Communication Complexity vs Randomness Complexity in Interactive Proofs

Benny Applebaum
Tel-Aviv University, Israel

Kaartik Bhushan
IIT Bombay, India

Manoj Prabhakaran
IIT Bombay, India

Abstract

In this work, we study the interplay between the communication from a verifier in a general private-coin interactive protocol and the number of random bits it uses in the protocol. Under worst-case derandomization assumptions, we show that it is possible to transform any I-round interactive protocol that uses ρ random bits into another one for the same problem with the additional property that the verifier's communication is bounded by $O(I \cdot \rho)$. Importantly, this is done with a minor, logarithmic, increase in the communication from the prover to the verifier and while preserving the randomness complexity. Along the way, we introduce a new compression game between computationally-bounded compressor and computationally-unbounded decompressor and a new notion of conditioned efficient distributions that may be of independent interest. Our solutions are based on a combination of perfect hashing and pseudorandom generators.

2012 ACM Subject Classification Theory of computation → Interactive proof systems

Keywords and phrases Interactive Proof Systems, Communication Complexity, Hash Functions, Pseudo-Random Generators, Compression

Digital Object Identifier 10.4230/LIPIcs.ITC.2024.2

Funding The first and second authors are supported by ISF grant no. 2805/21 and by the European Union (ERC, NFITSC, 101097959). Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them. The second and third authors are also supported by IIT Bombay Trust Lab. The third author is also supported by an Algorand Centre of Excellence grant by the Algorand Foundation.
Benny Applebaum: Supported by ISF grant no. 2805/21 and by the European Union (ERC-2022-ADG) under grant agreement no.101097959 NFITSC.
Kaartik Bhushan: Supported in part by IITB Trust Lab. Part of the research was done while visiting Tel Aviv Univerisity and was supported by ISF grant no. 2805/21 and by the European Union (ERC-2022-ADG) under grant agreement no.101097959 NFITSC. Also supported by the Prime Minister's Research Fellowship (PMRF), Government of India.
Manoj Prabhakaran: Supported in part by IITB Trust Lab and by an Algorand Centre of Excellence at IIT Bombay funded by the Algorand Foundation.

Acknowledgements We thank Gil Segev for valuable discussions about perfect hashing. Part of the research was done while the second author visited Tel Aviv University.

1 Introduction

Interactive probabilistic proof systems [15, 4] are central objects in cryptography and complexity theory. Roughly speaking, they allow a computationally unbounded prover to convince a computationally-bounded randomized verifier that a certain statement holds. The use of *interaction* and *randomness* extends the classical notion of NP problems (that

can be deterministically verified given a single message from a prover) all the way up to polynomial-space computable languages [23, 33]. If the verifier does not use randomness, clearly interaction is useless: the prover can compute all the verifier's queries on her own and just send the answers. That is, when the verifier uses 0 bits of randomness, it needs to communicate 0 bits. In this paper, we try to extend this observation: if the verifier uses only ρ random bits, how many bits does it need to communicate?

Intuitively, since the entropy of the verifier's messages is at most ρ, there is no point in communicating more than ρ bits. Indeed, if the proof system is a public-coin system (in which the verifier simply sends random coins in each round) this intuition holds, and the randomness complexity equals the communication complexity. However, in the general case of private-coin proof system, the verifier's messages may be much longer than their entropy. We note that although it is possible to transform a given private-coin proof system into a public-coin system [16, 13, 22], existing transformations *increase* the overall communication. In particular, a transcript of the modified public-coin proof system contains at least one copy of a transcript of the original private-coin proof system plus some additional overhead which is polynomial in the original communication (to certify that the transcript is "typical" or "heavy"). Furthermore, these transformations also increase the randomness complexity of the verifier either polynomially [16, 13] or by a constant factor [22]. Overall, the following question remains open:

> Is it possible to transform a proof system with randomness ρ in which the prover sends out $\mathsf{Com_P}$ bits, into a new proof system for the same problem in which the total communication depends on the verifier's randomness complexity in the original proof system rather than the verifier's outgoing communication, i.e., the total communication is $\mathsf{poly}(\rho, \mathsf{Com_P})$ or even $O(\rho + \mathsf{Com_P})$? Further, can we do this while preserving the randomness complexity?

Closely related questions were studied in [2] and [22]. Specifically, [2] studied the "converse question" of upper-bounding the randomness complexity in terms of the communication complexity (aka "randomness sparsification"), and [22] studied the question of bounding the *round complexity* in terms of the randomness complexity. The latter work shows [22, Thm A.1] that a proof system that uses $\rho(n)$ random bits for n-long instances can be converted into a proof system with $O(\rho(n)/\log n)$ rounds, while preserving the randomness complexity.[1] Unfortunately, this transformation increases the total communication complexity (as the prover guesses in every round many possible extensions to the current transcript). In particular, even for constant-round protocol this transformation may increase the communication by a $\mathsf{poly}(n)$ factor.

1.1 Our Results

We partially resolve the above question by relying on a complexity theoretic assumption. In particular, we prove the following main result. (See Section 2 for a formal presentation of the hardness assumption).

▶ **Theorem 1.** *Suppose that a promise problem* Π *has an interactive proof system* $\langle \mathsf{P}, \mathsf{V} \rangle$ *with round complexity* $I(n)$*, randomness complexity* $\rho(n)$*, verifier communication* $\mathsf{Com_V}(n)$*, prover communication* $\mathsf{Com_P}(n)$*, where* n *denotes the length of the instance. Then, assuming that*

[1] In fact, if one is willing to increase the randomness complexity by a constant factor, then it is possible to derive an $O(\rho(n)/\log n)$-round *public-coin* system [22, Thm 1.1].

$\mathsf{E} = \mathsf{DTime}(2^{O(n)})$ is hard for exponential-size non-deterministic NP-circuits, there exists an interactive proof system $\langle \mathsf{P}', \mathsf{V}' \rangle$ for Π in which the verifier and prover communication are

$$\mathsf{Com}_{\mathsf{V}'}(n) = O(I(n)\rho(n)), \text{ and}$$
$$\mathsf{Com}_{\mathsf{P}'}(n) = \mathsf{Com}_{\mathsf{P}}(n) + O(I(n)\log n).$$

The randomness complexity of the proof system remains unchanged, the round complexity grows by (at most) 1, and the completeness error grows additively by 0.1.

For the special case of constant-round protocols, the verifier communicates $O(\rho(n))$ bits and the prover's additive overhead is at most $O(\log n)$; if the original prover communicates in each round a logarithmic number of bits (resp., super-logarithmic number of bits), the additive overhead is linear (resp., sub-linear) in $\mathsf{Com}_{\mathsf{P}}(n)$. In the general case, when the prover may communicate as little as one bit per round, we still have $I(n) \leq \mathsf{Com}_{\mathsf{P}}(n)$; then the overhead for the prover communication is a multiplicative factor of $O(\log n)$, and the total communication is $O(\mathsf{Com}_{\mathsf{P}}(n) \cdot (\rho(n) + \log n))$. Note that in all cases, the total communication is independent of $\mathsf{Com}_{\mathsf{V}}(n)$, the verifier's original communication. The question of achieving a total communication of $O(\rho(n) + \mathsf{Com}_{\mathsf{P}}(n))$ for protocols with polynomially-many rounds remains an interesting open question.

Theorem 1 is based on a worst-case assumption. This assumption asserts that one cannot significantly speed-up (uniform) Exponential-Time problems by adding non-uniformity and two levels of non-determinism (non-deterministic NP-circuits are the non-uniform analogue of $\mathsf{NP}^{\mathsf{NP}}$; see Section 2 for details). This assumption is somewhat strong but widely believed as it reflects our current understanding of the relations between time, nonuniformity and nondeterminism. Similar assumptions have been extensively used in cryptography and complexity theory (see, e.g.,[9, 20, 24, 37, 30, 14, 17, 31, 6, 32, 8, 1, 2, 5, 29]).

