A PCP Theorem for Interactive Proofs

Gal Arnon
gal.arnon@weizmann.ac.il
Weizmann Institute

Alessandro Chiesa
alexch@berkeley.edu
UC Berkeley

Eylon Yogev
eylony@gmail.com
Tel Aviv University

July 6, 2021

Abstract

The celebrated PCP Theorem states that any language in NP can be decided via a verifier that reads $O(1)$ bits from a polynomially long proof. Interactive oracle proofs (IOP), a generalization of PCPs, allow the verifier to interact with the prover for multiple rounds while reading a small number of bits from each prover message. While PCPs are relatively well understood, the power captured by IOPs (beyond NP) has yet to be fully explored.

We present a generalization of the PCP theorem for interactive languages. We show that any language decidable by a $k(n)$-round IP has a $k(n)$-round public-coin IOP, where the verifier makes its decision by reading only $O(1)$ bits from each (polynomially long) prover message and $O(1)$ bits from each of its own (random) messages to the prover. Our proof relies on a new notion of PCPs that we construct called index-decodable PCPs, which may be of independent interest.

We are then able to bring transformations that previously applied only for IPs into the realm of IOPs. We show IOP-to-IOP transformations that preserve query complexity and achieve: (i) private-coins to public-coins; (ii) round reduction; and (iii) imperfect to perfect completeness.

Keywords: interactive proofs; probabilistically checkable proofs; interactive oracle proofs

Contents

1 Introduction

Probabilistic proofs play a central role in complexity theory and cryptography. In the past decades, probabilistic proofs have become powerful and versatile tools in these fields, leading to breakthroughs in zero-knowledge, delegation of computation, hardness of approximation, and other areas.

As an example, interactive proofs (IPs) [GMR89] allow proof-verification to be randomized and interactive, which seemingly confers them much more power than their deterministic (and non-interactive) counterparts. In a k-round IP, a probabilistic polynomial-time verifier exchanges k messages with an all-powerful prover and then accepts or rejects; $\mathrm{IP}[\mathsf{k}]$ is the class of languages decidable via a k-round interactive proof. Seminal results characterize the power of IPs ($\mathrm{IP}[\mathrm{poly}(n)] = \mathrm{PSPACE}$) [LFKN92; Sha92] and also achieve zero-knowledge [GMR89; GMW91].

The development of IPs, in turn, led to probabilistically checkable proofs (PCPs) [BFLS91; FGLSS96], where a probabilistic polynomial-time verifier has query access to a proof string. Here $\mathrm{PCP}[\mathsf{r}, \mathsf{q}]$ denotes the class of languages decidable by a PCP verifier that uses at most r bits of randomness and queries at most q bits of the proof string. A line of works culminated in the PCP Theorem [AS98; ALMSS98], which can be stated as $\mathrm{NP} = \mathrm{PCP}[O(\log n), O(1)]$; that is, every language in NP can be decided, with constant soundness error, by probabilistically examining only a constant number of bits in a polynomially long proof.

These advances in probabilistic proofs have reshaped theoretical computer science.

Interactive oracle proofs. More recently, researchers formulated *interactive oracle proofs* (IOPs) [BCS16; RRR16], a model of probabilistic proof that combines aspects of the IP and PCP models. A k-round IOP is a k-round IP where the verifier has PCP-like access to each prover message: the prover and verifier interact for k rounds, and after the interaction the verifier probabilistically reads a small number of bits from each prover message and decides to accept or reject based on the examined locations. The randomness used in the final phase is called *decision randomness* (which we distinguish from the random messages that the verifier sends to the prover during the interaction).

Recent work has constructed highly-efficient IOPs [BCGV16; Ben+17; BCGRS17; BBHR18; BCGGHJ17; BCRSVW19; BCGGRS19; BBHR19; BGKS19; COS20; RR20; BCG20; BCL20; BN21]. While the shortest PCPs known to date have quasi-linear length [BS08; Din07], IOPs can achieve linear proof length and fast provers. These developments are at the heart of recent constructions of non-interactive succinct arguments (SNARGs), and have facilitated their deployment in numerous real-world systems. IOPs are also used to construct IPs for delegating computation [RRR16].

IOPs beyond NP? Most research regarding IOPs has focused on understanding IOPs for languages in NP (and more generally various forms of non-deterministic computations) while using the additional rounds of interaction to achieve better efficiency compared to PCPs for those languages.

However, the power of IOPs for languages beyond NP is not well understood. We do know that IPs can express all languages in PSPACE for sufficiently large round complexity [LFKN92; Sha92]; moreover more rounds lead to more languages because, under plausible complexity assumptions, it holds that $\mathrm{IP}[\mathsf{k}] \not\subseteq \mathrm{IP}[o(\mathsf{k})]$ (while restricting to polynomial communication complexity) [GVW02]. But what can we say about the power of IOPs *with small query complexity (over the binary alphabet)*?[1]

Prior works imply certain facts about k-round IOPs for extreme settings of k.

- For non-deterministic languages, the answer is given by the PCP Theorem, which can be viewed as a "half-round" IOP with query complexity $O(1)$ and decision randomness $O(\log n)$.[2]

[1] An IP is an IOP where the verifier has large query complexity over the binary alphabet.

[2] The interaction solely consists of the prover sending a message, so we could write $\mathsf{k} = 0.5$.

- For languages that have a public-coin IP with $\mathsf{k} = 1$ round (a verifier message followed by a prover message), Drucker [Dru11a] proves a hardness of approximation result in the terminology of CSPs, which can be re-interpreted as an IOP where the verifier randomly checks a single constraint. That is, any public-coin one-round IP has a corresponding one-round IOP where the verifier sends a message, the prover sends a message, and the verifier to probabilistically reads $O(1)$ bits from both messages and decides (using $O(\log n)$ bits of decision randomness). However, Drucker's result does not extend to arbitrary many rounds.[3]
- When k can be polynomially large, we observe that constant-query IOPs for PSPACE can be obtained from [CFLS95; CFLS97],[4] which in turn provides such an IOP for every language having an IP. However, their result does not express the power of IOPs for a given number of rounds k. Other analogues of PCP have been given (e.g., [HRT07] applies to the polynomial hierarchy, [Dru11b] is also for PSPACE) but they do not seem to translate to IOPs.

In summary, not much is known about the power of general k-round IOPs, which leads us to ask:

What languages have a k-round IOP where the verifier decides by reading $O(1)$ bits from each prover message and each verifier message?

1.1 Our results

We answer the above question by showing that (informally) the power of IOPs with k rounds where the verifier reads $O(1)$ bits from communication round (both prover and verifier messages) is the same as if the verifier reads the entire protocol transcript (as in an IP). This can be seen as extending the PCP Theorem to interactive proofs, interpreted as "*you can be convinced by a conversation while barely listening (even to yourself)*".

To achieve this, our main result is a transformation from IPs to IOPs: we transform any IP into a corresponding IOP where the verifier reads $O(1)$ bits from each communication round and uses a total of $O(\log n)$ bits of decision randomness.[5] The round complexity is preserved, and other parameters are preserved up to polynomial factors. (A round is a verifier message followed by a prover message; after the interaction, the verifier's decision is probabilistic.)

Theorem 1 (IP → IOP)**.** *Let L be a language with a public-coin IP with k rounds and constant soundness error. Then L has an IOP with k rounds, constant soundness error, where the verifier decides by using $O(\log n)$ bits of decision randomness and reading $O(1)$ bits from each prover message and each verifier message. All other parameters are polynomially related.*

Comparison with known characterizations. Consider a language $L \in \mathrm{AM}[\mathsf{k}]$ for some (possibly non-constant) k. By definition, L has a public-coin k-round protocol, where the verifier reads the entire (polynomially long) transcript. On the other hand, since $\mathrm{AM}[\mathsf{k}] \subseteq \mathrm{NEXP}$, L has a PCP where the prover sends a single *exponentially-long* message from which the (polynomial-time) verifier reads $O(1)$ bits. But what can we say if we required the prover to send *polynomially long* messages? Using Theorem 1, we get the best of both solutions: k rounds and the verifier reads $O(1)$ bits from each round. See Figure 1 for a summary of this comparison.

[3] Round reduction [BM88] can reduce the number of rounds from any k to 1 with a blow-up in communication that is exponential in k. This does not work when k is super constant; see Section 2.2.3 for further discussion.

[4] Their result shows that PSPACE has what is known as a *probabilistically checkable debate system.* In their system, one prover plays a uniform random strategy. Thus one can naturally translate the debate system into an IOP.

[5] After the interaction, the verifier uses $O(\log n)$ random bits to decide which locations to read from all k rounds.

	complexity class	**model**	**proof length**	**alphabet**	**query complexity**	**round complexity**
[BGHSV05]	NEXP	PCP	$\exp(\|\mathbb{x}\|)$	$\{0,1\}$	$O(1)$	1
[CFLS97]	PSPACE	IOP	$\mathrm{poly}(\|\mathbb{x}\|)$	$\{0,1\}$	$O(1)$	$\mathrm{poly}(\|\mathbb{x}\|)$
implied by [BGHSV05]	AM[k]	PCP	$\exp(\|\mathbb{x}\|)$	$\{0,1\}$	$O(1)$	1
[Bab85; GMR89]	AM[k]	IP	k	$\{0,1\}^{\mathrm{poly}(\|\mathbb{x}\|)}$	1 per round	k
[this work]	AM[k]	IOP	$\mathrm{poly}(\|\mathbb{x}\|)$	$\{0,1\}$	$O(1)$ per round	k
[Dru11b]	AM	IOP	$\mathrm{poly}(\|\mathbb{x}\|)$	$\{0,1\}$	$O(1)$	1
[ALMSS98; AS98]	NP	PCP	$\mathrm{poly}(\|\mathbb{x}\|)$	$\{0,1\}$	$O(1)$	1

Figure 1: Classes captured by different types of probabilistic proofs (in the regime of constant soundness error). Here, $\mathbb{x}$ denotes the instance whose membership in the language the verifier is deciding. Here, AM stands for two-message public-coin protocols (a verifier random message followed by a prover message), and AM[k] is a k-round public-coin protocol.

1.1.1 Transformations for IOPs

As a corollary of Theorem 1, we get IOP analogues of classical IP theorems. We show IOP-to-IOP transformations, with small query complexity, and achieve classical results that were known for IPs, including: a private-coin to public-coin transformation (in the style of [GS86]); a round reduction technique (in the style of [BM88]); and a method to obtain perfect completeness (in the style of [FGMSZ89]). A graphic of this corollary is displayed in Figure 2.

Corollary 1. *Let L be a language with a k-round IOP with polynomial proof length over a binary alphabet. Then the following holds:*

1. **private-coins to public-coins:** *L has a $O(\mathsf{k})$-round public-coin IOP;*
2. **round reduction:** *for every constant $c \leq \mathsf{k}$, L has a k/c-round IOP;*
3. **perfect completeness:** *L has a perfectly complete k-round IOP.*

All resulting IOPs have polynomial proof length and $O(1)$ per-round query complexity over a binary alphabet; all other parameters are polynomially related to the original IOP.

Similar to the case with IPs, one can combine these transformations to get all properties at once. In particular, one can transform any IOP to be public-coin and have perfect completeness while preserving the round complexity.

1.1.2 Index-decodable PCPs

A building block behind Theorem 1 is a new notion of PCP that we call *index-decodable PCPs.* We discuss its definition and compare it with other notions of PCP in Section 2.3; here, we provide an intuitive description. An index-decodable PCP can be seen as a PCP on *maliciously encoded data.* The prover wishes to convince the verifier about a statement that regards k data segments $\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}]$ and an instance $\mathbb{x}$ (i.e., there exists a witness $\mathbb{w}$ such that $C(\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}], \mathbb{x}, \mathbb{w}) = 1$ for some circuit C) by providing a PCP string Π. The verifier receives as input only the instance $\mathbb{x}$, and

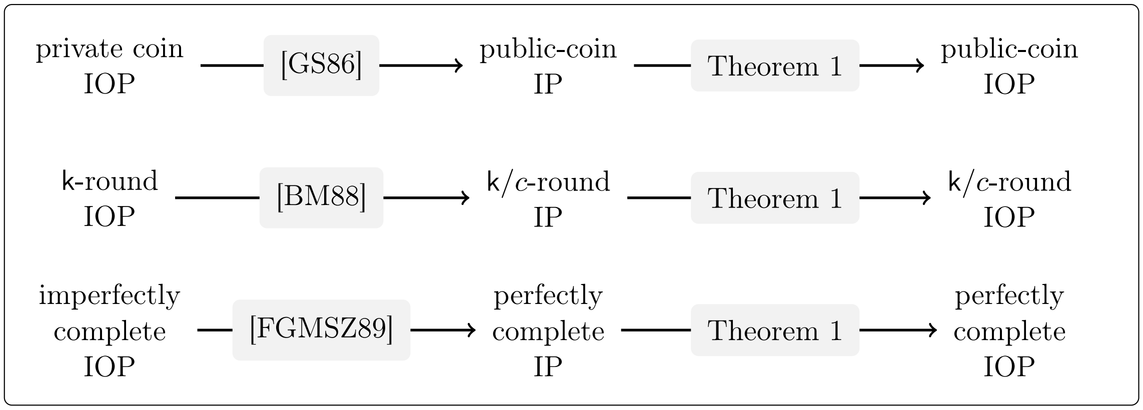

Figure 2: Using Theorem 1 to derive IOP analogues of classical IP theorems.

is given query access to an *encoding* of each data segment $\mathbb{i}[i]$ and query access to Π; this means that the verifier has query access to a total of $\mathsf{k}+1$ oracles.

The definition of an index-decodable PCP, to be useful, needs to take into account several subtle points (which, in fact, are crucial for our transformation in Theorem 1).

First, the encoding of each data segment must be computed independently of other data segments and even the instance. (Though the PCP string Π can depend on all data segments and the instance.)

Second, the verifier is not guaranteed that the k data oracles are valid encodings, in the sense that "security" is required to hold even against malicious provers that have full control of all $\mathsf{k}+1$ oracles (not just the PCP string oracle). In other words, we wish to formulate a security notion that is meaningful even for data that has been maliciously encoded.

The security notion that we use is *decodability*. Informally, we require that if the verifier accepts with high-enough probability a given set of (possibly malicious) data oracles and PCP string, then each data oracle can be individually decoded into a data segment such that, collectively, all the data segments and the instance form a true statement (there is a witness that makes the circuit accept them). We stress that the decoder algorithm must run on each data oracle separately from other data oracles and the instance (similarly as the encoder).

2 Techniques

We overview our ideas for transforming IPs into IOPs. We begin in Section 2.1 by describing a solution in [Dru11a] that works for a single round and explaining why it is challenging to extend it to work for multiple rounds. Then, we describe our transformation for many rounds in two steps. First, in Section 2.2 we describe how to make a verifier query each of its random messages at few locations. Next, in Section 2.3 we define our new notion of *index-decodable PCPs* and then in Section 2.4 describe how to use these to make the verifier query each prover message at few locations (without affecting the first step). We conclude with our construction of index-decodable PCPs: in Section 2.5 we outline a randomness-efficient index-decodable PCP that makes $O(1)$ queries to each of its oracles except for one; then in Section 2.6 we use proof composition to improve this query complexity.

Throughout, we call *interaction randomness* (or verifier random messages) the randomness sent by the verifier to the prover during the interaction, and *decision randomness* the randomness used by the verifier in the post-interaction decision stage.

2.1 The case of a single-round IP

The case of a single-round was settled by Drucker [Dru11a], whose work implies a transformation from a public-coin single-round IP to a single-round IOP where the verifier reads $O(1)$ bits from the communication transcript (here consisting of the prover message and the verifier message). His construction uses as building blocks the randomness-efficient amplification technique of [BGG90] and *PCPs of proximity* (PCPPs) [DR04; BGHSV06].[6] We give a high-level overview of his construction.

In a public-coin single-round IP, given a common input instance $\mathbb{x}$, the verifier $\mathbf{V}_{\mathsf{IP}}$ sends randomness ρ, the prover $\mathbf{P}_{\mathsf{IP}}$ sends a message a, and the verifier $\mathbf{V}_{\mathsf{IP}}$ decides whether to accept by applying a predicate to $(\mathbb{x}, \rho, a)$. Consider the non-deterministic machine $M_{\mathbb{x}}$ such that $M_{\mathbb{x}}(\rho) = 1$ if and only if there exists a such that $\mathbf{V}_{\mathsf{IP}}$ accepts $(\mathbb{x}, \rho, a)$. The constructed IOP works as follows:

1. the IOP verifier sends $\mathbf{V}_{\mathsf{IP}}$'s randomness ρ;
2. the IOP prover computes $\mathbf{P}_{\mathsf{IP}}$'s message a and produces a PCPP string Π for the claim "$M_{\mathbb{x}}(\rho) = 1$";
3. the IOP verifier checks Π using the PCPP verifier with explicit input $M_{\mathbb{x}}$ and implicit input ρ.

This IOP is sound if the underlying IP is "randomness-robust", which means that if $\mathbb{x}$ is not in the language then with high probability over ρ it holds that ρ is *far* from any accepting input for $M_{\mathbb{x}}$. Drucker achieves this property by using an amplification technique in [BGG90] that achieves soundness error $2^{-|\rho|}$ while using $O(|\rho|)$ random bits (standard amplification would, when starting with a constant-soundness protocol, result in $\omega(|\rho|)$ random bits). Thus, with high probability, ρ is not only a "good" random string (which holds for any single-round IP) but also is δ-far from any "bad" random string, for some small constant $\delta > 0$. This follows since the ball of radius δ around any bad random string has size $2^{\delta'|\rho|}$, for some small constant δ' that depends only on δ.

2.1.1 How to extend to multiple rounds?

We wish to obtain a similarly efficient transformation for a public-coin k-round IP where $\mathsf{k} = \mathrm{poly}(n)$.

[6] A PCPP is a PCP system where the verifier has oracle access to its input in addition to the prover's proof; the soundness guarantee is that if the input is *far* (in Hamming distance) from any input in the language, then the verifier accepts with small probability.

One possible approach would be to reduce the number of rounds of the given IP from k to 1 and then apply the transformation for single-round IPs. The round reduction of Babai and Moran [BM88] shows that any public-coin k-round IP can be transformed into a one-round IP where efficiency parameters grow by $n^{O(\mathsf{k})}$. This transformation, however, is not efficient for super-constant values of k. Moreover, it is undesirable even when k is constant because the transformation overhead is not a fixed polynomial (the exponent depends on k rather than being a fixed constant).

