Guest Column: Average-case Complexity Through the Lens of Interactive Puzzles[1]

R. Pass[2] ***M. Venkitasubramaniam***[3]

Abstract

We review a study of average-case complexity through the lens of *interactive puzzles*—interactive games between a computationally bounded Challenger and computationally-bounded Solver/Attacker. Most notably, we use this treatment to review a recent result showing that if NP is hard-on-the-average, then there exists a sampleable distribution over only *true* statements of an NP language, for which no probabilistic polynomial time algorithm can find witnesses. We also discuss connections to the problem of whether average-case hardness in NP implies average-case hardness in TFNP, or the existence of cryptographic one-way functions.

1 Introduction

The question whether $\mathsf{P} \neq \mathsf{NP}$ is arguably the most fundamental problem in computer science. But, even if $\mathsf{P} \neq \mathsf{NP}$, it could be that *in practice*, the NP instances that we encounter in "real life" come from from some distribution that make them easy to solve. Indeed, this motivated the complexity-theoretic study of average-case hardness of NP problems [47, 32, 6, 40].

It is worth repeating the following parable due to Impagliazzo from his 1995 essay [39] that framed the question with a human angle: Impagliazzo tells the story of Professor Grouse, young Gauss' teacher, who assigned Gauss' class the problem of summing up the numbers from 1 to 100. After Gauss solved this problem, Professor Grouse became obsessed with trying to humiliate Gauss by asking him questions he could not solve. While the story did not have a pleasant ending (with Grouse being admitted to a mental asylum), Impagliazzo uses the battle between Professor Grouse and young Gauss as a way to understand different possible worlds in average-case complexity. Consider, for instance, *Heuristica*—one of Impagliazzo's five hypothetical worlds—where NP is intractable in the worst-case, but tractable on the average for any sampleable distribution: In this

[2] Cornell Tech, NY, NY 10044, USA. rafael@cs.cornell.edu. Supported in part by NSF Award SATC-1704788, NSF Award RI-1703846, AFOSR Award FA9550-18-1-0267, a JP Morgan Faculty Award, and DARPA Award HR00110C0086.

[3] University of Rochester, Rochester, NY 14628, USA. muthuv@cs.rochester.edu. Supported by Google Faculty Research Grant, NSF Award CNS-1618884, Intelligence Advanced Research Projects Activity (IARPA) via 2019-19-020700009 and DIMACS Research Visit Program via DIMACS/Simons Collaboration in Cryptography.

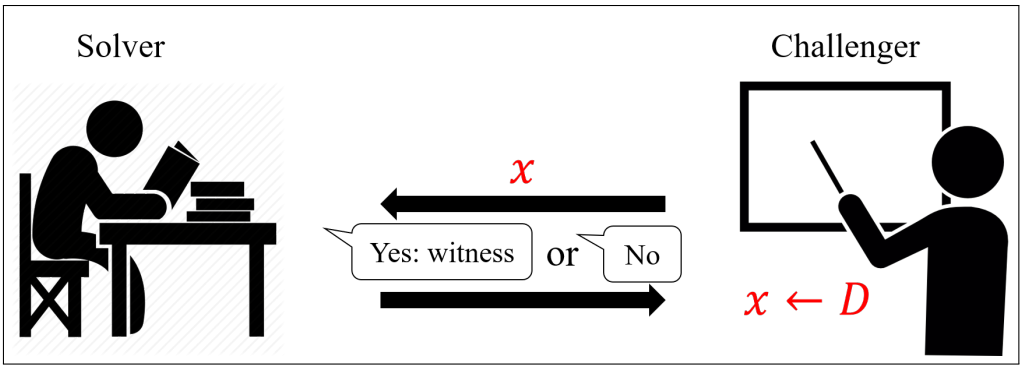

Figure 1: Challenger-Solver Game

world, while there exist instances on which (a computationally-bounded) Gauss will fail, Grouse does not have any efficient method of generating them.

Another appealing abstraction of an average-case analog of $\mathsf{P} \neq \mathsf{NP}$ was provided by Gurevich in his 1989 essay [31] through his notion of a *Challenger-Solver Game*. Gurevich outlines several classes of Challenger-Solver games; we here outline one particular instance of it, focusing on an NP search problem L. Consider a probabilistic polynomial-time *Challenger* C (the analog of Professor Grouse) who samples an instance x and provides it to the *Solver* S (the analog of Gauss). The solver S is supposed to find a witness to $x \in L$ and is said to win if either (1) the statement x chosen by the challenger is false (and therefore does not have a witness), or (2) S succeeds in finding a witness w for $x \in L$. We refer to the Challenger-Solver game as being *hard* if no probabilistic polynomial-time (PPT) solver succeeds in winning in the game with inverse polynomial probability. (In other words, such a game models a hard-on-average distributional search problem in NP.) In other words, the existence of a hard Challenger-Solver game means that there exists a way to efficiently sample mathematical statements x that no computationally bounded mathematician can find proofs for.

But How Do We Determine Who Won? An unappealing aspect of a Challenger-Solver game (which already goes back to the definition of distributional search problems [6]) is that checking whether the solver wins cannot necessarily be efficiently done, as it requires determining whether the sampled instance x is in the language.

This motivates the following fundamental question: Does the Challenger-Solver game become any easier if we restrict the challenger to *always* sample true statements x?[4] In other words, *"Is it easier to find proofs for efficiently-sampled mathematical statements that are guaranteed to be true?"* In complexity-theoretic terms:

> *Does the existence of an hard-on-average distributional search problem in* NP *imply the existence of a hard-on-average distributional search problem where the sampler* **only samples true statements***?*

We refer to distributional search problems where the sampler only samples true statements as *promise-true* distributional search problems. The above question, and the notion of a promise-true distributional search problems, actually predates the formal study of average-case complexity:

[4] Or equivalently, to distributions where one can efficiently check when the sampler outputs a false instance.

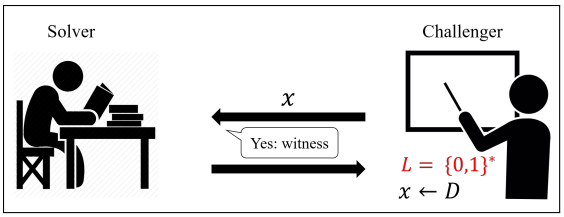

Figure 2: (a) Average-case hardness of TFNP

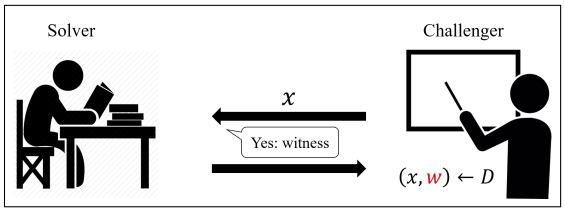

Figure 3: (b) Existence of OWFs

It was noted already by Even, Selman and Yacobi [17] in 1984 that for typical applications of (average-case) hardness for NP problems—in particular, for cryptographic applications—we need hardness for instances that are "promised" to be true. As they noted (following [18][5]), in the context of public-key encryption, security is only required for ciphertexts that are sampled as valid encryptions of some message. (This motivated [17] to introduce the concept of a promise problem; see also [25] for further discussion on this issue and the connection to average-case complexity.)

1.1 Connections to OWFs and TFNP

Intuitively, restricting to challengers that only sample true statements ought to make the job of the challenger a lot harder—it now needs to be sure that the sampled instance is true. There are two natural methods for the challenger to achieve this task:

(a) sampling the statement x *together with a witness* w (as this clearly enables the challenger to be sure that x is true); and,

(b) restricting to NP languages where *every* statement is true.

Connections to OWFs: As noted by Impagliazzo [31, 39], the existence of a Challenger-Solver game satisfying restriction (a) is equivalent to the existence of *one-way functions (OWFs)*—that is, a function f that can be computed in polynomial time but cannot be efficiently inverted. Such a function f directly yields the desired sampling method: pick a random string r and let $x = f(r)$ be the statement and r the witness. Conversely, to see why the existence of such a sampling method implies a one-way function, consider the function f that takes the random coins used by the sampling method and outputs the instance generated by it.

But whether the existence of a hard-on-average language in NP implies the existence of one-way functions is arguably the most important open problem in the foundations of Cryptography: One-way functions are both necessary [41] and sufficient for many of the central cryptographic tasks (e.g., pseudorandom generators [35], pseudorandom functions [26], private-key encryption [28, 3]). As far as we know, there are only two approaches towards demonstrating the existence of one-way functions from average-case NP hardness: (1) Ostrovsky and Wigderson [56] demonstrate such an implication assuming that NP has zero-knowledge proofs [27], (2) Komargodski et al. [46] demonstrate the implication (in fact, an even stronger implication, showing worst-case hardness of NP implies one-way functions) assuming the existence of *indistinguishability obfuscators* [4]. Both

[5] As remarked in [18], these type of "problems with a promise" can be traced back even further: they are closely related to what was referred to as a "birdy" problem in [22] and a "partial algorithm problem" in [69], in the study of context-free languages.

of these additional assumptions are not known to imply one-way functions on their own (in fact, they are unconditionally true if $\mathsf{NP} \subseteq \mathsf{BPP}$).