1.2 Technical Overview

Consider the simple case of a single-round protocol where the verifier's message is of length m bits that is much larger than the randomness complexity ρ. In this case the verifier is sending a long message that is sampled from a low-entropy distribution $\mathcal{D}$, and our goal is to reduce the communication. The crucial observation is that the prover has full knowledge of the distribution $\mathcal{D}$, and, being computationally unbounded, he can help the verifier to *compress* the message. Indeed, we can abstract this scenario as a special variant of the well-known *data compression* problem.

1.2.1 Single-Round Compression Game

In this data compression game there are two parties: a computationally bounded compressor CMP and a computationally-unbounded decompressor DCMP. At the beginning of the game, the compressor is given a string $x \in \{0,1\}^m$ that is efficiently sampled from a probability distribution $\mathcal{D}$ whose full description is given to the decompressor DCMP. The goal of the compressor is to deliver the string x to the decompressor with probability at least $1 - \epsilon$ taken over the choice of x. The parties are allowed to communicate in both directions, and the goal is to minimize the communication ideally up to the entropy of the distribution.

It is instructive to compare our game to a few other compression games. Shannon's original game [34] refers to compression of *multiple independent* samples from $\mathcal{D}$, and his celebrated source-coding theorem shows that the expected amortized communication

approaches the entropy. In contrast, our game involves a single-shot challenge and worst-case communication and, accordingly, allows some error (i.e., the scheme may be *lossy*). In computer science literature, Ta-Shama et al. [38] studied the problem of compressing "computationally-weak" sources by a computationally efficient compressor and decompressor and provided compression schemes whose communication complexity is close to the entropy for several classes of such distributions. In contrast, in our setting the decompressor is allowed to be computationally-unbounded, and as a result we can bypass some of the lower-bounds of [38] (e.g., we can hope to compress pseudorandom distributions). Finally, Orlitsky [26] considered a one-shot compression game in which the parties are computationally-unbounded but have some information gap captured by some auxiliary information y about x that is given to the decompressor and is unknown to the compressor. Notably, Orlitsky's schemes use interaction (like in our setting) whereas the schemes of [34, 38] are non-interactive (the compressor sends a single message to the decompressor).

Getting back to our compression game, let us further simplify the problem and assume that we care only about the communication from the compressor to the decompressor. In this case, there is a simple solution that is described in Lemma 14. Take $k = H(\mathcal{D})/\epsilon$ where $H(\cdot)$ denotes Shannon's entropy, and let DCMP send a description of hash function $f : \{0,1\}^m \to \{0,1\}^k$ that is 1-1 over the set of 2^k heaviest strings in $\mathcal{D}$. The compressor CMP responds with the "digest" $y = f(x)$, and DCMP outputs the "heaviest" string x' in $\mathcal{D}$ that is consistent with y. It is not hard to show that the error is at most ϵ (see Lemma 12) which is essentially the best that one can hope for.[2] Indeed, this approach can be viewed as one-shot analog of Shannon's celebrated Source coding theorem [34]. Unfortunately, the decompressor has to communicate the description of a hash function f which is taken from a family F of 2^k-*perfect hash function*. That is, the family F contains, for each 2^k-subset of strings $X \subset \{0,1\}^m$, a function f that is injective on X and so it cannot be too small. In fact, it is known that the description size must be at least $\Omega(2^k)$ bits [25]. The cost can be significantly reduced by allowing some slackness, i.e., by expanding the output length of f to $k' > k$ (e.g., $k' = 3k$). In this case, the description length can be reduced to $\Omega(m + k')$ bits by using existing families of perfect hash functions (e.g., [10]). However, this is still too expensive for our purposes.[3]

1.2.2 Focusing on efficiently samplable distribution

We note that when the distribution $\mathcal{D}$ is taken from a family of efficiently samplable distributions (i.e., there is an efficient algorithm that given a random tape outputs a sample from $\mathcal{D}$) it is possible to compress the description length of the hash function. Indeed, in this case our family should be injective only over "nice" sets that correspond to heavy strings in distributions that can be described by a polynomial-size circuit. Specifically, we begin with a standard, off-the-shelf, family $F = \{f_z\}_{z \in \{0,1\}^{m+k'}}$ (e.g., based on pair-wise independent hash functions [7]) in which each function is identified by a long string $z \in \{0,1\}^{m+k'}$. Next, we reduce the description length of the functions via the use of an appropriate pseudorandom

[2] Indeed, for every $\epsilon > 0$, there exists a distribution $\mathcal{D}$ such that any event of probability $1 - \epsilon$ must be supported over at least 2^k strings for $k = \Omega(H(\mathcal{D})/\epsilon)$. For example, consider the distribution $\mathcal{D}$ obtained by sampling, with probability 2ϵ, a uniform x from a set A of size $2^{\ell/2\epsilon}$, and, with probability $1 - 2\epsilon$, a uniform x from a disjoint set B of size 2^ℓ. The entropy of $\mathcal{D}$ is $\Theta(\ell)$ and in order to capture $1 - \epsilon$ of the mass, one must collect at least ϵ fraction of the strings in A, i.e., 2^k strings for $k \geq \ell/2\epsilon - \log(1/\epsilon) = \Omega(H(\mathcal{D})/\epsilon)$.

[3] To the best of our knowledge, the description length of all existing constructions of 2^k-perfect hash functions is either linear in 2^k or in the input-length m, see e.g., [27].

generator (PRG) $G : \{0,1\}^{\ell} \to \{0,1\}^{m+k'}$. That is, each function f'_s in the new family F' is indexed by a seed s of the PRG and is defined to be $f'_s = f_{G(s)}$. We show that if the PRG fools AM/poly adversaries the family F' can be used to compress $\mathcal{D}$. By using standard derandomization assumptions (slightly weaker than the one stated in Theorem 1), we get such a PRG with exponential stretch which allows us to reduce the communication from the decompressor to logarithmic. It should be mentioned that the idea of using a PRG to (partially) derandomize a probabilistic construction is not new. This paradigm was abstracted by [20], and was also used in many relevant works. Interestingly, the same paradigm was used in [1] for the contrary purpose of constructing so-called *incompressible functions*.

1.2.3 Back to interactive proofs

Let us move back to the case of multi-round interactive proofs. A natural strategy is to apply the above approach for each round. Roughly, in each round, we let w denote the partial transcript and let $\mathcal{D}_w$ denote the distribution of the next message of the verifier. Instead of letting the verifier send his message x, the parties will run a compression protocol in which the prover selects a hash function f that is injective on the 2^k heaviest strings, where k is about $H(\mathcal{D}_w)/\epsilon$ for some error parameter ϵ. The problem is that $\mathcal{D}_w$ is not efficiently samplable, rather it is obtained by feeding the verifier with the prover messages and random coins that are *conditioned* on generating the partial transcript w. We abstract this property via the notion of *conditioned efficient distributions*. This notion generalizes the notion of efficiently samplable distributions by allowing the sampler A to output a special failure symbol $\perp$, and by letting $\mathcal{D}$ denote the outcome of A applied to random coins conditioned on not outputting $\perp$. By using slightly stronger PRGs, we extend our compression schemes to the case of conditioned efficient distributions, and employ them to reduce the interaction of interactive proofs as stated in Theorem 1.

▶ Remark 2 (More on conditioned efficient distributions). An equivalent way to define a conditioned efficient distribution is by considering a pair of algorithms, Sampler S and Conditioner E, such that sampling from $\mathcal{D}$ boils down to sampling a random tape r conditioned on $E(r) = 1$ and outputting $S(r)$. Thus this new notion can be viewed as a combination of two well-studied classes of distributions: distributions over circuit's outputs (i.e., efficiently samplable distributions) and distributions over circuit's inputs that lead to a given result (aka *efficiently recognizable distributions* [28]). This new notion is natural and may prove to be useful elsewhere.

1.2.4 Organization

Following some preliminaries in Section 2, we construct compression schemes in Section 3 and use them to prove our main theorem in Section 4.