Therefore, we seek an approach that directly applies to a multi-round IP. Unfortunately, Drucker's approach for one-round IPs does not generalize to multiple-round IPs for several reasons. First, the corresponding machine $M_{\mathbb{x}}(\rho_1, \ldots, \rho_{\mathsf{k}})$ (which accepts if and only if there exist prover messages $a_1, \ldots, a_{\mathsf{k}}$ such that $\mathbf{V}_{\mathsf{IP}}$ accepts $(\mathbb{x}, \rho_1, a_1, \ldots, \rho_{\mathsf{k}}, a_{\mathsf{k}})$) does not capture the soundness of the interactive proof because it fails to capture interaction (a protocol may be sound according to the IP definition and, yet, for every $\mathbb{x}$ and $\rho_1, \ldots, \rho_{\mathsf{k}}$ it could be that $M_{\mathbb{x}}(\rho_1, \ldots, \rho_{\mathsf{k}}) = 1$). Moreover, it is not clear how to perform a randomness-efficient amplification for multiple rounds that makes the protocol sufficiently "randomness robust" for the use of a PCPP. The main reason is that to get soundness error 2^{-m} (as in [Dru11a]), the techniques of [BGG90] add $O(m)$ bits *per round*, which is too much when the protocol has many rounds (see Section 2.2.3 for a more detailed discussion on why this approach fails for many rounds).

We give a different solution that circumvents this step and works for any number of rounds. Our transformation from k-round IP to an IOP in two stages. In the first stage, we transform the IP into one in which the verifier reads only $O(1)$ bits from each random message it sends. In the second stage, we transform the IP into an IOP with $O(1)$ per-round query complexity, simultaneously for each prover message and each verifier message. We achieve this via a new notion of PCPs that we call *index-decodable PCPs*, and we describe in Section 2.3. First, we explain how to achieve the property that the verifier reads $O(1)$ bits from each of its random messages to the prover.

2.2 Local access to randomness

We transform a public-coin IP $(\mathbf{P}_{\mathsf{IP}}, \mathbf{V}_{\mathsf{IP}})$ into an IP $(\mathbf{P}'_{\mathsf{IP}}, \mathbf{V}'_{\mathsf{IP}})$ whose verifier (i) reads $O(1)$ bits from each of its random messages to the prover, and (ii) has logarithmic decision randomness (the randomness used by the verifier in the post-interaction decision stage). For now, the verifier reads in full every message received from the prover, and only later we discuss how to reduce the query complexity to prover messages while preserving the query complexity to the verifier random messages.

2.2.1 One-round public-coin proofs

In order to describe our ideas we begin with the simple case of one-round public-coin interactive proofs. Recall from Section 2.1 that this case is solved in [Dru11a], but we nevertheless first describe our alternative approach in this case and after that we will discuss the multiple-round case.

A strawman protocol. An idea would be for the prover to reply to the verifier with the received randomness, and the verifier to use this latter and test consistency with its own randomness. Given an instance $\mathbb{x}$: $\mathbf{V}'_{\mathsf{IP}}$ sends $\mathbf{V}_{\mathsf{IP}}$'s random message $\rho \in \{0,1\}^{\mathsf{r}}$; $\mathbf{P}'_{\mathsf{IP}}$ replies with $\rho' := \rho$ and the message $a := \mathbf{P}_{\mathsf{IP}}(\mathbb{x}, \rho)$; and $\mathbf{V}'_{\mathsf{IP}}$ checks that ρ and ρ' agree on a random location and that $\mathbf{V}_{\mathsf{IP}}(\mathbb{x}, \rho', a) = 1$.

This new IP is complete, and its verifier queries its random message at one location to conduct the consistency test. However, the protocol might not be sound, as we explain. Suppose that $\mathbb{x} \notin L$. Let r be the length of ρ, let β be the soundness error of the original IP, and let ν_{r} be the volume of the Hamming sphere of radius $\mathsf{r}/3$ in $\{0,1\}^{\mathsf{r}}$. A choice of verifier message ρ is *bad* if there exists a

such that $\mathbf{V}_{\mathsf{IP}}(\mathbb{x}, \rho, a) = 1$. By the soundness guarantee of $\mathbf{V}_{\mathsf{IP}}$, the fraction of bad choices of random verifier messages is at most β. A choice of verifier message ρ is *ball-bad* if there exist a bad ρ' that is 1/3-close to ρ. By the union bound, the fraction of ball-bad coins is at most $\gamma = \beta \cdot \nu_{\mathsf{r}}$.

Let E be the event over the choice of ρ that the prover sends ρ' that is 1/3-far from ρ.

- Conditioned on E occurring, $\mathbf{V}'_{\mathsf{IP}}$ rejects with probability at least 1/3 (whenever $\mathbf{V}'_{\mathsf{IP}}$ chooses a location on which ρ and ρ' disagree).
- Conditioned on E not occurring, $\mathbf{P}'_{\mathsf{IP}}$ cannot send any ρ' and a such that $\mathbf{V}_{\mathsf{IP}}(\mathbb{x}, \rho', a) = 1$ unless ρ is ball-bad, and so $\mathbf{V}'_{\mathsf{IP}}$ rejects with probability at least $1 - \gamma$.

Therefore, for the new IP to be sound, we need $\gamma = \beta \cdot \nu_{\mathsf{r}}$ to be small. Notice that $\nu_{\mathsf{r}} = 2^{c \cdot \mathsf{r}}$ (for some constant $0 < c < 1$) depends on r but not on β. Thus we need to achieve $\log 1/\beta > c \cdot \mathsf{r}$. As in Drucker's transformation, this can be done using the randomness-efficient soundness amplification of [BGG90], *but we deliberately take a different approach that will generalize for multiple rounds.*

Shrinking γ using extractors. Let Ext be an extractor with output length r, seed length $O(\log 1/\beta)$, and error β;[7] such extractors are constructed in [GUV09]. Suppose that the prover and verifier have access to a sample z from a source D with high min-entropy. Consider the following IP: $\mathbf{V}'_{\mathsf{IP}}$ sends s; $\mathbf{P}'_{\mathsf{IP}}$ replies with $s' := s$ and $a := \mathbf{P}_{\mathsf{IP}}(\mathbb{x}, \mathsf{Ext}(z, s'))$; $\mathbf{V}'_{\mathsf{IP}}$ checks that s and s' agree on a random location and that $\mathbf{V}_{\mathsf{IP}}(\mathbb{x}, \mathsf{Ext}(z, s'), a) = 1$.

At most a 2β-fraction of the seeds s are such that there exists a such that $\mathbf{V}_{\mathsf{IP}}(\mathbb{x}, \mathsf{Ext}(z, s), a) = 1$, because Ext is an extractor with error β and D is a distribution with high min-entropy. By an identical argument to the one done previously, either $\mathbf{P}'_{\mathsf{IP}}$ sends s' that is far from s and so $\mathbf{V}'_{\mathsf{IP}}$ rejects with constant probability, or $\mathbf{V}'_{\mathsf{IP}}$ rejects with probability at least $\gamma = 2\beta \cdot \nu_{\mathsf{r}'}$ where $\mathsf{r}' = |s| = O(\log 1/\beta)$. Thus we have that $\gamma = 2 \cdot \beta^{1-c}$, which is a constant fraction for small enough values of β (which can be achieved with standard parallel repetition).

Generating a source of high min-entropy. We describe how the prover and verifier can agree on a sample from a high-entropy source by leveraging the following observation: if z is a uniformly random string and z' is an arbitrary string that is close in Hamming distance to z, then z' has high min-entropy. Thus we can sample via similar ideas as above: $\mathbf{V}'_{\mathsf{IP}}$ samples and sends z; $\mathbf{P}'_{\mathsf{IP}}$ replies with $z' := z$; and $\mathbf{V}'_{\mathsf{IP}}$ checks that z and z' agree on a random location. (So $\mathbf{V}'_{\mathsf{IP}}$ reads one bit of its random message z.) If, with constant probability over z, $\mathbf{P}'_{\mathsf{IP}}$ sends z' that is far from z, then $\mathbf{V}'_{\mathsf{IP}}$ rejects with constant probability. Otherwise, we show that z' has high min-entropy because with high probability it agrees with z on most of its locations.

Putting it all together. Let $(\mathbf{P}_{\mathsf{IP}}, \mathbf{V}_{\mathsf{IP}})$ be a public-coin single-round IP with soundness β and randomness complexity r, and let Ext be an extractor with output length r, seed length $O(\log 1/\beta)$, and error β. The new IP $(\mathbf{P}'_{\mathsf{IP}}, \mathbf{V}'_{\mathsf{IP}})$ is as follows.

- *Sample high min-entropy source:* $\mathbf{V}'_{\mathsf{IP}}$ sends z and $\mathbf{P}'_{\mathsf{IP}}$ replies with $z' := z$.
- *Sample extractor seed:* $\mathbf{V}'_{\mathsf{IP}}$ sends s and $\mathbf{P}'_{\mathsf{IP}}$ replies with $s' := s$.
- *Prover message:* $\mathbf{P}'_{\mathsf{IP}}$ sends $a := \mathbf{P}_{\mathsf{IP}}(\mathbb{x}, \mathsf{Ext}(z', s'))$.
- *Verification:* $\mathbf{V}'_{\mathsf{IP}}$ checks that z and z' agree on a random location, s and s' agree on a random location, and $\mathbf{V}_{\mathsf{IP}}(\mathbb{x}, \mathsf{Ext}(z', s'), a) = 1$.

[7] A function $\mathsf{Ext}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is a (k, ε)-extractor if, for every random variable X over $\{0,1\}^n$ with min-entropy at least k, the statistical distance between $\mathsf{Ext}(X, U_d)$ and U_m is at most ε.

2.2.2 Extending to multiple rounds

In order to extend the previously described protocol to multiple rounds, we leverage the notion of *round-by-round soundness.* An IP for a language L has round-by-round soundness error β_{rbr} if there exists a "state" function such that: (i) for $\mathbb{x} \notin L$, the starting state is "doomed"; (ii) for every doomed state and next message that a cheating prover might send, with probability β_{rbr} over the verifier's next message, the protocol state will remain doomed; (iii) if at the end of interaction the state is doomed then the verifier rejects.

In the analysis of the one-round case there was an event (called *bad*) over the IP verifier's random message ρ such that if this event does not occur then the prover has no accepting strategy. This event can be replaced, in the round-by-round case, by the event that, in a given round, the verifier chooses randomness where the transcript remains doomed. This idea leads to a natural extension of the one-round protocol described in Section 2.2.1 to the multi-round case, which is our final protocol.

Let $(\mathbf{P}_{\mathsf{IP}}, \mathbf{V}_{\mathsf{IP}})$ be a public-coin k-round IP with round-by-round soundness β_{rbr} and randomness complexity r, and Ext an extractor with output length r, seed length $O(\log 1/\beta_{\text{rbr}})$, and error β_{rbr}.

- For each round $j \in [\mathsf{k}]$ of the original IP:
  1. *Sample high min-entropy source:* $\mathbf{V}'_{\mathsf{IP}}$ sends z_j and $\mathbf{P}'_{\mathsf{IP}}$ replies with $z'_j := z_j$.
  2. *Sample extractor seed:* $\mathbf{V}'_{\mathsf{IP}}$ sends s_j and $\mathbf{P}'_{\mathsf{IP}}$ replies with $s'_j := s_j$.
  3. *Prover message:* $\mathbf{P}'_{\mathsf{IP}}$ sends $a_j := \mathbf{P}_{\mathsf{IP}}(\mathbb{x}, \rho_1, \ldots, \rho_j)$ where $\rho_i := \mathsf{Ext}(z_i, s_i)$.
- $\mathbf{V}'_{\mathsf{IP}}$ accepts if and only if the following tests pass:
  1. Choose a random location and, for every $j \in [\mathsf{k}]$, test that z_j and z'_j agree on this location.
  2. Choose a random location and, for every $j \in [\mathsf{k}]$, test that s_j and s'_j agree on this location.
  3. For every $j \in [\mathsf{k}]$, compute $\rho_j := \mathsf{Ext}(z'_j, s'_j)$. Check that $\mathbf{V}_{\mathsf{IP}}(\mathbb{x}, \rho_1, a_1, \ldots, \rho_{\mathsf{k}}, a_{\mathsf{k}}) = 1$.

The soundness analysis of this protocol is similar to the one-round case. Suppose that $\mathbb{x} \notin L$. Then the empty transcript is "doomed". By an analysis similar to the one-round case, except where we set "bad" verifier messages to be ones where the transcript state switches from doomed to not doomed, if a round begins with a doomed transcript then except with probability $\gamma = 2 \cdot \beta_{\text{rbr}}^{1-c}$ (for some constant c) the transcript in the next round is also doomed. Thus, by a union bound, the probability that the transcript ends up doomed, and as a result the verifier rejects, is at least $1 - 2 \cdot \mathsf{k} \cdot \beta_{\text{rbr}}^{1-c}$. As shown in [CCHLRR18] round-by-round soundness error can be reduced via parallel repetition, albeit at a lower rate than regular soundness error. Thus, by doing enough parallel repetition before applying our transformation, the round-by-round soundness error β_{rbr} can be reduced enough so that the verifier rejects with constant probability.

The above protocol has $2\mathsf{k}_{\mathsf{IP}}$ rounds. The verifier reads 1 bit from each of its random messages, and has $O(\log |\mathbb{x}|)$ bits of decision randomness (to sample random locations for testing consistency between each z'_j and z_j and between each s'_j and s_j). To achieve k_{IP} rounds, we first apply the round reduction of [BM88] on the original IP to reduce to $\mathsf{k}_{\mathsf{IP}}/2$ rounds, and then apply our transformation.

2.2.3 Why randomness-efficient soundness amplification is insufficient

We briefly sketch why applying randomness-efficient soundness amplification in the style of [BGG90] is insufficient in the multi-round case, even if we were to consider round-by-round soundness. Recall that we wish for $\beta_{\text{rbr}} \cdot 2^{\Theta(\mathsf{r})}$ to be small, where β_{rbr} is the round-by-round soundness of the protocol and r is the number of random bits sent by the verifier in a single round. Bellare, Goldreich and Goldwasser [BGG90] show that, starting with constant soundness and randomness r, one can achieve soundness of 2^{-m} using $\mathsf{r}' = O(\mathsf{r} + m)$ random bits; they do this via m parallel repetitions where the randomness between repetitions is shared in a clever way. Using parallel repetition, achieving round-by-round soundness 2^{-m} requires m/k repetitions (see [CCHLRR18]). Thus, even if we were to show that the transformation of [BGG90] reduces round-by-round soundness at the same rate as standard parallel repetition (as it does for standard soundness), in order to get round-by-round soundness error 2^{-m}, we would need $\mathsf{r}' = O(\mathsf{r} + m \cdot \mathsf{k})$ bits of randomness. This would achieve $\beta_{\text{rbr}} \cdot 2^{\Theta(\mathsf{r}')} = 2^{-m} \cdot 2^{\Theta(\mathsf{r}+m\mathsf{k})}$, which, for super-constant values of k, is greater than 1 regardless of r.

2.3 Index-decodable PCPs

We introduce *index-decodable PCPs*, a notion of PCP that works on *multi-indexed relations*. A multi-indexed relation R is a set of tuples $(\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}], \mathbb{x}, \mathbb{w})$ where $(\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}])$ is the index vector, $\mathbb{x}$ the instance, and $\mathbb{w}$ the witness. As seen in the following definition, an index-decodable PCP treats the index vector $(\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}])$ and the instance $\mathbb{x}$ differently, which is why they are not "merged" into an instance $\mathbb{x}' = (\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}], \mathbb{x})$ (and why we do not consider standard relations).

Definition 1. *An* **index-decodable PCP** *for a multi-indexed relation* $R = \{ (\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}], \mathbb{x}, \mathbb{w}) \}$ *is a tuple of algorithms* $(\mathbf{I}_{\mathsf{PCP}}, \mathbf{P}_{\mathsf{PCP}}, \mathbf{V}_{\mathsf{PCP}}, \mathbf{D}_{\mathsf{PCP}})$*, where* $\mathbf{I}_{\mathsf{PCP}}$ *is the (honest) indexer,* $\mathbf{P}_{\mathsf{PCP}}$ *the (honest) prover,* $\mathbf{V}_{\mathsf{PCP}}$ *the verifier, and* $\mathbf{D}_{\mathsf{PCP}}$ *the decoder. The system has (perfect completeness and)* decodability bound κ_{PCP} *if the following conditions hold.*

- ***Completeness.*** *For every* $(\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}], \mathbb{x}, \mathbb{w}) \in R$*,*

$$\Pr_{\rho}\left[\mathbf{V}_{\mathsf{PCP}}^{\pi_1,\ldots,\pi_{\mathsf{k}},\Pi}(\mathbb{x};\rho) = 1 \;\middle|\; \begin{array}{r} \pi_1 \leftarrow \mathbf{I}_{\mathsf{PCP}}(\mathbb{i}[1]) \\ \vdots \\ \pi_{\mathsf{k}} \leftarrow \mathbf{I}_{\mathsf{PCP}}(\mathbb{i}[\mathsf{k}]) \\ \Pi \leftarrow \mathbf{P}_{\mathsf{PCP}}(\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}], \mathbb{x}, \mathbb{w}) \end{array} \right] = 1 \enspace .$$

- ***Decodability.*** *For every* $\mathbb{x}$*, malicious indexer proofs* $\tilde{\pi}_1, \ldots, \tilde{\pi}_{\mathsf{k}}$*, and malicious prover proof* $\tilde{\Pi}$*, if*

$$\Pr_{\rho}\left[\mathbf{V}_{\mathsf{PCP}}^{\tilde{\pi}_1,\ldots,\tilde{\pi}_{\mathsf{k}},\tilde{\Pi}}(\mathbb{x};\rho) = 1 \right] > \kappa_{\mathsf{PCP}}(|\mathbb{x}|)$$

then there exists $\mathbb{w}$ *such that* $\big(\mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_1), \ldots, \mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_{\mathsf{k}}), \mathbb{x}, \mathbb{w}\big) \in R$*.*

The indexer $\mathbf{I}_{\mathsf{PCP}}$ separately encodes each index, independent of indices and the instance, to obtain a corresponding *indexer proof.* The prover $\mathbf{P}_{\mathsf{PCP}}$ gets all the data as input (index vector, instance, and witness) and outputs a *prover proof.* The verifier $\mathbf{V}_{\mathsf{PCP}}$ gets the instance as input and has query access to $\mathsf{k}+1$ oracles (k indexer proofs and 1 prover proof), and outputs a bit.

The decodability condition warrants some discussion. The usual soundness condition of a PCP for a standard relation R has the following form: "if $\mathbf{V}_{\mathsf{PCP}}^{\tilde{\Pi}}(\mathbb{x})$ accepts with high-enough probability then there exists a witness $\mathbb{w}$ such that $(\mathbb{x}, \mathbb{w}) \in R$". For a multi-indexed relation it could be that for any given instance $\mathbb{x}$ there exist indexes $\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}]$ and a witness $\mathbb{w}$ such that $(\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}], \mathbb{x}, \mathbb{w}) \in R$. Since we do not trust the indexer's outputs, a soundness condition is not meaningful.