Connections to TFNP Hardness: A hard Challenger-Solver game satisfying restriction (b), on the other hand, is syntactically equivalent to a hard-on-average problem in the class TFNP [51]: the class TFNP (total function NP) is the search analog of NP with the additional guarantee that *any* instance has a solution. In other words, TFNP is the class of search problems in $\mathsf{NP} \cap \mathsf{coNP}$ (i.e., $F(\mathsf{NP} \cap \mathsf{coNP})$). In recent years, TFNP has attracted extensive attention due to its natural syntactic subclasses that capture the computational complexity of important search problems from algorithmic game theory, combinatorial optimization and computational topology—perhaps most notable among those are the classes PPAD [57, 23], which characterizes the hardness of computing Nash equilibrium [14, 11, 15], and PLS [43], which characterizes the hardness of local search. A central open problem is whether (average-case) NP hardness implies (average-case) TFNP hardness. A recent elegant result by Hubacek, Naor, and Yogev [38] shows that under certain "derandomization" assumptions [55, 42, 52, 5]—the existence of Nisan-Wigderson (NW) [55] type pseudorandom generators that fool circuits with oracle gates to languages in the second level of the polynomial hierarchy[6]—(almost everywhere) average-case hardness of NP implies average-case hardness of TFNP. [38] also show that average-case hardness of NP implies an average-case hard problem in $\mathsf{TFNP}/poly$ (i.e., TFNP with a *non-uniform verifier*). On a high level, this follows since non-uniformity enables unconditional derandomization; we provide more details on this in Section 8.2.

The above-mentioned works thus give complexity-theoretic assumptions (e.g., the existence of zero-knowledge proofs for NP, or strong derandomization assumptions) under which the above problem has a positive resolution.

1.2 New Results

In a recent paper [60], we provided a resolution to the above problem *without any complexity-theoretic assumption*:[7]

Theorem 1. *The existence of an almost-everywhere hard-on-average language in* NP[8] *implies the existence of a hard-on-average promise-true distributional search problem in* NP.

In fact, an even stronger statement was demonstrated: We showed that without loss of generality, the sampler/challenger of the distributional search problem needs to satisfy one of the above two "natural" restrictions:

Theorem 2. *The existence of an almost-everywhere hard-on-average language in* NP *implies either (a) the existence one-way functions, or (b) a hard-on-average* TFNP *problem.*

[6] Such PRGs are known under the assumption that $E = DTIME[2^{O(n)}]$ has no $2^{\epsilon n}$ sized Π_2-circuits, for all $\epsilon > 0$, where a Π_2-circuit is a standard circuit that can additionally perform oracle queries to any language $L \in \Pi_2$ (i.e., any language in the second level of the polynomial hierarchy).

[7] Pedantically, it is not a fully complete resolution as we start with an *almost-everywhere* hard problem and only get an *infinitely-often* hard problem. But, except for this minor issue, it is a complete resolution. We also note that earlier results [56, 38] also require starting off with an almost-everywhere hard-on-average language in NP.

[8] That is, a language in NP such that for every $\delta > 0$, no PPT attacker A can decide random instances with probability greater than $\frac{1}{2} + \delta$ for *infinitely many* (as opposed to all) $n \in N$. Such an "almost-everywhere" notion is more commonly used in the cryptographic literature.

In other words, in Impagliazzo's Pessiland [39] (a world where NP is hard-on-average, but one-way functions do not exist), TFNP is unconditionally hard (on average).

We will here provide a high level overview of the proof techniques involved in showing this results. Towards proving this result, we will pass through an alternative notion of a Challenger-Solver game, which we referred to as a *Interactive Puzzle*. Roughly speaking, interactive puzzles are Challenger-Solver games where the interactions between the Challenger (Professor Grouse) and the Solver (Gauss) can proceed in *many* rounds before determining who succeeded. In contrast to Challenger-Solver games, we will additionally impose the restriction that the transcript of the conversation efficiently determines whether the Solver won. We believe that such a notion of an interactive puzzle is interesting in its own, and provides a natural generalization of average-case hardness of NP. As we shall see, lifting the notion of a Challenger-Solver game to a more interactive setting will allows us to rely on techniques developed for interactive proof systems [29, 3]. Before introducing this notion, let us briefly recall some standard definitions from average-case complexity, to see how interactive puzzles generalize them.

2 Preliminaries on Average-case Complexity

We assume familiarity with basic concepts such as Turing machines, interactive Turing machine, polynomial-time algorithms, probabilistic polynomial-time algorithms (PPT), non-uniform polynomial-time and non-uniform PPT algorithms. A function $\mu(\cdot)$ is said to be *negligible* if for every polynomial $p(\cdot)$ there exists some n_0 such that for all $n > n_0$, $\mu(n) \leq \frac{1}{p(n)}$. An interactive protocol (P, V) is a pair of interactive Turing machine; we denote by $\langle P_1, P_2 \rangle(x)$ the output of P_2 in an interaction between P_1 and P_2 on common input x.

We refer to a relation $\mathcal{R}$ over pairs (x, y) as being *polynomially bounded* if there exists a polynomial $p(\cdot)$ such that for every $(x, y) \in \mathcal{R}$, $|y| \leq p(|x|)$. We denote by $L_{\mathcal{R}}$ the language characterized by the "witness relation" $\mathcal{R}$—i.e., $x \in L_{\mathcal{R}}$ iff there exists some y such that $(x, y) \in \mathcal{R}$. We say that a relation $\mathcal{R}$ is *polynomial-time* if $\mathcal{R}$ is polynomially-bounded and the language consisting of pairs $(x, y) \in \mathcal{R}$ is in P. A search problem $\mathcal{R}$ is simply a polynomially-bounded relation; an NP search problem $\mathcal{R}$ is a polynomial-time relation. An NP search problem $\mathcal{R}$ is *total* if for every $x \in \{0,1\}^*$ there exists some y such that $(x, y) \in \mathcal{R}$ (i.e., every instance has a witness). We refer to FNP (function NP) as the class of NP search problems and TFNP (total-function NP) as the class of total NP search problems.

2.1 One-way functions

We recall the definition of one-way functions (see e.g., [24]). Roughly speaking, a function f is one-way if it is polynomial-time computable, but hard to invert for PPT attackers. The standard (cryptographic) definition of a one-way function requires every PPT attacker to fail (with high probability) on all sufficiently large input lengths. We will also consider a weaker notion of an *infinitely-often* one-way function [56] which only requires the PPT attacker to fail for infinitely many inputs length (in other words, there is no PPT attacker that succeeds on all sufficiently large input lengths, analogously to complexity-theoretic notions of hardness).

Definition 3. *Let $f : \{0,1\}^* \to \{0,1\}^*$ be a polynomial-time computable function. f is said to be a* one-way function (OWF) *if for every* PPT *algorithm A, there exists a negligible function μ such*

that for all $n \in \mathbb{N}$*,*

$$\Pr[x \leftarrow \{0,1\}^n; y = f(x) : A(1^n, y) \in f^{-1}(f(x))] \leq \mu(n)$$

f is said to be an infinitely-often one-way function (ioOWF) *if the above condition holds for infinitely many $n \in \mathbb{N}$ (as opposed to all).*

2.2 Average-Case Complexity

We recall some basic notions from average-case complexity. A *distributional problem* is a pair $(L, \mathcal{D})$ where $L \subseteq \{0,1\}^*$ and $\mathcal{D}$ is a PPT; we say that $(L, \mathcal{D})$ is an NP distributional problem if $L \in \mathsf{NP}$. Roughly speaking, a distributional problem $(L, \mathcal{D})$ is hard-on-average if there does not exist some PPT algorithm that can decide instances drawn from $\mathcal{D}$ with probability significantly better than $1/2$.

Definition 4 (δ-hard-on-the-average)**.** *We say that a distributional problem $(L, \mathcal{D})$ is* δ-hard-on-the-average (δ-HOA) *if there does not exist some PPT A such that for every sufficiently large $n \in \mathbb{N}$,*

$$\Pr[x \leftarrow \mathcal{D}(1^n) : A(1^n, x) = L(x)] > 1 - \delta$$

We say that a distributional problem $(L, \mathcal{D})$ is simply hard-on-the-average (HOA) *if it is δ-HOA for some $\delta > 0$.*

The above notion of average-case hardness (traditionally used in the complexity-theory literature) is defined analogously to the notion of an *infinitely-often* one-way function: we simply require every PPT "decider" to fail for infinitely many $n \in \mathbb{N}$. For our purposes, we will also rely on an "almost-everywhere" notion of average-case hardness (similar to standard definitions in the cryptography, and analogously to the definition of a one-way function), where we require that every decider fails on *all* (sufficiently large) input lengths.

Definition 5 (almost-everywhere hard-on-the-average (aeHOA))**.** *We say that a distributional problem $(L, \mathcal{D})$ is* almost-everywhere δ hard-on-the-average (δ-aeHOA) *if there does not exist some PPT A such that for infinitely many $n \in \mathbb{N}$,*

$$\Pr[x \leftarrow \mathcal{D}(1^n) : A(1^n, x) = L(x)] > 1 - \delta$$

We say $(L, \mathcal{D})$ is almost-everywhere hard-on-the-average (aeHOA) *if $(L, \mathcal{D})$ is δ-aeHOA for some $\delta > 0$.*

We move on to defining hard-on-the-average *search problems*. A *distributional search problem* is a pair $(\mathcal{R}, \mathcal{D})$ where $\mathcal{R}$ is a search problem and $\mathcal{D}$ is a PPT. If $\mathcal{R}$ is an NP search problem we refer to $(\mathcal{R}, \mathcal{D})$ as a distributional NP search problem.