2 Preliminaries

2.1 Probability distributions

For a discrete probability distribution $\mathcal{D}$, let $\Pr_{\mathcal{D}}(x)$ denote the probability of the string x being sampled according to the distribution. Throughout the paper, we will only work with probability distributions that are supported on a finite set. The *Shannon entropy* (or simply entropy) $H(\mathcal{D})$ of a discrete probability distribution $\mathcal{D}$ supported on a finite set D is defined as the quantity

$$H(\mathcal{D}) = \sum_{x \in D} \Pr_{\mathcal{D}}(x) \log\left(\frac{1}{\Pr_{\mathcal{D}}(x)}\right).$$

▶ **Definition 3** (Efficient and conditioned efficient probability distributions). *A family of probability distributions* $\{\mathcal{D}_w\}$ *is* efficiently samplable *(or simply,* efficient*) if there exist parameters* ρ_w, m_w *denoted as the* randomness complexity *and the* domain bit-length*, and a PPT* sampling *algorithm* A *that given an index* $w \in \{0,1\}^*$ *and a random tape with* $r \in \{0,1\}^{\rho_w}$ *samples a random string* $x \in \{0,1\}^{m_w}$ *according to the distribution* $\mathcal{D}_w$*. The* complexity *of* $\{\mathcal{D}_w\}$ *is at most* T *if* $A(w;r)$ *runs in time* $T(|w|)$ *for every* w *and* r*.*

A family of distributions $\{\mathcal{D}_w\}$ *is said to be* conditioned efficiently samplable *(or simply,* conditioned efficient*) if, with parameters* ρ_w, m_w *as above, there exists a PPT algorithm* A *which, on input* $w \in \{0,1\}^*$ *and a uniformly random string* $r \in \{0,1\}^{\rho_w}$ *on its random tape, outputs an element* $x \in \{0,1\}^{m_w} \cup \{\perp\}$ *such that, conditioned on not being* $\perp$ *the output is distributed as* $\mathcal{D}_w$*. We will refer to such an algorithm* A *as a conditional sampler for* $\mathcal{D}_w$*.*

2.2 Promise problems

A promise problem Π consists of a pair of disjoint sets of strings $\Pi_{\mathsf{yes}}, \Pi_{\mathsf{no}} \subset \{0,1\}^*$. Strings in Π_{yes} are referred to as *yes* instances and strings in Π_{no} are referred to as *no* instances. The standard definition of a *language* corresponds to the case where every string is either a yes instance or a no instance, i.e., $\Pi_{\mathsf{yes}} \cup \Pi_{\mathsf{no}} = \{0,1\}^*$. (See [11] for a discussion and references.) For two parties A and B engaging in a protocol on common input x, let $\langle \mathsf{A}, \mathsf{B} \rangle(x)$ denote the final output of the protocol.

▶ **Definition 4** (Interactive Proofs). *An interactive proof system for a promise problem* $\Pi = (\Pi_{\mathsf{yes}}, \Pi_{\mathsf{no}})$ *is defined by a computationally bounded probabilistic verifier* V*, with a polynomial* T_{V} *such that the running time of* V *on common input* x *is upper-bounded by* $T_{\mathsf{V}}(|x|)$*, and an unbounded prover* P *satisfying the following properties:*

- *if* $x \in \Pi_{\mathsf{yes}}$*, then* $\Pr[\langle \mathsf{P}, \mathsf{V} \rangle(x) = 0] \leq \gamma$*, and*
- *if* $x \in \Pi_{\mathsf{no}}$*, then* $\forall \mathsf{P}^*, \Pr[\langle \mathsf{P}^*, \mathsf{V} \rangle(x) = 1] \leq \delta$*,*

where γ *and* δ *are constants in* $[0,1)$ *denoting the errors in completeness and soundness respectively. By default, we assume that* $\gamma = \delta = 0.1$*.*

2.3 Arthur-Merlin Proofs, and NP/Non-Deterministic Circuits

An AM protocol is a constant-round public-coin proof system and $\mathsf{AM/poly}$ is the non-uniform analog in which the verifier is implemented by a family of polynomial-sized probabilistic circuits. The complexity class $\mathsf{AM/poly}$ consists of all promise problems that admit $\mathsf{AM/poly}$ protocols. (See standard textbooks like [3, 12] for formal definition.) A *nondeterministic* circuit C has additional "nondeterministic input wires". We say that the circuit C evaluates to 1 on x iff there exist an assignment to the nondeterministic input wires that makes C output 1 on x. An *NP-circuit* C (resp., *nondeterministic NP-circuit*) is a standard circuit (resp., nondeterministic circuit) which in addition to the standard gates uses SAT gates, where a SAT gate gets a formula φ as an input and returns 1 iff the formula is satisifiable. The *size* of the circuit is the total number of wires and gates. Polynomial-size nondeterministic circuits, NP-circuits, non-deterministic NP-circuits are the non-uniform analogues of NP, P^{NP} and $\mathsf{NP}^{\mathsf{NP}} = \Sigma_2^P$, respectively.

The literature on complexity theory and derandomization contains various hardness assumptions against $\mathsf{AM/poly}$/nondeterministic/nondeterministic NP circuits and their generalizations to higher levels of the polynomial hierarchy. (See, e.g., [1] and references therein). Specifically, we will make use of the following result.

▶ **Theorem 5** (PRGs from hardness assumptions [18, 21, 30, 31]). *Suppose that* $\mathsf{E} = \mathsf{DTime}(2^{O(n)})$ *is hard for exponential-size non-deterministic circuits (resp., exponential-size non-deterministic NP-circuits), i.e., there exists a language* L *in* E *and a constant* $\beta > 0$*, such that for every sufficiently large* n*, circuits of size* $2^{\beta n}$ *fail to compute the characteristic function of* L *on inputs of length* n*.*

Then for every polynomial $T(\cdot)$ *and inverse polynomial* $\epsilon(\cdot)$*, for all sufficiently large* m*, there exists a pseudorandom generator* G *that stretches seeds of length* $\rho = O(\log m)$ *into a string of length* m *in time* $\mathsf{poly}(m)$ *such that* G *fools every promise problem* $\Pi = (\Pi_{\mathsf{yes}}, \Pi_{\mathsf{no}})$ *that can be decided by an* $\mathsf{AM}/\mathsf{poly}$ *proof system with a* T*-size verifier (resp., by a non-deterministic NP-circuit of size* T*) in the following sense. For every sufficiently large* m *and* $b \in \{\mathsf{yes}, \mathsf{no}\}$

$$\left| \Pr_{z \overset{R}{\leftarrow} U_m}[z \in \Pi_b] - \Pr_{z \overset{R}{\leftarrow} G(U_\rho)}[z \in \Pi_b] \right| \leq \epsilon(m).$$

As noted in [1], the above assumptions can be seen as the nonuniform and scaled-up versions of assumptions of the form Exponential-Time is not equal to NP or to Σ_2^P (which are widely believed in complexity theory). As such, these assumptions are very strong, and yet plausible - the failure of one of these assumptions will force us to change our current view of the interplay between time, nonuniformity and nondeterminism. As a secondary advantage (also noted in previous works), one can base the PRG on any concrete E-complete problem, and an explicit PRG whose security reduces to the underlying assumption. (We do not have to consider and evaluate various different candidate functions for the hardness assumption.)

2.4 Set-lower bound

We will make use of the set lower-bound protocol of [16].

▶ **Theorem 6** (Set lower-bound protocol[16]). *Let* $S \subset \{0,1\}^*$ *be an* NP *set (i.e., membership in* S *can be efficiently verified). Then there exists an* AM *protocol* $\langle \mathsf{P}, \mathsf{V} \rangle$ *such that given* $(1^n, k)$ *as common inputs the following holds,*

- *if* $|S \cap \{0,1\}^n| \geq k$*, then* $\Pr[\langle \mathsf{P}, \mathsf{V} \rangle(1^n, k) = 1] \geq 0.9$*,*
- *if* $|S \cap \{0,1\}^n| \leq k/2$ *then for every prover* P^* *it holds that* $\Pr[\langle \mathsf{P}^*, \mathsf{V} \rangle(1^n, k) = 1] < 0.1$*.*

2.5 Approximate counting

We say that a number p is an ϵ-relative approximation to q if $(1-\epsilon) \cdot p \leq q \leq (1+\epsilon) \cdot p$. It is useful to note that if p' is an ϵ-approximation to p and q' is an ϵ-approximation to q, then a p'/q' is a 2ϵ-approximation to p/q. We use the following classical result on approximate counting of satisfying assignments.