Instead, the decodability condition that we consider has the following form: "if $\mathbf{V}_{\mathsf{PCP}}^{\tilde{\pi}_1,\ldots,\tilde{\pi}_{\mathsf{k}},\tilde{\Pi}}(\mathbb{x})$ accepts with high-enough probability then there exists a witness $\mathbb{w}$ such that $(\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}], \mathbb{x}, \mathbb{w}) \in R$ where $\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}]$ are the decoded indices respectively found in $\tilde{\pi}_1, \ldots, \tilde{\pi}_{\mathsf{k}}$". It is crucial that the decoder receives as input the relevant indexer proof but not also the instance, or else the decodability condition would be trivially satisfied (the decoder could output the relevant index of the lexicographically first index vector putting the instance in the relation). This ensures that the proofs collectively convince the verifier not only that there exists an index vector and witness that place the instance in the relation, but that there exists a witness that, along with index vector obtainable from the index oracles via the decoder, places the instance in the relation.

We do not require the indexer or the decoder to be efficient. However, in some applications, it is useful to have an efficient indexer and decoder, and indeed we construct an index-decodable PCP with an efficient indexer and decoder.

Remark 2.1 (comparison with holography)**.** We compare index-decodable PCPs and holographic PCPs, which also work for indexed relations (see [CHMMVW20] and references therein). In both cases, an indexer produces an encoding of the index (independent of the instance). However, there are key differences between the two: (i) in an index-decodable PCP the indexer works separately on each entry of the index vector, while in a holographic PCP there is a single index; moreover, (ii) in a holographic PCP the indexer is trusted in the sense that security is required to hold only when the verifier has oracle access to the honest indexer's output, but in an index-decodable PCP, the indexer is **not trusted** in the sense that the cheating prover can choose encodings for all of the indices. *Both differences are essential properties for our transformation of IPs into IOPs.*

Our main technical result is an index-decodable PCPs with $O(1)$ query complexity per oracle.

Theorem 2. *Any multi-indexed relation $R = \{(\mathbb{i}[1], \dots, \mathbb{i}[\mathsf{k}], \mathbb{x}, \mathbb{w})\}$ to which membership can be verified in polynomial time has a non-adaptive index-decodable PCP with the following parameters:*

Index-Decodable PCP for $(\mathbb{i}[1], \dots, \mathbb{i}[\mathsf{k}], \mathbb{x}, \mathbb{w}) \in R$	
Indexer proof length (per proof)	$O(\lvert\mathbb{i}[i]\rvert)$
Prover proof length	$\mathrm{poly}(\lvert\mathbb{x}\rvert)$
Alphabet size	2
Queries per oracle	$O(1)$
Randomness	$O(\log\lvert\mathbb{x}\rvert)$
Decodability bound	$O(1)$
Indexer running time	$\tilde{O}(\lvert\mathbb{i}[i]\rvert)$
Prover running time	$\mathrm{poly}(\lvert\mathbb{x}\rvert)$
Verifier running time	$\mathrm{poly}(\lvert\mathbb{x}\rvert)$
Decoder running time	$\tilde{O}(\lvert\mathbb{i}[i]\rvert)$

Our construction achieves optimal parameters similar to the PCP theorem: it has $O(1)$ query complexity (per oracle) over a binary alphabet, and the randomness complexity is logarithmic, *independent* of the number of indexes k. Achieving small randomness complexity is challenging and useful for applications. First, it facilitates proof composition (where a prover writes a proof for every possible random string), which is common when constructing zero-knowledge PCPs (e.g., [IW14]). Second, small randomness complexity is necessary for hardness of approximation results.

A similar notion is (implicitly) considered in [ALMSS98] but their construction does not achieve the parameters we obtain in Theorem 2 (most crucially, they do not achieve small randomness).

2.4 Local access to prover messages

We show how to transform an IP into an IOP by eliminating the need of the verifier to read more than a few bits of each prover message. This transformation preserves the number of bits read by the verifier to its own interaction randomness. Thus, combining it with the transformation described in Section 2.2, this completes the proof (overview) of Theorem 1.

We transform any public-coin IP into an IOP by using an index-decodable PCP. In a public-coin k-round IP, the prover $\mathbf{P}_{\mathsf{IP}}$ and verifier $\mathbf{V}_{\mathsf{IP}}$ receive as input an instance $\mathbb{x}$ and then, in each round i, the verifier $\mathbf{V}_{\mathsf{IP}}$ sends randomness ρ_i and the prover replies with a message $a_i \leftarrow \mathbf{P}_{\mathsf{IP}}(\mathbb{x}, \rho_1, \ldots, \rho_i)$; after the interaction, the verifier $\mathbf{V}_{\mathsf{IP}}$ runs an efficient probabilistic algorithm with decision randomness ρ_{dc} on the transcript $(\mathbb{x}, \rho_1, a_1, \ldots, \rho_{\mathsf{k}}, a_{\mathsf{k}})$ to decide whether to accept or reject.

The IP verifier $\mathbf{V}_{\mathsf{IP}}$ defines a multi-indexed relation $R(\mathbf{V}_{\mathsf{IP}})$ consisting of tuples

$$\Big(\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}], \mathbb{x}, \mathbb{w}\Big) = \Big(a_1, \ldots, a_{\mathsf{k}}, (\mathbb{x}, \rho_1, \ldots, \rho_{\mathsf{k}}, \rho_{\mathsf{dc}}), \bot\Big)$$

such that the IP verifier $\mathbf{V}_{\mathsf{IP}}$ accepts the instance $\mathbb{x}$, transcript $(\rho_1, a_1, \ldots, \rho_{\mathsf{k}}, a_{\mathsf{k}})$, and decision randomness ρ_{dc}. (Here we do not rely on witnesses although the definition of index-decodable PCPs supports this.)

From IP to IOP. Let $(\mathbf{I}_{\mathsf{PCP}}, \mathbf{P}_{\mathsf{PCP}}, \mathbf{V}_{\mathsf{PCP}}, \mathbf{D}_{\mathsf{PCP}})$ be an index-decodable PCP for the relation $R(\mathbf{V}_{\mathsf{IP}})$. We construct the IOP as follows. The IOP prover and IOP verifier receive an instance $\mathbb{x}$. In round $i \in [\mathsf{k}]$, the IOP verifier sends randomness ρ_i (just like the IP verifier $\mathbf{V}_{\mathsf{IP}}$) and the (honest) IOP prover sends the indexer proof $\pi_i := \mathbf{I}_{\mathsf{PCP}}(a_i)$ where $a_i \leftarrow \mathbf{P}_{\mathsf{IP}}(\mathbb{x}, \rho_1, \ldots, \rho_i)$. In a final additional message (which can be sent at the same time as the last indexer proof π_{k}), the IOP prover sends $\Pi := \{\Pi_{\rho_{\mathsf{dc}}}\}_{\rho_{\mathsf{dc}}}$ where, for every possible choice of decision randomness ρ_{dc}, $\Pi_{\rho_{\mathsf{dc}}}$ is an index-decodable PCP prover proof to the fact that $\big(a_1, \ldots, a_{\mathsf{k}}, (\mathbb{x}, \rho_1, \ldots, \rho_{\mathsf{k}}, \rho_{\mathsf{dc}}), \bot\big) \in R(\mathbf{V}_{\mathsf{IP}})$. After the interaction, the IOP verifier samples IP decision randomness ρ_{dc} and checks that $\mathbf{V}_{\mathsf{PCP}}^{\tilde{\pi}_1, \ldots, \tilde{\pi}_{\mathsf{k}}, \tilde{\Pi}}\big((\mathbb{x}, \rho_1, \ldots, \rho_{\mathsf{k}}, \rho_{\mathsf{dc}})\big) = 1$.

Proof sketch. Completeness follows straightforwardly from the construction. We now sketch a proof of soundness. Letting L be the language decided by $(\mathbf{P}_{\mathsf{IP}}, \mathbf{V}_{\mathsf{IP}})$, fix an instance $\mathbb{x} \notin L$ and a malicious IOP prover $\tilde{\mathbf{P}}_{\mathsf{IOP}}$. Given interaction randomness $\rho_1, \ldots, \rho_{\mathsf{k}}$, consider the messages $\tilde{\pi}_1, \ldots, \tilde{\pi}_{\mathsf{k}}$ output by $\tilde{\mathbf{P}}_{\mathsf{IOP}}$ in the relevant rounds ($\tilde{\pi}_i$ depends on $\rho_1, \ldots, \rho_i$) and the message $\tilde{\Pi} = \{\tilde{\Pi}_{\rho_{\mathsf{dc}}}\}_{\rho_{\mathsf{dc}}}$ output by $\tilde{\mathbf{P}}_{\mathsf{IOP}}$ in the last round (this message depends on $\rho_1, \ldots, \rho_{\mathsf{k}}$). We consider two complementary options of events over the IOP verifier's randomness $(\rho_1, \ldots, \rho_{\mathsf{k}}, \rho_{\mathsf{dc}})$.

1. With high probability the proofs $\tilde{\pi}_1, \ldots, \tilde{\pi}_{\mathsf{k}}$ and $\tilde{\Pi}_{\rho_{\mathsf{dc}}}$ generated while interacting with $\tilde{\mathbf{P}}_{\mathsf{IOP}}$ using randomness $\rho_1, \ldots, \rho_{\mathsf{k}}$ and ρ_{dc} are such that

$$\Big(\mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_1), \ldots, \mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_{\mathsf{k}}), (\mathbb{x}, \rho_1, \ldots, \rho_{\mathsf{k}}, \rho_{\mathsf{dc}}), \bot\Big) \notin R(\mathbf{V}_{\mathsf{IP}}) \ .$$

   If this is true, then, by the decodability property of the index-decodable PCP, the IOP verifier must reject with high probability over the choice of randomness for $\mathbf{V}_{\mathsf{PCP}}$.

2. With high probability the proofs $\tilde{\pi}_1, \ldots, \tilde{\pi}_{\mathsf{k}}$ and $\tilde{\Pi}_{\rho_{\mathsf{dc}}}$ generated while interacting with $\tilde{\mathbf{P}}_{\mathsf{IOP}}$ using randomness $\rho_1, \ldots, \rho_{\mathsf{k}}$ and ρ_{dc} are such that

$$\Big(\mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_1), \ldots, \mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_{\mathsf{k}}), (\mathbb{x}, \rho_1, \ldots, \rho_{\mathsf{k}}, \rho_{\mathsf{dc}}), \bot\Big) \in R(\mathbf{V}_{\mathsf{IP}}) \ .$$

   We prove that this case cannot occur by showing that it contradicts the soundness of the original IP. Suppose towards contradiction that the above is true. We use $\tilde{\mathbf{P}}_{\mathsf{IOP}}$ and the decoder of the index-decodable PCP, $\mathbf{D}_{\mathsf{PCP}}$, to construct a malicious IP prover for the original IP as follows.

In round i, the transcript $(\rho_1, a_1, \dots, \rho_{i-1}, a_{i-1})$ has already been set during previous interaction. The IP verifier sends randomness ρ_i. The IP prover sends $a_i := \mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_i)$ to the IP verifier, where $\tilde{\pi}_i := \tilde{\mathbf{P}}_{\mathsf{IOP}}(\rho_1, \dots, \rho_i)$. Recall that $\big(\mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_1), \dots, \mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_{\mathsf{k}}), (\mathbb{x}, \rho_1, \dots, \rho_{\mathsf{k}}, \rho_{\mathsf{dc}}), \bot\big) \in R(\mathbf{V}_{\mathsf{IP}})$ if and only if the IP verifier accepts given instance $\mathbb{x}$, randomness $(\rho_1, \dots, \rho_{\mathsf{k}}, \rho_{\mathsf{dc}})$, and prover messages $\mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_1), \dots, \mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_{\mathsf{k}})$, which is precisely what the IP prover supplies it with. Since the event that $\big(\mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_1), \dots, \mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_{\mathsf{k}}), (\mathbb{x}, \rho_1, \dots, \rho_{\mathsf{k}}, \rho_{\mathsf{dc}}), \bot\big) \in R(\mathbf{V}_{\mathsf{IP}})$ happens with high probability, this implies that with high probability the IP verifier will accept, contradicting soundness of the original IP. Here we crucially used the fact that the decoder $\mathbf{D}_{\mathsf{PCP}}$ does not depend on the instance of the index-decodable PCP (which consists of $\mathbb{x}$ and all of the IP verifier's randomness $\rho_1, \dots, \rho_{\mathsf{k}}, \rho_{\mathsf{dc}}$) or on the other indexer messages.

The resulting IOP has k rounds, exactly as in the original IP. The IOP verifier uses as much randomness as the original IP verifier with the addition of the randomness used by the index-decodable PCP. The query complexity is that of the underlying verifier of the index-decodable PCP. The proof length and alphabet are the same as those of the index-decodable PCP.

Preserving local access to randomness. The transformation described above can be modified to preserve the query complexity of the verifier to its own interaction randomness if the verifier is non-adaptive with respect to its queries to its random messages (i.e., the choice of bits that it reads depends only on $\mathbb{x}$ and ρ_{dc}). We can redefine the multi-indexed relation $R(\mathbf{V}_{\mathsf{IP}})$ to have as explicit inputs the instance $\mathbb{x}$, decision randomness ρ_{dc}, and the bits of $\rho_1, \dots, \rho_{\mathsf{k}}$ that the verifier needs to read to decide whether to accept or reject (rather than the entire interaction randomness strings). In more detail, suppose that the verifier reads q bits from its own interaction randomness. Then the new multi-indexed relation consists of tuples:

$$\Big(\mathbb{i}[1], \dots, \mathbb{i}[\mathsf{k}], \mathbb{x}, \mathbb{w}\Big) = \Big(a_1, \dots, a_{\mathsf{k}}, (\mathbb{x}, b_1, \dots, b_{\mathsf{q}}, \rho_{\mathsf{dc}}), \bot\Big)$$

such that given decision randomness ρ_{dc} the IP verifier $\mathbf{V}_{\mathsf{IP}}$ accepts given instance $\mathbb{x}$, decision randomness ρ_{dc}, prover messages $(a_1, \dots, a_{\mathsf{k}})$, and $(b_1, \dots, b_{\mathsf{q}})$ as answers to its q queries to $\rho_1, \dots, \rho_{\mathsf{k}}$.

Given a multi-indexed PCP for this relation, the IP to IOP transformation is identical to the one described above, except that after the interaction, the IOP verifier samples IP decision randomness, queries its own interaction randomness to get answers $b_1, \dots, b_{\mathsf{q}}$, and these replace $\rho_1, \dots, \rho_{\mathsf{k}}$ as explicit inputs to the index-decodable PCP verifier $\mathbf{V}_{\mathsf{PCP}}$.

2.5 Constructing index-decodable PCPs

We outline a construction of an index-decodable PCP with $O(1)$ query complexity to each indexer proof and to the prover proof, and where the prover proof is over a large alphabet (of size 2^{k}). Later on, in Section 2.6, we compose this index-decodable PCP with an inner PCP to reduce the alphabet of the prover proof to binary. To achieve this composition while preserving polynomial proof length, here we additionally require that the verifier has logarithmic randomness complexity.

Building blocks. In our construction we rely on variants of PCPPs. Recall that a PCPP is a PCP system where the verifier has oracle access to its input in addition to the prover's proof; the soundness guarantee is that if the input is *far* (in Hamming distance) from any input in the language, then the verifier accepts with small probability.

We use PCPPs that are *multi-input* and *oblivious*. We explain each of these properties.

- A PCPP is multi-input if the verifier has oracle access to multiple (oracle) inputs. The soundness guarantee is that, for every vector of inputs that satisfy the circuit in question, if at least one input oracle is far from the respective satisfying input, then the verifier accepts with small probability.
- A (non-adaptive) PCPP is oblivious for a circuit family $\mathcal{C} = \{C_i\}_{i \in [\mathsf{k}]}$ if the queries made by the verifier to its oracles depend only on $\mathcal{C}$ and its randomness. In particular they do not depend on i. This property will be used later to facilitate bundling queries. We will have k PCPs, each with a different C_i, but the verifier will use the same randomness in each test. Since the PCPPs are oblivious, this means that the verifier makes the same queries for every test. Thus we can group together the k proofs into a single proof with larger alphabet and maintain good query complexity on this proof. This property is important in order to achieve our final parameters.

See Section 5.1 for definitions for the above notions, and how to obtain them from standard PCPPs. Henceforth, all PCPPs that we use will be over the binary alphabet and have constant proximity, constant soundness error, constant query complexity, and logarithmic randomness complexity.

The construction. We construct an index-decodable PCP for a multi-indexed relation $R = \{(\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}], \mathbb{x}, \mathbb{w})\}$ whose membership can be verified efficiently.

The indexer encodes each index via an error-correcting code with (constant) relative distance greater than the (constant) proximity parameter of the PCPP used later. The prover uses PCPPs to prove that there exist indexes and a witness that put the given instance in the relation and adds consistency checks to prove that the indices are consistent with those encoded by the indexer. The verifier checks each of these claims. The decoder decodes the indexer proofs using the same code.

In slightly more detail, the index-decodable PCP is as follows.

- $\mathbf{I}_{\mathsf{PCP}}(\mathbb{i}[i])$: Encode the index $\mathbb{i}[i]$ as π_i using an error-correcting code.
- $\mathbf{P}_{\mathsf{PCP}}(\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}], \mathbb{x}, \mathbb{w})$:
  1. *Encoding the indexes*: Compute Π_*, an encoding of the string $(\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}], \mathbb{w})$.
  2. *Membership of encoding*: Compute a PCPP string Π_{mem} for the claim that $C_{*,\mathbb{x}}(\Pi_*) = 1$ where $C_{*,\mathbb{x}}$ checks that Π_* is a valid encoding of indexes and a witness that put $\mathbb{x}$ in R.
  3. *Consistency of encoding*: For every $i \in [\mathsf{k}]$, compute a PCPP string Π_i for the claim that $C_i(\pi_i, \Pi_*) = 1$ where C_i checks that π_i and Π_* are valid encodings and that the string $\mathbb{i}[i]$ encoded within π_i is equal to the matching string encoded within Π_*.
  4. Output $(\Pi_*, \Pi_{\mathsf{mem}}, \Pi_{\mathbb{i}})$ where $\Pi_{\mathbb{i}}$ are the proofs $\Pi_1, \ldots, \Pi_{\mathsf{k}}$ "bundled" together into symbols of k bits such that $\Pi_{\mathbb{i}}[q] = (\Pi_1[q], \ldots, \Pi_{\mathsf{k}}[q])$.
- $\mathbf{V}_{\mathsf{PCP}}^{\tilde{\pi}_1,\ldots,\tilde{\pi}_{\mathsf{k}},(\tilde{\Pi}_*,\tilde{\Pi}_{\mathsf{mem}},\tilde{\Pi}_{\mathbb{i}})}(\mathbb{x})$: Check that all the tests below pass.
  1. *Membership*: Run the PCPP verifier on the claim that $C_{*,\mathbb{x}}(\tilde{\Pi}_*) = 1$ using proof oracle $\tilde{\Pi}_{\mathsf{mem}}$.
  2. *Consistency*: For every $i \in [\mathsf{k}]$, run the PCPP verifier on the claim that $C_i(\tilde{\pi}_i, \tilde{\Pi}_*) = 1$ using proof oracle $\tilde{\Pi}_i$. These k tests are run *with the same randomness*. Since the PCPP is oblivious and randomness is shared, the queries made by the PCPP verifier in each test are identical, and so each query can be made by reading the appropriate k-bit symbols from $\tilde{\Pi}_{\mathbb{i}}$.
- $\mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_i)$: Output the codeword closest to $\tilde{\pi}_i$ (in the error-correcting code).

