, both in the one-page conference version and in the two-page full version [53], gives few details about the intuition behind the definitions and the possibility of generalized or alternative definitions.

Ben-David *et al.* [12] consider two issues not addressed in Levin's article. One issue is the class of distributions to consider. Levin restricts his attention to the class of "polynomial time computable distributions" that includes several natural distributions but that excludes, for example, the output of a pseudorandom generator and other natural distributions. Ben David et al. observe that the more general class of "efficiently samplable" distributions is a better formalization of the notion of natural distribution and formulate the question of whether Levin's completeness result can be extended to the corresponding class $(\mathrm{NP}, \mathrm{PSamp})$ of distributional problems $(L, \mathcal{D})$ such that L is in NP and $\mathcal{D}$ is samplable. Another issue studied in [12] is the average-case complexity of *decision* versus *search* problems, and their main result shows that if every decision problem in NP can be solved efficiently with respect to the uniform distribution, then every search problem in NP can also be solved efficiently with respect to the uniform distribution. Impagliazzo and Levin [42], solving the main open question formulated in [12], prove that there is a problem that is complete for $(\mathrm{NP}, \mathrm{PSamp})$.

[7] Interestingly, around the same time (mid-1970s), another approach was studied to "cope" with the intractability of NP-complete optimization problems, namely, to design provably efficient *approximate* algorithms that deliver near-optimal solutions, and the question was asked of when not even such algorithms exist. In the 1990s, the theory of probabilistically checkable proofs gave a powerful tool to prove intractability of approximation problems. A satisfactory and general theory to prove average-case intractability, unfortunately, does not exist yet.

1.2.3 Average-case intractability and derandomization

Yao [76] proves that the existence of pseudorandom generators implies the possibility of derandomizing probabilistic algorithms, and that pseudorandom generators can be constructed using one-way permutations. (Håstad *et al.* [39] later proved that the existence of one-way functions is sufficient.) The existence of a one-way permutation f can be stated as the average-case intractability of the *distributional search problem* of inverting f on a random input, so Yao's result proves that a specific average-case assumption (for certain search problems within NP) implies derandomization of probabilistic algorithms. The connection between average-case complexity and derandomization became more direct, simpler, and more general in the work of Nisan and Wigderson [60]. Their work requires the existence of hard-on-average distributional *decision* problems in EXP. The work of Nisan and Wigderson raised the question of whether derandomization could be based on *worst-case* assumptions about problems in EXP instead of average-case assumptions. The question led to the study of worst-case versus average-case complexity in EXP, and to such tools as *random self-reduction* [10], *amplification of hardness* [41, 44], and *error-correcting codes* [69]. As a result of this decade-long investigation, we now know that worst-case and average-case are equivalent in complexity classes such as EXP and PSPACE. The interested reader can find an account of such results in survey articles by Trevisan [71] (see, in particular, Chapter 4) and by Kabanets [45].

1.2.4 Worst-case versus average case within NP

The proofs of the worst-case and average-case equivalence for complete problems in EXP, PSPACE, and other classes raise the question of whether a similar worst-case and average-case equivalence also holds for intractable problems within NP. This is related to fundamental questions in the foundations of cryptography: Is it possible to base one-way functions on NP-completeness? If so, what about one-way permutations or public key encryption?

On the one hand, it is easy to see that one-way permutations cannot be based on NP-completeness, unless NP = coNP (or AM = coAM,

if one allows randomized reductions, or NP/poly = coNP/poly, if one allows non-uniform reductions). Not even the intractability of *worst-case* inversion can be based on NP-completeness (see Section 7.2).

On the other hand, it is possible to define "one-way functions" that are computable in polynomial time and that cannot have a "worst-case inverter" (i.e., a polynomial time inverter that works on all inputs) unless P = NP. For this reason, when we ask whether the existence of one-way functions (under the standard, average-case, definition) can be based on NP-completeness, we are asking a question about the *average-case complexity* of inverters.

To clarify before we continue: The existence of one-way permutations implies the existence of one-way functions, which implies the existence of hard-on-average distributional problems in $(\mathrm{NP}, \mathrm{PSAMP})$,[8] which implies that P is different from NP. We do not know how to prove the inverse of any of those implications, even though we believe that all the statements are true, and so they all imply each other vacously.