▶ **Theorem 7** (approximate counting,[19, 36, 35]). *For every sufficiently large* s *and every* $\epsilon > 0$*, there is an NP-circuit of size* $\mathsf{poly}(s/\epsilon)$ *that given a (standard) circuit* C *of size* s *outputs an* ϵ*-approximation of* $|\{x : C(x) = 1\}|$*.*

2.6 Hashing

We say that a function family $\mathcal{F}_{m,d}$ is (δ, s)-injective if for every set $S \subset \{0,1\}^m$ of size at most 2^s, a random member $f \overset{R}{\leftarrow} \mathcal{F}_{m,d}$ is injective over S with probability at least $1 - \delta$. It is well known that pair-wise independent hash functions [7] have this property. Formally, the following statement follows from [10] where the pair-wise independent hash family is instantiated with hash functions that are based on, say, Toeplitz matrices.

▶ **Lemma 8** (Good hashing from 2-wise independence). *There exists a family of hash functions $\mathcal{F}$ such that for every δ, s and $d = 2s + \lceil \log\left(\frac{1}{\delta}\right) \rceil$, the restriction $\mathcal{F}_{m,d}$ of $\mathcal{F}$ to functions from m-bits to d-bits is (δ, s)-injective. Moreover, (1) Functions in $\mathcal{F}_{m,d}$ are indexed by strings $z \in \{0,1\}^L$ where $L = m + 2d$; (2) There exists an efficient universal evaluation algorithm F that given $(1^m, 1^d)$, an index $z \in \{0,1\}^L$, and an input $x \in \{0,1\}^m$ outputs $f_z(x)$; and (3) for $z \xleftarrow{R} \{0,1\}^{m+2d}$, the function $F(1^m, 1^d, z, \cdot)$ is uniform over $\mathcal{F}_{m,d}$.*

3 Hashing-Based Solution for the Compression Problem

In this section we construct compression schemes. We begin with a formal definition.

▶ **Definition 9** (Interactive Compression). *An interactive compression scheme for a family of distributions $\{\mathcal{D}_w\}$ with error $\epsilon(\cdot)$ is defined by a computationally bounded compressor* CMP *and a computationally-unbounded decompressor* DCMP *satisfying the following property: For every $w \in \{0,1\}^*$ given to* DCMP*, and $x \xleftarrow{R} \mathcal{D}_w$ given to* CMP *the probability that* DCMP *outputs x is at least $1 - \epsilon(w)$, where the probability is taken over the choice of x and the randomness of the parties (if the parties are randomized). The communication complexity of the decompressor and compressor, $\mathsf{Com}_{\mathsf{DCMP}}(w)$ and $\mathsf{Com}_{\mathsf{CMP}}(w)$, are defined to be the maximal number of bits communicated by the decompressor and compressor when the decompressor's input is w.*

It is natural to solve the compression problem by letting the compressor hash the string x to a shorter string. If the hash function is injective over a "heavy set" of strings then the decompressor will be able to recover x from the hash of x, with a low error probability. This idea resembles Shannon's celebrated source coding theorem [34] except that we use a single instance of the source and accordingly rely on a weaker concentration of measure results (Markov's inequality as opposed to Chernoff). We formalize this approach starting with the notion of "heavy strings," which will form our heavy set.

▶ **Definition 10** (set of heavy strings). *For a distribution $\mathcal{D}$ over m-bit strings and an error parameter ϵ, we define the set of ϵ-*heavy *strings, $\mathscr{X}(\mathcal{D}, \epsilon)$, to be the set of all strings whose weight under $\mathcal{D}$ is at least $2^{-h/\epsilon}$ where $h = H(\mathcal{D})$ is the Shannon's entropy of $\mathcal{D}$, i.e.,*

$$\mathscr{X}(\mathcal{D}, \epsilon) := \{x \in \{0,1\}^m \mid \Pr_{\mathcal{D}}(x) \geq 2^{-H(\mathcal{D})/\epsilon}\}.$$

We also define $\ell(\mathcal{D}, \epsilon) := \lceil \log |\mathscr{X}(\mathcal{D}, \epsilon)| \rceil$.

It is not hard to see that the set of heavy strings cannot be too large and also that it is a heavy set – i.e., it contains at least $1 - \epsilon$ mass of the distribution. Specifically, we record the following observations.

▶ **Lemma 11** (Set of heavy strings is small). *For every $\mathcal{D}$ and $\epsilon > 0$, it holds that $\ell(\mathcal{D}, \epsilon) \leq \lceil H(\mathcal{D})/\epsilon \rceil$.*

Proof. Let $h = H(\mathcal{D})$. For every string $x \in \mathscr{X}(\mathcal{D}, \epsilon)$ it holds that $\Pr_{\mathcal{D}}(x) \geq 2^{-h/\epsilon}$, and therefore

$$2^{-h/\epsilon} |\mathscr{X}(\mathcal{D}, \epsilon)| \leq \sum_{x \in \mathscr{X}(\mathcal{D}, \epsilon)} \Pr_{\mathcal{D}}(x) \leq 1.$$

It follows that $|\mathscr{X}(\mathcal{D}, \epsilon)| \leq 2^{h/\epsilon}$ which further implies that $\ell(\mathcal{D}, \epsilon) = \lceil \log |\mathscr{X}(\mathcal{D}, \epsilon)| \rceil \leq \lceil h/\epsilon \rceil$. ◀

▶ **Lemma 12** (Set of heavy strings is heavy). *For every $\mathcal{D}$ and $\epsilon > 0$, it holds that $\sum_{x \in \mathscr{X}(\mathcal{D},\epsilon)} \Pr_{\mathcal{D}}(x) \geq 1 - \epsilon$.*

Proof. For a string x, let $p_x := \Pr_{\mathcal{D}}(x)$ denote the weight of x under $\mathcal{D}$. Sample $x \xleftarrow{R} \mathcal{D}$ and consider the random variable $k_x = \log(1/p_x)$. By definition, the expected value of k_x is simply the entropy of $\mathcal{D}$, i.e., $\mathsf{E}_x[k_x] = h$, and so, by Markov's inequality, $\Pr_{x \xleftarrow{R} \mathcal{D}}[k_x \geq h/\epsilon] \leq \frac{\mathsf{E}_x[k_x]}{h/\epsilon} = \epsilon$. That is, at least $1-\epsilon$ of the mass belongs to elements x for which $\log(1/p_x) \leq h/\epsilon$, or equivalently, to elements whose weight is at least $2^{-h/\epsilon}$, which is nothing but our desired set $\mathscr{X}(\mathcal{D}, \epsilon)$. ◀

▶ **Definition 13** (good hash functions). *Let $\mathcal{D} = \{\mathcal{D}_w\}$ be a family of distributions where $\mathcal{D}_w$ is supported over m_w-long strings and has entropy of h_w. We say that a function family $\mathcal{F}$ is (δ, ϵ)-*good *for $\mathcal{D}$ if there exists a length function $d(h_w, \epsilon, \delta)$ such that for every distribution $\mathcal{D}_w$ in the family:*

$$\Pr_{f \xleftarrow{R} \mathcal{F}_{m,d}} [f \text{ is injective over } \mathscr{X}(\mathcal{D}_w, \epsilon)] \geq 1 - \delta,$$

where $\mathcal{F}_{m,d}$ denotes the restriction of $\mathcal{F}$ to functions from $\{0,1\}^m$ to $\{0,1\}^d$, $m = m_w$ and $d = d(h_w, \epsilon, \delta)$. We refer to d as the compression *of h. We say that $\mathcal{F}$ has a* representation size *of $L(m, d)$ if each function from m bits to d bits can be represented by $L(m, d)$ bits. We also assume that the family is efficiently computable, i.e., given an index z and an input x one can evaluate $f_z(x)$ in polynomial time.*

Recalling that $\mathscr{X}(\mathcal{D}_w, \epsilon)$ is of size at most 2^s for $s = h_w/\epsilon + 1$, we can use Lemma 8 to derive a (δ, ϵ)-good family with compression $d = 2s + \left\lceil \log\left(\frac{1}{\delta}\right) \right\rceil = 2h_w/\epsilon + \log(1/\delta) + O(1)$ and description $L = m + 2d$.

