Complexity of Distributions and Average-Case Hardness*

Dmitry Itsykson[1], Alexander Knop[2], and Dmitry Sokolov[3]

1 **St. Petersburg Department of V. A. Steklov Institute of Mathematics of the Russian Academy of Sciences, St. Petersburg, Russia**
dmitrits@pdmi.ras.ru

2 **St. Petersburg Department of V. A. Steklov Institute of Mathematics of the Russian Academy of Sciences, St. Petersburg, Russia**
aaknop@gmail.com

3 **St. Petersburg Department of V. A. Steklov Institute of Mathematics of the Russian Academy of Sciences, St. Petersburg, Russia**
sokolov.dmt@gmail.com

Abstract

We address the following question in the average-case complexity: does there exists a language L such that for all easy distributions D the distributional problem (L, D) is easy on the average while there exists some more hard distribution D' such that (L, D') is hard on the average? We consider two complexity measures of distributions: the complexity of sampling and the complexity of computing the distribution function.

For the complexity of sampling of distribution, we establish a connection between the above question and the hierarchy theorem for sampling distribution recently studied by Thomas Watson. Using this connection we prove that for every $0 < a < b$ there exist a language L, an ensemble of distributions D samplable in $n^{\log^b n}$ steps and a linear-time algorithm A such that for every ensemble of distribution F that samplable in $n^{\log^a n}$ steps, A correctly decides L on all inputs from $\{0,1\}^n$ except for a set that has infinitely small F-measure, and for every algorithm B there are infinitely many n such that the set of all elements of $\{0,1\}^n$ for which B correctly decides L has infinitely small D-measure.

In case of complexity of computing the distribution function we prove the following tight result: for every $a > 0$ there exist a language L, an ensemble of polynomial-time computable distributions D, and a linear-time algorithm A such that for every computable in n^a steps ensemble of distributions F, A correctly decides L on all inputs from $\{0,1\}^n$ except for a set that has F-measure at most $2^{-n/2}$, and for every algorithm B there are infinitely many n such that the set of all elements of $\{0,1\}^n$ for which B correctly decides L has D-measure at most 2^{-n+1}.

1998 ACM Subject Classification F.1.3 Complexity Measures and Classes

Keywords and phrases average-case complexity, hierarchy theorem, sampling distributions, diagonalization

Digital Object Identifier 10.4230/LIPIcs.ISAAC.2016.38

1 Introduction

This paper is devoted to average-case complexity. In the average case settings, every computational problem is supplied with an ensemble of distributions on inputs. The problem

* The research presented in Section 3 was supported by Russian Science Foundation (project 16-11-10123). The third author was partially supported by "Dynasty" foundation.

is easy on the average if it can be solved efficiently on all but a small fraction (according to the distribution) of the inputs.

The paper [6] gave an example of a noncomputable ensemble of distributions such that every language with that ensemble of distributions is easy on the average iff it is easy in the worst case. This explains why the average-case complexity studies not all but only feasible ensembles of distributions. The most natural class of ensembles of distributions is the class of polynomial-time samplable distributions. Such distributions are distributions of outputs of polynomial-time randomized algorithms. The second important class of ensembles of distributions is the class of polynomial-time computable ensembles of distributions. An ensemble of distributions is computable in polynomial time if it's cumulative distribution function is computable in polynomial time. It is known that every polynomial-time computable ensemble of distribution is polynomial-time samplable but the opposite is not true if one-way functions exist [2].

It is well known that several hard problems can be efficiently solved on almost all inputs for some natural distributions. For example the **NP**-complete problem Hamiltonian Path is decidable in a linear time on almost all inputs according to the uniform distributions on the graphs [3]. Another interesting example is the Graph Isomorphism problem that is solvable in linear time on almost all inputs in the case of the uniform distribution on the inputs [1], while there exists much more tricky distribution (see for example [8]) such that there are no known polynomial-time algorithms that solve the graph isomorphism problem with high probability.

Statement of the problem. The standard time hierarchy theorem in the average-case settings states that for all $a > 0$ there exists a distributional problem (L, D) such that every $g(n)$-time algorithm errs on almost all (or on a significant fraction of) inputs according to D but there exists an algorithm with slightly bigger running time $f(n)$ that correctly solves the problem on almost all inputs according to D. For deterministic algorithms, it is easy to show by the straightforward diagonalization that there exists a language L that is decidable in n^a steps but every algorithm with running time $O(n^{a-\epsilon})$ gives incorrect answer on all sufficiently large inputs for $\epsilon > 0$. For randomized algorithms with bounded error Pervyshev [7] showed that for all $b > 0$ there exists a language L with the uniform distribution that is decidable in randomized polynomial time with a bounded error on all but ϵ fraction of inputs but every randomized algorithm with running time n^b gives incorrect answer with high probability on at least $\frac{1}{2} - \epsilon$ fraction of inputs for all $\epsilon > 0$. The paper [5] showed that the fraction of hard instances in the Pervyshev's result may be improved to $1 - \frac{1}{k} - \epsilon$ if we switch from languages to k-valued functions.

In this paper we study the dual question: is it possible that a language suddenly transits from very average-case easy to very average-case hard if we slightly increase the complexity of the distribution? Namely, we study the following question: does there exists a language L such that for all distributions E of complexity $g(n)$ the distributional problem (L, E) is easy on the average, but there exist an ensemble of distributions D of complexity $f(n)$ such that the distributional problem (L, D) is hard on the average?

- We consider two complexity measures of distributions:
  1. time complexity for sampling;
  2. time complexity of computing the distribution function.
- We say that a distributional problem (L, D) is easy on the average if there is a linear-time algorithm that for all n gives correct answer on $1 - \alpha(n)$ fraction (according to D) of inputs of length n, where $\alpha(n) = o(1)$. (It seems that we may claim the existence of a polynomial-time algorithm instead of linear-time, but it turns out that if there is

an example with a polynomial-time algorithm, then there is also an example with a linear-time algorithm).

- We consider two variants of the notion that (L, D) is hard on the average:
  1. strong hardness: every algorithm for infinitely many n give a correct answer on at most $\beta(n)$ fraction (according to D) of inputs of length n, where $\beta(n) = o(1)$; (It seems that we may claim this condition only for polynomial-time algorithms but it turns out that if there is an example that is hard for polynomial-time algorithms, then there is also an example that is hard for all algorithms.)
  2. weak hardness: every algorithm for infinitely many n give a correct answer on at most $1 - \beta(n)$ fraction (according to D) of inputs of length n. In this case it is reasonable to assume that $\alpha(n) = o(\beta(n))$.
- It is desirable for $f(n)$ to be not much larger than $g(n)$. In tight results $f(n)$ would be at most polynomial in $g(n)$; in other results $f(n)$ is bounded by a quasipolynomial in $g(n)$.