Finally, we say that a distributional search problem $(\mathcal{R}, \mathcal{D})$ is *promise-true* if for every n and every x in the support of $\mathcal{D}(1^n)$, it holds that $x \in L_{\mathcal{R}}$. (That is, $\mathcal{D}$ only samples true instances.)

Definition 6 (hard-on-the-average search (SearchHOA))**.** *We say that a distributional search problem $(\mathcal{R}, \mathcal{D})$ is* δ-hard-on-the-average (δ-SearchHOA) *if there does not exist some PPT A such that for every sufficiently large $n \in N$,*

$$\Pr[x \leftarrow \mathcal{D}(1^n); (w, x) \leftarrow A(1^n, x) : \\ ((L_{\mathcal{R}}(x) = 1) \Rightarrow (x, w) \in \mathcal{R})] > 1 - \delta$$

$(\mathcal{R}, \mathcal{D})$ is simply SearchHOA *if there exists $\delta > 0$ such that $(\mathcal{R}, \mathcal{D})$ is δ-SearchHOA.*

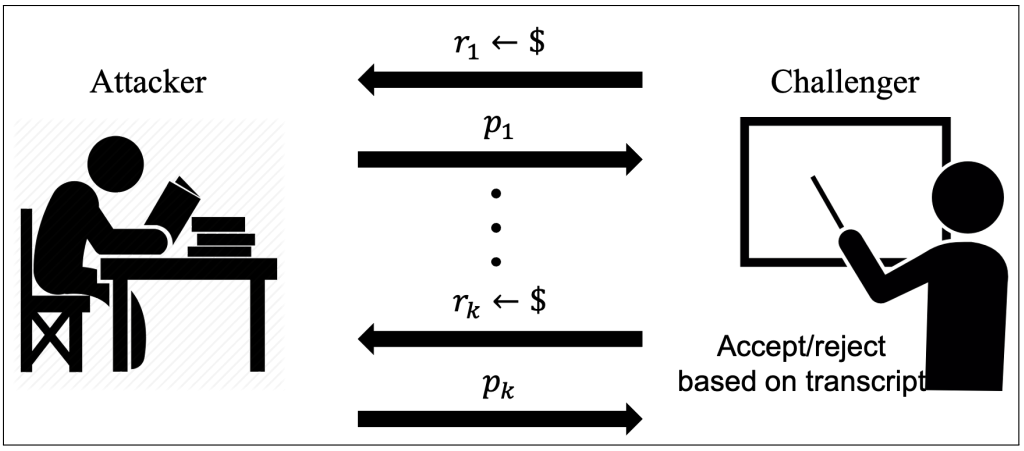

Figure 4: Interactive Puzzles

3 Interactive Puzzles

Let us introduce the notion of an *interactive puzzle* [60]: a 2-player interactive game between a polynomial-time Challenger $\mathcal{C}$ and a Solver/Attacker[9] satisfying the following properties:

- **Computational Soundness:** There does not exist a *probabilistic polynomial-time (PPT)* attacker $\mathcal{A}^*$ and polynomial p such that $\mathcal{A}^*(1^n)$ succeeds in making $\mathcal{C}(1^n)$ output 1 with probability $\frac{1}{p(n)}$ for all sufficiently large $n \in N$.
- **Completeness/Non-triviality:** There exists a negligible function μ and an *inefficient* attacker $\mathcal{A}$ that on input 1^n succeeds in making $\mathcal{C}(1^n)$ output 1 with probability $1 - \mu(n)$ for all $n \in N$.
- **Public Verifiability:** Whether $\mathcal{C}$ accepts should just be a deterministic function of the transcript.

In other words, (a) no *polynomial-time* attacker, $\mathcal{A}^*$, can make C output 1 with inverse polynomial probability, yet (b) there exists a *computationally unbounded* attacker $\mathcal{A}$ that makes C output 1 with overwhelming probability. We refer to $\mathcal{C}$ as a $k(\cdot)$-*round computational puzzle* (or simply a $k(\cdot)$-round puzzle) if $\mathcal{C}$ satisfies the above completeness and computational soundness conditions, while restricting $\mathcal{C}(1^n)$ to communicate with $\mathcal{A}$ in $k(n)$ rounds. More formally:

Definition 7 (interactive puzzle)**.** *An interactive algorithm $\mathcal{C}$ is referred to as a $k(\cdot)$-*round puzzle *if the following conditions hold:*

- **$k(\cdot)$-round, publicly-verifiability:** *$\mathcal{C}$ is an (interactive)* PPT *that on input 1^n (a) only communicates in $k(n)$ communication rounds, and (b) only performs some deterministic computation as a function of the transcript to determine its final verdict.*
- **Completeness/Non-triviality:** *There exists a (possibly unbounded) Turing machine $\mathcal{A}$ and a negligible function $\mu(\cdot)$ such that for all $n \in \mathbb{N}$,*

$$\Pr[\langle \mathcal{A}, \mathcal{C} \rangle(1^n) = 1] \geq 1 - \mu(n)$$

[9] Following the nomenclature in the cryptographic literature, we use the name Attacker instead of Solver.

- **Computational Soundness:** *There does not exist a* PPT *machine* $\mathcal{A}^*$ *and polynomial* $p(\cdot)$ *such that for all sufficiently large* $n \in \mathbb{N}$,

$$\Pr[\langle \mathcal{A}^*, \mathcal{C} \rangle(1^n) = 1] \geq \frac{1}{p(n)}$$

On Public-coins and Perfect-completeness: We mostly restrict our attention to *public-coin* puzzles, where the Challenger's messages are simply random strings—more formally, $\mathcal{C}$ simply sends the outcomes of its coin tosses. Additionally, we say that a puzzle $\mathcal{C}$ has *perfect completeness* if the "completeness error", $\mu(n)$, is 0—in other words, the completeness condition holds with probability 1.

As an example of a 2-round public-coin puzzle, let f be a one-way permutation and consider a game where $\mathcal{C}(1^n)$ samples a random $y \in \{0,1\}^n$ and requires the adversary to output a pre-image x such that $f(x) = y$. Since f is a permutation, this puzzle has "perfect" completeness—an unbounded attacker $\mathcal{A}$ can always find a pre-image x. By the one-wayness of f (and the permutation property of f), we also have that no PPT adversary $\mathcal{A}^*$ can find such an x (with inverse polynomial probability), and thus soundness holds. If however, f had only been a one-way function and not a permutation, then we may no longer be able to sample a uniform y, but rather have $\mathcal{C}$ first sample a random x and next output $y = f(x)$. This 2-round puzzle does not satisfy the public-coin property, but it still has perfect completeness.

On 2-round Public-coin Puzzles and Average-case hardness of NP: It is not hard to see that the existence of 2-round (public-coin) puzzles is "essentially" equivalent to the existence of an average-case hard problem in NP: any 2-round public-coin puzzle trivially implies a hard-on-average search problem (w.r.t. the uniform distribution) in NP and thus by [40] also a hard-on-average decision problem in NP. Furthermore, "almost-everywhere" hard-on-average languages in NP w.r.t. the uniform distribution (which by [40] is implied by the existence of a hard-on-average language in NP w.r.t. any sampleable distribution) also imply the existence of a 2-round puzzle (by simply sampling many random instances x and asking the attacker to provide a witness for at least, say, 1/3 of the instances).[10]

Proposition 3.1. *The existence of an (almost-everywhere) hard-on-average language in* NP *implies the existence of a 2-round public-coin puzzle. Furthermore, the existence of a 2-round public-coin puzzle implies the existence of a hard-on-average language in* NP*.*

Thus, 2-round public-coin puzzles are "morally" (up to the infinitely-often/almost-everywhere issue) equivalent to the existence of a hard-on-average language in NP. Since 2-round puzzles capture average-case hardness of NP, $k(\cdot)$-round public-coin puzzles thus provide a natural generalization thereof.

On Weaker Soundness and Completeness: One can consider a more relaxed notion of a public-coin $(c(\cdot), s(\cdot))$-puzzles for $c(n) > s(n) + \frac{1}{\mathsf{poly}(n)}$, where the completeness condition is required to hold with probability $c(\cdot)$ for sufficiently large $n \in \mathbb{N}$, and the soundness condition holds with probability $s(\cdot)$ for sufficiently large $n \in \mathbb{N}$. But, by "Chernoff-type" parallel-repetition theorems for computationally-sound public-coin protocols [59, 36, 12, 13], the existence of such a $k(\cdot)$-round public-coin $(c(\cdot), s(\cdot))$-puzzle implies the existence of a $k(\cdot)$-round public-coin puzzle.

[10] The reason we need the language to be *almost-everywhere* hard-on-average is to guarantee that YES instances exists for every sufficiently large input length, or else completeness would not hold.

Capturing TFNP and Promise-True Distributional Search Problems: Towards the goal of linking puzzles and the questions raised in the introduction, we remark that natural syntactic restrictions of 2-round puzzles capture natural subclasses of distributional problems in NP:

- the existence of a hard-on-average problem in TFNP is syntactically equivalent to the existence of a 2-round *public-coin* puzzle *with perfect completeness*.
- the existence of a hard-on-average *promise-true* distributional search problem is syntactically equivalent to a 2-round (private-coin) puzzle *with perfect completeness*.