Completeness follows straightforwardly from the construction. We now sketch decodability.

Decodability. Fix an instance $\mathbb{x}$, indexer proofs $\tilde{\pi}_1, \ldots, \tilde{\pi}_{\mathsf{k}}$, and prover proof $(\tilde{\Pi}_*, \tilde{\Pi}_{\mathsf{mem}}, \tilde{\Pi}_{\mathbb{i}})$. Suppose that the verifier accepts with high-enough probability. We argue that this implies that there exists

$\mathbb{w}$ such that $\big(\mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_1), \ldots, \mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_{\mathsf{k}}), \mathbb{x}, \mathbb{w}\big) \in R$. Specifically, we argue that $\tilde{\Pi}_*$ encodes indices $\tilde{\mathbb{i}}[1], \ldots, \tilde{\mathbb{i}}[\mathsf{k}]$ and witness $\tilde{\mathbb{w}}$ that place $\mathbb{x}$ in R and, additionally, each $\tilde{\pi}_i$ is an encoding of $\tilde{\mathbb{i}}[i]$. This completes the proof of decodability because $\mathbf{D}_{\mathsf{PCP}}$ decodes each $\tilde{\pi}_i$ to $\tilde{\mathbb{i}}[i]$, and these strings together with $\tilde{\mathbb{w}}$ put $\mathbb{x}$ in the multi-indexed relation R.

Let δ_{PCPP} be the PCPP's proximity and δ_{ECC} the code's (relative) distance; recall that $\delta_{\mathsf{PCPP}} \leq \delta_{\mathsf{ECC}}$.

- *Membership*: We claim that there exist strings $\tilde{\mathbb{i}}[1], \ldots, \tilde{\mathbb{i}}[\mathsf{k}]$ and $\tilde{\mathbb{w}}$ that place $\mathbb{x}$ in R and whose encoding has Hamming distance at most δ_{PCPP} from $\tilde{\Pi}_*$; since $\delta_{\mathsf{PCPP}} \leq \delta_{\mathsf{ECC}}$, this implies that $\tilde{\Pi}_*$ decodes to $(\tilde{\mathbb{i}}[1], \ldots, \tilde{\mathbb{i}}[\mathsf{k}], \tilde{\mathbb{w}})$. Suppose towards contradiction that there are no such strings. In other words, for every codeword $\widehat{\Pi}_*$ that is close in Hamming distance to $\tilde{\Pi}_*$ we have that $C_{*,\mathbb{x}}(\widehat{\Pi}_*) = 0$. As a result the PCPP verifier must reject with high probability, which contradicts our assumption that $\mathbf{V}_{\mathsf{PCP}}$ (which runs the PCPP verifier) *accepts* with high probability.

- *Consistency*: We claim that there exist strings $\tilde{\mathbb{i}}[1], \ldots, \tilde{\mathbb{i}}[\mathsf{k}]$ and $\tilde{\mathbb{w}}$ such that their collective encoding is close to $\tilde{\Pi}_*$ and that, for every $i \in [\mathsf{k}]$, $\tilde{\pi}_i$ is close to the encoding of $\mathbb{i}_*[i]$. As before, since the proximity parameter of the PCPP is smaller than the distance of the code, this implies that $\tilde{\Pi}_*$ decodes to $(\tilde{\mathbb{i}}[1], \ldots, \tilde{\mathbb{i}}[\mathsf{k}], \tilde{\mathbb{w}})$ and that $\tilde{\pi}_i$ decodes to $\tilde{\mathbb{i}}[i]$. Suppose towards contradiction that for some $i \in [\mathsf{k}]$ the above condition does not hold: for every $\widehat{\pi}_i$ and $\widehat{\Pi}$ such that $\widehat{\pi}_i$ is close to $\tilde{\pi}_i$ and $\widehat{\Pi}_*$ is close to $\tilde{\Pi}_*$ it holds that $C_i(\widehat{\pi}_i, \widehat{\Pi}_*) = 0$. By the soundness of the (*multi-input*) PCPP, the PCPP verifier must reject with high probability, which contradicts our assumption that $\mathbf{V}_{\mathsf{PCP}}$ (which runs the PCPP verifier) *accepts* with high probability.

Complexity measures. The above construction is an index-decodable PCP with polynomial-length proofs and where the verifier makes $O(1)$ queries to each indexer proof and makes $O(1)$ queries to the prover proof. Moreover, the prover proof has alphabet size 2^{k} since the prover bundles the PCPP consistency test proofs into k-bit symbols; this bundling is possible because the verifier shares randomness between all of the (oblivious) PCPPs in the consistency test. Since the PCPPs are oblivious to the index i, and they share randomness, they all must make the same queries to their oracles. The verifier uses $O(\log |\mathbb{x}|)$ bits of randomness: $O(\log |\mathbb{x}|)$ for the membership test, and $O(\log |\mathbb{x}|)$ for all k consistency test (which all share the same randomness).

2.6 Achieving constant query complexity

We describe how to achieve an index-decodable PCP with constant query complexity per proof over the binary alphabet. The main tool is proof composition, which we review in Section 2.6.1. Then in Section 2.6.2, we define and construct a robust variant of index-decodable PCPs, which we use as the outer PCP in proof composition.

2.6.1 Proof composition

Proof composition is a technique to lower the query complexity of PCPs [AS98] and IOPs [BCGRS17]. In proof composition, an "inner" PCP is used to prove that a random execution of the "outer" PCP would have accepted. The inner PCP needs to be a PCPP. Recall that a PCPP is a PCP system where the verifier has oracle access to its input in addition to the prover's proof, and the soundness guarantee is that if the input is *far* from any input in the language, then the verifier accepts with small probability. To match this, the outer PCP must be *robust*, which means that the soundness guarantee ensures that when the instance is not in the language then not only is a random local view of the verifier rejecting but it is also far (in Hamming distance) from any accepting local view.

Typically the robust outer PCP has small proof length but large query complexity, while the inner PCPP has small query complexity but possibly a large proof length. Composition yields a PCP with small query complexity and small proof length.

We observe that proof composition *preserves decodability* (see Section 6.1): if the outer PCP in the composition is index-decodable, then the composed PCP is index-decodable. This is because the composition operation does not change the outer PCP proof and only adds a verification layer to show that the outer verifier accepts.

We thus apply proof composition as follows: the outer PCP is a robust variant of the index-decodable PCP from Section 2.5 that we discuss in Section 2.6.2; and the inner PCP is a standard PCPP with polynomial proof length. This will complete the proof sketch of Theorem 2.

2.6.2 Robust index-decodable PCPs

Our goal is to perform proof composition where the outer PCP is index-decodable. As mentioned above, this requires the PCP to be robust. Our starting point is the index-decodable PCP from Section 2.5. This PCP does have large query complexity over the binary alphabet ($O(\mathsf{k})$ queries to the prover proof). However, the fact that its queries to the prover proof are already bundled into a constant number of locations over an alphabet of size 2^{k} implies that we do not have to worry about a "generic" query bundling step and instead only have to perform a (tailored) robustification step prior to composition. Accordingly, the robustness definition below focuses on the prover proof, and so is the corresponding construction described after.

Definition 2. *A non-adaptive*[8] *index-decodable PCP* $(\mathbf{I}_{\mathsf{PCP}}, \mathbf{P}_{\mathsf{PCP}}, (\mathbf{V}^{\mathsf{qry}}_{\mathsf{PCP}}, \mathbf{V}^{\mathsf{dc}}_{\mathsf{PCP}}), \mathbf{D}_{\mathsf{PCP}})$ *for a multi-indexed relation* R *is* **prover-robustly index-decodable** *with decodability bound* κ_{PCP} *and robustness* σ_{PCP} *if for every* $\mathbb{x}$ *and proofs* $\tilde{\mathbf{\Pi}}_{\mathbb{i}} = (\tilde{\pi}_1, \ldots, \tilde{\pi}_{\mathsf{k}})$ *and* $\tilde{\Pi}$ *if*

$$\Pr_{\rho}\left[\exists A' \text{ s.t. } \mathbf{V}^{\mathsf{dc}}_{\mathsf{PCP}}(\mathbb{x}, \rho, \tilde{\mathbf{\Pi}}_{\mathbb{i}}[Q_{\mathbb{i}}], A') = 1 \ \wedge\ \Delta(A', A) \le \sigma_{\mathsf{PCP}}(|\mathbb{x}|) \;\middle|\; \begin{array}{l} (Q_{\mathbb{i}}, Q_*) \leftarrow \mathbf{V}^{\mathsf{qry}}_{\mathsf{PCP}}(\mathbb{x}, \rho) \\ A := \{\, \tilde{\Pi}[q] \mid q \in Q_* \,\} \end{array} \right] > \kappa_{\mathsf{PCP}}(|\mathbb{x}|)$$

then there exists $\mathbb{w}$ *such that* $\big(\mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_1), \ldots, \mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_{\mathsf{k}}), \mathbb{x}, \mathbb{w}\big) \in R$. *Above* $Q_{\mathbb{i}}$ *and* Q_* *are the queries made to the indexer proofs the prover proof respectively and* $\Delta(A', A)$ *is the relative distance between* A' *and* A.

In other words, if there is no witness $\mathbb{w}$ such that $\big(\mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_1), \ldots, \mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_{\mathsf{k}}), \mathbb{x}, \mathbb{w}\big) \in R$ then with high probability not only will the verifier reject but also any set of answers *from the prover proof* that are close in Hamming distance to the real set of answers will also be rejecting.

We now outline how we transform the index-decodable PCP constructed in Section 2.5 into a *robust* index-decodable PCP. The techniques follow the robustification step in [BGHSV06]. The transformation preserves the verifier's randomness complexity $O(\log |\mathbb{x}|)$, which facilitates using this modified PCP as the outer PCP in proof composition.

Robustification. We apply an error-correcting code separately to each symbol of the prover proof. When the verifier wants to read a symbol from this proof, it reads the codeword encoding the symbol, decodes it, and then continues. It reads the indexer proofs as in the original PCP. This makes the

[8] A PCP verifier is non-adaptive if it can be split into two algorithms: $\mathbf{V}^{\mathsf{qry}}_{\mathsf{PCP}}$ chooses which locations to query without accessing its oracles; and $\mathbf{V}^{\mathsf{dc}}_{\mathsf{PCP}}$ receives the results of the queries and decides whether to accept or reject.

PCP robust because if a few bits of the codeword representing a symbol are corrupted, then it will still be decoded to the same value. The robustness, however, degrades with the number of queries. If the relative distance of the error-correcting code is δ and the original verifier reads q symbols from the prover proof, then the resulting PCP will have robustness $O(\delta/q)$.

Indeed, let $c_1, \ldots, c_q$ be the codewords read by the new PCP verifier from the prover proof, and let $a_1, \ldots, a_q$ be such that a_i is the decoding of c_i. In order to change the decoding into some other set of strings $a'_1, \ldots, a'_q$ that, when received by the verifier, may induce a different decision than $a_1, \ldots, a_q$, it suffices (in the worst case) to change a single codeword to decode to a different value. Since the relative distance of the code is δ, to do this, one must change at least a δ-fraction of the bits of a single codeword, c_i. A δ-fraction of a single codeword is a δ/q-fraction of the whole string of q codewords, $c_1, \ldots, c_q$.

In sum, to achieve constant robustness, *we need to begin with an index-decodable PCP with a small number of queries to the prover proof*, but possibly with a large alphabet. It is for this reason that we required this property in Section 2.5.

3 Preliminaries

3.1 Relative distance

Let $f, g \colon \Sigma_1 \to \Sigma_2$ be functions. The relative distance between f and g, denoted by $\Delta(f, g)$ is equal to the relative number of locations in which f and g disagree:

$$\Delta(f, g) = \frac{|\{x \in \Sigma_1 \mid f(x) \neq g(x)\}|}{|\Sigma_1|} \ .$$

We say that f and g are δ-far if $\Delta(f, g) > \delta$, and if $\Delta(f, g) \leq \delta$ then the functions are δ-close.

Similarly, the relative distance between two strings $x, y \in \Sigma^m$ is the relative distance between the functions $f, g \colon [m] \to \Sigma$ such that $f(i) = x_i$ and $g(i) = y_i$.

3.2 Relations

We consider proof systems for *binary* relations and for *multi-indexed* relations.

- A binary relation R is a set of tuples $(\mathbb{x}, \mathbb{w})$ where $\mathbb{x}$ is the instance and $\mathbb{w}$ the witness. The corresponding language $L(R)$ is the set of $\mathbb{x}$ for which there exists $\mathbb{w}$ such that $(\mathbb{x}, \mathbb{w}) \in R$.
- A multi-indexed relation R is a set of tuples $(\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}], \mathbb{x}, \mathbb{w})$ where $\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}]$ are the indexes, $\mathbb{x}$ the instance, and $\mathbb{w}$ the witness.

3.3 Interactive oracle proofs

Interactive Oracle Proofs (IOPs) [BCS16; RRR16] are information-theoretic proof systems that combine aspects of Interactive Proofs [Bab85; GMR89] and Probabilistically Checkable Proofs [BFLS91; FGLSS91; AS98; ALMSS98], and also generalize the notion of Interactive PCPs [KR08]. Below we describe *public-coin* IOPs.

A $\mathsf{k}_{\mathsf{IOP}}$-round public-coin IOP works as follows. For each round $i \in [\mathsf{k}_{\mathsf{IOP}}]$, the verifier sends a uniformly random message $\boldsymbol{\rho}_i$ to the prover; then the prover sends a proof string Π_i to the verifier. After $\mathsf{k}_{\mathsf{IOP}}$ rounds of interaction, the verifier makes some queries to the proof strings $\Pi_1, \ldots, \Pi_{\mathsf{k}_{\mathsf{IOP}}}$ sent by the prover, and then decides if to accept or to reject.

In more detail, let $\mathsf{IOP} = (\mathbf{P}_{\mathsf{IOP}}, \mathbf{V}_{\mathsf{IOP}})$ be a tuple where $\mathbf{P}_{\mathsf{IOP}}$ (the prover) is an interactive algorithm, and $\mathbf{V}_{\mathsf{IOP}}$ (the verifier) is an interactive oracle algorithm. We say that IOP is a *public-coin IOP* for a binary relation R with $\mathsf{k}_{\mathsf{IOP}}$ rounds and soundness error β_{IOP} if the following holds.

- **Completeness.** For every $(\mathbb{x}, \mathbb{w}) \in R$,

$$\Pr_{\rho_1, \ldots, \rho_{\mathsf{k}_{\mathsf{IOP}}}, \rho_{\mathsf{dc}}} \left[\mathbf{V}_{\mathsf{IOP}}^{\Pi_1, \ldots, \Pi_{\mathsf{k}_{\mathsf{IOP}}}, \rho_1, \ldots, \rho_{\mathsf{k}_{\mathsf{IOP}}}}(\mathbb{x}; \rho_{\mathsf{dc}}) = 1 \ \middle| \ \begin{array}{r} \Pi_1 \leftarrow \mathbf{P}_{\mathsf{IOP}}(\mathbb{x}, \mathbb{w}, \rho_1) \\ \vdots \\ \Pi_{\mathsf{k}_{\mathsf{IOP}}} \leftarrow \mathbf{P}_{\mathsf{IOP}}(\mathbb{x}, \mathbb{w}, \rho_1, \ldots, \rho_{\mathsf{k}_{\mathsf{IOP}}}) \end{array} \right] = 1 \ .$$

- **Soundness.** For every $\mathbb{x} \notin L(R)$ and unbounded malicious prover $\tilde{\mathbf{P}}_{\mathsf{IOP}}$,

$$\Pr_{\rho_1, \ldots, \rho_{\mathsf{k}_{\mathsf{IOP}}}, \rho_{\mathsf{dc}}} \left[\mathbf{V}_{\mathsf{IOP}}^{\tilde{\Pi}_1, \ldots, \tilde{\Pi}_{\mathsf{k}_{\mathsf{IOP}}}, \rho_1, \ldots, \rho_{\mathsf{k}_{\mathsf{IOP}}}}(\mathbb{x}; \rho_{\mathsf{dc}}) = 1 \ \middle| \ \begin{array}{r} \tilde{\Pi}_1 \leftarrow \tilde{\mathbf{P}}_{\mathsf{IOP}}(\rho_1) \\ \vdots \\ \tilde{\Pi}_{\mathsf{k}_{\mathsf{IOP}}} \leftarrow \tilde{\mathbf{P}}_{\mathsf{IOP}}(\rho_1, \ldots, \rho_{\mathsf{k}_{\mathsf{IOP}}}) \end{array} \right] \leq \beta_{\mathsf{IOP}}(|\mathbb{x}|) \ .$$

Complexity measures. We consider several complexity measures beyond soundness error. All of these complexity measures are implicitly functions of the instance $\mathbb{x}$.