1.1 Our results

Samplable distributions. The most interesting complexity measure of distributions is the complexity of sampling. In Section 3.1 we consider the statement with the strong notion of hardness. We show that in this case the affirmative answer to our question is *equivalent* to the following *hierarchy for sampling distributions*: there exists a distribution D that is samplable in $f(n)$ steps such that for every distribution F that is samplable in $g(n)$ steps, the statistical distance between D and F is at least $1 - o(1)$.

Watson [9] recently proved the similar (but weaker for our goals) theorem:

▶ **Theorem** ([9]). *For all $a > 0$, $\epsilon > 0$ and $k \in \mathbb{N}$ there exists an ensemble $D \in$ **PSamp** such that:*

- *for all n the distribution D_n is concentrated on $\{1, 2, \dots, k\}$;*
- *for every ensemble of distributions $F \in$ **Samp**$[n^a]$ there exist infinitely many n such that statistical distance between D_n and F_n is at least $1 - \frac{1}{k} - \epsilon$.*

Watson's theorem is not sufficient for our goals since we need the statistical distance $1 - o(1)$ while the theorem gives the statistical distance $1 - \delta$ for all constants $\delta > 0$. We stress that from the equivalence result the tending of the statistical distance to 1 is necessary in order to get an example of a language that is easy according to easy distributions and hard for some more complicated distribution. The proof of Watson's theorem is based on the tree-like diagonalization; we explain (see details in the end of Section 3.1) why the tree-like diagonalization can not be used to get a statistical distance $1 - o(1)$ for polynomial f and g. The statistical distance in Watson's theorem is optimal for distributions concentrated on k values. Thus to get a statistical distance $1 - o(1)$ the function k should be increasing and thus the diagonalization tree should have outgoing degree at least $k(n)$ and this condition makes the diagonalization tree too large and it is impossible to layout it. We show that it is possible to layout the tree in the case when f and g differ quasipolynomially. We prove the hierarchy for sampling distributions for $f(n) = n^{\log^b n}$ and $g(n) = n^{\log^a n}$ for all $0 < a < b$. Our proof uses the proof strategy that is similar to Watson's theorem but our proof is significantly simpler and in particular, we do not use list-decodable error-correcting codes for the transmitting information. From the hierarchy for sampling distributions and the equivalence result we get the following theorem.

▶ **Theorem 1.** *For all $\epsilon > 0$ and $c > 0$ there exist a language L and a linear-time algorithm A such that for every polynomial-time samplable ensemble of distributions F and all n,*

$\Pr_{x\leftarrow F_n}[A(x) = L(x)] \geq 1 - \frac{1}{2^{(\log\log\log n)^c}}$ *and there exists* $D \in \mathbf{Samp}[n^{\log^\epsilon n}]$ *such that for every algorithm* B *for infinitely many* n, $\Pr_{x\leftarrow D_n}[B(x) = L(x)] \leq \frac{1}{2^{(\log\log\log n)^c}}$.

Note that although Theorem 1 argues only about deterministic algorithms B, it implies that for any probabilistic algorithm B' with running time bounded by a computable function, $\Pr_{x\leftarrow D_n}[\Pr_{B'}[B'(x) = L(x)] > 1/2] \leq \frac{1}{2^{(\log\log\log n)^c}}$, where the inner probability is taken over the random bits of algorithm B'. It is interesting to compare Theorem 1 with the result of Gutfreund, Shaltiel and Ta-Shma [4]; they proved that for every $\alpha(n) = o(1)$ there is a distribution D that is samplable in quasipolynomial-time such that for every **NP**-complete language L every polynomial-time randomized algorithm fails to compute L with probability at least $\alpha(n)$ for infinitely many n unless $\mathbf{NP} \subseteq \mathbf{BPP}$. In contrast to [4] Theorem 1 is unconditional, uses the strong notion of hardness and additionally states that L is easy for all polynomial-time samplable distributions, on the other hand the distribution from [4] is the same for all **NP**-complete languages and **NP**-complete languages are important while a language from Theorem 1 is an artificial language based on the tricky diagonalization.

In Section 3.2 we consider the weak notion of hardness and $f(n) = poly(g(n))$. Analogously to the strong hardness we show that in this case the affirmative answer to our question is equivalent to the following conjecture:

▶ **Conjecture 2.** *There exist infinitely small functions* $\beta(n)$ *and* $\alpha(n) = o(\beta(n))$ *such that for all integer* $a > 0$ *and* $b > 0$ *there exist an ensemble of distributions* $D \in \mathbf{PSamp}$*, an increasing sequence of integers* l_n *and a sequence of sets* $S_n \subseteq \{0,1\}^{l_n}$ *such that the following holds:* $D(S_n) > \beta(l_n)$ *for all* n*; for all* $F \in \mathbf{Samp}[n^a]$*,* $F(S_n) \leq \alpha(l_n)$ *for infinitely many* n*.*

Nontrivial condition on this condition is that $\alpha(n)$ is infinitely small. For constants α and β ($\alpha < \beta$) the statement follows from Watson's theorem. For infinitely small α the statement is nontrivial even in the case $\alpha(n) = \beta(n)$. We prove the following theorem.

▶ **Theorem 3.** *For all integer* $a > 0$ *and* $b > 0$ *there exist an ensemble of distributions* $D \in \mathbf{PSamp}$*, a sequence of integers* l_n *and a sequence of sets* $S_n \subseteq \{0,1\}^{l_n}$ *such that the following holds:*

- $D(S_n) > \frac{1}{l_n^b}$ *for all* n*;*
- *For all* $F \in \mathbf{Samp}[n^a]$*,* $F(S_n) \leq \frac{1}{l_n^b}$ *for infinitely many* n*.*

Computable distributions. In Section 4 we consider a complexity of a distribution as the complexity of computing the distribution function. In case of computable distributions in contrast to samplable ones it is possible to find an element with small probability using binary search in polynomial time. However there is the following difficulty: it is not clear how to verify efficiently whether an algorithm computes a distributional function or not. This difficulty prevents to construct the universal computable distribution that dominates all other computable distributions while it is possible to do in the samplable case. We overcome this difficulty and prove the following result for the strong hardness and $f(n) = poly(g(n))$:

▶ **Theorem 4.** *For every* $a > 0$ *there exists a language* L *and an ensemble of polynomial-time computable distributions* D *such that:*

- *there exists a linear-time algorithm* A *such that* $\Pr_{x\leftarrow E_n}[A(x) \neq L(x)] = O(2^{-n})$ *for all* E *that are computable in* $O(n^a)$ *steps;*
- *for every algorithm* A *and for all* n*,* $\Pr_{x\leftarrow D_n}[A(x) \neq L(x)] > 1 - \frac{1}{2^{n-1}}$.