We can ask, however, whether reverse implications can be proved via *reductions*, that is, for example, whether there is a distributional problem $(L, \mathcal{D})$ in $(\mathrm{NP}, \mathrm{PSAMP})$ and a reduction R such that, for every algorithm A that solves $(L, \mathcal{D})$ well on average, the reduction R plus the algorithm A give a worst-case algorithm for 3SAT.

Feigenbaum and Fortnow [27] study a special case of the above question. They consider the case in which R is a "non-adaptive random self-reduction." They show that the existence of such a reduction implies the collapse of the polynomial hierarchy (which contradicts standard conjectures). The result of Feigenbaum and Fortnow rules out a certain way of proving equivalence of worst-case and average-case for NP-complete problems, including the way used in the work on EXP and PSPACE [10, 41, 44, 69] (see Section 7.3).

In a celebrated breakthrough, Ajtai [5], describes a distributional problem in $(\mathrm{NP}, \mathrm{PCOMP})$ whose average-case complexity is at least as high as the worst-case complexity of a related (promise) problem in NP–a version of the shortest vector problem for lattices in $\mathbb{R}^n$. Ajtai also proves the existence of one-way functions that are based on the

[8] This implication is non-trivial; see Section 4.3.

worst-case complexity of problems in NP. Ajtai and Dwork [7] present a public key cryptosystem based on a worst-case assumption, and Micciancio and Regev [57, 59, 64] present various improvements.

The security of the cryptosystems of Ajtai, Dwork, Micciancio, and Regev relies on the worst-case complexity of problems that are not known to be NP-complete and, in fact, are in NP $\cap$ coNP. It remains an open question whether these techniques can be refined and improved to the point where cryptography primitives can be constructed that rely on the worst-case complexity of an NP-complete problem.

Bogdanov and Trevisan [15] prove that no non-adaptive worst-case to average-case reduction exists for NP-complete problems unless NP/poly = coNP/poly. Akavia *et al.* [9] prove that one-way functions cannot be based on NP-complete problems via non-adaptive reductions unless AM = coAM (see Section 7.3).

It seems likely that reductions cannot relate worst-case and average-case hardness in NP. What about different degrees of average-case intractability? For instance, if there exist distributional problems in NP that are hard on some non-negligible fraction of instances, does it follow that there are distributional problems in NP that are hard on almost all instances? These questions have been answered in the affirmative by O'Donnell [61] and Healy, Vadhan, and Viola [40] in the non-uniform setting and by Trevisan [70, 72] in the uniform setting (see Chapter 6).

References

[1] D. Achlioptas and C. Moore, "The asymptotic order of the k-SAT threshold," in *Proceedings of the 43rd IEEE Symposium on Foundations of Computer Science*, pp. 779–788, 2002.

[2] D. Achlioptas and Y. Peres, "The threshold for random k-SAT is $2^k \log 2 - O(k)$," *Journal of the AMS*, vol. 17, no. 4, pp. 947–973, 2004.

[3] L. Adleman, "Two theorems on random polynomial time," in *Proceedings of the 19th IEEE Symposium on Foundations of Computer Science*, pp. 75–83, 1978.

[4] D. Aharonov and O. Regev, "Lattice Problems in NP∩coNP," *Journal of the ACM*, vol. 52, no. 5, pp. 749–765, Preliminary version in Proceedings of FOCS 2004, 2005.

[5] M. Ajtai, "Generating hard instances of lattice problems," in *Proceedings of the 28th ACM Symposium on Theory of Computing*, pp. 99–108, 1996.

[6] M. Ajtai, "The Shortest Vector Problem in ℓ_2 is NP-hard for Randomized Reductions," in *Proceedings of the 30th ACM Symposium on Theory of Computing*, pp. 10–19, 1998.

[7] M. Ajtai and C. Dwork, "A Public-Key Cryptosystem with Worst-Case/Average-Case Equivalence," in *Proceedings of the 29th ACM Symposium on Theory of Computing*, pp. 284–293, 1997.

[8] M. Ajtai, R. Kumar, and D. Sivakumar, "A sieve algorithm for the shortest lattice vector problem," in *Proceedings of the 33rd ACM Symposium on Theory of Computing*, pp. 601–610, 2001.

[9] A. Akavia, O. Goldreich, S. Goldwasser, and D. Moshkovitz, "On basing one-way functions on NP-hardness," in *Proceedings of the 38th ACM Symposium on Theory of Computing*, 2006.