3.1 Hashing-based compression

Let $\mathcal{D} = \{\mathcal{D}_w\}$ be a collection of distributions and assume that $\mathcal{F}$ is (δ, ϵ)-good for $\mathcal{D}$ with compression d. Define the single-round compression protocol $P_{\mathcal{F}}$ as follows: Given the index w of the distribution, the de-compressor specifies a hash function $f_z \in \mathcal{F}_{m,d}$ where $m = m_w$ and $d = d(h_w, \epsilon, \delta)$ that is injective over the set $\mathscr{X}(\mathcal{D}_w, \epsilon')$ and sends the description z of f_z to the compressor, who sends back the value of $y = f_z(x)$. The computationally unbounded de-compressor then checks if y has a pre-image x' in $\mathscr{X}(\mathcal{D}, \epsilon')$, and if so it outputs the (unique) preimage x'. Otherwise, the de-compressor outputs a failure symbol $\perp$.

Assuming that $\delta < 1$, the de-compressor always finds a function f_z which is injective over the set $\mathscr{X}(\mathcal{D}, \epsilon')$, and so the protocol errs only if x falls out of $\mathscr{X}(\mathcal{D}, \epsilon')$. By Lemma 12, this happens with a probability of at most ϵ. Summarizing the above discussion, we get the following lemma.

▶ **Lemma 14** (compression from hashing). *Assuming that $\mathcal{F}$ is a (δ, ϵ)-good hash family for $\mathcal{D} = \{\mathcal{D}_w\}$ for some $\delta < 1$, the protocol $P_{\mathcal{F}}$ is a compression protocol for $\mathcal{D}$ with an error ϵ. For a distribution specified by w, the compressor communicates $d = d(h_w, \epsilon, \delta)$ bits and the de-compressor communicates $L = L(m_w, d)$ bits, and the compressor's computational complexity is $\mathsf{poly}(m_w, d)$, where d is the compression parameter of $\mathcal{F}$ and L is the description length. For the special case of pair-wise independent hashing and $\delta = 0.1$, we get $d = 2h_w/\epsilon + O(1)$ and $L = O(m_w + 2h_w/\epsilon)$.*

The factor of 2 overhead in the compressor's communication can be improved to $1 + o(1)$ by using better hash functions (e.g., two-level hashing [10]). We omit the details since it hardly changes the final results of the paper.

The above lemma yields a protocol that obtains the desired compression for the communication from the compressor, but suffers from a high overhead on the communication from the decompressor's end. We will improve this in the next section.

3.2 Improving De-Compressor Communication

In order to improve the communication of the prover, we construct a succinct hash family that is good for efficiently samplable distributions. To sample a hash function we choose a random seed for a PRG, expand it to a long string and use this string to specify a hash function from a family of pair-wise independent hash functions. We show that a PRG against $\mathsf{AM/poly}$ allows us to exponentially compress the description length of a hash function $f : \{0,1\}^m \to \{0,1\}^d$ from $\Omega(m + d)$ to $O(\log(m + d))$. We also extend this result to the case of conditioned efficient distributions at the expense of using a slightly stronger hardness assumption. This extension will be useful for the proof of Theorem 1.

▶ **Theorem 15** (succinct hashing for efficient and conditioned efficient distributions)**.** *Suppose that $E = \mathsf{DTime}(2^{O(n)})$ is hard for exponential-size non-deterministic circuits. Let $\mathcal{D} = \{\mathcal{D}_w\}$ be an efficient family of distributions over m_w-bit long strings having entropy h_w, and let $\epsilon(w)$ be inverse polynomial error parameter. Then, for any constant $\delta < 1$, there exists an efficient family of hash functions that is (δ, ϵ)-good for $\mathcal{D}$ with compression parameter $d = 2h_w/\epsilon + O(1)$ and description size of $L(m, d) = O(\log(m + d))$ bits.*

Moreover, the theorem extends to the case where $\mathcal{D} = \{\mathcal{D}_w\}$ is a family of conditioned efficient distributions, assuming that E is hard for exponential-size non-deterministic NP-circuits.

The high-level idea is to show that when $\mathcal{D}$ is efficiently samplable, there is an $\mathsf{AM/poly}$ protocol for checking whether a given hash function is injective over the set $\mathscr{X}(\mathcal{D}, \epsilon)$. Therefore, if we use a PRG that fools $\mathsf{AM/poly}$ to sample a hash function f from a collection $\mathcal{F}$, the probability that f will be injective over $\mathscr{X}(\mathcal{D}, \epsilon)$ is almost the same as the probability that a *random* member of $\mathcal{F}$ will be injective. The theorem then follows by taking $\mathcal{F}$ to be a family of pair-wise independent hash functions (for which we know that a random member is injective whp). Unfortunately, we do not know how to construct an $\mathsf{AM/poly}$ protocol that certifies injectivity, however, we can use an approximate version of this property that suffices for our purposes. We continue with formal proof.

Proof of Theorem 15. Let $T_1(|w|)$ be the complexity of $\mathcal{D}$ and let $T_2(|w|)$ denote the complexity of evaluating the pair-wise independent hash functions $\mathcal{F}_{m,d} = \{f_z\}_{z \in \{0,1\}^L}$ promised in Lemma 8 where $m = m_w$ and $d = 2h_w/\epsilon + \left\lceil \log\left(\frac{2}{\delta}\right) \right\rceil + 2$ and $L = m + 2d$. Note that for these parameters $\mathcal{F}_{m,d}$ is $(\delta/2, 1 + h/\epsilon)$-injective (Lemma 8). Let T be some fixed polynomial in $T_1(|w|) + T_2(|w|)$ whose value will be determined later, and let $\mathsf{G} : \{0,1\}^k \to \{0,1\}^L$ be a PRG that $\frac{\delta}{2}$-fools T-size $\mathsf{AM/poly}$ with seed length $k = O(\log L)$ whose existence is promised by Theorem 5. Consider the family of functions $\mathcal{F}'_{m,d}$ whose members f'_s are identified by an index $s \in \{0,1\}^k$, and are defined by $f'_s = f_{\mathsf{G}(s)}$ where f_z is the function from $\mathcal{F}_{m,d}$ whose index is z. Note that $\mathcal{F}'_{m,d}$ is computable in time $\mathsf{poly}(|w|)$. We claim that $\mathcal{F}'_{m,d}$ is (δ, ϵ)-good for $\mathcal{D}_w$ for every w.

Fix some w. We begin by introducing a promise problem Π_w over L-bit strings. (Recall that $L = m_w + 2d(w)$.)

- Yes instance: A string $z \in \{0,1\}^L$ is a yes instance if the function $f_z : \{0,1\}^m \to \{0,1\}^d$ is *not injective* over the set $\mathscr{X}(\mathcal{D}_w, \epsilon)$.
- No instance: A string $z \in \{0,1\}^L$ is a No instance if the function $f_z : \{0,1\}^m \to \{0,1\}^d$ is *injective* over the set $\mathscr{X}'(\mathcal{D}_w, \epsilon) = \{x \in \{0,1\}^m \mid \Pr_{\mathcal{D}_w}(x) \geq 0.5 \cdot 2^{-(h_w/\epsilon)}\}$.

We show that the above promise problem admits an AM/poly proof system. Let ρ_w denote the randomness complexity of the distribution $\mathcal{D}_w$. Let $A(w;\cdot)$ be the PPT algorithm for sampling from $\mathcal{D}_w$. Consider the following protocol with prover P and verifier V and common input z:

1. P sends two strings $(x_0, x_1) \in \{0,1\}^m \times \{0,1\}^m$ to V.
2. V checks if $x_0 \neq x_1$ and $f_z(x_0) = f_z(x_1)$. If the checks fail, then it aborts with output 0. Otherwise, the parties proceed further.
3. For $b \in \{0,1\}$, the parties run the following set membership protocol for the string x_b:
   a. Consider the set $R_{x_b} = \{r \in \{0,1\}^{\rho_w} \mid A(w;r) = x_b\}$.
   b. Run a set lower-bound protocol for set R_{x_b} with size parameter $\alpha = 2^{\rho_w} \cdot 2^{-h/\epsilon}$. For membership queries to the set R_{x_b} for a string r, just check whether $A(w;r) = x_b$.
4. V outputs 1 if both the checks succeed. Otherwise, it outputs 0.