2 Preliminaries

An ensemble of distributions is a sequence $\{D_n\}_{n=1}^{\infty}$, where D_n is a probability distribution on $\{0,1\}^n$. Sometimes it is convenient to assume that D_n is concentrated on $\{0,1,\dots,2^n-1\}$.

For two distributions A and B on $\{0,1\}^n$ the statistical distance between them is $\Delta(A,B) = \max_{S\subseteq\{0,1\}^n} |\Pr_{x\leftarrow A}[x\in S] - \Pr_{x\leftarrow B}[x\in S]|$.

A distributional problem is a pair (L,D) that consists of the language L and the ensemble of distributions D. Let $\delta:\mathbb{N}\to[0,1]$ be a function. We say that a distributional problem (L,D) is heuristically decidable it time $t(n)$ with error $\delta(n)$ if there exists an algorithm A such that A runs in $O(t(n))$ steps on the inputs of length n and the following holds: $\Pr_{x\leftarrow D_n}[A(x)\neq L(x)]\leq\delta(n)$ for all n. We denote it as $(L,D)\in \mathrm{Heur}_{\delta(n)}\mathbf{DTime}[t(n)]$. We also define a class of distributional problems $\mathrm{Heur}_{\delta(n)}\mathbf{P} = \bigcup_{c>0}\mathrm{Heur}_{\delta(n)}\mathbf{DTime}[n^c]$.

We also define a class $\mathrm{Heur}_{\delta(n)}\mathbf{R}$ that consists of all distributional problems (L,D) such that there exists an algorithm A such that $\Pr_{x\leftarrow D_n}[A(x)\neq L(x)]\leq\delta(n)$ for all n.

We say that an ensemble of distributions D is samplable in time $t(n)$ if there exists a randomized algorithm S that on the input 1^n runs in at most $O(t(n))$ steps and $S(1^n)$ is distributed accordingly D_n. The set of all ensembles that are samplable in time $t(n)$ we denote as $\mathbf{Samp}[t(n)]$. We consider the set $\mathbf{PSamp} = \bigcup_{c>0}\mathbf{Samp}[n^c]$ of all polynomial-time samplable ensembles.

3 Samplable distributions

3.1 Strong hardness

▶ **Definition 5.** We say that time constructible functions f and g satisfy the *hierarchy property of sampling distributions* with parameter $\lambda(n)$ if there exists an ensemble of distributions $D\in\mathbf{Samp}[f(n)]$ such that for every ensemble of distributions $F\in\mathbf{Samp}[g(n)]$, there exist infinitely many numbers n such that the statistical distance between D_n and F_n is at least $1-\lambda(n)$.

▶ **Definition 6.** We say that time constructible functions f and g satisfy the *hierarchy property on complexity of distributional problems* with parameters $\alpha(n)>0$ and $\beta(n)>0$ if there exist a language L and an ensemble of distributions $D\in\mathbf{Samp}[f(n)]$ steps such that:

- $(L,F)\in\mathrm{Heur}_{\alpha(n)}\mathbf{P}$ for all $F\in\mathbf{Samp}[g(n)]$;
- $(L,D)\notin\mathrm{Heur}_{1-\beta(n)}\mathbf{P}$.

We say that f and g satisfy *strong* hierarchy property on complexity of distributional problems if the conditions are formulated as:

- There is a linear-time algorithm A such that for all $F\in\mathbf{Samp}[g(n)]$ $\Pr_{x\leftarrow F_n}[A(x)=L(x)]\geq 1-\alpha(n)$ for all n large enough;
- $(L,D)\notin\mathrm{Heur}_{1-\beta(n)}\mathbf{R}$.

▶ **Lemma 7.** *For every time constructible functions $f(n)$, $h(n)$ and $g(n)\geq n$ if f and h satisfy the hierarchy property on sampling distributions with parameter $\lambda(n)$ and $g(n)\log g(n) = o(h(n))$ then f and g satisfy the strong hierarchy property on complexity of distributional problems with parameters $\alpha(n)$ and $\lambda(n)$ for $\alpha(n)=\omega(\lambda(n))$.*

Proof. Let A_i be an enumeration of all randomized algorithms supplied with an alarm clock that interrupt their executions after $O(g(n))$ steps. We will think about A_i as algorithms that sample distributions; that is the output of $A_i(1^n)$ we interpret as a string from $\{0,1\}^n$ by some fixed way. Let B be an algorithm that samples a distribution as follows: on input

1^n with probability $\frac{1}{2}$ it executes $A_1(1^n)$ (and returns its result), with probability $\frac{1}{2^2}$ it executes $A_2(1^n)$, ..., with probability $\frac{1}{2^{n-1}}$ it executes $A_{n-1}(1^n)$ and with probability $\frac{1}{2^{n-1}}$ executes $A_n(1^n)$. Let B define an ensemble of distributions E. It is straightforward that $E \in \mathbf{Samp}[h(n)]$.

Since f and h satisfy the hierarchy property of sampling distributions, there exists an ensemble $D \in \mathbf{Samp}[f(n)]$ such that $\Delta(D_n, E_n) \geq 1 - \lambda(n)$ for infinitely many numbers n. We denote the set of all such n as $I = \{n_1, n_2, \dots\}$. For $n \in I$ there exists a set $S_n \subseteq \{0,1\}^n$ such that $D_n(S_n) - E_n(S_n) \geq 1 - \lambda(n)$, hence $E_n(S_n) \leq \lambda(n)$.

We will define a language L such that $L \subseteq \bigcup_{n \in I} S_n$. Let T_i be an enumeration of all algorithms. We define L such that for every $x \in S_{n_k}$, $x \in L$ if and only if T_k does not stop on the input x or rejects it. By the construction $(L, D) \notin \mathrm{Heur}_{1-\lambda(n)}\mathbf{R}$.