[10] L. Babai, L. Fortnow, N. Nisan, and A. Wigderson, "BPP has subexponential time simulations unless EXPTIME has publishable proofs," *Computational Complexity*, vol. 3, no. 4, pp. 307–318, 1993.

[11] P. Beame, R. Karp, T. Pitassi, and M. Saks, "On the complexity of unsatisfiability proofs for random k-CNF formulas," in *Proceedings of the 30th ACM Symposium on Theory of Computing*, 1998.

[12] S. Ben-David, B. Chor, O. Goldreich, and M. Luby, "On the theory of average case complexity," *Journal of Computer and System Sciences*, vol. 44, no. 2, pp. 193–219, 1992.

[13] E. Ben-Sasson and A. Wigderson, "Short proofs are narrow: Resolution made simple," *Journal of the ACM*, vol. 48, no. 2, 2001.

[14] M. Blum and S. Micali, "How to generate cryptographically strong sequences of pseudorandom bits," *SIAM Journal on Computing*, vol. 13, no. 4, pp. 850–864, Preliminary version in *Proceedings of FOCS'82*, 1984.

[15] A. Bogdanov and L. Trevisan, "On wosrt-case to average-case reductions for NP problems," in *Proceedings of the 44th IEEE Symposium on Foundations of Computer Science*, pp. 308–317, 2003.

[16] D. Boneh and R. J. Lipton, "Amplification of weak learning under the uniform distribution," in *Proceedings of the 6th ACM Conference on Computational Learning Theory*, pp. 347–351, 1993.

[17] G. Brassard, "Relativized cryptography," in *Proceedings of the 20th IEEE Symposium on Foundations of Computer Science*, pp. 383–391, 1979.

[18] V. Chvatal and E. Szemeredi, "Many hard examples for resolution," *Journal of the ACM*, vol. 35, no. 4, pp. 759–768, 1998.

[19] A. Coja-Oghlan, A. Goerdt, A. Lanka, and F. Schdlich, "Certifying unsatisfiability of random 2k-SAT formulas using approximation techniques," in *Proceedings of 14th Symposium on Foundations of Computation Theory*, pp. 15–26, LNCS 2751, 2003.

[20] W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, 1976.

[21] S. Even, A. Selman, and Y. Yacobi, "The complexity of promise problems with applications to public-key cryptography," *Information and Computation*, vol. 61, no. 2, pp. 159–173, 1984.

[22] S. Even and Y. Yacobi, "Cryptography and NP-completeness," in *Proceedings of the 7th International Colloquium on Automata, Languages and Programming*, pp. 195–207, Springer-Verlag, 1980.

[23] U. Feige, "Relations between average case complexity and approximation complexity," in *Proceedings of the 34th ACM Symposium on Theory of Computing*, pp. 534–543, 2002.

[24] U. Feige, J. Kim, and E. Ofek, "Witnesses for non-satisfiability of dense random 3CNF formulas," in *Proceedings of the 47th IEEE Symposium on Foundations of Computer Science*, 2006. To appear.

[25] U. Feige and E. Ofek, "Easily refutable subformulas of random 3CNF formulas," in *Proceedings of the 31st International Colloquium on Automata, Languages and Programming*, pp. 519–530, 2004.

[26] U. Feige and E. Ofek, "Random 3CNF formulas elude the Lovasz theta function," Tech. Rep. TR06-043, Electronic Colloquium on Computational Complexity, 2006.

[27] J. Feigenbaum and L. Fortnow, "Random-self-reducibility of complete sets," *SIAM Journal on Computing*, vol. 22, pp. 994–1005, 1993.

[28] E. Friedgut, "Necessary and sufficient conditions for sharp thresholds of graph properties and the k-SAT problem," *Journal of the AMS*, vol. 12, pp. 1017–1054, 1999.

[29] A. Goerdt and M. Krivelevich, "Efficient recognition of random unsatisfiable k-SAT instances by spectral methods," in *Proceedings of the 18th Symposium on Theoretical Aspects of Computer Science*, pp. 294–304, 2001.

[30] O. Goldreich and S. Goldwasser, "On the limits of non-approximability of lattice problems," in *Proceedings of the 30th ACM Symposium on Theory of Computing*, pp. 1–9, 1998.