▷ Claim 16. The above protocol is an AM/poly protocol for Π_w.

Proof. Suppose that z is a Yes instance. That is, f_z is not injective over the set $\mathscr{X}(\mathcal{D}_w, \epsilon)$. Then, an honest P will be able to find two strings $(x_0, x_1) \in \mathscr{X}(\mathcal{D}_w, \epsilon) \times \mathscr{X}(\mathcal{D}_w, \epsilon)$ such that $x_0 \neq x_1$ and $f_z(x_0) = f_z(x_1)$. In our protocol, the checks $x_0 \neq x_1$ and $f_z(x_0) = f_z(x_1)$ will always succeed in this case. Since both x_0 and x_1 belong to the set $\mathscr{X}(\mathcal{D}_w, \epsilon)$, this implies that both the sets R_{x_0} and R_{x_1} have size at least α according to the definition of $\mathscr{X}(\mathcal{D}_w, \epsilon)$. Hence, both the set lower-bound protocols correspond to YES instances and either of these will fail with probability at most 0.1. It follows that the total failure probability is at most 0.2.

We move on to the case where z is a No instance, i.e., the function f_z is injective over the set $\mathscr{X}'(\mathcal{D}_w, \epsilon)$ for all $w \in \{0,1\}^*$. Fix the pair $(x_0, x_1) \in \{0,1\}^m \times \{0,1\}^m$ that the prover sends in the first step and assume that the verifier did not reject in the second step, i.e., $x_0 \neq x_1$ and $f_z(x_0) = f_z(x_1)$. Then, at least one of x_0 or x_1 must lie outside the set $\mathscr{X}'(\mathcal{D}_w, \epsilon)$ since f_z is injective over this set. Therefore, either the size of R_{x_0} or that of R_{x_1} must be smaller than $\alpha/2 = 2^{\rho_w} \cdot 0.5 \cdot 2^{-(h/\epsilon)}$. It follows that, except with probability 0.1, the verifier rejects in at least one of the set lower-bound protocols. ◁

By hard-wiring w and h_w, we implement the verifier by a non-uniform circuit of size polynomial in $T_1(|w|) + T_2(|w|)$. We can therefore take $T(|w|)$ to be complexity of the verifier, and conclude that

$$\begin{aligned}\Pr_s[f_{\mathsf{G}(s)} \text{ is injective over } \mathscr{X}(\mathcal{D}_w, \epsilon)] &> \Pr_s[f_{\mathsf{G}(s)} \text{ is injective over } \mathscr{X}'(\mathcal{D}_w, \epsilon)] \\ &> \Pr_z[f_z \text{ is injective over } \mathscr{X}'(\mathcal{D}_w, \epsilon)] - \frac{\delta}{2} > 1 - \frac{\delta}{2} - \frac{\delta}{2} = 1 - \delta.\end{aligned}$$

The last inequality follows by recalling that $\mathcal{F}_{m,d}$ is $(\delta/2, 1 + h_w/\epsilon)$-injective and since that set $\mathscr{X}'(\mathcal{D}_w, \epsilon)$ is of size at most $2^{1+h_w/\epsilon}$. We conclude that $\mathcal{F}'_{m,d}$ is (δ, ϵ)-good for $\mathcal{D}_w$, as required. The first part of the theorem follows.

The "Moreover" part

The proof of the second part is similar with the following modification. We take G to be a PRG that 0.1-fools T-size non-deterministic NP-circuits (whose existence follows from the underlying assumption via Theorem 5), and show that Π_w can be decided by such

circuits. Given a string z, the circuit C_w non-deterministically guesses a pair of m-bit strings (x_0, x_1) and verifies that $x_0 \neq x_1$ and $f_z(x_0) = f_z(x_1)$. (If any of these conditions fail, the circuit rejects.) Next, C_w derives for $b \in \{0,1\}$, an α-approximation q_b for the quantity $p_b = \Pr[\mathcal{D}_w = x_b]$ for $\alpha = 0.2$, and accepts if and only if q_0 and q_1 are both larger than $0.7 \cdot 2^{-(h_w/\epsilon)}$. (Here h_w is hard-wired to C_w.) The approximation q_b is obtained by using the Approximate Counting algorithm (Theorem 7) as follows. Recall that $\mathcal{D}$ is defined by a PPT conditional sampler $A(w;\cdot)$ with randomness complexity ρ_w such that

$$p_b = \Pr[\mathcal{D}_w = x_b] = \frac{|\{r \in \{0,1\}^{\rho_w} : A(w;r) = x_b\}|}{|\{r \in \{0,1\}^{\rho_w} : A(w;r) \neq \perp\}|}.$$

Hence, to derive an α-approximation of p_b it suffices to get a $\alpha/2$-approximation of both the denominator and numerator. This can be done by a polynomial-size NP-circuit since these sets are recognizable by polynomial-size circuits (whose size is the sum of the complexity of A). It remains to prove the following claim.

▷ Claim 17. The circuit C_w accepts Yes instances and rejects No instances of Π_w.

Proof. Suppose that z is a Yes instance. That is, f_z is not injective over the set $\mathscr{X}(\mathcal{D}_w, \epsilon)$. Then there exists an f_z-collision $x_0 \neq x_1 \in \mathscr{X}(\mathcal{D}_w, \epsilon) \times \mathscr{X}(\mathcal{D}_w, \epsilon)$. Since both x_0 and x_1 belong to the set $\mathscr{X}(\mathcal{D}_w, \epsilon)$, this implies that p_0 and p_1 are at least $2^{-h_w/\epsilon}$ and so q_0 and q_1 are larger than $0.8 \cdot 2^{-h_w/\epsilon}$ and C_w accepts.

We move on to the case where z is a No instance, i.e., the function f_z is injective over the set $\mathscr{X}'(\mathcal{D}_w, \epsilon)$ for all $w \in \{0,1\}^*$. Then, for any f_z-collision $x_0 \neq x_1$ either $p_0 < 0.5 \cdot 2^{-h_w/\epsilon}$ or $p_1 < 0.5 \cdot 2^{-h_w/\epsilon}$. This means that either q_0 or q_1 must be smaller than $1.2 \cdot 0.5 \cdot 2^{-h_w/\epsilon} \leq 0.6 \cdot 2^{-h_w/\epsilon}$, and C_w rejects. ◁

The rest of the argument is identical to the proof of the first part of the theorem. ◀

Together with Lemma 14, we derive the following theorem.

▶ **Theorem 18.** *Suppose that $E = \mathsf{DTime}(2^{O(n)})$ is hard for exponential-size non-deterministic circuits. Let $\mathcal{D} = \{\mathcal{D}_w\}$ be an efficient family of distributions over m_w-bit long strings having entropy h_w, and let $\epsilon(w)$ be inverse polynomial error parameter. Then, there exists a single-round compression protocol for $\mathcal{D}$ with an error ϵ and communication of $d = 2h_w/\epsilon + O(1)$ for the compressor and $L = O(\log(m_w + h_w/\epsilon))$ for the de-compressor. Furthermore, the compressor is efficient.*

Moreover, the theorem extends to the case where $\mathcal{D} = \{\mathcal{D}_w\}$ is a family of conditioned efficient distributions, assuming that E is hard for exponential-size non-deterministic NP-circuits.

4 Reducing the Communication in Interactive Proofs

In this section, we prove the main theorem (Theorem 1). Roughly speaking, we compress each message that the verifier sends by using a properly chosen hash function. We begin with some notations and definitions.