The Complexity of Puzzles: While the game-based modeling in the notion of a puzzle is common in the cryptographic literature—most notably, it is commonly used to model cryptographic assumptions [53, 58, 21], complexity-theoretic consequences or properties of puzzles have remained largely unexplored. We will here initially review such a treatment. Additionally, we will show that such an interactive treatment of average-case complexity leads to a new tool set also for answering "classic" questions regarding average-case hardness in NP.

The two main problems that we will consider are (1) **round-complexity**—to what extent does adding more round yields more power, and (2) **perfect completeness**—are interactive puzzles with perfect completeness easier to solve? Our main theorems (mentioned in the introduction) will next follow as corollaries from answering these questions about round-complexity and perfect completeness.

4 The Round-Complexity of Puzzles

Perhaps the most basic question regarding the existence of interactive puzzles is whether the existence of a k-round puzzle is actually a weaker assumption than the existence of a $k-1$ round puzzle. In particular, do interactive puzzles actually generalize beyond just average-case hardness in NP:

Does the existence of a k-round puzzle imply the existence of $(k-1)$-round puzzle?

We here focus our attention only on public-coin puzzles. At first sight, one would hope the classic "round-reduction" theorem due to Babai-Moran (BM) [3] can be applied to collapse any $O(1)$-round puzzle into a 2-round puzzle (i.e., a hard-on-average NP problem). Unfortunately, while BM's round reduction technique indeed works for all *information-theoretically* sound protocols, Wee [70] demonstrated that BM's round reduction fails for computationally sound protocols. In particular, Wee shows that black-box proofs of security cannot be used to prove that BM's transformation preserves soundness even when applied to just 3-round protocols, and demonstrates (under computational assumptions) a concrete 4-round protocol for which BM's round-reduction results in an unsound protocol.

In contrast to this negative result, the central technical result in [60] provides an affirmative answer to the above question—we demonstrates a round-reduction theorem for puzzles.

Theorem 8. *For every constant c, the existence of a $k(\cdot)$-round public-coin puzzle is equivalent to the existence of a $(k(\cdot)-c)$-round public-coin puzzle.*

In particular, as corollary of this result, we get that the assumption that a $O(1)$-round public-coin puzzle exists is *not* weaker than the assumption that average-case hardness in NP exists:

Corollary 9. *The existence of a $O(1)$-round public-coin puzzle implies the existence of a hard-on-average problem in* NP.

Perhaps paradoxically, we strongly rely on BM's round reduction technique, yet we rely on a *non-black-box* security analysis. The main technical lemma shows that if *infinitely-often one-way functions* do not exist (i.e., if we can invert any function for all sufficiently large input lengths), then BM's round reduction actually works:

Lemma 4.1. *Either infinitely-often one-way functions exist, or BM's round-reduction transformation turns a $k(\cdot)$-round public-coin puzzle into a $(k(\cdot)-1)$-round public-coin puzzle.*

We provide a proof outline of Lemma 4.1 in Section 4.1. The proof of Theorem 8 now easily follows by considering two cases:

Case 1: (Infinitely-often) one-way functions exists. In such a world, we can rely on Rompel's construction of a universal one-way hash function (UOWHFs) [54, 62] from one-way functions to get a 2-round puzzle. More precisely, a OWHF is a family of functions such that no PPT attacker can, given a uniformly sampled function h in the family, and a uniform input $x \in \{0,1\}^n$, find a "second-preimage" x' (a.k.a a collision) such that $h(x') = h(x)$. Given such a collection of hash functions, the puzzle challenger simply selects random h, x and the solver wins if it finds a second-preimage x'.

Case 2: (Infinitely-often) one-way functions does not exist. In such a world, by Lemma 4.1, BM's round reduction preserves soundness of the underlying protocol and thus we have gotten a puzzle with one round less. We can next iterate BM's round reduction any constant number of times.

We highlight that the above proof is highly non black-box in nature. In case OWFs do not exists we do rely on the original puzzle (but require applying a OWF inverter on the code of the original puzzle), and in case OWF do exist, we simply observe that 2-round puzzles exist.

4.1 Proof Overview of Lemma 4.1

We here provide a proof overview of the round-collapse theorem. As mentioned, we shall show that if one-way functions do not exist, then Babai-Moran's round reduction method actually works.

Some Technical Tools: Towards this we will rely on two tools:

- *Pre-image sampling.* By the result of Impagliazzo and Levin [40], the existence of so-called "distributional one-way functions" (function for which it is hard to sample a uniform pre-image) imply the existence of one-way function. So if one-way functions do not exist, we have that for every efficient function f, given a sample $f(x)$ for a random input x, we can efficiently sample a (close to random) pre-image x'.

- *Raz's sampling lemma* (from the literature on parallel repetition for 2-prover games and interactive arguments [61, 36, 13]). This lemma states that if we sample ℓ uniform n-bit random variables $R_1, R_2, \ldots R_\ell$ conditioned on some event W that happens with sufficiently large probability ϵ, then the conditional distribution R_i of a randomly selected index i will be

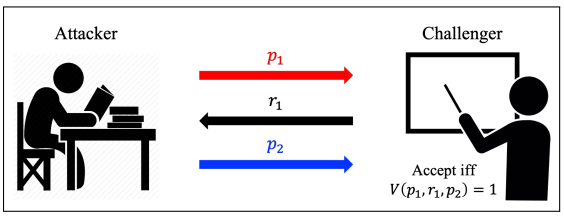

Figure 5: (a) Three-round protocol

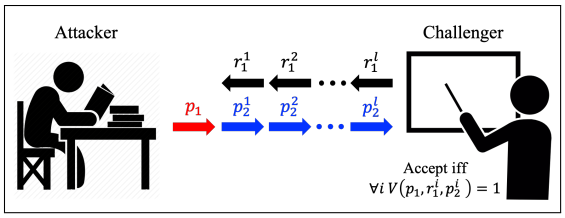

Figure 6: (b) Babai-Moran transformation

close to uniform. More precisely, the statistical distance will be $\sqrt{\frac{\log(\frac{1}{\epsilon})}{\ell}}$, so even if ϵ is tiny, as long as we have sufficiently many repetitions ℓ, the distance will be small.[11]

The BM Round-collapse Transformation: To see how we will use these tools, let us first recall the BM transformation (and its proof for the case of information-theoretically sound protocols). To simplify our discussion, we here focus on showing how to collapse a 3-round public-coin protocol between a prover P and a public-coin verifier V into a 2-round protocol. We denote a transcript of the 3-round protocol (p_1, r_1, p_2) where p_1 and p_2 are the prover messages and r_1 is the randomness of the verifier; see Figure 5. Let $n = |p_1|$ be the length of the prover message. The BM transformation collapses this protocol into a 2-round protocol in the following two steps:

- **Step 1: Reducing soundness error:** First, use a form of parallel repetition to make the soundness error 2^{-n^2} (i.e., *extremely small*). More precisely, consider a 3-round protocol where P first still send just p_1, next the verifier picks $\ell = n^2$ random strings $\vec{r} = (r_1^1, \ldots, r_1^\ell)$, and finally P needs to provide accepting answers $\vec{p_2} = (p_2^1, \ldots, p_2^\ell)$ to all of the queries $\vec{r}$ (so that for every $i \in [\ell]$, (p_1, r_1^i, p_2^i) is accepting transcript).

- **Step 2: Swap order of messages:** Once the soundness error is small, yet the length of the first message is short, we can simply allow the prover to pick its first message p_1 after having $\vec{r}$. In other words, we now have a 2-round protocol where V first picks $\vec{r}$, then the prover responds by sending $p_1, \vec{p_2}$; see Figure 6.

  This swapping preserves soundness by a simple union bound: since (by soundness) for every string p_1, the probability over $\vec{r}$ that there exists some accepting response $\vec{r}$ is 2^{-n^2}, it follows that with probability at most $2^n \times 2^{-n^2} = 2^{-n}$ over $\vec{r}$, *there exists some* p_1 that has an accepting $\vec{p_2}$ (as the number of possible first messages p_1 is 2^n). Thus soundness still holds (with a 2^n degradation) if we allow P to choose p_1 after seeing $\vec{r}$.

Analyzing Soundness: For the case of computationally sound protocols, the "logic" behind both steps fail: (1) it is not known how to use parallel repetition to reduce soundness error beyond being negligible, (2) the union bound cannot be applied since, for computationally sound protocols, it is not the case that responses $\vec{p_2}$ do not exist, rather, they are just hard to find. Yet, as we shall see, using the above tools, we present a different proof strategy. More precisely, to capture computational hardness, we show a reduction from any polynomial-time attacker A that breaks soundness of the collapsed protocol with some inverse polynomial probability ϵ, to a polynomial-time attacker B that breaks soundness of the original 3-round protocol.

[11] Earlier works [36, 13] always used Raz's lemma when ϵ was non-negligible. In contrast, we will here use it also when ϵ is actually negligible.