We consider an algorithm that returns 0 on every input. If $R \in \mathbf{Samp}[g(n)]$, then there exists i such that A_i samples R. For $n \geq i$ for every set $S \subseteq \{0,1\}^n$ the following inequality holds: $E(S) \geq 2^{-i}R(S)$. Hence for every ensemble R from $\mathbf{Samp}[g(n)]$ this algorithm has error at most $c\lambda(n)$, where c is a constant that depends only on the ensemble R; $c\lambda(n) < \alpha(n)$ for n large enough. ◀

We also prove the opposite implication.

▶ **Lemma 8.** *If f and g satisfy the hierarchy property of complexity of distributional problems with parameters $\alpha(n)$ and $\beta(n)$ then f and g satisfy the sampling hierarchy property with parameter $\alpha + \beta$.*

Proof. For all $F \in \mathbf{Samp}[g(n)]$ there exists a polynomial time algorithm A that solves (L, F) in $\mathrm{Heur}_{\alpha(n)}\mathbf{P}$ and also $(L, D) \notin \mathrm{Heur}_{1-\beta(n)}\mathbf{P}$. Let S_n be set of all $x \in \{0,1\}^n$ such that $A(x) = L(x)$. We know that $F_n(S_n) \geq 1 - \alpha(n)$ for all n and $D_n(S_n) \leq \beta(n)$ for for infinitely many n. Hence $\Delta(D_n, F_n) \geq F_n(S_n) - D_n(S_n) \geq 1 - \alpha(n) - \beta(n)$ for infinitely many n. ◀

Lemma 7 and Lemma 8 implies that if f and g satisfy the hierarchy property of the complexity of distributional problems with two infinitely small parameters then f and $g/\log^2 g$ satisfy the strong hierarchy property on the complexity of distributional problems with two infinitely small parameters. As we mentioned Watson [9] proved that for every $a > 0$, $\epsilon > 0$ and every constant k there exists $b > 0$ such that n^a and n^b satisfy the hierarchy property on sampling distributions with parameter $\frac{1}{k} + \epsilon$. In fact Watson proved the stronger statement since ensemble D is concentrated on k inputs. Watson conjectured that for every $a > 0$ there exists infinitely small function $\alpha(n)$ there exists $b > 0$ such that n^a and n^b satisfy the hierarchy property on sampling distributions with parameter $\alpha(n)$. This statement is still an open question. We prove the following theorem:

▶ **Theorem 9.** *For every a, b, c such that $0 < a < b$ and $c > 0$ functions $f(n) = n^{\log^b n}$ and $g(n) = n^{\log^a n}$ satisfies the sampling hierarchy property with the parameter $\lambda(n) = \frac{1}{2^{(\log\log\log n)^c}}$.*

▶ **Corollary 10.** *For every a, b, c such that $0 < a < b$ and $c > 0$ functions $f(n) = n^{\log^b n}$ and $g(n) = n^{\log^a n}$ satisfies the strong hierarchy property on complexity of distributional problems with parameters $\alpha(n) = \beta(n) = \frac{1}{2^{(\log\log\log n)^c}}$.*

Note that Theorem 1 stated in the introduction follows from Corollary 10.

Before giving a formal proof of Theorem 9 we present an idea of the proof.

In the following, we assume that random variables and elements of ensembles of distributions take values from the set $\{0, 1, \dots, 2^n - 1\}$ instead of $\{0,1\}^n$.

Our proof like a proof of the Watson's theorem is based on the tree-like diagonalization. We construct a distribution D and diagonalize over all distributions samplable in $O(g(n))$

steps by the enumeration of their generators A_i. For the i-th distribution we will prove that the statistical distance between D and $A_i(1^n)$ is large for some n from $[n_i, n_i^*]$, where n_i^* is significantly more than n_i. For every i we construct a tree T_i with vertices uniquely marked with numbers from $[n_i, n_i^*]$. The root of T_i is marked by n_i^* and leaves of T_i are marked with numbers that are about n_i. The number of a parent is greater than the number of a child also the number of a parent is bounded by a quasipolynomial in numbers of its child. Let t be an element from $\{0, 1, \dots, 2^{n_i} - 1\}$ such that in all leaves A_i-probability of t is less then $\lambda(m_i)$, where m_i is the maximum leaf. Such t exists since there are not too many leaves, the possible values of distributions is at least 2^{n_i} and for every distribution the number of elements with probability at least $\lambda(n)$ is at most $\frac{1}{\lambda(n)}$. The distribution $D_{n_i^*}$ is concentrated on t. We assume that for all $n \in [n_i; n_i^*]$ the statistical distance between $A_i(1^n)$ and D_n is less than $1 - \lambda(n)$. Our goal is to define D in such a way that in at least one leaf D is concentrated on t. This will contradict our assumption and the definition of t.

We will transmit information about t from a parent to at least one of its children. The distribution D on the children of p has the following property: if D_p is concentrated (with probability $1 - \epsilon$) on some element, then D_n is concentrated on the same element for at least one child n of p. From the assumption about statistical distances we have that $\Pr[A_i(1^p) = t] \geq \lambda(p) - \epsilon$, hence there are at most $\frac{2}{\lambda}$ candidates on the role of t if we have an access to $A_i(1^p)$. We generate a list of all elements with $A_i(1^p)$-probability at least $\lambda(p) - \epsilon$. In the first child of p we make D concentrated on the first element of the list, on the second child on the second element and so on. There is a problem that there are possibly different lists will be generated in different children; we solve this problem by using several thresholds for frequencies. Formally we do it in the following lemma:

▶ **Lemma 11.** *There is an algorithm $C^\bullet(n, i, \delta, \lambda)$ that has an oracle access to some random variable γ taking values in $\{0, 1, \dots, 2^n - 1\}$ such that for all positive integer n and $\delta, \lambda \in (0, 1]$ if $\Pr[\gamma = t] \geq \lambda$ for some t, then there is some integer $0 \leq i \leq \lceil 1 + \frac{1}{\lambda} \rceil^2$ such that $\Pr[C^\gamma(n, i, \delta, \lambda) = t] \geq 1 - \delta$ and $C^\bullet$ runs at most $\mathrm{poly}(n, \log \frac{1}{\delta}, \frac{1}{\lambda})$ steps.*

Proof. Consider the following algorithm $C^\gamma(n, i, \delta, \lambda)$:

1. Let $k = \lceil \frac{1}{\lambda} + 1 \rceil$ and $\epsilon = \frac{\lambda^3}{10k}$;
2. We interpret i as a pair (a, b), where $a, b \in [k]$;
3. Request the oracle for $N = \lceil \frac{2(n+1+\log \frac{1}{\delta})}{\epsilon^2} \rceil$ samples of γ;
4. Consider the list $y_1, \dots, y_s$ of all elements with frequency at least $\lambda - \epsilon a$;
5. Return y_b if $b \leq s$ or 0 otherwise.