Let $\langle \mathsf{P}, \mathsf{V} \rangle$ be an interactive proof for a promise problem Π. For an input x, let $T(|x|)$ and $\rho(|x|)$ denote the running-time and randomness complexity of the verifier.We assume that the parties speak in alternating turns and that the prover sends the first and last message. (The latter assumption always holds and the former can be guaranteed at the expense of adding an additional empty round). Letting $I'(|x|)$ denote the number of rounds in which

the verifier speaks, we get that the the total number of rounds is $I(|x|) = 2I'(|x|) + 1$. We let a_i and b_i denote the ith message of the prover and verifier, respectively, and let $b_{I(|x|)+1}$ denote the final verdict of the verifier (accept/reject).

We can think of the V as a machine that takes an input x a random tape r and a sequence of prover's messages $a = (a_i)_{1 \le i \le k}, k \le I' + 1$ and outputs the message b_k. For a partial transcript $w = (x, a = (a_i)_{i \in [k]}, b = (b_i)_{i \in [k-1]})$, consider the probability distribution $\mathcal{D}_w$ of the verifier's next message b_k conditioned on seeing the partial transcript w. We can describe $\mathcal{D}_w$ as the output of the following randomized process: Sample a random tape $r \xleftarrow{R} \{0,1\}^{\rho(|x|)}$ conditioned on the event

$$\bigwedge_{1 \le j \le k-1} \mathsf{V}(x, a_1, \ldots, a_j; r) = b_j \tag{1}$$

and output the string $b_k = \mathsf{V}(x, a_1, \ldots, a_{k-1}; r)$. Note that $\mathcal{D} = \{\mathcal{D}_w\}$ is *conditioned efficient*: consider a PPT conditional sampler A, which on input w as above and random tape $r \in \{0,1\}^{\rho(|x|)}$, outputs $\perp$ if (1) does not hold, and outputs b_k otherwise. Let h_w and m_w denote the entropy and domain bit-length of $\mathcal{D}_w$, and let $\epsilon(w) = 0.01/I(|x|)$ where x is the first entry of w. Let $\mathcal{F}$ denote a family of hash functions (promised by the second part of Theorem 15) which is $(0.2, \epsilon)$-good for $\mathcal{D}$ and for $\mathcal{D}_w$ achieves compression of $2h_w/\epsilon + O(1)$ and description size of $O(\log(m_w + h_w/\epsilon)) < O(\log(m_w/\epsilon)) < O(\log m_w) + O(\log I(|x|))$ where the first inequality follows by noting that $h_w \le m_w$.

The new proof system

We define the new proof system $\langle \mathsf{P}', \mathsf{V}' \rangle$ as follows. Given x as a common input and randomness ρ for the verifier, the parties initialize an "emulated" transcript $w = x$ and proceed for $i = 1, \ldots, I' - 1$ rounds as follows.

1. P': Compute a_i by calling $\mathsf{P}(w)$ and locally update $w = (w, a_i)$. Choose hash function f_i from $\mathcal{F}$ that is injective over $\mathscr{X}(\mathcal{D}_w, \epsilon)$, and send (a_i, f_i). (Recall that $\mathscr{X}(\mathcal{D}_w, \epsilon)$ is the set of strings whose weight under $\mathcal{D}_w$ is at least $2^{-h/\epsilon}$ where h is the Shannon's entropy of $\mathcal{D}_w$; See Definition 10.)
2. V': Compute b_i by calling $\mathsf{V}(x, a_1, \ldots, a_i; \rho)$ where a_i is the ith message sent by the prover, and send $b'_i = f_i(b_i)$.
3. Before proceeding to the next iteration the prover locally computes b_i by choosing the unique string in $\mathscr{X}(\mathcal{D}_w, \epsilon)$ that maps to b'_i. If this string is not unique the prover sends a special abort symbol and the verifier terminates with rejection. Otherwise, the prover updates its view to $w := (w, b_i)$.

At the last round, the prover sends $a_{I'}$ by calling $\mathsf{P}(w)$ and the verifier outputs its verdict $b_{I'+1}$ by calling $\mathsf{V}(x, a_1, \ldots, a_t; \rho)$.

Completeness and Soundness

Let γ be the completeness error of the original proof system. For a yes instance x, it holds that

$$\Pr[\langle \mathsf{P}', \mathsf{V}' \rangle(x) = 1] \ge \Pr_r[\langle \mathsf{P}, \mathsf{V} \rangle(x) = 1] - \Pr_r[\text{decoding failure}] \ge 1 - \gamma - \epsilon \cdot (I' - 1) \ge 1 - \gamma - 0.01,$$

where the second inequality follows from a union-bounds over all the rounds. For soundness, fix a No instance x, and observe that any cheating strategy for the prover P' in the new proof system translates to a cheating strategy in the original proof system. Indeed, if for each $i \in [I']$, P' maliciously chooses a_i and f_i based on b'_{i-1} and its internal state

$S = (x, a = (a_j)_{j<i}, f = (f_j)_{j<i}, b' = (b'_j)_{j<i-1})$, we can define a cheating prover for the original system that maintains the same state S and given b_{i-1} executes P' on the state S and on $b'_{i-1} = f_{i-1}(b_{i-1})$. The probability that the verifier V accepts is exactly the same probability that V' does. Therefore the soundness error of the new system is the same as the soundness error of the original system.

Communication complexity

We begin by analyzing the expected communication complexity under the assumption that P is honest. Fix x, and let $I = I(|x|)$,$I' = I'(|x|)$. Let $w = (x, (a_i, b_i)_{i\in[I']})$ denote the random variable that describes a random transcript and let $w[k] = (x, (a_i, b_i)_{i\in[k]})$ denote the kth prefix of the transcript. We can assume that the honest prover is deterministic (i.e., a_i is a deterministic function of $x, (b_j)_{j<i}$) and so all the randomness is due to the b part. Recall that in each round i, the verifier sends a string b'_i of length $2h_{w[i]}/\epsilon + O(1)$ where $h_{w[i]}$ is the entropy of $\mathcal{D}_{w[i]}$. Note that $h_{w[i]}$ is a random variable (since $w[i]$ is a random variable). Thus, the communication complexity of the verifier is given by the following random variable

$$\sum_{i\in[I']} 2h_{w[i]}/\epsilon + O(1) = O(I') + 2/\epsilon \sum_{i\in[I']} h_{w[i]} = O(I) + O(I) \sum_{i\in[I']} h_{w[i]}.$$

By the chain rule, the expected value of $\sum_{i\in[I']} h_{w[i]}$ is the entropy of w which is at most the randomness complexity of the verifier. Overall, the expected communication complexity of the verifier is $O(I \cdot \rho)$. The communication of the prover in the ith iteration consists of the original communication (the a_i part) and the description of the hash functions which is of length $O(\log m_{w[i]}) + O(\log I(|x|))$. Overall the prover communication grows by at most $O(I(\log I + \log m))$ where m is the maximal length of the verifier's message in the original scheme. Since m and I are polynomially bounded in n, this can be written as $O(I(\log n))$.

Deriving Theorem 1

The communication analysis is only on expectation and it assumes that the prover is honest. To get a worst case bound, we slightly modify the proof system by letting the verifier halt the interaction (with rejection) if she communicates more than, say 100 times the expected communication complexity. By Markov's inequality, this increases the completeness error by at most 0.01. Theorem 1 follows.

5 Conclusion

Compressing interactive protocols is a problem of fundamental nature in information theory. When computational constraints are also involved, it leads to a question that combines complexity theory and information theory into a natural, yet difficult problem. In this paper, we partially answered the problem posed at the beginning: whether the communication from a verifier in an interactive proof system can always be reduced to the level of randomness used by the verifier, without increasing the verifier's randomness, the round complexity or the prover's communication significantly. We leave it as an open question if such a result is possible without relying on complexity assumptions (or using weaker ones), and if quantitative improvements can be achieved over our result.

En-route to our main result, we encounter several interesting problems. While our focus is on proof systems, the compression results here extend to any 2-party protocol where one party is computationally unbounded, and the other party is randomized but has no private

inputs. Further, the special case of the single-round compression problem is of significance in its own right. The notion of efficiently conditional distributions that we introduced, being natural, could be of independent interest.