B starts by sampling a random string $\vec{r'}$ and computes A's response given this challenge $(p'_1, \vec{p'_2}) \leftarrow A(\vec{r'})$. If the response is not an accepting transcript, simply abort; otherwise, take p'_1 and forward externally as B's first message. (Since A is successful in breaking soundness, we have that B won't abort with probability ϵ.) Next, B gets a verifier challenge r from the external verifier and needs to figure out how to provide an answer to it. If B is lucky and r is one of the challenges r'_i in $\vec{r'}$, then B could provide the appropriate p_2 message, but this unfortunately will only happen with negligible probability. Rather, B will try to get A to produce another accepting transcript $(p''_1, \vec{r''}, \vec{p''_2})$ that (1) still contains p'_1 as the prover's first message (i.e., $p''_1 = p'_1$), and (2) contains r in some coordinate i of $\vec{r''}$. To do this, B will consider the function $f(\vec{r}, z, i)$—which runs $(p_1, \vec{p_2}) \leftarrow A(\vec{r}; z)$ (i.e., A has its randomness fixed to z) and outputs (p_1, r_i) if $(p_1, \vec{r}, \vec{p_2})$ is accepting and $\perp$ otherwise—and runs the pre-image sampler for this function f on (p'_1, r) to recover some new verifier challenge, randomness, index tuple $(\vec{r''}, z, i)$ which leads $A(\vec{r''}; z)$ to produce a transcript $(p'_1, \vec{r''}, \vec{p''_2})$ of the desired form, and B can subsequently forward externally the i'th coordinate of $\vec{p''_2}$ as its response and convince the external verifier.

So, as long as the pre-image sampler indeed succeeds with high enough probability, we have managed to break soundness of the original 3-round protocol. The problem is that the pre-image sampler is only required to work given outputs that are correctly distributed over the range of the function f, and the input (p_1, r) that we now feed it may not be so—for instance, perhaps $A(\vec{r})$ chooses the string p_1 as a function of $\vec{r}$. So, whereas the marginal distribution of both p_1 and r are correct, the *joint* distribution is not. In particular, the distribution of r conditioned on p_1 may be off. We, however, show how to use Raz's lemma to argue that if the number of repetitions ℓ is sufficiently bigger than the length of p_1, the conditional distribution of r cannot be too far off from being uniform (and thus the pre-image sampler will work). On a high-level, we proceed as follows:

- Note that in the one-way function experiment, we can think of the output distribution (p_1, r) of f on a random input, as having been produced by first sampling p_1 and next, if $p_1 \neq \perp$, sampling $\vec{r}$ conditioned on the event W_{p_1} that A generates a successful transcript with first-round prover message p_1, and finally sampling a random index i and outputting p_1 and r_i (and otherwise output $\perp$).

- Note that by an averaging argument, we have that with probability at least $\frac{\epsilon}{2}$ over the choice of p_1, $\Pr[W_{p_1}] \geq \frac{\epsilon}{2^{n+1}}$ (otherwise, the probability that A succeeds would need to be smaller than $\frac{\epsilon}{2} + 2^n \times \frac{\epsilon}{2^{n+1}} = \epsilon$, which is a contradiction).

- Thus, whenever we pick such a "good" p_1 (i.e., a p_1 such that $\Pr[W_{p_1}] \geq \frac{\epsilon}{2^{n+1}}$), by Raz's lemma the distribution of r_i for a random i can be made $\frac{1}{p(n)}$ close to uniform for any polynomial p by choosing ℓ to be sufficiently large (yet polynomial). Note that even though the lower bound on $\Pr[W_{p_1}]$ is negligible, the key point is that it is independent of ℓ and as such we can still rely on Raz's lemma by choosing a sufficiently large ℓ. (As we pointed out above, this usage of Raz's lemma even on very "rare" events—with negligible probability mass—is different from how it was previously applied to argue soundness for computationally sound protocols [36, 13].)

- It follows that conditioned on picking such a "good" p_1, the pre-image sampler will also successfully generate correctly distributed preimages if we feed him p_1, r where r is randomly sampled. But this is exactly the distribution that B feeds to the pre-image sampler, so we

conclude that with probability $\frac{\epsilon}{2}$ over the choice of p_1, B will manage to convince the outside verifier with probability close to 1.

This concludes the proof overview for 3-round protocols. When the protocol has more than 3 rounds, we can apply a similar method to collapse the last rounds of the protocol. The analysis now needs to be appropriately modified to condition also on the prefix of the partial execution up until the last rounds.

5 The Complexity of Puzzles with Many Rounds

A natural question is whether we can collapse more than a constant number of rounds. Our next result—which characterizes the existence of $\mathsf{poly}(n)$-round puzzles—shows that this is unlikely. (This result is not pertinent to the above-mentioned main problem, but is interesting in its own right to understand the power of puzzles.)

Theorem 10. *For every $\epsilon > 0$, there exists an n^ϵ-round (public-coin) puzzle if and only if* $\mathsf{PSPACE} \not\subseteq \mathsf{BPP}$.

In particular, if n^ϵ-round public-coin puzzles imply $O(1)$-round public-coin puzzles, then by combining Theorem 8 and Theorem 10, we have that $\mathsf{PSPACE} \not\subseteq \mathsf{BPP}$ implies the existence of a hard-on-average problem in NP, which seems unlikely. Theorem 10 also shows that the notion of an interactive puzzle (with a super constant-number of rounds) indeed is a non-trivial generalization of average-case hardness in NP. Theorem 10 follows using mostly standard techniques, which we now outline:

Solving Puzzles using a PSPACE Oracle: Any puzzle $\mathcal{C}$ can be broken using a PSPACE oracle (as the optimal strategy can be found using a PSPACE oracle), so if $\mathsf{PSPACE} \subseteq \mathsf{BPP}$, it can also be broken by a probabilistic polynomial-time algorithm.

A Public-coin Puzzle assuming $\mathsf{PSPACE} \not\subseteq \mathsf{BPP}$: For the other direction, recall that worst-case to average-case reductions are known for PSPACE [19, 2]. In other words, there exists a language $L \in \mathsf{PSPACE}$ that is hard-on-average assuming $\mathsf{PSPACE} \not\subseteq \mathsf{BPP}$. Additionally, recall that PSPACE is closed under complement. We then construct a public-coin puzzle where $\mathcal{C}$ first samples a hard instance for L and then asks $\mathcal{A}$ to determine whether $x \in L$ and next provide an interactive proof—using [65, 50] which is public-coin—for containment or non containment in L. This puzzle clearly satisfies the completeness condition. Computational soundness, on the other hand, follows directly from the hard-on-average property of L (and the unconditional soundness of the interactive proof of [65]). Let us remark that the worst-case to average-case reduction known for PSPACE only yield a a "weakly" hard-on-the-average problem with $\delta = \frac{1}{poly}$ and thus the resulting soundness error of the puzzle will only be $1 - \frac{1}{poly}$; however, as we remarked in Section 3, a public-coin puzzle satisfiying this weaker form of soundness implies the standard form of a public-coin puzzle by relying on parallel repetition.

6 From Imperfect to Perfect Completeness

We remark that a standard technique from the literature on interactive proofs (namely the result of [20]) can be used to show that any 2-round public-coin puzzle can be transformed into a 3-

round public-coin puzzle with *perfect completeness*. In more detail, Furer et al. [20] showed how to transform any 2-round public-coin proof system into a 3-round public-coin proof system with perfect completeness. We will rely on the same protocol transformation to transform any 2-round puzzle into a 3-round puzzle with perfect completeness. The perfect completeness condition will follow directly from their proof; we simply must argue that the transformation also preserves *computational* soundness (as they only showed that it preserves information-theoretic soundness).

Theorem 11. *Suppose there exists* 2*-round public-coin puzzle. Then there exists a* 3*-round public-coin puzzle with perfect completeness.*

Let us outline the transformation of Furer et al. [20] and explain why it works. Let $\mathcal{C}$ be a 2-round public-coin puzzle; see Figure 7. Let ℓ_c, ℓ_a be polynomials such that the message from $\mathcal{C}(1^n)$ is of length $\ell_c(n)$ and the message from $\mathcal{A}(1^n)$ is of length $\ell_a(n)$; we assume without loss of generality that $\ell_c(n) > 2$. When the security parameter n is clear from the context we will omit it and let $\ell_c(n) = \ell_c$ and $\ell_a(n) = \ell_a$.