Note that for $\lambda \in (0, 1]$

$$k(\lambda - \epsilon(2k)) \geq (\frac{1}{\lambda} + 1)(\lambda - \lambda^3/5) = 1 + \lambda - \lambda^2/5 - \lambda^3/5 > 1. \tag{1}$$

Hence the number of elements x such that $\Pr[\gamma = x] > \lambda - \epsilon k$ is less than k; by the similar reasons $s < k$, where s the size of the list in the 4-th step of the algorithm C.

Consider intervals $I_j = [\lambda - \epsilon j - \epsilon/2; \lambda - \epsilon j + \epsilon/2]$. There is $a \in [k]$ such that $\Pr[\gamma = x] \notin I_a$ for all x since otherwise $1 = \sum_x \Pr[\gamma = x] \geq k(\lambda - \epsilon k - \epsilon/2)$ that contradicts inequality (1). Hence there is $a \in [k]$ such that $|\Pr[\gamma = x] - \lambda - \epsilon a| > \epsilon/2$ for all x.

Let $x_1, \dots, x_l$ be the list of all elements x such that $\Pr[\gamma = x] > \lambda - \epsilon a$. We know that if $\Pr[\gamma = x] > \lambda - \epsilon a$, then $\Pr[\gamma = x] > \lambda - \epsilon a + \epsilon/2$ and also if $\Pr[\gamma = x] \leq \lambda - \epsilon a$, then $\Pr[\gamma = x] < \lambda - \epsilon a - \epsilon/2$. For given a for every $j \in [l]$, x_j appears in the list from 4th step of algorithm C with probability at least $1 - 2e^{-\epsilon^2 N/2}$. If $\Pr[\gamma = x] \leq \lambda - \epsilon a$ then by Chernoff bound x does not appear in the list from the 4th step of the algorithm C with

probability at least $1-2e^{-\epsilon^2 N/2}$. Since γ is concentrated on the set of size 2^n with probability at least $1-2^{n+1}e^{-\epsilon^2 N/2} \geq 1-\delta$ the list generated on 4th step of algorithm C is precisely the list $x_1,\dots,x_l$. Since $\Pr[D=t]>\lambda$, there is b such that $x_b = t$. Hence if $i=(a,b)$ then $\Pr[C^\gamma(n,i,\delta,\lambda)=t] \geq 1-\delta$. ◀

Proof of Theorem 9. Our proof is based on the tree-like delayed diagonalization. We diagonalize against all randomized algorithms supplied with a $O(g(n))$-alarm clock, we interpret them as samplers of distributions. Let $A_1, A_2, \dots$ be an enumeration of all randomized algorithms supplied with a $O(g(n))$-alarm clock.

Let us consider an $\epsilon > 0$ such that $(1+a)(1+\epsilon) < (1+b)$ and fix some c. We define integer sequences n_i and n_i^* such that $n_1 = 1$, $n_i^* = 2^{(\log n_i)^{(1+\epsilon)^{d_i}}}$, where $d_i = \lceil \log_{1+\epsilon} 2 \rceil \lceil (\log\log n_i)^2 \rceil$ and $n_{i+1} = n_i^* + 1$. For every i we define an ensemble of distributions D_n for $n \in \{n_i, n_i+1, \dots, n_i^*\}$ such that there exists $k \in \{n_i, n_i+1, \dots, n_i^*\}$ such that $\Delta(D_k, A_i(1^k)) \geq 1-\lambda(k)$.

▶ **Lemma 12.** *For every $\epsilon > 0$ there exists a family of trees T_i such that:*

1. *The set of vertices of T_i is a subset of $\{n_i, n_i+1, \dots, n_i^*\}$;*
2. *n_i^* is the root of T_i;*
3. *All leaves of T_i have numbers at most $m_i = 2n_i$;*
4. *The depth of T_i is $d_i = \lceil \log_{1+\epsilon} 2 \rceil \lceil (\log\log n_i)^2 \rceil$;*
5. *If p is a parent of n then $p \leq 2^{\log^{1+\epsilon} n}$;*
6. *There is an algorithm that for any vertex n of T_i outputs the parent p of n and the number of children of p that are less than n in poly(n) steps;*
7. *For every inner vertex v of T_i, v has $k = \lceil \frac{1}{\lambda(n_i^*)} + 1 \rceil^2$ children.*

Proof. Let us denote $\delta = \lceil \log_{1+\epsilon} 2 \rceil$. We define a tree T_i as a complete balanced tree with depth d_i. The number of leaves in the tree can be estimated as follows: $k^{d_i} \leq (2^{(\log\log\log n_i^*)^{3c}})^{\delta(\log\log n_i)^2} \leq (2^{(\log\log n_i)^{12c}})^{\delta(\log\log n_i)^2} = 2^{\delta(\log\log n_i)^{24c}} \leq n_i$.

The root n_i^* is the only vertex on the zero level. There are exactly k^s vertices on s-th level. Let $a_{i,j} = 2^{(\log n_i)^{(1+\epsilon)^j}}$, where $j \in \{0,1,2,\dots\}$. Vertices of T_i on level $(d_i - s)$ are $[a_{i,s}; a_{i,s} + k^{d_i - s} - 1]$.

Note that $a_{i,s+1} - a_{i,s} \geq a_{i,1} - a_{i,0} = 2^{(\log n_i)^{(1+\epsilon)}} - n_i \geq 2^{(\log n_i)^{(1+\epsilon)-1}} > n_i \geq k^{d_i} \geq k^{d_i - s}$. Hence on all levels there is enough place for vertices.

The parent of j-th vertex on s-th level has number $\lfloor \frac{j}{k} \rfloor$. Let $h(n) = n^{\log^\epsilon n}$. Since $h(n+k) \geq h(n) + k$ we have $h(2^{\log^{(1+\epsilon)^s} n_i} + j) \geq h(2^{\log^{(1+\epsilon)^s} n_i}) + j \geq 2^{\log^{(1+\epsilon)^{s+1}} n_i} + j/k$, therefore the property 5 is satisfied. The verification of other properties is straightforward. ◀

Now we describe an algorithm that samples D_n for $n \in \{n_i, \dots, n_i^*\}$ in $O(f(n))$ steps.