References

1 Benny Applebaum, Sergei Artemenko, Ronen Shaltiel, and Guang Yang. Incompressible functions, relative-error extractors, and the power of nondeterministic reductions. *Comput. Complex.*, 25(2):349–418, 2016. doi:10.1007/S00037-016-0128-9.

2 Benny Applebaum and Eyal Golombek. On the Randomness Complexity of Interactive Proofs and Statistical Zero-Knowledge Proofs. In Stefano Tessaro, editor, *2nd Conference on Information-Theoretic Cryptography (ITC 2021)*, volume 199 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 4:1–4:23, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ITC.2021.4.

3 S. Arora and B. Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2006. URL: https://theory.cs.princeton.edu/complexity/book.pdf.

4 László Babai and Shlomo Moran. Arthur-merlin games: A randomized proof system, and a hierarchy of complexity class. *J. Comput. Syst. Sci.*, 36(2):254–276, April 1988. doi:10.1016/0022-0000(88)90028-1.

5 Marshall Ball, Ronen Shaltiel, and Jad Silbak. Non-malleable codes with optimal rate for poly-size circuits. *Electron. Colloquium Comput. Complex.*, TR23-167, 2023. arXiv:TR23-167.

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

7 J.Lawrence Carter and Mark N. Wegman. Universal classes of hash functions. *Journal of Computer and System Sciences*, 18(2):143–154, 1979. doi:10.1016/0022-0000(79)90044-8.

8 Andrew Drucker. Nondeterministic direct product reductions and the success probability of SAT solvers. In *FOCS*, pages 736–745. IEEE Computer Society, 2013.

9 Uriel Feige and Carsten Lund. On the hardness of computing the permanent of random matrices. *Computational Complexity*, 6(2):101–132, 1997.

10 Michael L. Fredman, János Komlós, and Endre Szemerédi. Storing a sparse table with O(1) worst case access time. *J. ACM*, 31(3):538–544, June 1984. doi:10.1145/828.1884.

11 Oded Goldreich. On promise problems: A survey. In Oded Goldreich, Arnold L. Rosenberg, and Alan L. Selman, editors, *Theoretical Computer Science, Essays in Memory of Shimon Even*, volume 3895 of *Lecture Notes in Computer Science*, pages 254–290. Springer, 2006. doi:10.1007/11685654_12.

12 Oded Goldreich. *Computational complexity - a conceptual perspective*. Cambridge University Press, 2008. doi:10.1017/CBO9780511804106.

13 Oded Goldreich and Maya Leshkowitz. *On Emulating Interactive Proofs with Public Coins*, pages 178–198. Springer International Publishing, Cham, 2020. doi:10.1007/978-3-030-43662-9_12.

14 Oded Goldreich and Avi Wigderson. Derandomization that is rarely wrong from short advice that is typically good. In *RANDOM*, volume 2483 of *Lecture Notes in Computer Science*, pages 209–223. Springer, 2002.

15 S Goldwasser, S Micali, and C Rackoff. The knowledge complexity of interactive proof-systems. In *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing*, STOC '85, pages 291–304, New York, NY, USA, 1985. Association for Computing Machinery. doi:10.1145/22145.22178.

16 S Goldwasser and M Sipser. Private coins versus public coins in interactive proof systems. In *Proceedings of the Eighteenth Annual ACM Symposium on Theory of Computing*, STOC '86, pages 59–68, New York, NY, USA, 1986. Association for Computing Machinery. doi:10.1145/12130.12137.

17 Dan Gutfreund, Ronen Shaltiel, and Amnon Ta-Shma. Uniform hardness versus randomness tradeoffs for arthur-merlin games. *Computational Complexity*, 12(3-4):85–130, 2003.

18 Russell Impagliazzo and Avi Wigderson. P = bpp if e requires exponential circuits: Derandomizing the xor lemma. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, STOC '97, pages 220–229, New York, NY, USA, 1997. Association for Computing Machinery. doi:10.1145/258533.258590.

19 Mark Jerrum, Leslie G. Valiant, and Vijay V. Vazirani. Random generation of combinatorial structures from a uniform distribution. *Theor. Comput. Sci.*, 43:169–188, 1986.

20 Adam Klivans and Dieter van Melkebeek. Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses. *SIAM J. Comput.*, 31(5):1501–1526, 2002.

21 Adam R. Klivans and Dieter van Melkebeek. Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses. *SIAM Journal on Computing*, 31(5):1501–1526, 2002. doi:10.1137/S0097539700389652.

22 Maya Leshkowitz. Round complexity versus randomness complexity in interactive proofs. *Theory Comput.*, 18:1–65, 2022. URL: https://theoryofcomputing.org/articles/v018a013/.

23 Carsten Lund, Lance Fortnow, Howard Karloff, and Noam Nisan. Algebraic methods for interactive proof systems. *J. ACM*, 39(4):859–868, October 1992. doi:10.1145/146585.146605.

24 Peter Bro Miltersen and N. V. Vinodchandran. Derandomizing arthur-merlin games using hitting sets. *Computational Complexity*, 14(3):256–279, 2005.

25 Moni Naor, Leonard J. Schulman, and Aravind Srinivasan. Splitters and near-optimal derandomization. In *36th Annual Symposium on Foundations of Computer Science, Milwaukee, Wisconsin, USA, 23-25 October 1995*, pages 182–191. IEEE Computer Society, 1995. doi:10.1109/SFCS.1995.492475.

26 A. Orlitsky. Worst-case interactive communication. i. two messages are almost optimal. *IEEE Transactions on Information Theory*, 36(5):1111–1126, 1990. doi:10.1109/18.57210.

27 Rasmus Pagh. Hash and displace: Efficient evaluation of minimal perfect hash functions. In Frank K. H. A. Dehne, Arvind Gupta, Jörg-Rüdiger Sack, and Roberto Tamassia, editors, *Algorithms and Data Structures, 6th International Workshop, WADS '99, Vancouver, British Columbia, Canada, August 11-14, 1999, Proceedings*, volume 1663 of *Lecture Notes in Computer Science*, pages 49–54. Springer, 1999. doi:10.1007/3-540-48447-7_5.

28 Ronen Shaltiel. Weak derandomization of weak algorithms: Explicit versions of yao's lemma. *Comput. Complex.*, 20(1):87–143, 2011. doi:10.1007/S00037-011-0006-4.

29 Ronen Shaltiel and Jad Silbak. Explicit codes for poly-size circuits and functions that are hard to sample on low entropy distributions. *Electron. Colloquium Comput. Complex.*, TR23-149, 2023. arXiv:TR23-149.

30 Ronen Shaltiel and Christopher Umans. Simple extractors for all min-entropies and a new pseudorandom generator. *J. ACM*, 52(2):172–216, 2005.

31 Ronen Shaltiel and Christopher Umans. Pseudorandomness for approximate counting and sampling. *Computational Complexity*, 15(4):298–341, 2006.

32 Ronen Shaltiel and Christopher Umans. Low-end uniform hardness versus randomness tradeoffs for AM. *SIAM J. Comput.*, 39(3):1006–1037, 2009.

33 Adi Shamir. Ip = pspace. *J. ACM*, 39(4):869–877, October 1992. doi:10.1145/146585.146609.

34 Claude E. Shannon. A mathematical theory of communication. *Bell Syst. Tech. J.*, 27(3):379–423, 1948. doi:10.1002/J.1538-7305.1948.TB01338.X.

35 Michael Sipser. A complexity theoretic approach to randomness. In *STOC*, pages 330–335. ACM, 1983.

36 Larry J. Stockmeyer. The complexity of approximate counting (preliminary version). In *STOC*, pages 118–126. ACM, 1983.

37 Luca Trevisan and Salil P. Vadhan. Extracting randomness from samplable distributions. In *FOCS*, pages 32–42. IEEE Computer Society, 2000.

38 Luca Trevisan, Salil P. Vadhan, and David Zuckerman. Compression of samplable sources. *Comput. Complex.*, 14(3):186–227, 2005. doi:10.1007/S00037-005-0198-6.