[31] O. Goldreich, *The Foundations of Cryptography - Volume 1*. Cambridge University Press, 2001.

[32] O. Goldreich, "On promise problems (a survey in memory of Shimon Even [1935–2004])," Tech. Rep. TR05-018, Electronic Colloquium on Computational Complexity, 2005.

[33] O. Goldreich and S. Goldwasser, "On the possibility of basing Cryptography on the assumption that $P \neq NP$," 1998. Unpublished manuscript.

[34] O. Goldreich, N. Nisan, and A. Wigderson, "On Yao's XOR Lemma," Tech. Rep. TR95-50, Electronic Colloquium on Computational Complexity, 1995.

[35] S. Goldwasser and S. Micali, "Probabilistic encryption," *Journal of Computer and System Sciences*, vol. 28, no. 2, pp. 270–299, Preliminary Version in *Proceedings of STOC'82*, 1984.

[36] D. Gutfreund, R. Shaltiel, and A. Ta-Shma, "If NP languages are hard on the worst-case then it is easy to find their hard instances," in *Proceedings of the 20th IEEE Conference on Computational Complexity*, 2005.

[37] D. Gutfreund and A. Ta-Shma, "New connections between derandomization, worst-case complexity and average-case complexity," Tech. Rep. TR06-108, Electronic Colloquium on Computational Complexity, 2006.

[38] J. Håstad, "Some optimal inapproximability results," *Journal of the ACM*, vol. 48, no. 4, pp. 798–859, 2001.

[39] J. Håstad, R. Impagliazzo, L. Levin, and M. Luby, "A pseudorandom generator from any one-way function," *SIAM Journal on Computing*, vol. 28, no. 4, pp. 1364–1396, 1999.

[40] A. Healy, S. Vadhan, and E. Viola, "Using nondeterminism to amplify hardness," in *Proceedings of the 36th ACM Symposium on Theory of Computing*, pp. 192–201, 2004.

[41] R. Impagliazzo, "Hard-core distributions for somewhat hard problems," in *Proceedings of the 36th IEEE Symposium on Foundations of Computer Science*, pp. 538–545, 1995.

[42] R. Impagliazzo and L. Levin, "No Better Ways to Generate Hard NP Instances than Picking Uniformly at Random," in *Proceedings of the 31st IEEE Symposium on Foundations of Computer Science*, pp. 812–821, 1990.

[43] R. Impagliazzo and M. Luby, "One-way Functions are Essential for Complexity Based Cryptography," in *Proceedings of the 30th IEEE Symposium on Foundations of Computer Science*, pp. 230–235, 1989.

[44] R. Impagliazzo and A. Wigderson, "$P = BPP$ unless E has sub-exponential circuits," in *Proceedings of the 29th ACM Symposium on Theory of Computing*, pp. 220–229, 1997.

[45] V. Kabanets, "Derandomization: A brief overview," *Bulletin of the European Association for Theoretical Computer Science*, vol. 76, pp. 88–103, 2002.

[46] R. Karp, "Probabilistic Analysis of Partitioning Algorithms for the Traveling-Salesman Problem in the Plane," *Mathematics of Operations Research*, vol. 2, no. 3, pp. 209–224, 1977.

[47] R. Karp, J. Lenstra, C. McDiarmid, and A. R. Kan, "Probabilistic analysis," in *Combinatorial Optimization: An Annotated Bibliography*, (M. O'hEigeartaigh, J. Lenstra, and A. R. Kan, eds.), pp. 52–88, Wiley, 1985.

[48] S. Khot, "Hardness of approximating the shortest vector problem in lattices," 2004. Manuscript.

[49] D. Knuth, *The Art of Computer Programming*. Vol. 3, Addison-Wesley, 1973.

[50] R. Kumar and D. Sivakumar, "On polynomial-factor approximations to the shortest lattice vector length," *SIAM Journal on Discrete Mathematics*, vol. 16, no. 3, pp. 422–425, Preliminary version in Proceedings of SODA 2001, 2003.

[51] A. Lempel, "Cryptography in transition," *Computing Surveys*, vol. 11, no. 4, pp. 215–220, 1979.

[52] A. Lenstra, H. Lenstra, and L. Lovasz, "Factoring polynomials with rational coefficients," *Mathematische Annalen*, vol. 261, pp. 515–534, 1982.