- *proof length* $\mathsf{l}_{\mathsf{IOP}}$: the total number of bits in $\Pi_1, \ldots, \Pi_{\mathsf{k}_{\mathsf{IOP}}}$.
- *proof queries* $\mathsf{q}_{\mathsf{IOP},\Pi}$: the number of bits read by the verifier from $\Pi_1, \ldots, \Pi_{\mathsf{k}_{\mathsf{IOP}}}$.
- *interaction randomness length* $\mathsf{r}_{\mathsf{IOP},\mathsf{int}}$: the total number of bits in $\rho_1, \ldots, \rho_{\mathsf{k}_{\mathsf{IOP}}}$.
- *interaction randomness queries* $\mathsf{q}_{\mathsf{IOP},\mathsf{int}}$: the number of bits read by the verifier from $\rho_1, \ldots, \rho_{\mathsf{k}_{\mathsf{IOP}}}$.
- *decision randomness length* $\mathsf{r}_{\mathsf{IOP},\mathsf{dc}}$: The number of bits in ρ_{dc}.
- *prover time* $\mathsf{pt}_{\mathsf{IOP}}$: $\mathbf{P}_{\mathsf{IOP}}$ runs in time $\mathsf{pt}_{\mathsf{IOP}}$.
- *verifier time* $\mathsf{vt}_{\mathsf{IOP}}$: $\mathbf{V}_{\mathsf{IOP}}$ runs in time $\mathsf{vt}_{\mathsf{IOP}}$.

PCPs and IPs. A probabilistically checkable proof (PCP) is an IOP where the prover sends a single message and then the verifier probabilistically reads it (it is not exactly the case where $\mathsf{k}_{\mathsf{IOP}} = 1$, as the prover goes first, where we defined the verifier to speak first).

An interactive proof (IP) is an IOP in which the verifier reads every symbol of the proofs sent to it by the prover. More formally, it is an IOP with proof length $\mathsf{l}_{\mathsf{IOP}} = \mathrm{poly}(|\mathbb{x}|)$ and query complexity equal to $\mathsf{l}_{\mathsf{IOP}}$. Unless explicitly stated otherwise, we assume that IPs have no decision randomness.

Notation. We sometimes denote with bold letters a combination of proofs. For example, we let $\mathbf{\Pi} = (\pi_1, \ldots, \pi_{\mathsf{k}}, \Pi)$ denote the set of oracles received by the verifier. Given a set of queries Q to these oracles, $\mathbf{\Pi}[Q]$ is the set of symbols written in the appropriate oracles.

Non-adaptive verifiers. A public-coin IOP is *non-adaptive* if the algorithm run by $\mathbf{V}_{\mathsf{IOP}}$ after the interactive phase can be written as two algorithms $\mathbf{V}_{\mathsf{IOP}}^{\mathsf{query}}$ and $\mathbf{V}_{\mathsf{IOP}}^{\mathsf{dec}}$ such that:

- $\mathbf{V}_{\mathsf{IOP}}^{\mathsf{query}}$: Given $\mathbb{x}$ and randomness ρ, outputs Q, the set of queries made to the oracles of $\mathbf{V}_{\mathsf{IOP}}$ on the same instance and randomness.

- $\mathbf{V}_{\mathsf{IOP}}^{\mathsf{dec}}$: Given $\mathbb{x}$, ρ and a set A of query answers, outputs the decision that $\mathbf{V}_{\mathsf{IOP}}$ makes given instance $\mathbb{x}$, randomness ρ and A as the set of answers to its queries.

- Efficiency: Running $\mathbf{V}_{\mathsf{IOP}}^{\mathsf{query}}$ and $\mathbf{V}_{\mathsf{IOP}}^{\mathsf{dec}}$ one after the other has identical running time to running $\mathbf{V}_{\mathsf{IOP}}$ on the same instance and randomness.

That is, for every instance $\mathbb{x}$, randomness $\rho_1, \ldots, \rho_{\mathsf{k}_{\mathsf{IOP}}}, \rho_{\mathsf{dc}}$ and oracles $\tilde{\mathbf{\Pi}} = (\tilde{\Pi}_1, \ldots, \tilde{\Pi}_k)$:

$$\mathbf{V}_{\mathsf{IOP}}^{\tilde{\mathbf{\Pi}}}(\mathbb{x}, \rho_1, \ldots, \rho_{\mathsf{k}_{\mathsf{IOP}}}, \rho_{\mathsf{dc}}) = \mathbf{V}_{\mathsf{IOP}}^{\mathsf{dec}}\left(\mathbb{x}, \rho_1, \ldots, \rho_{\mathsf{k}_{\mathsf{IOP}}}, \rho_{\mathsf{dc}}, \tilde{\mathbf{\Pi}}\left[\mathbf{V}_{\mathsf{IOP}}^{\mathsf{query}}\left(\mathbb{x}, \rho_1, \ldots, \rho_{\mathsf{k}_{\mathsf{IOP}}}, \rho_{\mathsf{dc}}\right)\right]\right) \quad .$$

3.4 Round-by-round soundness

Definition 3.1 (State function)**.** *Let* $(\mathbf{P}_{\mathsf{IOP}}, \mathbf{V}_{\mathsf{IOP}})$ *be an IOP for a relation* R*. A state function for* $(\mathbf{P}_{\mathsf{IOP}}, \mathbf{V}_{\mathsf{IOP}})$ *is a deterministic (possibly inefficient) function state that receives as input an instance* $\mathbb{x}$ *and a transcript* $\mathsf{tr} = (\rho, \Pi)$ *and outputs a bit for which the following holds.*

- *Empty transcript: if* $\mathsf{tr} = \emptyset$ *is the empty transcript then* $\mathsf{state}(\mathbb{x}, \mathsf{tr}) = 0$*.*
- *Prover moves: if* tr *is a transcript where the prover is about to move and* $\mathsf{state}(\mathbb{x}, \mathsf{tr}) = 0$*, then for any* Π*,* $\mathsf{state}(\mathbb{x}, \mathsf{tr}||\Pi) = 0$*.*
- *Full transcript: if* tr *is a full transcript and* $\mathsf{state}(\mathbb{x}, \mathsf{tr}) = 0$*, then* $\mathbf{V}_{\mathsf{IOP}}^{\Pi}(\mathbb{x}; \rho) = 0$*.*

Definition 3.2 (Round-by-round soundness)**.** *An IOP* $(\mathbf{P}_{\mathsf{IOP}}, \mathbf{V}_{\mathsf{IOP}})$ *with* $\mathsf{k}_{\mathsf{IOP}}$ *rounds for a relation* R *has round-by-round soundness error* $\beta_{\mathsf{IOP},\mathrm{rbr}}$ *if there exists a state function* state *such that for all*

$\mathbb{x} \notin L(R)$*, every* $i \in [\mathsf{k}_{\mathsf{IOP}}]$*, and every transcript* tr *of the first* i *rounds where the verifier is about to move and* $\mathsf{state}(\mathbb{x}, \mathsf{tr}) = 0$ *it holds that*

$$\Pr_{\rho}[\mathsf{state}(\mathbb{x}, \mathsf{tr}||\rho) = 1] \leq \beta_{\mathsf{IOP},\mathrm{rbr}} \enspace .$$

Fact 3.3 ([CCHLRR18], Corollary 5.7)**.** *Let IP be an interactive proof with soundness* $O(1)$ *and* k_{IP} *rounds. Then the* m*-fold parallel repetition of IP has round-by-round soundness* $O(2^{-m/\mathsf{k}_{\mathsf{IP}}})$*.*

3.5 Error correcting codes

A pair of efficient deterministic algorithms $\mathsf{ECC} = (\mathbf{Enc}, \mathbf{Dec})$ is a $(r, \delta_{\mathsf{ECC}})$-code if for every k: (i) $\mathbf{Enc}\colon \{0,1\}^k \to \{0,1\}^{r(k)}$; (ii) $\mathbf{Dec}\colon \{0,1\}^{r(k)} \to \{0,1\}^k$; (iii) for every m and C' such that $\Delta(\mathbf{Enc}(m), C') \leq \delta_{\mathsf{ECC}}$ it holds that $\mathbf{Dec}(C') = m$.

For $m = (m_1, \ldots, m_\ell) \in (\{0,1\}^k)^\ell$ we denote $\mathbf{Enc}(m, \ell) = \mathbf{Enc}(m_1), \ldots, \mathbf{Enc}(m_\ell)$ and similarly for $C = (C_1, \ldots, C_\ell) \in (\{0,1\}^{r(k)})^\ell$ with, $\mathbf{Dec}(C, \ell) = \mathbf{Dec}(C_1), \ldots, \mathbf{Dec}(C_\ell)$. We call r the rate of ECC. The following theorem follows from various previous works (e.g., [GI05]).

Theorem 3.4. *There exists a* $(r, \delta_{\mathsf{ECC}})$*-code where* $r(k) = O(k)$ *and* $\delta_{\mathsf{ECC}} = \Omega(1)$*. Encoding and decoding* k*-bit strings takes time* $\tilde{O}(k)$*.*

3.6 PCPs of proximity for circuit satisfiability

Let $\mathsf{PCP} = (\mathbf{P}_{\mathsf{PCP}}, \mathbf{V}_{\mathsf{PCP}})$ be a tuple where $\mathbf{P}_{\mathsf{PCP}}$ is a deterministic algorithm and $\mathbf{V}_{\mathsf{PCP}}$ is a randomized oracle algorithm. PCP is a *PCP of proximity (PCPP) for circuit satisfiability* with soundness error β_{PCP} and proximity δ_{PCP} if the following hold for every boolean circuit $C\colon \{0,1\}^n \to \{0,1\}$ and $\mathbb{x} \in \{0,1\}^n$:

- **Completeness.** If $C(\mathbb{x}) = 1$ then

$$\Pr\left[\; \mathbf{V}_{\mathsf{PCP}}^{\mathbb{x},\Pi}(C) = 1 \;\middle|\; \Pi \leftarrow \mathbf{P}_{\mathsf{PCP}}(C, \mathbb{x}) \;\right] = 1 \enspace .$$

- **Soundness.** If $\mathbb{x}$ is δ_{PCP}-far from any $\mathbb{x}'$ such that $C(\mathbb{x}') = 1$ then for any $\tilde{\Pi}$

$$\Pr\left[\; \mathbf{V}_{\mathsf{PCP}}^{\mathbb{x},\tilde{\Pi}}(C) = 1 \;\right] \leq \beta_{\mathsf{PCP}} \enspace .$$

Theorem 3.5 (PCP of proximity [Mie09])**.** *There exists a non-adaptive PCP of proximity for circuit satisfiability such that:*

PCP of Proximity for $C(\mathbb{x}) = 1$	
Proof length	$\mathrm{poly}(\lvert C\rvert)$
Alphabet size	2
Queries	$O(1)$
Randomness	$O(\log \lvert C\rvert)$
Proximity	$O(1)$
Soundness error	$O(1)$
Prover running time	$\mathrm{poly}(\lvert C\rvert)$
Verifier running time	$\tilde{O}(\lvert C\rvert)$

3.7 Extractors

Definition 3.6. *The min-entropy of a random variable X is*

$$H_{\mathsf{min}}(X) = \min_{x \in \mathsf{supp}(X)} -\log \Pr[X = x]$$

Definition 3.7. *A function $\mathsf{Ext}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is a (k, ε)-extractor if for every X with min-entropy at least k, $\mathsf{SD}(\mathsf{Ext}(X, U_d), U_m) \leq \varepsilon$ (where SD is the statistical distance). An extractor is explicit if it is computable in polynomial time.*

We use the following explicit construction of extractors with tight parameters.

Theorem 3.8 ([GUV09])**.** *For every constant $\alpha > 0$, and all positive integers n, k and all $\varepsilon > 0$, there is an explicit construction of a (k, ε)-extractor $\mathsf{Ext}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ with $d = O(\log n + \log(1/\varepsilon))$, and $m \geq (1 - \alpha)k$.*

Setting specific parameters, we will use this simpler version of the theorem.

Theorem 3.9. *For all positive integers m, and $\ell \geq \log m$ there is an explicit construction of a $(2m, 2^{-\ell})$-extractor $\mathsf{Ext}\colon \{0,1\}^{3m} \times \{0,1\}^d \to \{0,1\}^m$ with $d = O(\ell)$.*

Fact 3.10. *For all $n \in \mathbb{N}$, all $x \in \{0,1\}^n$ and $0 < \delta < 1$ we have that*

$$|\{x' \in \{0,1\}^n : \Delta(x, x') \leq \delta)\}| \leq 2^{n \cdot H(\delta)} \ .$$

(here H is the entropy function $H(p) = -p\log(p) - (1-p)\log(1-p)$).

4 Index-decodable PCPs

Let $\mathsf{PCP} = (\mathbf{I}_{\mathsf{PCP}}, \mathbf{P}_{\mathsf{PCP}}, \mathbf{V}_{\mathsf{PCP}}, \mathbf{D}_{\mathsf{PCP}})$ be a tuple where $\mathbf{I}_{\mathsf{PCP}}$ (the indexer) is a deterministic algorithm, $\mathbf{P}_{\mathsf{PCP}}$ (the prover) is a deterministic algorithm, $\mathbf{V}_{\mathsf{PCP}}$ (the verifier) is a randomized oracle algorithm, and $\mathbf{D}_{\mathsf{PCP}}$ (the decoder) is a (possibly inefficient) deterministic algorithm. PCP is an *index-decodable PCP* for a multi-indexed relation R with decodability bound κ_{PCP} if the following holds.

- **Completeness.** For every $(\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}], \mathbb{x}, \mathbb{w}) \in R$,

$$\Pr_{\rho}\left[\mathbf{V}_{\mathsf{PCP}}^{\pi_1,\ldots,\pi_{\mathsf{k}},\Pi}(\mathbb{x};\rho) = 1 \;\middle|\; \begin{array}{r} \pi_1 \leftarrow \mathbf{I}_{\mathsf{PCP}}(\mathbb{i}[1]) \\ \vdots \\ \pi_{\mathsf{k}} \leftarrow \mathbf{I}_{\mathsf{PCP}}(\mathbb{i}[\mathsf{k}]) \\ \Pi \leftarrow \mathbf{P}_{\mathsf{PCP}}(\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}], \mathbb{x}, \mathbb{w}) \end{array} \right] = 1 \ .$$

- **Decodability.** For every $\mathbb{x}$ and malicious proofs $\tilde{\pi}_1, \ldots, \tilde{\pi}_{\mathsf{k}}, \tilde{\Pi}$, if

$$\Pr_{\rho}\left[\mathbf{V}_{\mathsf{PCP}}^{\tilde{\pi}_1,\ldots,\tilde{\pi}_{\mathsf{k}},\tilde{\Pi}}(\mathbb{x};\rho) = 1 \right] > \kappa_{\mathsf{PCP}}(|\mathbb{x}|)$$

  then there exists $\mathbb{w}$ such that $(\mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_1), \ldots, \mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_{\mathsf{k}}), \mathbb{x}, \mathbb{w}) \in R$.

The proofs $\pi_1, \ldots, \pi_{\mathsf{k}}$ are called *indexer proofs* and the proof Π is called the *prover proof.* We refer to Section 2.3 for an intuitive overview of this notion and further discussion.

Complexity measures. In addition to the standard complexity measures mentioned in Section 3.3 we consider several additional measures for index-decodable PCPs. All of these complexity measures are implicitly functions of the instance $\mathbb{x}$.

- *Indexer proof length* $\mathsf{l}_{\mathsf{PCP},\mathbf{I}}$: the number of symbols in a single indexer proof π_i.
- *Indexer proof alphabet* $\Sigma_{\mathsf{PCP},\mathbf{I}}$: the alphabet of the indexer proofs.
- *Prover proof length* $\mathsf{l}_{\mathsf{PCP},\mathbf{P}}$: the number of symbols in a single indexer proof Π.
- *Prover proof alphabet* $\Sigma_{\mathsf{PCP},\mathbf{P}}$: the alphabet of the prover's proof.
- *Indexer time* $\mathsf{it}_{\mathsf{PCP}}$: $\mathbf{I}_{\mathsf{PCP}}$ runs in time $\mathsf{it}_{\mathsf{PCP}}$.
- *Decoder time* $\mathsf{dt}_{\mathsf{PCP}}$: $\mathbf{D}_{\mathsf{PCP}}$ runs in time $\mathsf{dt}_{\mathsf{PCP}}$.

We sometimes refer separately to the number of queries done to the indexer proofs (per proof) and and prover proof. If these are not listed separately, then the number is asymptotically identical.

Remark 4.1 (index-decodable IOPs)**.** The definition of index-decodable PCPs can be extended in a straightforward way to allow interaction, thereby obtaining *index-decodable IOPs.* While not used in this paper, this extended notion is likely to achieve better parameters (e.g., linear proof length) via interactive tools (e.g., interactive proof composition [BCGRS17] instead of non-interactive proof composition as in Section 6.1) and is likely to be of further interest. We leave the exploration of this notion to future work.

Remark 4.2 (comparison with decodable PCPs)**.** Despite the similar names, index-decodable PCPs and *decodable PCPs* [DH13] are different notions: a decodable PCP is a standard PCP with a "PCP decoder" that list-decodes a random location in the NP witness from a given PCP string.

Prover-robust index-decodable PCPs. Let $\mathsf{PCP} = (\mathbf{I}_{\mathsf{PCP}}, \mathbf{P}_{\mathsf{PCP}}, (\mathbf{V}_{\mathsf{PCP}}^{\mathsf{qry}}, \mathbf{V}_{\mathsf{PCP}}^{\mathsf{dc}}), \mathbf{D}_{\mathsf{PCP}})$ be a non-adaptive index-decodable PCP for a multi-indexed relation R. We say that PCP is *prover-robust* with decodability bound κ_{PCP} and robustness σ_{PCP} if for every $\mathbb{x}$ and proofs $\tilde{\mathbf{\Pi}}_{\mathbb{i}} = (\tilde{\pi}_1, \ldots, \tilde{\pi}_{\mathsf{k}})$ and $\tilde{\Pi}$ if

$$\Pr_{\rho}\left[\exists\, A' \text{ s.t. } \mathbf{V}^{\mathsf{dc}}_{\mathsf{PCP}}(\mathbb{x}, \rho, \tilde{\mathbf{\Pi}}_{\mathbb{i}}[Q_{\mathbb{i}}], A') = 1 \;\wedge\; \Delta(A', A) \leq \sigma_{\mathsf{PCP}}(|\mathbb{x}|) \;\middle|\; \begin{array}{l} (Q_{\mathbb{i}}, Q_*) \leftarrow \mathbf{V}^{\mathsf{qry}}_{\mathsf{PCP}}(\mathbb{x}, \rho) \\ A := \{\, \tilde{\Pi}[q] \mid q \in Q_* \,\} \end{array} \right] > \kappa_{\mathsf{PCP}}(|\mathbb{x}|) \enspace ,$$

where $Q_{\mathbb{i}}$ are the queries made to the indexer proofs and Q_* are the queries made to the prover proof. Then there exists $\mathbb{w}$ such that $(\mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_1), \ldots, \mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_{\mathsf{k}}), \mathbb{x}, \mathbb{w}) \in R$.

The notion of robustness essentially says that for any set of proofs whose decoding is not in the relation, with probability κ_{PCP} the Hamming ball of radius σ_{PCP} around the answers to the verifier's queries to the prover proof do not make the verifier accept.