We now apply the Furer et al. [20] transformation to this puzzle to create a 3-round puzzle $\widetilde{\mathcal{C}}$. The puzzle will proceed by first having the adversary sending ℓ_c "pads" $z_1, \ldots, z_{\ell_c} \in \{0,1\}^{\ell_c}$ to $\widetilde{\mathcal{C}}$; $\widetilde{\mathcal{C}}$ next responds with a random message $r_{\widetilde{\mathcal{C}}} \in \{0,1\}^{\ell_c}$, and the adversary is next supposed to find a response i, p such that $(r_{\widetilde{\mathcal{C}}} \oplus z_i, p)$ is a valid transcript for the original puzzle (i.e., the adversary needs to win in one of the parallel "padded" instances of the original puzzle); see Figure 8. More formally, $\widetilde{\mathcal{C}}(1^n, (z_1, \ldots, z_{\ell_c}), (i, p); r_{\widetilde{\mathcal{C}}}) = 1$ if and only if $\mathcal{C}(1^n, p; r_{\widetilde{\mathcal{C}}} \oplus z_i)$ outputs 1. To show perfect completeness of $\widetilde{\mathcal{C}}$, we proceed just as in the elegant original proof by [20], which we recall: From the (imperfect) completeness of $\mathcal{C}$, we have that there exists some adversary $\mathcal{A}$ such that $\Pr[\langle \mathcal{A}, \mathcal{C} \rangle(1^n) = 1] \geq 1 - \frac{1}{n}$ for all sufficiently large n; without loss of generality A is deterministic. Fix some $n > 2$ for which this holds. Let $S \subseteq \{0,1\}^{\ell_c}$ be the set of challenges for which $\mathcal{A}$ provides an accepting response; the probability that a random challenge $z \in \{0,1\}^{\ell_c}$ is inside S is thus at least $1 - \frac{1}{n}$. We will show that there exists "pads" $z_1, \ldots, z_{\ell_c}$ such that for every $r \in \{0,1\}^{\ell_c}$, there exists some i such that $r \oplus z_i \in S$, which concludes that an unbounded attacker $\widetilde{\mathcal{A}}$ can succeed with probability 1 (by selecting those pads and next providing an accepting response). Note that for every fixed r, for a *randomly* chosen pad z_i, the probability that $r \oplus z_i \notin S$ is at most $\frac{1}{n}$; and thus the probability over randomly chosen pads $z_1, \ldots, z_{\ell_c}$ that $r \oplus z_i \notin S$ *for all* i is at most $\frac{1}{n^{\ell_c}}$. We conclude, by a union bound, that the probability over randomly chosen pads $z_1, \ldots, z_{\ell_c}$ that *there exists* some $r \in \{0,1\}^{\ell_c}$ such that $r \oplus z_i \notin S$ for all i is at most $\frac{2^{\ell_c}}{n^{\ell_c}} < 1$. Thus, there exists pads $z_1, \ldots, z_{\ell_c}$ such that *for every* $r \in \{0,1\}^{\ell_c}$ there exists some i such that $r \oplus z_i \notin S$, which concludes perfect completeness.

We now turn to proving computational soundness. Consider some adversary $\widetilde{\mathcal{A}}^*$ that succeeds in convincing $\widetilde{\mathcal{C}}$ with probability $\epsilon(n)$ for all $n \in \mathbb{N}$. We construct an adversary $\mathcal{A}^*$ that convinces $\mathcal{C}$ with probability $\frac{\epsilon(n)}{\ell_c}$, which is a contradiction. $\mathcal{A}^*(1^n)$ picks a random tape $r_{\widetilde{\mathcal{A}}^*}$ for $\widetilde{\mathcal{A}}^*$, lets $(z_1, \ldots, z_{\ell_c}) = \widetilde{\mathcal{A}}^*(1^n; r_{\widetilde{\mathcal{A}}^*})$, picks a random index $i \in [\ell_c]$ and outputs z_i. Upon receiving a "challenge" r, it lets $(j, p) = \widetilde{\mathcal{A}}^*(1^n, r \oplus z_i; r_{\widetilde{\mathcal{A}}^*})$ outputs p if $i = j$ and $\perp$ otherwise. First, note that in the emulation by $\mathcal{A}^*$, $\mathcal{A}^*$ feeds $\widetilde{\mathcal{A}}^*$ the same distribution of messages as $\widetilde{\mathcal{A}}^*$ would see in a "real" interaction with $\widetilde{\mathcal{C}}$; thus, we have that the (j, p) is an accepting message (w.r.t., the challenge $r \oplus z_i$) with probability ϵ. Additionally, since $r \oplus z_i$ information-theoretically hides i (as r is completely random), we have that the probability that $i = j$ is $\frac{1}{\ell_c}$ and furthermore, the event that this happens is independent of whether the message (j, p) is accepting. We conclude that $\mathcal{A}^*$ convinces $\mathcal{C}$ with

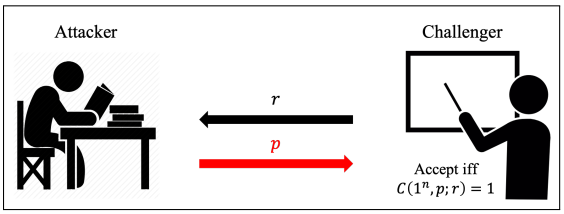

Figure 7: (a) Two-round public coin puzzle

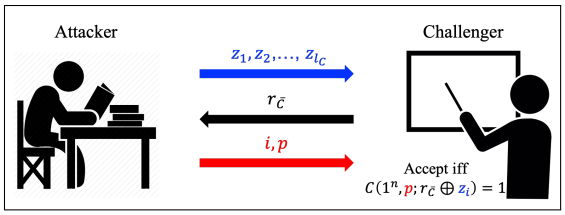

Figure 8: (b) Furer et al. transformation

probability $\frac{\epsilon(n)}{\ell_c}$, which concludes the soundness proof.

7 Avg-case Hardness of NP implies OWFs or TFNP-Hardness

We now outline how to combine the round-reduction with the perfect-completeness transformation to prove Theorem 2 (i.e., to show that average-case hardness of NP implies either OWFs or average-case hardness of TFNP):

- As noted in Proposition 3.1, an almost-everywhere hard-on-average problem in NP yields a 2-round puzzle.
- We next apply the perfect-completeness transformation (using Theorem 11) to turn this puzzle into a 3-round puzzle with *perfect completeness*.
- We next observe that the BM transformation preserves perfect completeness of the protocol. Thus, by Lemma 4.1, either infinitely-often one-way functions exist, or we can get a 2-round public-coin puzzle with perfect completeness.
- Finally, as observed above, the existence of a 2-round public-coin puzzle with perfect completeness is syntactically equivalent to the existence of a hard-on-average problem in TFNP (with respect to the uniform distribution on instances).

The above proof approach actually only concludes a slightly weaker form of Theorem 2—we only show that either TFNP is hard or *infinitely-often* one-way functions exist. As infinitely-often one-way functions directly imply 2-round *private-coin* puzzles with perfect completeness, which (as observed above) are syntactically equivalent to hard-on-average *promise-true* distributional search problems, this however already suffices to prove Theorem 1 (that is, average-case hardness of NP implies the existence of an average-case hard promise-true distributional search problem).

We can get the proof also of the stronger conclusion of Theorem 2 (i.e., conclude the existence of standard (i.e., "almost-everywhere") one-way functions), by noting that an almost-everywhere hard-on-average language in NP actually implies an 2-round puzzle satisfying a "almost-everywhere" notion of soundness, and for such "almost-everywhere puzzles", Lemma 4.1 can be strengthened to show that either one-way functions exist, or BM's round-reduction works.[12]

[12] More precisely, the variant of Lemma 4.1 says that either one-way functions exist, or the existence of a k-round almost-everywhere puzzle yields the existence of a $k-1$-round puzzle (with the standard, infinitely-often, notion of soundness).

8 Towards Stronger Implications from Avg-case Hardness of NP

Our results (i.e., Theorem 2) show that average-case of NP implies *either* (1) the existence of OWFs, *or* (2) average-case hardness of TFNP. Ideally, we would like to show that average-case hardness of NP implies OWF *and* average-case hardness of TFNP. Or at the very least, unconditionally show one of the implications (1) average-case hardness of NP implies OWF, or (2) average-case hardness of NP implies average-case hardness of TFNP. As mentioned in the introduction, both of these are long-standing open problems. In this final section, we recall some recent progress on these problems.

8.1 Towards OWFs from Avg-case Hardness of NP

As mentioned in the introduction, the question of whether the existence of a hard-on-average language in NP implies the existence of one-way functions is arguably the most important open problem in the foundations of Cryptography. This question dates back to the seminal work of Diffie and Hellman [16] from 1976, but so far most results in the literature have been negative.

Barriers to Basing OWF on NP-Hardness: Notably, starting with the work by Brassard [9] in 1983, a long sequence of works have shown various types of black-box separations between *restricted* types of OWF (e.g., one-way permutations) and NP-hardness (see e.g., [9, 8, 1, 30, 49, 33, 7]). We emphasize, however, that these results only show limited separations: they either consider restricted types of one-way functions, or restricted classes of black-box reductions.

By Theorem 2, to prove the existence of OWFs from average-case hardness of NP, it suffices to prove that average-case hardness of TFNP (rather than NP) implies the existence of OWFs. Thus, it suffices to construct a OWF starting from average-case hardness of a structured class of problems (namely problems in TFNP). We highlight that this difference is quite substantial. Known black-box separations typically are of the form: "If OWFs (with some additional structure, e.g., being a permutation) can be based on the hardness of a language L, then $L \in \mathsf{AM} \cap \mathsf{coAM}$, which is unlikely for any NP-complete language (in particular it implies the collapse of the Polynomial Hierarchy). However, if $L \in \mathsf{TFNP}$, then L is trivially in $\mathsf{AM} \cap \mathsf{coAM}$, so no unexpected collapse happens!

On OWFs and Time-bounded Kolmogorov-complexity: While the question of whether OWFs can be based on the average-case hardness of NP is still wide open, a recent result by Liu and Pass [48] takes us a steps towards it: it demonstrates the first *natural* NP problem L such that average-case hardness of L is *equivalent* to the existence of OWFs. (This problem, however, is not known to be average-case complete for NP.) The problem, which dates back to the 1960s, is the so-called the *time-bounded Kolmogorov complexity* problem.