1. If $n = n_i^*$ then output the minimal $t_i \in \{0,1,\dots,2^{n_i}-1\}$ such that for all $l \in [n_i; m_i]$ we have that $\Pr[A_i(1^l) = t_i] < \lambda(n_i)/2$. Such t_i indeed exists since for every l there are at most $\frac{2}{\lambda(n_i)}$ elements with $A_i(1^l)$-probability at least $\lambda(n_i)/2$ and $\frac{2}{\lambda(n_i)} m_i \leq 2^{n_i}$. Such t_i can be found in at most $m_i c_i g(m_i) 2^{c_i g(m_i)}$ steps by brute force search over all possible random bits, where c_i is a constant that depends on i.

$$m_i c_i g(m_i) 2^{c_i g(m_i)} \leq 2^{m_i g(m_i)} \leq 2^{2n_i g(2n_i)} \leq 2^{2n_i (2n_i)^{2\log^a n_i}} < 2^{2^{4\log^{(1+a)} n_i}} \leq 2^{2^{2^{4(1+a)\log\log n_i}}} \leq 2^{2^{2^{(\log\log n_i)^2}}} < n_i^* = o(f(n_i^*))$$

2. If n is not a vertex of T_i then return 0.

3. Otherwise, let p be the parent of n and j is a number of n in the list of all children of p. By the property of T_i, $p \leq 2^{\log^{1+\epsilon} n}$ and such p can be found in $poly(n)$ steps. We return $C^{A_i(1^p)}(p, j, \lambda(n)/2, \lambda(p)/2)$, where C is the algorithm from Lemma 11. By Lemma 11 C runs at most $\mathrm{poly}(p)$ steps and on every step the simulation of $A_i(1^p)$ occupies at most $c_i g(p)$ steps. Note that $c_i g(p)\mathrm{poly}(p) < 2^{2\log^{a+1} p} < 2^{2\log^{1+a}(2^{\log^{1+\epsilon} n})} = 2^{2\log^{(1+a)(1+\epsilon)} n} < 2^{\log^{(1+b)} n} = f(n)$.

For the sake of contradiction we assume that for all $n \in \{n_i, \dots, n_i^*\}$, $\Delta(D_n, A_i(1^n)) < 1 - \lambda(n)$. By induction on the level s of T_i we prove that there is a vertex v of level s in T_i such that $D_v(t_i) \geq 1 - \lambda(v)/2$. If $D_v(t_i) \geq 1 - \lambda(v)$ for some leaf v then $\Pr[A_i(1^v) = t_i] \geq (1 - \lambda(v)/2) - (1 - \lambda(v)) = \lambda(v)/2$ but we define t_i such that $\Pr[A_i(1^v) = t_i] < \lambda(v)/2$. Hence we will get a contradiction in leaves.

The base of induction follows from the construction of $D_{n_i^*}$. Let us prove the inductive step from s to $s + 1$. Let v be a vertex of level s such that $D_v(t_i) \geq 1 - \lambda(v)/2$. If v is a leaf then we are done. Otherwise $\Pr[A_i(1^v) = t_i] > \lambda(v)/2$ since $\Delta(D_v, A_i(1^v)) < 1 - \lambda(v)$. Hence by Lemma 11 there is a child u with number j among the all children of v such that $\Pr[C^{A_i(1^v)}(v, j, \lambda(u)/2, \lambda(v)/2) = t_i] > 1 - \lambda(u)/2$. ◀

Our proof in contrast to Watson's proof does not use error correcting codes with list decoding. This is because we find one element that has a small probability for all leaves of the tree. This trick was impossible in Watson's case since all distributions were concentrated on a constant number of points. In Watson's proof, there were a lot of information transmitted from the root to leaves, and parts of this information were stored in different vertices. Watson used list error decoding codes in order to prevent information distortion.

Now we show why this approach cannot be adapted to the case of $g(n) = n^a$ and polynomial $f(n)$. The problem is the following: for nonconstant $\lambda(n)$ the tree T_i should have nonconstant degree: every inner vertex has at least k_i children, where k_i goes to infinity. In the root of the tree, we have to make exponential in any leaf number of steps; and the parent of every node n should be at most polynomial of every children. Thus for every leaf l the distance between root and l is at least $\Omega(\log \ell)$. Let m_i be the leaf with the maximal number; then the distance from the root to m_i is at least $L = \Omega(\log m_i)$. Let S be the set of vertices such that their numbers are less then m_i but the numbers of their parents are more then m_i. Note that all vertices on the distance L from the root must either be in S or have a descendent in S. Therefore the size of S should be at least k_i^L that is greater then m_i for large i, since k_i goes to infinity. But this is a contradiction since S is set of vertices with numbers less then m_i.

3.2 Weak hardness

In this section we consider statement of the problem with the weak notion of hardness and tight hierarchy ($f(n) = poly(g(n))$). We start from equivalent formulations:

▶ **Proposition 13.** *The following statements are equivalent:*

1. *There exists infinitely small functions $\beta(n)$ and $\alpha(n) = o(\beta(n))$ such that for all $a > 0$ there exists an ensemble of distributions $D \in$ **PSamp** and a language L such that the following holds:*
   - $(L, F) \in \mathrm{Heur}_{\alpha(n)}\mathbf{P}$ *for all* $F \in \mathbf{Samp}[n^a]$;
   - $(L, D) \notin \mathrm{Heur}_{\beta(n)}\mathbf{P}$.

2. *There exists infinitely small functions $\beta(n)$ and $\alpha(n) = o(\beta(n))$ such that for all $a > 0$ there exist an ensemble of distributions $D \in$ **PSamp**, an increasing sequence of integers l_n and a sequence of sets $S_n \subseteq \{0,1\}^{l_n}$ such that the following holds:*
   - *$D(S_n) > \beta(l_n)$ for all n;*
   - *For all $F \in$ **Samp**$[n^a]$, $F(S_n) \leq \alpha(n)$ for infinitely many n.*
3. *There exists infinitely small functions $\beta(n)$ and $\alpha(n) = o(\beta(n))$ such that for all $a > 0$ there exists an ensemble of distributions $D \in$ **PSamp** and a language L such that the following holds:*
   - *there exists linear-time algorithm A such that for all $F \in$ **Samp**$[n^a]$, $\Pr_{x \leftarrow F_n}[L(x) = A(x)] \geq 1 - \alpha(n)$ for all n large enough;*
   - *$(L, D) \notin \mathrm{Heur}_{\beta(n)}\mathbf{R}$.*

We prove the statement that is weaker than statement 2 from Proposition 13. Namely we prove it in the case $\alpha(n) = \beta(n) = \frac{1}{n^b}$. By the similar way it is possible to prove it for other infinitely small functions: $\frac{1}{2^n}$, $\frac{1}{\log n}$ etc.