5 Constructing index-decodable PCPs

In our construction of index-decodable PCPs, we will use slight variations of the notion of PCPs of proximity that suit our setting. We define these notions and show how to construct them (based on standard PCPPs) in Section 5.1, and then in Section 5.2 we show the actual construction. We achieve the following theorem:

Theorem 5.1. *Let $R = \{(\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}], \mathbb{x}, \mathbb{w})\}$ be a multi-indexed relation whose membership can be verified by a circuit C. Then R has a non-adaptive index-decodable PCP $\mathsf{PCP} = (\mathbf{I}_{\mathsf{PCP}}, \mathbf{P}_{\mathsf{PCP}}, \mathbf{V}_{\mathsf{PCP}}, \mathbf{D}_{\mathsf{PCP}})$ for R with the parameters below.*

Index-Decodable PCP for $(\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}], \mathbb{x}, \mathbb{w}) \in R$	
Indexer proof length (per proof)	$O(\lvert\mathbb{i}[i]\rvert)$
Prover proof length	$\mathrm{poly}(\lvert C\rvert)$
Indexer alphabet size	2
Prover alphabet size	2^{k}
Queries to proofs	$O(1)$
Randomness	$O(\log \lvert C\rvert)$
Decodability bound	$O(1)$
Indexer running time	$\tilde{O}(\lvert\mathbb{i}[i]\rvert)$
Prover running time	$\mathrm{poly}(\lvert C\rvert)$
Verifier running time	$\tilde{O}(\lvert C\rvert)$
Decoder running time	$\tilde{O}(\lvert\mathbb{i}[i]\rvert)$

Proof. We use the construction described in Theorem 5.7, plugging in the oblivious multi-input PCPP of Lemma 5.5 and the constant-rate and constant-distance error-correcting code described in Theorem 3.4. □

5.1 Building blocks

Definition 5.2 (Multi-input PCP of proximity)**.** *A PCP of proximity system for circuit satisfiability of a k-input circuit $C : \{0,1\}^{n_1} \times \cdots \times \{0,1\}^{n_k} \to \{0,1\}$ is* multi-input *with soundness β_{PCPP} and proximity δ_{PCPP} if the soundness condition is replaced with the following:*

Let $\mathbb{x}_1, \ldots, \mathbb{x}_k$ be the inputs to the circuit. Suppose that for every set of inputs $\mathbb{x}'_1, \ldots, \mathbb{x}'_k$ such that $C(\mathbb{x}'_1, \ldots, \mathbb{x}'_k) = 1$ there exists i such that $\mathbb{x}_i$ is δ_{PCP}-far from $\mathbb{x}'_i$. Then for any $\tilde{\Pi}$:

$$\Pr\left[\ \mathbf{V}_{\mathsf{PCP}}^{\mathbb{x}_1,\ldots,\mathbb{x}_k,\tilde{\Pi}}(C) = 1\ \right] \leq \beta_{\mathsf{PCP}}\ .$$

Definition 5.3 (Oblivious PCP of proximity)**.** *Let $\mathcal{C} = \{C_z\}_{z\in\{0,1\}^m}$ be a family of circuits where for every z, $C_z \colon \{0,1\}^{n_1} \times \cdots \times \{0,1\}^{n_k} \to \{0,1\}$, and suppose that there exists a circuit $C' \colon \{0,1\}^{n_1} \times \cdots \times \{0,1\}^{n_k} \times \{0,1\}^m \to \{0,1\}$ such that for every $\mathbb{x}_1, \ldots, \mathbb{x}_k$ and z: $C'(\mathbb{x}_1, \ldots, \mathbb{x}_k, z) = C_z(\mathbb{x}_1, \ldots, \mathbb{x}_k)$. An oblivious multi-input PCP of proximity system for circuit satisfiability of $\mathcal{C}$ is PCPP that is complete and sound for proving satisfiability of C_z $z \in \{0,1\}^m$. Additionally, the verifier's queries depend only on $\mathcal{C}$ and on the verifier's randomness. In particular, its queries do not depend on z, or on its oracles.*

We now show how to generically construct these PCPPs from standard PCPPs (with minimal overhead).

Lemma 5.4 (Existence of multi-input PCPPs)**.** *There exists a multi-input PCP of proximity for circuit satisfiability of circuits* $C : \{0,1\}^{n_1} \times \cdots \times \{0,1\}^{n_k} \to \{0,1\}$ *where* k *is a constant with the following parameters:*

Multi-input PCP of Proximity for $C(\mathbb{x}_1, \ldots, \mathbb{x}_k) = 1$	
Proof length	$\mathrm{poly}(\lvert C\rvert)$
Alphabet size	2
Queries	$O(1)$
Randomness	$O(\log \lvert C\rvert)$
Proximity	$O(1)$
Soundness error	$O(1)$
Prover running time	$\mathrm{poly}(\lvert C\rvert)$
Verifier running time	$\tilde{O}(\lvert C\rvert)$

Proof. Let $(\mathbf{P}_{\mathsf{PCPP}}, \mathbf{V}_{\mathsf{PCPP}})$ be the PCPP system of Theorem 3.5 with soundness β_{PCPP} and proximity parameter δ_{PCPP}, where β_{PCPP} and δ_{PCPP} are small enough constants satisfying $\delta_{\mathsf{PCPP}} \cdot k < 1/3$. If this is not the case, one can first improve proximity using standard techniques (e.g., [RR20, Lemma 8.6]) with minimal overhead. Let $n_{\mathsf{max}} := \max\{n_i\}$ For every $i \in [k]$ let $t_i := n_{\mathsf{max}}/n_i$. Assume without loss of generality that these values are integers (otherwise, padding each input to the next power of 2 will suffice). For an instance $\mathbb{x}_i$, let $\mathbb{x}_i^{t_i}$ be the t_i-wise repetition of $\mathbb{x}_i$.

Consider the circuit C' that on inputs $(\widehat{\mathbb{x}}_1, \ldots, \widehat{\mathbb{x}}_k) \in \{0,1\}^{k \cdot n_{\mathsf{max}}}$ outputs 1 if and only if the following tests pass:
1. *Encoding validity*: For every i, there exists a string $\mathbb{x}_i$ such that $\widehat{\mathbb{x}} = \mathbb{x}_i^{t_i}$.
2. *Satisfiability*: Let $\mathbb{x}_1, \ldots, \mathbb{x}_k$ be the strings from the previous test. Test that $C(\mathbb{x}_1 \ldots, \mathbb{x}_k) = 1$.
We now describe the PCPP. On input a circuit $C : \{0,1\}^{n_1} \times \cdots \times \{0,1\}^{n_k} \to \{0,1\}$, and inputs $(\mathbb{x}_1, \ldots, \mathbb{x}_k) \in \{0,1\}^{n_1} \times \cdots \times \{0,1\}^{n_k}$:

- The prover sends $\Pi := \mathbf{P}_{\mathsf{PCPP}}(C', \mathbb{x}_1^{t_1}, \ldots, \mathbb{x}_k^{t_k})$.
- The verifier, given oracle access to the inputs $\mathbb{x}_1, \ldots, \mathbb{x}_k$ and to $\tilde{\Pi}$ emulates the verification of:

$$\mathbf{V}_{\mathsf{PCP}}^{\mathbb{x}_1^{t_1}, \ldots, \mathbb{x}_k^{t_k}, \mathbb{x}, \tilde{\Pi}}(C') = 1 \ ,$$

as follows: Queries to $\tilde{\Pi}$ are forwarded to the relevant oracle. For a query $q \in [t_i \cdot |\mathbb{x}_i|]$ to $\mathbb{x}_i^{t_i}$, return $\mathbb{x}_i$ at location $(q \mod t_i)$.

Completeness is immediate by the perfect completeness of the PCP of proximity and the fact that the verifier correctly returns queries to $\mathbb{x}_i^{t_i}$. We show soundness by proving the contrapositive. Assume that the verifier accepts with high probability. We show that there must exist strings that satisfy the circuit and are individually close to each input. Fix inputs $\mathbb{x}_1, \ldots, \mathbb{x}_k$ and (possibly malicious) proof $\tilde{\Pi}$ such that the verifier, given these inputs and proof, accepts with probability greater than β_{PCPP}. Then we have that with probability greater than β_{PCPP}:

$$\mathbf{V}_{\mathsf{PCP}}^{\mathbb{x}_1^{t_1}, \ldots, \mathbb{x}_k^{t_k}, \mathbb{x}, \tilde{\Pi}}(C') = 1 \ .$$

This implies, by soundness of the PCPP, that there exist $\widehat{\mathbb{x}}'_1, \ldots, \widehat{\mathbb{x}}'_k$ such that $C'(\widehat{\mathbb{x}}'_1, \ldots, \widehat{\mathbb{x}}'_k) = 1$ and

$$\Delta((\mathbb{x}_1^{t_1}, \ldots, \mathbb{x}_k^{t_k}), (\widehat{\mathbb{x}}'_1, \ldots, \widehat{\mathbb{x}}'_k)) < \delta_{\mathsf{PCPP}} \ .$$

Since $C'(\widehat{\mathbb{x}}'_1, \ldots, \widehat{\mathbb{x}}'_k) = 1$, we have that for every i there exists $\mathbb{x}'_i$ such that $\widehat{\mathbb{x}}'_i = \mathbb{x}'^{t_i}_i$. Moreover $C(\mathbb{x}'_1, \ldots, \mathbb{x}'_k) = 1$.

Notice that for every i: $|\mathbb{x}^{t_i}_i| = |\mathbb{x}'^{t_i}_i| = n_{\mathsf{max}}$. Therefore, by a counting argument,

$$\Delta((\mathbb{x}^{t_1}_1, \ldots, \mathbb{x}^{t_k}_k), (\mathbb{x}'^{t_1}_1, \ldots, \mathbb{x}'^{t_k}_k)) < \delta_{\mathsf{PCPP}} \ ,$$

implies that for every i: $\Delta(\mathbb{x}^{t_i}_i, \mathbb{x}'^{t_i}_i) < k \cdot \delta_{\mathsf{PCPP}}$. It follows that $\Delta(\mathbb{x}_i, \mathbb{x}'_i) < k \cdot \delta_{\mathsf{PCPP}} = O(1)$ since if $\Delta(\mathbb{x}_i, \mathbb{x}'_i) = m$, then every one of the differences between the two strings propagates t_i times, and so the relative number of times it occurs is still m: $\Delta(\mathbb{x}^{t_i}_i, \mathbb{x}'^{t_i}_i) = m$.

We now analyze the parameters of the new PCPP. Notice that $|C'| = O(|C|)$. Thus the new prover has running time $\mathrm{poly}(|C|)$, which is also the proof length. The verifier uses the same number of random bits as $\mathbf{V}_{\mathsf{PCPP}}$, and runs in time $\tilde{O}(|C'|) = \tilde{O}(|C|)$. If the original verifier was non-adaptive, then so is the new verifier. □

Lemma 5.5 (Existence of oblivious multi-input PCPPs)**.** *Let $\mathcal{C} = \{C_z\}_{z\in\{0,1\}^m}$ be a circuit family with $C_z\colon \{0,1\}^{n_1} \times \ldots, \times \{0,1\}^{n_k}$ for constant k and suppose there exists $C'\colon \{0,1\}^{n_1} \times \cdots \times \{0,1\}^{n_k} \times \{0,1\}^m \to \{0,1\}$ such that for every $\mathbb{x}_1, \ldots, \mathbb{x}_k$ and z: $C'(\mathbb{x}_1, \ldots, \mathbb{x}_k, z) = C_z(\mathbb{x}_1, \ldots, \mathbb{x}_k)$. There exists an oblivious multi-input PCP of proximity for circuit satisfiability of $\mathcal{C}$ with the following parameters:*

Oblivious PCP of Proximity for $C_z(\mathbb{x}_1, \ldots, \mathbb{x}_k) = 1$	
Proof length	$\mathrm{poly}(\lvert C'\rvert)$
Alphabet size	2
Queries	$O(1)$
Randomness	$O(\log \lvert C'\rvert)$
Proximity	$O(1)$
Soundness error	$O(1)$
Prover running time	$\mathrm{poly}(\lvert C'\rvert)$
Verifier running time	$\tilde{O}(\lvert C'\rvert)$

Proof. Let $(\mathbf{P}_{\mathsf{PCPP}}, \mathbf{V}_{\mathsf{PCPP}})$ be the multi-input system of Lemma 5.4 and $(\mathbf{Enc}, \mathbf{Dec})$ be the error-correcting code of Theorem 3.4.

Consider the $(k+1)$-input circuit C'' that on inputs $\mathbb{x}_1, \ldots, \mathbb{x}_k$ and $\widehat{z}$ outputs 1 if and only if the following tests pass:

1. *Encoding validity*: $\mathbf{Enc}(\mathbf{Dec}(\widehat{z})) = \widehat{z}$.
2. *Satisfiability*: $C'(\mathbb{x}_1, \ldots, \mathbb{x}_k, \mathbf{Dec}(\widehat{z})) = 1$.

We now describe the PCPP. On explicit input C_z, and oracle inputs $\mathbb{x}_1, \ldots, \mathbb{x}_k$:

- The prover sends $\Pi := \mathbf{P}_{\mathsf{PCPP}}(C'', (\mathbb{x}_1, \ldots, \mathbb{x}_k, \mathbf{Enc}(z)))$.
- The verifier, given oracle access to $\mathbb{x}$ and to $\tilde{\Pi}$ computes $\mathbf{Enc}(z)$ and verifies

$$\mathbf{V}_{\mathsf{PCP}}^{(\mathbb{x}_1,\ldots,\mathbb{x}_k,\mathbf{Enc}(z)),\tilde{\Pi}}(C'') = 1 \ .$$

Completeness is immediate by the perfect completeness of the multi-input PCPP and the fact that the verifier correctly returns queries to $\widehat{z}$. We show that multi-input soundness holds with respect to the circuit C_z. We do this via the contrapositive - we show that if the verifier accepts with high probability, then the instances $\mathbb{x}_1, \ldots, \mathbb{x}_k$ are close to satisfying C_z.

Fix z and instances $\mathbb{x}_1, \ldots, \mathbb{x}_k$ and a proof $\tilde{\Pi}$. Suppose that, given z explicitly and oracle access to $\mathbb{x}_1, \ldots, \mathbb{x}_k$ and $\tilde{\Pi}$, the verifier accepts with probability greater than β_{PCP}. Then we have that there exist $\mathbb{x}'_1, \dots, \mathbb{x}'_k$ and $\widehat{z}'$ such that: (i) $C''(\mathbb{x}'_1, \dots, \mathbb{x}'_k, \widehat{z}') = 1$, (ii) For every i: $\Delta(\mathbb{x}_i, \mathbb{x}'_i) < \delta_{\mathsf{PCPP}}$ and, (iii) $\Delta(\widehat{z}, \widehat{z}') < \delta_{\mathsf{PCPP}}$. This must be true since otherwise we have a contradiction to the soundness of the multi-input PCPP. This, in turn, implies that $C'(\mathbb{x}'_1, \dots, \mathbb{x}'_k, \mathbf{Dec}(\widehat{z}')) = 1$. Since $\delta_{\mathsf{PCPP}} < \delta_{\mathsf{ECC}}$, we have that $\mathbf{Dec}(\widehat{z}') = \mathbf{Dec}(\widehat{z}) = z$.

We now analyze the parameters of the new PCPP. Notice that $|C''| = \tilde{O}(|C'|)$. Thus the new prover has running time $\mathrm{poly}(|C'|)$, which is also the proof length. The verifier uses the same number of random bits as $\mathbf{V}_{\mathsf{PCPP}}$, and runs in time $\tilde{O}(|C''|) = \tilde{O}(|C'|)$. Since $\mathbf{V}_{\mathsf{PCPP}}$ is non-adaptive, the queries it makes do not depend on its oracles. This includes the oracle to $\widehat{z}$, and so the new verifier's queries do not depend on z. □

5.2 The construction

We begin by giving a construction that achieves all of the parameters of our final index-decodable PCP except that the number of queries made by the verifier to the prover proof is $O(\mathsf{k})$ rather than $O(1)$. We define the circuit $C_{*,\mathbb{x}}$ and circuit family $\mathcal{C} = \{C_i\}_{i \in [\mathsf{k}]}$. Later on, in the construction, we will have a (standard) PCPP proof for satisfiability of $C_{*,\mathbb{x}}$, and a proof for an oblivious multi-input PCPP for satisfiability of each of the circuits $C_i \in \mathcal{C}$.

Definition 5.6. *Let $R = \{(\mathbb{i}[1], \dots, \mathbb{i}[\mathsf{k}], \mathbb{x}, \mathbb{w})\}$ be a multi-indexed relation, and $(\mathbf{Enc}, \mathbf{Dec})$ be an error-correcting code. Let $\mathbb{x}$ be an instance. Define boolean circuits $C_{*,\mathbb{x}}$ and a circuit family $\mathcal{C} = \{C_i\}_{i \in [\mathsf{k}]}$ as follows:*

- $C_{*,\mathbb{x}}(\Pi_*) = 1$ *if and only if the following tests pass:*
  1. Encoding validity: *Test that* $\mathbf{Enc}(\mathbf{Dec}(\Pi_*)) = \Pi_*$.
  2. Membership: *Test that* $(\mathbb{i}_*[1], \dots, \mathbb{i}_*[\mathsf{k}], \mathbb{x}, \mathbb{w}_*) \in R$, *where* $(\mathbb{i}_*[1], \dots, \mathbb{i}_*[\mathsf{k}], \mathbb{w}_*) := \mathbf{Dec}(\Pi_*)$.
- $C_i(\pi, \Pi_*) = 1$ *if and only if the following tests pass:*
  1. Encoding validity: *Test that:*
     - $\mathbf{Enc}(\mathbf{Dec}(\pi)) = \pi$.
     - $\mathbf{Enc}(\mathbf{Dec}(\Pi_*)) = \Pi_*$.
  2. Consistency: *Let* $\tilde{\mathbb{i}}[i] := \mathbf{Dec}(\pi)$ *and* $(\mathbb{i}_*[1], \dots, \mathbb{i}_*[\mathsf{k}], \mathbb{w}_*) := \mathbf{Dec}(\Pi_*)$. *Test that* $\tilde{\mathbb{i}}[i] = \mathbb{i}_*[i]$.

From the definition it is easy to see that there exists C' such that $C'(\pi, \Pi_*, i) = C_i(\pi, \Pi_*)$ for every π, Π_* where $|C'| = O(|C_i|)$. This facilitates using this circuit family with an oblivious PCPP.