Let us briefly recall the notion of (time-bounded) Kolmogorov complexity and their result: What makes the string 12121212121212121 less random than 604848506683403574 92? The notion of *Kolmogorov complexity* (K-complexity), introduced by Solomonoff [67], Kolmogorov [45] and Chaitin [10], provides an elegant method for measuring the amount of "randomness" in individual strings: The K-complexity of a string is the length of the shortest program (to be run on some fixed universal Turing machine U) that outputs the string x. From a computational point of view, however, this notion is unappealing as there is no efficiency requirement on the program. The notion of *$t(\cdot)$-time-bounded Kolmogorov Complexity (K^t-complexity)* overcomes this issue: $K^t(x)$ is defined as the length of the shortest program that outputs the string x within time $t(|x|)$. As surveyed by Trakhtenbrot [68], the problem of efficiently determining the K^t-complexity for $t(n) = \mathsf{poly}(n)$

predates the theory of NP-completeness and was studied in the Soviet Union since the 60s as a candidate for a problem that requires "brute-force search" (see Task 5 on page 392 in [68]). The modern complexity-theoretic study of this problem goes back to Sipser [66], Ko [44] and Hartmanis [34]. Trakhtenbrot also notes that a "frequential" version of this problem was considered in the Soviet Union in the 60s: the problem of finding an algorithm that succeeds for a "high" fraction of strings x—in more modern terms from the theory of average-case complexity [47], whether K^t can be computed by a heuristic algorithm with inverse polynomial error, over random inputs x. We say that K^t is *mildly hard-on-average (mildly HoA)* if there exists some polynomial $p(\cdot) > 0$ such that every PPT fails in computing $K^t(\cdot)$ for at least a $\frac{1}{p(\cdot)}$ fraction of n-bit strings x for all sufficiently large n, and that K^{poly} is mildly HoA if there exists some polynomial $t(n) > 0$ such that K^t is mildly HoA. Liu and Pass [48] show that mild average-case hardness of K^{poly} is equivalent to the existence of OWFs.

Theorem 12 ([48])**.** *The following are equivalent:*

- *One-way functions exist.*
- K^{poly} *is mildly hard-on-average.*

8.2 Towards TFNP-Hardness from Avg-case Hardness of NP

We finally recall some recent results towards showing TFNP-hardness from average-case hardness of NP.

TFNP/poly-Hardness from Average-case Hardness of NP: As mentioned in the introduction, Hubacek, Naor, and Yogev [38] show that under certain "derandomization" assumptions [55, 42, 52, 5]—the existence of Nisan-Wigderson (NW) [55] type pseudorandom generators that fool circuits with oracle gates to languages in the second level of the polynomial hierarchy, (almost everywhere) average-case hardness of NP implies average-case hardness of TFNP. [38] also show unconditionally that average-case hardness of NP w.r.t. non-uniform PPT attackers implies an average-case hard problem in TFNP/poly (i.e,. TFNP with a *non-uniform verifier*) .

Theorem 13 ([38])**.** *The existence of an almost-everywhere hard-on-average language in* NP *(where hardness holds also w.r.t., non-uniform PPT attackers) implies the existence of a hard-on-average search problem in* TFNP/poly*.*

Let us briefly outline how this can be proved using our language of puzzles. As we have seen (see Proposition 3.1), the existence of an almost-everywhere hard-on-average language in NP implies the existence of a 2-round public-coin puzzle, which by the transformation by Fürer et al. [20] (See Theorem 11) implies a 3-round public coin puzzle with perfect completeness; furthemore, if starting with a problem in NP that is average-case hard w.r.t. non-uniform PPT attackers, the resulting 3-round puzzle will preserve hardness also w.r.t. non-uniform PPT attackers. Given a first message z, a second message x and a third message w, let $V_z(x, w)$ denote the challenger's acceptance predicate. Let us now argue that this acceptance predicate V_z yields a hard-on-the-average search problem in TFNP/poly, when appropriately picking z as the non-uniform advice.[13]

[13]Let us note that whereas [38] does not explicitly rely on the construction from [20], the final TFNP/poly problem they consider becomes exactly the same.

- **Totality:** Note that by perfect completeness of the puzzles, there exist some (first-message) string z such that for every (second-message) string x, there exists a response w that makes the Challenger accept; that is, there exists some z such that for every x, there exists a witness w such that $V_z(x, w) = 1$. Thus, for such a "good" z, V_z is a total relation.
- **Hard-on-average:** Additionally, for every sequence $\{z_n\}_{n\in N}$, we claim that $\{V_{z_n}\}_{n\in N}$ is a hard-on-average search problem. This follows since given any PPT attacker A that succeeds in breaking the search problem, we can get a *non-uniform* PPT attacker A' (which has $\{z_n\}_{n\in N}$ as non-uniform advice) that succeeds in breaking the 3-round puzzle with the same probability (by simply sending its non-uniform advice as the first message and next using A).

Barriers Towards Showing TFNP-Hardness: By Theorem 2, to establish average-case TFNP-Hardness from just average-case hardness of NP, it suffices to show that OWFs imply TFNP-hardness of NP. This, intuitively, should make the task quite a bit easier. So far, however, only negative results have been established:

- Rosen et al. [63] show a black-box separation of TFNP with a small number of witnesses—*bounded* TFNP—from OWFs. In fact, they prove a stronger separation, separating bounded-TFNP from injective trapdoor functions. They accomplish this by generalizing the technique of Rudich [64] to construct an oracle relative to which injective trapdoor-functions exist, yet bounded-TFNP are easy to solve. Their result highlights that bounded-TFNP behaves quite differently from TFNP; indeed, in our construction of a hard-on-average TFNP program, the problem has a large number of witnesses.
- A very recent result by Hubacek et al. [37] addresses the question of whether TFNP (potentially with a large number of witnesses) can be constructed from OWFs in a black-box way. They present some limitations on black-box constructions of a worst-case hard TFNP problem from injective one-way functions. However, their results only applies to quite restrictive forms of black-box constructions/reductions. In particular, they only rule out reductions that are *non-adaptive* and *oblivious* of the underlying one-way function; we refer the reader to [37] for further details on these restrictions.

Summarizing, the restrictions in known impossibility results are quite severe. It is an intriguing open problem to either extend these impossibility results to apply for general black-box constructions, or to simply overcome the barrier and present an average-case hard TFNP problem from OWFs (which combined with Theorem 2 gives average-case hardness of TFNP from just average-case hardness of NP). We are optimistic and believe that a construction of an average-case hard TFNP problem from OWFs will be found.

References

[1] Adi Akavia, Oded Goldreich, Shafi Goldwasser, and Dana Moshkovitz. On basing one-way functions on NP-hardness. In *STOC*, pages 701–710, 2006.

[2] László Babai, Lance Fortnow, Noam Nisan, and Avi Wigderson. BPP has subexponential time simulations unless EXPTIME has publishable proofs. *Computational Complexity*, 3:307–318, 1993.

[3] László Babai and Shlomo Moran. Arthur-Merlin games: A randomized proof system, and a hierarchy of complexity classes. *J. Comput. Syst. Sci.*, 36(2):254–276, 1988.

[4] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. In *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*, pages 1–18, 2001.

[5] Boaz Barak, Shien Jin Ong, and Salil P. Vadhan. Derandomization in cryptography. *SIAM J. Comput.*, 37(2):380–400, 2007.

[6] 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.

[7] Andrej Bogdanov and Christina Brzuska. On basing size-verifiable one-way functions on NP-hardness. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part I*, volume 9014 of *Lecture Notes in Computer Science*, pages 1–6. Springer, 2015.

[8] Andrej Bogdanov and Luca Trevisan. On worst-case to average-case reductions for NP problems. In *FOCS*, pages 308–317, 2003.

[9] Gilles Brassard. Relativized cryptography. *IEEE Transactions on Information Theory*, 29(6):877–893, 1983.

[10] Gregory J. Chaitin. On the simplicity and speed of programs for computing infinite sets of natural numbers. *J. ACM*, 16(3):407–422, 1969.

[11] Xi Chen, Xiaotie Deng, and Shang-Hua Teng. Settling the complexity of computing two-player Nash equilibria. *J. ACM*, 56(3):14:1–14:57, 2009.

[12] Kai-Min Chung and Feng-Hao Liu. Parallel repetition theorems for interactive arguments. In *Theory of Cryptography, 7th Theory of Cryptography Conference, TCC 2010, Zurich, Switzerland, February 9-11, 2010. Proceedings*, pages 19–36, 2010.

[13] Kai-Min Chung and Rafael Pass. Tight parallel repetition theorems for public-coin arguments using KL-divergence. In *Theory of Cryptography - 12th Theory of Cryptography Conference, TCC 2015, Warsaw, Poland, March 23-25, 2015, Proceedings, Part II*, pages 229–246, 2015.

[14] Constantinos Daskalakis, Paul W. Goldberg, and Christos H. Papadimitriou. The complexity of computing a Nash equilibrium. *Commun. ACM*, 52(2):89–97, 2009.

[15] Constantinos Daskalakis and Christos H. Papadimitriou. Continuous local search. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011, San Francisco, California, USA, January 23-25, 2011*, pages 790–804, 2011.

[16] Whitfield Diffie and Martin Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[17] Shimon Even, Alan L. Selman, and Yacov Yacobi. The complexity of promise problems with applications to public-key cryptography. *Information and Control*, 61(2):159–173, 1984.

[18] Shimon Even and Yacov Yacobi. Cryptocomplexity and NP-completeness. In *Automata, Languages and Programming, 7th Colloquium, Noordweijkerhout, The Netherlands, July 14-18, 1980, Proceedings*, pages 195–207, 1980.