[53] L. Levin, "Average case complete problems," *SIAM Journal on Computing*, vol. 15, no. 1, pp. 285–286, 1986.

[54] L. Levin, "One-Way Functions and Pseudorandom Generators," *Combinatorica*, vol. 7, no. 4, pp. 357–363, 1987.

[55] M. Li and P. M. B. Vitányi, "Average case complexity under the universal distribution equals worst-case complexity," *IPL*, vol. 42, no. 3, pp. 145–149, 1992.

[56] D. Micciancio, "The shortest vector problem is NP-hard to approximate to within some constant," *SIAM Journal on Computing*, vol. 30, no. 6, pp. 2008–2035, 2001.

[57] D. Micciancio, "Almost perfect lattices, the covering radius problem, and applications to Ajtai's connection factor," *SIAM Journal on Computing*, vol. 34, no. 1, pp. 118–169, 2004.

[58] D. Micciancio and S. Goldwasser, *Complexity of Lattice Problems*. Norwell, MA, USA: Kluwer Academic Publishers, 2002.

[59] D. Micciancio and O. Regev, "Worst-case to average-case reductions based on gaussian measure," in *Proceedings of the 45th IEEE Symposium on Foundations of Computer Science*, pp. 372–381, 2004.

[60] N. Nisan and A. Wigderson, "Hardness vs randomness," *Journal of Computer and System Sciences*, vol. 49, pp. 149–167, Preliminary version in *Proc. of FOCS'88*, 1994.

[61] R. O'Donnell, "Hardness amplification within NP," in *Proceedings of the 34th ACM Symposium on Theory of Computing*, pp. 751–760, 2002.

[62] R. Ostrovsky, "One-way functions, hard on average problems and statistical zero-knowledge proofs," in *STRUCTURES91*, pp. 51–59, 1991.

[63] O. Regev, "New lattice based cryptographic constructions," in *Proceedings of the 35th ACM Symposium on Theory of Computing*, pp. 407–416, 2003.

[64] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," in *Proceedings of the 37th ACM Symposium on Theory of Computing*, pp. 84–93, 2005.

[65] O. Regev, "Lattice-based cryptography," in *Advances in Cryptology (CRYPTO)*, pp. 131–141, 2006.

[66] O. Regev and R. Rosen, "Lattice problems and norm embeddings," in *Proceedings of the 38th ACM Symposium on Theory of Computing*, pp. 447–456, 2006.

[67] C. Schnorr, "A hierarchy of polynomial time lattice basis reduction algorithms," *Theoretical Computer Science*, vol. 53, pp. 201–224, 1987.

[68] A. Shamir, "On the cryptocomplexity of knapsack systems," in *Proceedings of the 11th ACM Symposium on Theory of Computing*, pp. 118–129, 1979.

[69] M. Sudan, L. Trevisan, and S. Vadhan, "Pseudorandom generators without the XOR Lemma," *Journal of Computer and System Sciences*, vol. 62, no. 2, pp. 236–266, 2001.

[70] L. Trevisan, "List-decoding using the XOR Lemma," in *Proceedings of the 44th IEEE Symposium on Foundations of Computer Science*, pp. 126–135, 2003.

[71] L. Trevisan, "Some Applications of coding theory in computational complexity," *Quaderni di Matematica*, vol. 13, pp. 347–424, arXiv:cs.CC/0409044, 2004.

[72] L. Trevisan, "On uniform amplification of hardness in NP," in *Proceedings of the 37th ACM Symposium on Theory of Computing*, pp. 31–38, 2005.

[73] L. G. Valiant and V. V. Vazirani, "NP is as easy as detecting unique solutions," *Theoretical Computer Science*, vol. 47, pp. 85–93, 1986.

[74] E. Viola, "The Complexity of constructing pseudorandom generators from hard functions," *Computational Complexity*, vol. 13, no. 3-4, pp. 147–188, 2004.

[75] E. Viola, "On Constructing Parallel Pseudorandom Generators from One-Way Functions," in *Proceedings of the 20th IEEE Conference on Computational Complexity*, 2005.

[76] A. C. Yao, "Theory and applications of trapdoor functions," in *Proceedings of the 23th IEEE Symposium on Foundations of Computer Science*, pp. 80–91, 1982.