Theorem 5.7. *Let $R = \{(\mathbb{i}[1], \dots, \mathbb{i}[\mathsf{k}], \mathbb{x}, \mathbb{w})\}$ be a multi-indexed relation. Let $\mathsf{PCPP} = (\mathbf{P}_{\mathsf{PCPP}}, \mathbf{V}_{\mathsf{PCPP}})$ be a (oblivious multi-input) PCP of proximity for circuit satisfiability with proximity parameter δ_{PCPP} using a binary alphabet and $(\mathbf{Enc}, \mathbf{Dec})$ be an error-correcting code with distance $\delta_{\mathsf{PCPP}} \leq \delta_{\mathsf{ECC}}$. Then Construction 5.8 is a non-adaptive index-decodable PCP $\mathsf{PCP} = (\mathbf{I}_{\mathsf{PCP}}, \mathbf{P}_{\mathsf{PCP}}, \mathbf{V}_{\mathsf{PCP}}, \mathbf{D}_{\mathsf{PCP}})$ for R with the parameters below.*

PCPP for satisfiability of circuits $\mathcal{C}$ and $C_{*,\mathbb{x}}$	
Proof length	$\mathsf{l}_{\mathsf{PCPP}}$
Alphabet size	2
Queries	$\mathsf{q}_{\mathsf{PCPP}}$
Randomness	$\mathsf{r}_{\mathsf{PCPP}}$
Proximity	δ_{PCPP}
Soundness error	β_{PCPP}
Prover running time	$\mathsf{pt}_{\mathsf{PCPP}}$
Verifier running time	$\mathsf{vt}_{\mathsf{PCPP}}$

$+$

Error correcting code	
Distance	δ_{ECC}
Rate	$\mathsf{rt}_{\mathsf{ECC}}$
Encoding time	$\mathsf{et}_{\mathsf{ECC}}$
Decoding time	$\mathsf{dt}_{\mathsf{ECC}}$

Index-Decodable PCP for $(\mathbb{i}[1],\dots,\mathbb{i}[\mathsf{k}],\mathbb{x},\mathbb{w}) \in R$	
Indexer proof length (per proof)	$\mathsf{rt}_{\mathsf{ECC}}(\lvert\mathbb{i}[i]\rvert)$
Prover proof length	$2 \cdot \mathsf{l}_{\mathsf{PCPP}} + \mathsf{rt}_{\mathsf{ECC}}(\lvert\mathbb{w}\rvert + \sum_{i=1}^{\mathsf{k}} \lvert\mathbb{i}[i]\rvert)$
Indexer alphabet size	2
Prover alphabet size	2^{k}
Queries to indexer proof (per proof)	$\mathsf{q}_{\mathsf{PCPP}}$
$\longrightarrow$ Queries to prover proof	$2 \cdot \mathsf{q}_{\mathsf{PCPP}}$
Randomness	$2 \cdot \mathsf{r}_{\mathsf{PCPP}}$
Decodability bound	β_{PCPP}
Indexer running time	$\mathsf{et}_{\mathsf{ECC}}(\lvert\mathbb{i}[i]\rvert)$
Prover running time	$\mathsf{et}_{\mathsf{ECC}}(\lvert\mathbb{w}\rvert + \sum_{i=1}^{\mathsf{k}} \lvert\mathbb{i}[i]\rvert) + (\mathsf{k}+1) \cdot \mathsf{pt}_{\mathsf{PCPP}}$
Verifier running time	$\mathsf{k} \cdot \mathsf{vt}_{\mathsf{PCPP}}$
Decoder running time	$\mathsf{dt}_{\mathsf{ECC}}(\lvert\mathbb{i}[i]\rvert)$

We now describe the construction (see Section 2.5 for an overview), and then prove the theorems.

Construction 5.8. We describe the index-decodable PCP for R.

- $\mathbf{I}_{\mathsf{PCP}}(\mathbb{i}[i])$: Output $\pi_i := \mathbf{Enc}(\mathbb{i}[i])$.
- $\mathbf{P}_{\mathsf{PCP}}(\mathbb{i}[1],\dots,\mathbb{i}[\mathsf{k}],\mathbb{x},\mathbb{w})$:
  1. Compute $\Pi_* := \mathbf{Enc}(\mathbb{i}[1],\dots,\mathbb{i}[\mathsf{k}],\mathbb{w})$.
  2. *Proof of membership*: Generate a proof $\Pi_{\mathsf{mem}} := \mathbf{P}_{\mathsf{PCPP}}(C_{*,\mathbb{x}},\Pi_*)$ (using a standard PCPP).
  3. *Proofs of consistency*: For every $i \in [\mathsf{k}]$, compute $\Pi_i := \mathbf{P}_{\mathsf{PCPP}}(C_i,(\pi_i,\Pi_*))$ where $\pi_i := \mathbf{I}_{\mathsf{PCP}}(\mathbb{i}[i])$ (using the oblivious PCPP for $\mathcal{C}$).
  4. *Query bundling*: Let $\Pi_{\mathbb{i}} := \{\Pi_i\}_{i\in[\mathsf{k}]}$ be a proof such that $\Pi_{\mathbb{i}}[q] = (\Pi_1[q],\dots,\Pi_{\mathsf{k}}[q])$. That is, in location q of $\Pi_{\mathbb{i}}$ write a k-bit symbol consisting of the q-th bit of each of the proofs $\{\Pi_i\}_{i\in[\mathsf{k}]}$.
  5. Output $(\Pi_*,\Pi_{\mathsf{mem}},\Pi_{\mathbb{i}})$.
- $\mathbf{V}_{\mathsf{PCP}}^{\tilde{\pi}_1,\dots,\tilde{\pi}_{\mathsf{k}},\tilde{\Pi}_*,\tilde{\Pi}_{\mathsf{mem}},\tilde{\Pi}_{\mathbb{i}}}(\mathbb{x})$: Accept if and only if all of the following test accept.
  1. *Membership test*: $\mathbf{V}_{\mathsf{PCPP}}^{\tilde{\Pi}_*,\tilde{\Pi}_{\mathsf{mem}}}(C_{*,\mathbb{x}}) = 1$.
  2. *Consistency test*: Parse $\tilde{\Pi}_{\mathbb{i}} = \{\tilde{\Pi}_i\}_{i\in[\mathsf{k}]}$. Choose PCPP verifier randomness ρ. For every $i \in [\mathsf{k}]$ test that $\mathbf{V}_{\mathsf{PCPP}}^{(\tilde{\pi}_i,\tilde{\Pi}_*),\tilde{\Pi}_i}(C_i;\rho) = 1$. (Using the verifier of the oblivious PCPP for $\mathcal{C}$).
- $\mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_i)$: Output $\mathbf{Dec}(\tilde{\pi}_i)$.

Proof. We prove completeness, then decodability, and finally analyze complexity measures.

Completeness. Fix $(\mathbb{i}[1],\dots,\mathbb{i}[\mathsf{k}],\mathbb{x},\mathbb{w}) \in R$. We show that both of $\mathbf{V}_{\mathsf{PCP}}$'s tests pass with probability 1 and therefore $\mathbf{V}_{\mathsf{PCP}}$ always accepts. Let $\pi_1,\dots,\pi_{\mathsf{k}}$ and $\Pi_*,\Pi_{\mathsf{mem}},\Pi_{\mathbb{i}}$ be the proofs generated by the indexer and prover respectively where $\Pi_{\mathbb{i}} := \{\Pi_i\}_{i\in[\mathsf{k}]}$.

1. *Membership test*: Since $(\mathbb{i}[1],\dots,\mathbb{i}[\mathsf{k}],\mathbb{x},\mathbb{w}) \in R$ and $\Pi_* = \mathbf{Enc}(\mathbb{i}[1],\dots,\mathbb{i}[\mathsf{k}],\mathbb{w})$ we have that $C_{*,\mathbb{x}}(\Pi_*) = 1$. By the perfect completeness of the PCP of proximity, since $\Pi_{\mathsf{mem}} = \mathbf{P}_{\mathsf{PCPP}}(C_{*,\mathbb{x}},\Pi_*)$, we have that
$$\Pr\left[\,\mathbf{V}_{\mathsf{PCPP}}^{\Pi_*,\Pi_{\mathsf{mem}}}(C_{*,\mathbb{x}}) = 1\,\right] = 1 \ .$$
2. *Consistency test*: Since $\Pi_* = \mathbf{Enc}(\mathbb{i}[1],\dots,\mathbb{i}[\mathsf{k}],\mathbb{w})$ and for every $i \in [\mathsf{k}]$, $\pi_i = \mathbf{Enc}(\mathbb{i}[i])$, we have that for every i, $C_i(\pi_i,\Pi_*) = 1$. Hence, since $\Pi_i = \mathbf{P}_{\mathsf{PCPP}}(C_i,(\pi_i,\Pi_*))$, by the perfect completeness of the PCP of proximity:
$$\Pr\left[\,\mathbf{V}_{\mathsf{PCPP}}^{(\pi_i,\Pi_*),\Pi_i}(C_i) = 1\,\right] = 1 \ .$$

Decodability. Fix $\mathbb{x}$, $\tilde{\pi}_1, \ldots, \tilde{\pi}_{\mathsf{k}}, \tilde{\Pi}_*, \tilde{\Pi}_{\mathsf{mem}}$ and $\tilde{\Pi}_{\mathbb{i}} = \{\tilde{\Pi}_i\}_{i \in [\mathsf{k}]}$. Suppose that:

$$\Pr\left[\, \mathbf{V}_{\mathsf{PCP}}^{\tilde{\pi}_1,\ldots,\tilde{\pi}_{\mathsf{k}},\tilde{\Pi}_*,\tilde{\Pi}_{\mathsf{mem}},\tilde{\Pi}_{\mathbb{i}}}(\mathbb{x}) = 1 \,\right] > \beta_{\mathsf{PCPP}} \enspace .$$

We show that there exists $\mathbb{w}$ such that $(\mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_1), \ldots, \mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_{\mathsf{k}}), \mathbb{x}, \mathbb{w}) \in R$. To do so we give two claims, each relating to a different test done by the verifier. The first says that the verifier's membership test implies that $\tilde{\Pi}_*$ encodes strings that put $\mathbb{x}$ in R.

Claim 5.9. *There exist* $\tilde{\mathbb{i}}[1], \ldots, \tilde{\mathbb{i}}[\mathsf{k}]$ *and* $\tilde{\mathbb{w}}$ *such that:*
1. Valid encoding*:* $(\tilde{\mathbb{i}}[1], \ldots, \tilde{\mathbb{i}}[\mathsf{k}], \tilde{\mathbb{w}}) = \mathbf{Dec}(\tilde{\Pi}_*)$*.*
2. Membership*:* $(\tilde{\mathbb{i}}[1], \ldots, \tilde{\mathbb{i}}[\mathsf{k}], \mathbb{x}, \tilde{\mathbb{w}}) \in R$*.*

Proof. We show that there exist $\tilde{\mathbb{i}}[1], \ldots, \tilde{\mathbb{i}}[\mathsf{k}]$ and $\tilde{\mathbb{w}}$ such that $\Delta(\widehat{\Pi}_*, \tilde{\Pi}_*) \leq \delta_{\mathsf{PCPP}}$ where $\widehat{\Pi}_* := \mathbf{Enc}(\tilde{\mathbb{i}}[1], \ldots, \tilde{\mathbb{i}}[\mathsf{k}], \tilde{\mathbb{w}})$ and which place $\mathbb{x}$ in the relation. This will imply the claim since the distance of the error-correcting code is at most $\delta_{\mathsf{PCP}} < \delta_{\mathsf{ECC}}$, and so $\widehat{\Pi}_*$ and $\tilde{\Pi}_*$ decode to the same value.

Suppose towards contradiction that for every $\tilde{\mathbb{i}}[1], \ldots, \tilde{\mathbb{i}}[\mathsf{k}]$ and $\tilde{\mathbb{w}}$ such that $\Delta(\widehat{\Pi}_*, \tilde{\Pi}_*) \leq \delta_{\mathsf{PCPP}}$ (where $\widehat{\Pi}_*$ is defined as before) we have that $(\tilde{\mathbb{i}}[1], \ldots, \tilde{\mathbb{i}}[\mathsf{k}], \mathbb{x}, \tilde{\mathbb{w}}) \notin R$. This means that $\tilde{\Pi}_*$ has distance greater than δ_{PCPP} from every $\widehat{\Pi}_*$ such that $C_{*,\mathbb{x}}(\widehat{\Pi}_*) = 1$. Hence by soundness of the PCPP system:

$$\Pr\left[\, \mathbf{V}_{\mathsf{PCPP}}^{\tilde{\Pi}_*,\tilde{\Pi}_{\mathsf{mem}}}(C_{*,\mathbb{x}}) = 1 \,\right] \leq \beta_{\mathsf{PCPP}} \enspace .$$

$\mathbf{V}_{\mathsf{PCP}}$ runs this test in Item 1 and hence will accept with probability at most β_{PCPP} in contradiction to the assumption that

$$\Pr\left[\, \mathbf{V}_{\mathsf{PCP}}^{\tilde{\pi}_1,\ldots,\tilde{\pi}_{\mathsf{k}},\tilde{\Pi}_*,\tilde{\Pi}_{\mathsf{mem}},\tilde{\Pi}_{\mathbb{i}}}(\mathbb{x}) = 1 \,\right] > \beta_{\mathsf{PCPP}} \enspace .$$

□

We now show that the indexer proofs $\tilde{\pi}_1, \ldots, \tilde{\pi}_{\mathsf{k}}$ must be consistent with the encoding $\tilde{\Pi}_*$.

Claim 5.10. *There exist* $\tilde{\mathbb{i}}[1], \ldots, \tilde{\mathbb{i}}[\mathsf{k}]$ *and* $\tilde{\mathbb{w}}$ *such that:*
1. Valid encoding*:* $(\tilde{\mathbb{i}}[1], \ldots, \tilde{\mathbb{i}}[\mathsf{k}], \tilde{\mathbb{w}}) = \mathbf{Dec}(\tilde{\Pi}_*)$*.*
2. Consistency*: For every* $i \in [\mathsf{k}]$*,* $\tilde{\mathbb{i}}[i] := \mathbf{Dec}(\tilde{\pi}_i)$*.*

Proof. We show that there exist $\tilde{\mathbb{i}}[1], \ldots, \tilde{\mathbb{i}}[\mathsf{k}]$ and $\tilde{\mathbb{w}}$ such that $\Delta(\widehat{\Pi}_*, \tilde{\Pi}_*) \leq \delta_{\mathsf{PCPP}}$ where $\widehat{\Pi}_* := \mathbf{Enc}(\tilde{\mathbb{i}}[1], \ldots, \tilde{\mathbb{i}}[\mathsf{k}], \tilde{\mathbb{w}})$. Additionally we have that for every $i \in [\mathsf{k}]$, $\Delta(\widehat{\pi}_i, \tilde{\pi}_i) \leq \delta_{\mathsf{PCPP}}$ where $\widehat{\pi}_i := \mathbf{Enc}(\tilde{\mathbb{i}}[i])$. This will imply the claim since the distance of the error-correcting code is at most $\delta_{\mathsf{PCP}} < \delta_{\mathsf{ECC}}$, and so $\widehat{\Pi}_*$ and $\tilde{\Pi}_*$ decode to the same value. Similarly $\widehat{\pi}_i$ and $\tilde{\pi}_i$ decode to the same value.

Suppose towards contradiction that there exists $i \in [\mathsf{k}]$ such that for every pair $\widehat{\pi}_i$ and $\widehat{\Pi}_*$ such that $C_i(\widehat{\pi}_i, \widehat{\Pi}_*) = 1$ (which implies that they have the required consistency), at least one of the following holds: (i) $\Delta(\widehat{\pi}_i, \tilde{\pi}_i) > \delta_{\mathsf{PCPP}}$, (ii) $\Delta(\widehat{\Pi}_*, \tilde{\Pi}_*) > \delta_{\mathsf{PCPP}}$. Then by the soundness property of the *multi-input* PCPP system:

$$\Pr\left[\, \mathbf{V}_{\mathsf{PCPP}}^{(\tilde{\pi}_i,\tilde{\Pi}_*),\tilde{\Pi}_i}(C_i) = 1 \,\right] \leq \beta_{\mathsf{PCPP}} \enspace ,$$

in contradiction to the assumption that $\mathbf{V}_{\mathsf{PCP}}$, that runs the above test in Item 2, accepts with probability greater than β_{PCPP}. □

We now prove decodability. Under the assumption that the verifier accepts with probability greater than β_{PCPP}, by Claim 5.9 and Claim 5.10, there exist $\tilde{\mathbb{i}}[1], \ldots, \tilde{\mathbb{i}}[\mathsf{k}]$ and $\tilde{\mathbb{w}}$ such that:

1. $(\tilde{\mathbb{i}}[1], \ldots, \tilde{\mathbb{i}}[\mathsf{k}], \mathbb{x}, \tilde{\mathbb{w}}) \in R$.
2. $(\tilde{\mathbb{i}}[1], \ldots, \tilde{\mathbb{i}}[\mathsf{k}], \tilde{\mathbb{w}}) = \mathbf{Dec}(\tilde{\Pi}_*)$.
3. For every $i \in [\mathsf{k}]$: $\tilde{\mathbb{i}}[i] = \mathbf{Dec}(\tilde{\pi}_i)$.

Putting the above items together with the fact that the decoder simply decodes the encoded index that it receives, we have that:

$$(\mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_1), \ldots, \mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_{\mathsf{k}}), \mathbb{x}, \tilde{\mathbb{w}}) = (\mathbf{Dec}(\tilde{\pi}_1), \ldots, \mathbf{Dec}(\tilde{\pi}_{\mathsf{k}}), \mathbb{x}, \tilde{\mathbb{w}}) = (\tilde{\mathbb{i}}[1], \ldots, \tilde{\mathbb{i}}[\mathsf{k}], \mathbb{x}, \tilde{\mathbb{w}}) \in R \ .$$

Efficiency. We analyze the efficiency parameters of the PCP:

- *Indexer alphabet.* The indexer alphabet size is 2.

- *Prover alphabet.* The prover writes its proof in groups of k bits. The alphabet size is 2^{k}.

- *Indexer proof length.* $\mathbf{I}_{\mathsf{PCP}}$ uses **Enc** on a bit-string of length $||\mathbb{i}[i]|$, so the proof length is $\mathsf{rt}_{\mathsf{ECC}}(|\mathbb{i}[i]|)$.

- *Prover proof length.* $\mathbf{P}_{\mathsf{PCP}}$ outputs the encoding of the string $\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}], \mathbb{w}$, and outputs $\mathsf{k}+1$ proofs for the PCP of proximity. The proofs in the consistency test part are interleaved into symbols. Thus the proof has length $2 \cdot \mathsf{l}_{\mathsf{PCPP}} + \mathsf{rt}_{\mathsf{ECC}}(|\mathbb{w}| + \sum_{i=1}^{\mathsf{k}} |\mathbb{i}[i]|)$.