Now we are ready for proving Theorem 3. We start from the intuition of the proof. For simplicity we start from the proof of the other statement with threshold $\frac{1}{2}$ instead of $\frac{1}{l_n^b}$. We use the delayed diagonalization; we consider integer sequences n_i and n_i^* such that $n_1 = 1$, $n_i = n_i^* + 1$, $n_i^* = 2^{n_i^a}$. Let F_i be enumeration of all randomized algorithms with alarm clock n^{a+1} such that every algorithm appears infinitely many times in this enumeration; we consider F_i as samplers of distributions.

We denote by $T_{0,n}$ and $T_{1,n}$ the set of binary strings of length n starting with 0 and 1 respectively. Consider the following sampler of the distribution D_n: if $n = n_i^*$, we find $t_i \in \{0,1\}$ such that $\Pr[F_i(1^{n_i}) \in T_{t_i,n_i}] \leq \frac{1}{2}$ and return the random element from T_{t_i,n_i^*}; if $n_i \leq n < n_i^*$ we execute $F_i(1^{n+1})$ if it returns a string starting with $s \in \{0,1\}$ we return a random element from $T_{s,n}$.

Assume that there exists $E \in$ **Samp**$[n^a]$ such that for all $n > n_0$ if for some $S \subseteq \{0,1\}^n$, $\Pr[D_n \in S] > \frac{1}{2}$, then $\Pr[F_i(1^n) \in S] > \frac{1}{2}$. Let F_i is a sampler for E and $i > n_0$. We know that $\Pr[D_{n_i^*} \in T_{t_i,n_i^*}] = 1$, then $\Pr[F_i(1^{n_i^*}) \in T_{t_i,n_i^*}] > \frac{1}{2}$, thus $\Pr[D_{n_i^*-1} \in T_{t_i,n_i^*-1}] > \frac{1}{2}$ and so on. Finally we get $\Pr[F_i(1^{n_i}) \in T_{t_i,n_i}] > \frac{1}{2}$ and this contradicts the definition of t_i.

For threshold $\frac{1}{k}$ the proof will be the same but we split $\{0,1\}^n$ into $\log k$ parts. In case of threshold $\frac{1}{n^b}$ we will have a different number of parts for different n, and we will use trees of intervals instead of chains.

Proof of Theorem 3. Consider an enumeration of all randomized algorithms F_i with alarm clock n^{a+1} such that every algorithm appears infinitely many times in this enumeration; we consider F_i as samplers of distributions. We define integer sequences n_i and n_i^* such that $n_1 = 1$, $n_i^* = 2^{n_i^a}$, and $n_{i+1} = 2n_i^*$.

Split all strings of length n on n^b nonempty sets; we call them intervals and denote by $T_{j,n}$ for $j \in \{1, 2, \ldots, n^b\}$. For $n \in [n_i; n_i^*]$ we define a graph (it will be a forest) as follows:

- The set of vertexes of the graph is the set of all intervals $T_{j,n}$ for $n = 2^k$ and $n_i \leq n \leq n_i^*$;
- All elements of T_{j,n_i} are roots of trees of the forest;
- For $n \in \{n_i, 2n_i, 4n_i, \ldots, n_i^*/2\}$, $T_{j,n}$ has 2^b children: $\{T_{j',2n} \mid 2^b(j-1) \leq j' \leq 2^b j - 1\}$.
- All elements of T_{j,n_i^*} are leaves of trees of the forest;

We define a sampler for D as follows. It gets on the input 1^n:

- If $n = n_i^*$ for some i, then find an interval T_{j,n_i} with the smallest probability according to $F_i(1^{n_i})$. If there are several such T_{j,n_i}, we take one with the minimal j. (Note that this can be done in $poly(n_i^*)$ time by brute-force). Then chose random descendent of T_{j,n_i} on length n_i^* and return some string form this descendent. Note that $\Pr[F_i(1^{n_i}) \in T_{j,n_i}] \leq \frac{1}{n_i^b}$;

- If $n_i \le n < n_i^*$ for some i, then run $F_i(1^{2n})$ and if the result belongs to a descendent of $T_{j,n}$ for some j, then return random string from $T_{j,n}$.

Let us prove that for all i there exists j and $n \in [n_i; n_i^*]$ such that $\Pr[D_n \in T_{j,n}] > \frac{1}{n^b}$ and $\Pr[F_i(1^n) \in T_{j,n}] \le \frac{1}{n^b}$. (This will conclude the proof of the theorem if we choose $S_i = T_{j,n}$.) Assume the opposite; that is for all j and $n \le n_i^*$ if $\Pr[F_i(1^n) \in T_{j,n}] \le \frac{1}{n^b}$, then $\Pr[D_n \in T_{j,n}] < \frac{1}{n^b}$. Let T_{j,n_i} be an interval with the smallest probability according to $F_i(1^{n_i})$, hence $\Pr[F_i(1^{n_i}) \in T_{j,n_i}] \le \frac{1}{n_i^b}$. By induction on l we prove that for all $n = 2^l n_i$ (and $n \le n_i^*$) there exists k such that $T_{k,n}$ is a descendant of T_{j,n_i} and $\Pr[D_n \in T_{k,n}] \le \frac{1}{n^b}$. The base case $l = 0$ is already proved. Let us prove the inductive step from l to $l+1$. Let $n = 2^l n_i$. Assume that $\Pr[D_n \in T_{k,n}] \le \frac{1}{n^b}$ then by the pigeonhole principle and construction of D there is one of 2^b children of $T_{k',2n}$ such that $\Pr[F_i(1^{2n}) \in T_{k',2n}] \le \frac{1}{(2n)^b}$ and hence by assumption $\Pr[D_{2n} \in T_{k',2n}] \le \frac{1}{(2n)^b}$. Therefore there exists k such that $\Pr[D_{n_i^*} \in T_{k,n_i^*}] \le \frac{1}{(n_i^*)^b}$ and T_{k,n_i^*} is a descendant of T_{j,n_i}, but the construction of D implies that the D-probability of every descendant of T_{j,n_i} on length n_i^* is equal to $\frac{n_i^b}{(n_i^*)^b} > \frac{1}{(n_i^*)^b}$. ◀