[19] Joan Feigenbaum and Lance Fortnow. Random-self-reducibility of complete sets. *SIAM Journal on Computing*, 22(5):994–1005, 1993.

[20] Martin Fürer, Oded Goldreich, Yishay Mansour, Michael Sipser, and Stathis Zachos. On completeness and soundness in interactive proof systems. *Advances in Computing Research*, 5:429–442, 1989.

[21] Craig Gentry and Daniel Wichs. Separating succinct non-interactive arguments from all falsifiable assumptions. In *STOC*, pages 99–108, 2011.

[22] Seymour Ginsburg. *The Mathematical Theory of Context-Free Languages*. McGraw-Hill, Inc., USA, 1966.

[23] Paul W. Goldberg and Christos H. Papadimitriou. Towards a unified complexity theory of total functions. Unpublished manuscript, 2016.

[24] Oded Goldreich. *Foundations of Cryptography — Basic Tools*. Cambridge University Press, 2001.

[25] Oded Goldreich. On promise problems: A survey. In *Theoretical Computer Science, Essays in Memory of Shimon Even*, pages 254–290, 2006.

[26] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. On the cryptographic applications of random functions. In *CRYPTO*, pages 276–288, 1984.

[27] Oded Goldreich, Silvio Micali, and Avi Wigderson. Proofs that yield nothing but their validity for all languages in NP have zero-knowledge proof systems. *J. ACM*, 38(3):691–729, 1991.

[28] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984.

[29] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1):186–208, 1989.

[30] S. Dov Gordon, Hoeteck Wee, David Xiao, and Arkady Yerukhimovich. On the round complexity of zero-knowledge proofs based on one-way permutations. In *LATINCRYPT*, pages 189–204, 2010.

[31] Yuri Gurevich. The challenger-solver game: variations on the theme of P=NP. In *Logic in Computer Science Column, The Bulletin of EATCS*. 1989.

[32] Yuri Gurevich. Average case completeness. *J. Comput. Syst. Sci.*, 42(3):346–398, 1991.

[33] Iftach Haitner, Mohammad Mahmoody, and David Xiao. A new sampling protocol and applications to basing cryptographic primitives on the hardness of NP. In *IEEE Conference on Computational Complexity*, pages 76–87, 2010.

[34] J. Hartmanis. Generalized kolmogorov complexity and the structure of feasible computations. In *24th Annual Symposium on Foundations of Computer Science (sfcs 1983)*, pages 439–445, Nov 1983.

[35] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999.

[36] Johan Håstad, Rafael Pass, Douglas Wikström, and Krzysztof Pietrzak. An efficient parallel repetition theorem. In *Theory of Cryptography, 7th Theory of Cryptography Conference, TCC 2010, Zurich, Switzerland, February 9-11, 2010. Proceedings*, pages 1–18, 2010.

[37] Pavel Hubácek, Chethan Kamath, Karel Král, and Veronika Slívová. On average-case hardness in TFNP from one-way functions. In Rafael Pass and Krzysztof Pietrzak, editors, *Theory of Cryptography - 18th International Conference, TCC 2020, Durham, NC, USA, November 16-19, 2020, Proceedings, Part III*, volume 12552 of *Lecture Notes in Computer Science*, pages 614–638. Springer, 2020.

[38] Pavel Hub'avcek, Moni Naor, and Eylon Yogev. The journey from NP to TFNP hardness. In *8th Innovations in Theoretical Computer Science Conference, ITCS 2017, January 9-11, 2017, Berkeley, CA, USA*, pages 60:1–60:21, 2017.

[39] Russell Impagliazzo. A personal view of average-case complexity. In *Structure in Complexity Theory '95*, pages 134–147, 1995.

[40] Russell Impagliazzo and Leonid A. Levin. No better ways to generate hard NP instances than picking uniformly at random. In *31st Annual Symposium on Foundations of Computer Science, St. Louis, Missouri, USA, October 22-24, 1990, Volume II*, pages 812–821, 1990.

[41] Russell Impagliazzo and Michael Luby. One-way functions are essential for complexity based cryptography (extended abstract). In *30th Annual Symposium on Foundations of Computer Science, Research Triangle Park, North Carolina, USA, 30 October - 1 November 1989*, pages 230–235, 1989.

[42] Russell Impagliazzo and Avi Wigderson. P = BPP if E requires exponential circuits: Derandomizing the xor lemma. In *STOC*, pages 220–229, 1997.

[43] David S. Johnson, Christos H. Papadimitriou, and Mihalis Yannakakis. How easy is local search? (extended abstract). In *26th Annual Symposium on Foundations of Computer Science, Portland, Oregon, USA, 21-23 October 1985*, pages 39–42, 1985.

[44] Ker-I Ko. On the notion of infinite pseudorandom sequences. *Theor. Comput. Sci.*, 48(3):9–33, 1986.

[45] A. N. Kolmogorov. Three approaches to the quantitative definition of information. *International Journal of Computer Mathematics*, 2(1-4):157–168, 1968.

[46] Ilan Komargodski, Tal Moran, Moni Naor, Rafael Pass, Alon Rosen, and Eylon Yogev. One-way functions and (im)perfect obfuscation. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pages 374–383. IEEE Computer Society, 2014.

[47] Leonid A. Levin. Average case complete problems. *SIAM J. Comput.*, 15(1):285–286, 1986.

[48] Yanyi Liu and Rafael Pass. On one-way functions and Kolmogorov complexity. In *IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, 2020.

[49] Noam Livne. On the construction of one-way functions from average case hardness. In Andrew Chi-Chih Yao, editor, *Innovations in Computer Science - ICS 2010, Tsinghua University, Beijing, China, January 5-7, 2010. Proceedings*, pages 301–309. Tsinghua University Press, 2010.

[50] Carsten Lund, Lance Fortnow, Howard J. Karloff, and Noam Nisan. Algebraic methods for interactive proof systems. *J. ACM*, 39(4):859–868, 1992.

[51] Nimrod Megiddo and Christos H. Papadimitriou. On total functions, existence theorems and computational complexity. *Theor. Comput. Sci.*, 81(2):317–324, 1991.

[52] Peter Bro Miltersen and N. V. Vinodchandran. Derandomizing Arthur-Merlin games using hitting sets. *Computational Complexity*, 14(3):256–279, 2005.

[53] Moni Naor. On cryptographic assumptions and challenges. In *Advances in Cryptology - CRYPTO 2003, 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 2003, Proceedings*, pages 96–109, 2003.

[54] Moni Naor and Moti Yung. Universal one-way hash functions and their cryptographic applications. In *STOC*, pages 33–43, 1989.

[55] Noam Nisan and Avi Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994.

[56] Rafail Ostrovsky and Avi Wigderson. One-way fuctions are essential for non-trivial zero-knowledge. In *ISTCS*, pages 3–17, 1993.

[57] Christos H. Papadimitriou. On the complexity of the parity argument and other inefficient proofs of existence. *J. Comput. Syst. Sci.*, 48(3):498–532, 1994.

[58] Rafael Pass. Limits of provable security from standard assumptions. In *STOC*, pages 109–118, 2011.

[59] Rafael Pass and Muthuramakrishnan Venkitasubramaniam. A parallel repetition theorem for constant-round Arthur-Merlin proofs. *TOCT*, 4(4):10:1–10:22, 2012.

[60] Rafael Pass and Muthuramakrishnan Venkitasubramaniam. A round-collapse theorem for computationally-sound protocols; or, TFNP is hard (on average) in pessiland. In *IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, 2020.

[61] Ran Raz. A parallel repetition theorem. *SIAM Journal on Computing*, 27(3):763–803, 1998.

[62] John Rompel. One-way functions are necessary and sufficient for secure signatures. In *STOC*, pages 387–394, 1990.

[63] Alon Rosen, Gil Segev, and Ido Shahaf. Can PPAD hardness be based on standard cryptographic assumptions? In Yael Kalai and Leonid Reyzin, editors, *Theory of Cryptography - 15th International Conference, TCC 2017, Baltimore, MD, USA, November 12-15, 2017, Proceedings, Part II*, volume 10678 of *Lecture Notes in Computer Science*, pages 747–776. Springer, 2017.

[64] Steven Rudich. *Limits on the Provable Consequences of One-way Functions*. PhD thesis, EECS Department, University of California, Berkeley, 1988.

[65] Adi Shamir. IP = PSPACE. *J. ACM*, 39(4):869–877, 1992.

[66] Michael Sipser. A complexity theoretic approach to randomness. In *Proceedings of the 15th Annual ACM Symposium on Theory of Computing, 25-27 April, 1983, Boston, Massachusetts, USA*, pages 330–335. ACM, 1983.

[67] R.J. Solomonoff. A formal theory of inductive inference. part i. *Information and Control*, 7(1):1 – 22, 1964.

[68] Boris A Trakhtenbrot. A survey of russian approaches to perebor (brute-force searches) algorithms. *Annals of the History of Computing*, 6(4):384–400, 1984.

[69] Joseph S. Ullian. Partial algorithm problems for context free languages. *Information and Control*, 11(1/2):80–101, 1967.

[70] Hoeteck Wee. Finding Pessiland. In *Theory of Cryptography, Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006, Proceedings*, pages 429–442, 2006.