▶ **Corollary 14.** *For all $a > 0$ and $b > 0$ there exists a ensemble of distributions $D \in \mathbf{PSamp}$, a language L and a linear-time algorithm A such that the following holds:* $\Pr_{x \leftarrow F_n}[A(x) \neq L(x)] = O(\frac{1}{n^b})$ *for all $F \in \mathbf{Samp}[n^a]$; $(L, D) \notin \mathrm{Heur}_{\frac{1}{n^b}} \mathbf{R}$.*

4 Computable distributions

Ensemble of distributions D_n is computable in time $t(n)$ if for all n probabilities of all elements according to D_n are binary rational numbers and there exists an algorithm $A(x)$ that runs in $O(t(|x|))$ steps and computes the cumulative distribution function of D_n (i.e. $\sum_{y \le x} D_n(x)$, where $\le$ is lexicographical order). The set of all ensembles that are computable in time $t(n)$ we denote as $\mathbf{Comp}[t(n)]$. The set $\mathbf{PComp} = \bigcup_{c>0} \mathbf{Comp}[n^c]$ is the set of all ensembles computable in polynomial time.

▶ **Lemma 15.** *If an ensemble $D \in \mathbf{PSamp}$ and for all n the distribution D_n is concentrated on one element, then $D \in \mathbf{PComp}$.*

It is possible to prove the statement that is analogous to hierarchy property of n^a and n^b of sampling distributions but for computable distributions.

▶ **Proposition 16.** *For all $a > 0$ there exists an ensemble $D \in \mathbf{PComp}$ such that for all ensembles $F \in \mathbf{Comp}[n^a]$ there are infinitely many numbers n such that $\Delta(D_n, F_n) \ge 1 - 2^{-n}$.*

Now we prove Theorem 4 that is similar to hierarchy property of n^a and n^b on the complexity of distributional problems but for computable distributions.

Proof. We cannot literally repeat the proof of Lemma 7 regardless of we have even already proved Proposition 16. The reason is the following: not every algorithm computes the distributional function, it is not necessary that it computes even monotonic function. And it is not easy to verify that algorithms compute a distribution function.

Let A_i be an enumeration of all algorithms supplied with alarm-clock Cn^a, where C is some constant. We interpret them as algorithms that compute distribution functions. However, we remember that it is not necessary that all of them computes a correct distribution function. We interpret the result of $A_i(x)$ as a binary real number between 0 and 1.

For every n we will show that it is possible in $\mathrm{poly}(n)$ time to find $x_n \in \{0,1\}^n$ such that if $i \in \{1, 2, \ldots, n\}$ and A_i is distributional function, then the A_i-probability of x_n is at most

2^{i-n}. The distribution D_n would be concentrated on x_n; the resulting ensemble is computable in polynomial time by Lemma 15. If for all n we find such x_n, then we may define L similarly to the proof of Lemma 7. Namely we will choose L such that $L \subseteq \bigcup_n \{x_n\}$ and $x_n \in L$ if and only if n-th algorithm in the enumeration of all algorithms rejects x_n. For all $F \in \mathbf{Comp}[n^a]$ the algorithm that returns 0 on all inputs decides (L, F) in $\mathrm{Heur}_{2^{i-n}}\mathbf{DTime}[n]$, if F is computable by A_i in our enumeration. By the construction $(L, D) \notin \mathrm{Heur}_{1-\frac{1}{2^{n-1}}}\mathbf{P}$.

Now we describe the procedure of finding strings x_n. Initially $I = \{1, 2, \ldots, n\}$, we will delete element i from I if we discover that A_i is not a distribution function on $\{0,1\}^n$. On each iteration we define $F(x) = \sum_{i \in I} \frac{1}{2^i} A_i(x)$. By binary search we try to find such element $x \in \{0,1\}^n$ that $F(x) - F(x') \leq 2^{-n}$, where x' is lexicographical predecessor of x and $F(x') = 0$ if $x = 0^n$. If binary search succeeds, then $x_n := x$. If binary search fails then it means that we discover nonmonotonicity of $F(x)$, using this we may find $i \in I$ such that A_i is nonmonotonic and exclude all such i from I and start new iteration. If $I = \emptyset$ then choose $x_n = 0^n$, in other cases for all $i \in I$ if A_i computes a correct distribution function then x_n has probability at most 2^{i-n}. ◀

Acknowledgements. The authors are grateful to anonymous reviewers for useful comments.

References

1 László Babai, Paul Erdős, and Stanley Selkow. Random graph isomorphism. *SIAM J. Comput.*, 9(3):628–635, 1980.

2 Shai Ben-David, Benny Chor, Oded Goldreich, and Michael Luby. On the theory of average case complexity. *J. Comput. Syst. Sci.*, 44(2):193–219, 1992. doi:10.1016/0022-0000(92)90019-F.

3 Yuri Gurevich and Saharon Shelah. Expected computation time for hamiltonian path problem. *SIAM J. Comput.*, 16(3):486–502, 1987.

4 Dan Gutfreund, Ronen Shaltiel, and Amnon Ta-Shma. If NP languages are hard on the worst-case, then it is easy to find their hard instances. *Computational Complexity*, 16(4):412–441, 2007. doi:10.1007/s00037-007-0235-8.

5 Dmitry Itsykson, Alexander Knop, and Dmitry Sokolov. Heuristic time hierarchies via hierarchies for sampling distributions. In *Algorithms and Computation – 26th International Symposium, ISAAC 2015, Nagoya, Japan, December 9-11, 2015, Proceedings*, pages 201–211, 2015.

6 Ming Li and Paul M.B. Vitanyi. Average case complexity under the universal distribution equals worst case complexity. *Information Processing Letters*, 42:145–149, 1992.

7 Konstantin Pervyshev. On heuristic time hierarchies. In *IEEE Conference on Computational Complexity*, pages 347–358, 2007. doi:10.1109/CCC.2007.20.

8 E. R. van Dama and M. Muzychuk. Some implications on amorphic association schemes. *Journal of Combinatorial Theory, Series A*, 117:111–127, 2010.

9 Thomas Watson. Time hierarchies for sampling distributions. *SIAM J. Comput.*, 43(5):1709–1727, 2014. doi:10.1137/120898553.