- *Query complexity.* $\mathbf{V}_{\mathsf{PCP}}$ makes $\mathsf{q}_{\mathsf{PCPP}}$ queries to each of the indexer proofs in the consistency tests. The consistency check is done k times with the same randomness, and the same circuit family $\mathcal{C}$. The PCPP system is oblivious, and so all of these PCPPs make queries to exactly the same locations – which are bundled together into one symbol by the prover. Thus this test makes only $\mathsf{q}_{\mathsf{PCP}}$ queries. The verifier additionally makes $\mathsf{q}_{\mathsf{PCPP}}$ queries to the prover proof in the membership test – a total of $2 \cdot \mathsf{q}_{\mathsf{PCPP}}$.

- *Randomness complexity.* $\mathbf{V}_{\mathsf{PCP}}$ runs the membership test with randomness $\mathsf{r}_{\mathsf{PCPP}}$, chooses new PCPP randomness and runs each of the consistency checks *with the same randomness.* Therefore it uses $2 \cdot \mathsf{r}_{\mathsf{PCPP}}$ random bits.

- *Indexer running time.* $\mathbf{I}_{\mathsf{PCP}}$ encodes $\mathbb{i}[i]$ in time $\mathsf{et}_{\mathsf{ECC}}(|\mathbb{i}[i]|)$.

- *Prover running time.* $\mathbf{P}_{\mathsf{PCP}}$ encodes a string of length $|\mathbb{w}| + \sum_{i=1}^{\mathsf{k}} |\mathbb{i}[i]|$ in time $\mathsf{et}_{\mathsf{ECC}}(|\mathbb{w}| + \sum_{i=1}^{\mathsf{k}} |\mathbb{i}[i]|)$ and computes $\mathsf{k}+1$ PCP of proximity proofs, each in time $\mathsf{pt}_{\mathsf{PCPP}}$. All together time $\mathsf{et}_{\mathsf{ECC}}(|\mathbb{w}| + \sum_{i=1}^{\mathsf{k}} |\mathbb{i}[i]|) + (\mathsf{k}+1) \cdot \mathsf{pt}_{\mathsf{PCPP}}$.

- *Verifier running time.* $\mathbf{V}_{\mathsf{PCP}}$ runs the PCPP verifier $\mathsf{k}+1$ times, taking time $(\mathsf{k}+1) \cdot \mathsf{vt}_{\mathsf{PCPP}}$.

- *Decoder running time.* $\mathbf{D}_{\mathsf{PCP}}$ uses **Dec** to decode the encoding of $\mathbb{i}[i]$ a string and so its running time is $\mathsf{dt}_{\mathsf{ECC}}(|\mathbb{i}[i]|)$.

- *Adaptivity.* If $\mathbf{V}_{\mathsf{PCPP}}$ is non-adaptive then so is $\mathbf{V}_{\mathsf{PCP}}$.

□

6 Achieving constant query complexity per oracle

In this section we construct index-decodable PCPs that make $O(1)$ queries to every oracle. We begin in Section 6.1 by showing that proof composition preserves index-decodability when the outer index-decodable PCP is prover-robust. Then, in Section 6.2, we show how to transform the index-decodable PCP constructed in Section 5 into a prover-robust index-decodable PCP. Putting all this together, we prove the following theorem:

Theorem 6.1 (restatement of Theorem 2)**.** *Let $R = \{(\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}], \mathbb{x}, \mathbb{w})\}$ be a multi-indexed relation whose membership can be verified by a circuit C. Then R has a non-adaptive index-decodable PCP $\mathsf{PCP} = (\mathbf{I}_{\mathsf{PCP}}, \mathbf{P}_{\mathsf{PCP}}, \mathbf{V}_{\mathsf{PCP}}, \mathbf{D}_{\mathsf{PCP}})$ for R with the parameters below.*

Index-Decodable PCP for $(\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}], \mathbb{x}, \mathbb{w}) \in R$	
Indexer proof length (per proof)	$O(\lvert\mathbb{i}[i]\rvert)$
Prover proof length	$\mathrm{poly}(\lvert C\rvert)$
Alphabet size	2
Queries per oracle	$O(1)$
Randomness	$O(\log\lvert C\rvert)$
Decodability bound	$O(1)$
Indexer running time	$\tilde{O}(\lvert\mathbb{i}[i]\rvert)$
Prover running time	$\mathrm{poly}(\lvert C\rvert)$
Verifier running time	$\tilde{O}(\lvert C\rvert)$
Decoder running time	$\tilde{O}(\lvert\mathbb{i}[i]\rvert)$

Proof. We take the robust index-decodable PCP of Theorem 6.4 and compose it using Theorem 6.2 with the PCP of proximity achieved by Theorem 3.5. □

6.1 Proof composition preserves index-decodability

We show that, in proof composition of PCPs [AS98], if the outer PCP of the proof is index-decodable then the composed PCP is also index-decodable (given the outer index-decodable PCP has good enough prover-robustness).

Theorem 6.2. *Let $\mathsf{PCP}_{\mathsf{out}} = (\mathbf{I}_{\mathsf{out}}, \mathbf{P}_{\mathsf{out}}, (\mathbf{V}^{\mathsf{qry}}_{\mathsf{out}}, \mathbf{V}^{\mathsf{dc}}_{\mathsf{out}}), \mathbf{D}_{\mathsf{out}})$ be a non-adaptive index-decodable PCP for a relation R with prover-robustness σ_{out} and $\mathsf{PCP}_{\mathrm{in}} = (\mathbf{P}_{\mathrm{in}}, \mathbf{V}_{\mathrm{in}})$ be a non-adaptive PCP of proximity for NP with proximity $\delta_{\mathrm{in}} \leq \sigma_{\mathsf{out}}$. Then Construction 6.3 is a non-adaptive index-decodable PCP $\mathsf{PCP} = (\mathbf{I}_{\mathsf{PCP}}, \mathbf{P}_{\mathsf{PCP}}, \mathbf{V}_{\mathsf{PCP}}, \mathbf{D}_{\mathsf{PCP}})$ for R with the parameters below.*

Index-Decodable PCP for $(\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}], \mathbb{x}, \mathbb{w}) \in R$	
Indexer proof length	$\mathsf{l}_{\mathsf{out},\mathbf{I}}$
Proof length	$\mathsf{l}_{\mathsf{out},\mathbf{P}}$
Alphabet size	λ_{out}
Queries to indexer proof	$\mathsf{iq}_{\mathsf{out}}$
Queries to prover proof	$\mathsf{pq}_{\mathsf{out}}$
Randomness	$\mathsf{r}_{\mathsf{out}}$
Prover-robustness	σ_{out}
Decodability bound	κ_{out}
Indexer running time	$\mathsf{it}_{\mathsf{out}}$
Prover running time	$\mathsf{pt}_{\mathsf{out}}$
Verifier running time	$\mathsf{vt}_{\mathsf{out}}$
Decoder running time	$\mathsf{dt}_{\mathsf{out}}$

+

PCPP for $\mathbf{V}^{\mathsf{dc}}_{\mathsf{out}}$	
Proof length	l_{in}
Alphabet size	λ_{in}
Queries	q_{in}
Randomness	r_{in}
Proximity	δ_{in}
Soundness error	β_{in}
Prover running time	$\mathsf{pt}_{\mathrm{in}}$
Verifier running time	$\mathsf{vt}_{\mathrm{in}}$

$\longrightarrow$

Index-Decodable PCP for $(\mathbb{i}[1],\ldots,\mathbb{i}[\mathsf{k}],\mathbb{x},\mathbb{w}) \in R$	
Indexer proof length	$\mathsf{l}_{\mathsf{out},\mathbf{I}}$
Prover proof length	$\mathsf{l}_{\mathsf{out},\mathbf{P}} + 2^{\mathsf{r}_{\mathsf{out}}} \cdot \mathsf{l}_{\mathrm{in}}$
Alphabet size	$\max\{\lambda_{\mathsf{out}}, \lambda_{\mathrm{in}}\}$
Queries to indexer proof	$\mathsf{iq}_{\mathsf{out}}$
Queries to prover proof	q_{in}
Randomness	$\mathsf{r}_{\mathsf{out}} + \mathsf{r}_{\mathrm{in}}$
Decodability bound	$\kappa_{\mathsf{out}} + (1-\kappa_{\mathsf{out}}) \cdot \beta_{\mathrm{in}}$
Indexer running time	$\mathsf{it}_{\mathsf{out}}$
Prover running time	$\mathsf{pt}_{\mathsf{out}} + 2^{\mathsf{r}_{\mathsf{out}}} \cdot (\mathsf{vt}_{\mathsf{out}} + \mathsf{pt}_{\mathrm{in}})$
Verifier running time	$\mathsf{vt}_{\mathsf{out}} + \mathsf{vt}_{\mathrm{in}}$
Decoder running time	$\mathsf{dt}_{\mathsf{out}}$

Construction 6.3. We construct the index-decodable PCP $\mathsf{PCP} = (\mathbf{I}_{\mathsf{PCP}}, \mathbf{P}_{\mathsf{PCP}}, \mathbf{V}_{\mathsf{PCP}}, \mathbf{D}_{\mathsf{PCP}})$ below.

- $\mathbf{I}_{\mathsf{PCP}}(\mathbb{i}[i])$: Output $\pi_i := \mathbf{I}_{\mathsf{out}}(\mathbb{i}[i])$.
- $\mathbf{P}_{\mathsf{PCP}}(\mathbb{i}[1],\ldots,\mathbb{i}[\mathsf{k}],\mathbb{x},\mathbb{w})$:
  1. For every $i \in [\mathsf{k}]$, compute the indexer proof $\pi_i := \mathbf{I}_{\mathsf{PCP}}(\mathbb{i}[i])$.
  2. Compute the prover proof $\Pi_{\mathsf{out}} := \mathbf{P}_{\mathsf{out}}(\mathbb{i}[1],\ldots,\mathbb{i}[\mathsf{k}],\mathbb{x},\mathbb{w})$ and set $\mathbf{\Pi}_{\mathbb{i}} := (\pi_1,\ldots,\pi_{\mathsf{k}})$.
  3. For every $\rho_{\mathsf{out}} \in \{0,1\}^{\mathsf{r}_{\mathsf{out}}}$, compute $(Q_{\mathbb{i}}, Q_*) := \mathbf{V}^{\mathsf{qry}}_{\mathsf{out}}(\mathbb{x},\rho_{\mathsf{out}})$ and set $C_{\mathrm{in}} := \mathbf{V}^{\mathsf{dc}}_{\mathsf{out}}(\mathbb{x},\rho_{\mathsf{out}},\mathbf{\Pi}_{\mathbb{i}}[Q_{\mathbb{i}}],\cdot)$ ($\mathbf{V}^{\mathsf{dc}}_{\mathsf{out}}$ with instance, randomness and answers to queries to indexer proofs fixed to $\mathbb{x}$, ρ_{out} and $\mathbf{\Pi}_{\mathbb{i}}[Q_{\mathbb{i}}]$) and $\mathbb{x}_{\mathrm{in}} := \Pi[Q_*]$. Compute the PCPP string $\Pi_{\mathrm{in}}[\rho_{\mathsf{out}}] := \mathbf{P}_{\mathrm{in}}(C_{\mathrm{in}}, \mathbb{x}_{\mathrm{in}})$.
  4. Output $\Pi := (\Pi_{\mathsf{out}}, \Pi_{\mathrm{in}})$ where $\Pi_{\mathrm{in}} := \{\Pi_{\mathrm{in}}[\rho_{\mathsf{out}}]\}_{\rho_{\mathsf{out}} \in \{0,1\}^{\mathsf{r}_{\mathsf{out}}}}$.
- $\mathbf{V}^{\tilde{\pi}_1,\ldots,\tilde{\pi}_{\mathsf{k}},\tilde{\Pi}}_{\mathsf{PCP}}(\mathbb{x})$:
  1. Parse $\tilde{\Pi} = (\tilde{\Pi}_{\mathsf{out}}, \{\tilde{\Pi}_{\mathrm{in}}[\rho_{\mathsf{out}}]\}_{\rho_{\mathsf{out}} \in \{0,1\}^{\mathsf{r}_{\mathsf{out}}}})$ and set $\tilde{\mathbf{\Pi}}_{\mathbb{i}} := (\tilde{\pi}_1,\ldots,\tilde{\pi}_{\mathsf{k}})$ for convenience.
  2. Sample randomness $\rho_{\mathsf{out}} \leftarrow \{0,1\}^{\mathsf{r}_{\mathsf{out}}}$ and compute the query sets $(Q_{\mathbb{i}}, Q_*) := \mathbf{V}^{\mathsf{qry}}_{\mathsf{out}}(\mathbb{x},\rho_{\mathsf{out}})$.
  3. Set $C_{\mathrm{in}} := \mathbf{V}^{\mathsf{dc}}_{\mathsf{out}}(\mathbb{x},\rho_{\mathsf{out}},\mathbf{\Pi}_{\mathbb{i}}[Q_{\mathbb{i}}],\cdot)$ and check that $\mathbf{V}^{\tilde{\Pi}_{\mathsf{out}}[Q_*],\tilde{\Pi}_{\mathrm{in}}[\rho_{\mathsf{out}}]}_{\mathrm{in}}(C_{\mathrm{in}}) = 1$.
- $\mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_i)$: Output $\tilde{\pi}_i := \mathbf{D}_{\mathsf{out}}(\tilde{\pi}_i)$.

Proof of Theorem 6.2. First we argue completeness, then argue decodability, and, finally, analyze efficiency measures of the resulting PCP.

Completeness. Fix $(\mathbb{i}[1],\ldots,\mathbb{i}[\mathsf{k}],\mathbb{x},\mathbb{w}) \in R$ and let $\mathbf{\Pi}_{\mathbb{i}} = (\pi_1,\ldots,\pi_{\mathsf{k}})$, Π_{out} and $\{\Pi_{\mathrm{in}}[\rho_{\mathsf{out}}]\}_{\rho_{\mathsf{out}} \in \{0,1\}^{\mathsf{r}_{\mathsf{out}}}}$ be the proofs output by the honest indexer $\mathbf{I}_{\mathsf{PCP}}$ and prover $\mathbf{P}_{\mathsf{PCP}}$. By the (perfect) completeness of the outer PCP,

$$\Pr\left[\, \mathbf{V}^{\mathsf{dc}}_{\mathsf{out}}(\mathbb{x},\rho_{\mathsf{out}},\mathbf{\Pi}_{\mathbb{i}}[Q_{\mathbb{i}}],\Pi_{\mathsf{out}}[Q_*]) = 1 \mid (Q_{\mathbb{i}},Q_*) \leftarrow \mathbf{V}^{\mathsf{qry}}_{\mathsf{PCP}}(\mathbb{x},\rho) \,\right] = 1 \ .$$

Hence, for every $\rho_{\mathsf{out}} \in \{0,1\}^{\mathsf{r}_{\mathsf{out}}}$, by the (perfect) completeness of the inner PCP (of proximity), for $\Pi_{\mathrm{in}}[\rho_{\mathsf{out}}]$ output by $\mathbf{P}_{\mathrm{in}}$ given circuit $C_{\mathrm{in}} := \mathbf{V}^{\mathsf{dc}}_{\mathsf{out}}(\mathbb{x},\rho_{\mathsf{out}},\mathbf{\Pi}_{\mathbb{i}}[Q_{\mathbb{i}}],\cdot)$ and input $\Pi_{\mathsf{out}}[Q_*]$, it holds that

$$\Pr_{\rho_{\mathrm{in}}}\left[\, \mathbf{V}^{\Pi_{\mathsf{out}}[Q_*],\Pi_{\mathrm{in}}[\rho_{\mathsf{out}}]}_{\mathrm{in}}(C_{\mathrm{in}};\rho_{\mathrm{in}}) = 1 \,\right] = 1 \ .$$

We conclude that the composed PCP also has perfect completeness:

$$\Pr\left[\, \mathbf{V}^{\pi_1,\ldots,\pi_{\mathsf{k}},\Pi}_{\mathsf{PCP}}(\mathbb{x}) = 1 \,\right] = 1 \ .$$

Decodability. Fix $\mathbb{x}$ and malicious proofs $\tilde{\mathbf{\Pi}}_{\mathbb{i}} = (\tilde{\pi}_1, \ldots, \tilde{\pi}_{\mathsf{k}})$ and $\tilde{\Pi} = (\tilde{\Pi}_{\mathsf{out}}, \{\tilde{\Pi}_{\mathrm{in}}[\rho_{\mathsf{out}}]\})$. Suppose that:

$$\Pr\left[\, \mathbf{V}_{\mathsf{PCP}}^{\tilde{\pi}_1,\ldots,\tilde{\pi}_{\mathsf{k}},\tilde{\Pi}}(\mathbb{x}) = 1 \,\right] > \kappa_{\mathsf{out}} + (1-\kappa_{\mathsf{out}}) \cdot \beta_{\mathrm{in}} \ \ .$$

For every choice of randomness $\rho_{\mathsf{out}} \in \{0,1\}^{\mathsf{r}_{\mathsf{out}}}$ let $(Q_{\mathbb{i}}, Q_*) := \mathbf{V}_{\mathsf{out}}^{\mathsf{qry}}(\mathbb{x}, \rho_{\mathsf{out}})$ and set $\tilde{A}_{\rho_{\mathsf{out}}} := \tilde{\Pi}_{\mathsf{out}}[Q_*]$. We consider the two possibilities for $\tilde{A}_{\rho_{\mathsf{out}}}$.

1. There exists some $\tilde{A}'$ such that $\Delta(\tilde{A}', \tilde{A}_{\rho_{\mathsf{out}}}) \le \sigma_{\mathsf{out}}$ and $\mathbf{V}_{\mathsf{out}}^{\mathsf{dc}}(\mathbb{x}, \rho_{\mathsf{out}}, \mathbf{\Pi}_{\mathbb{i}}[Q_{\mathbb{i}}], \tilde{A}') = 1$. In this case, we cannot rule out that $\mathbf{V}_{\mathrm{in}}$ accepts with high probability. So we can trivially write
$$\Pr_{\rho_{\mathrm{in}}}\left[\, \mathbf{V}_{\mathrm{in}}^{\tilde{A}_{\rho_{\mathsf{out}}},\tilde{\Pi}_{\mathrm{in}}[\rho_{\mathsf{out}}]}(\mathbf{V}_{\mathsf{out}}^{\mathsf{dc}}(\mathbb{x}, \rho_{\mathsf{out}}, \mathbf{\Pi}_{\mathbb{i}}[Q_{\mathbb{i}}], \cdot); \rho_{\mathrm{in}}) = 1 \,\right] \le 1 \ \ .$$

2. There *does not* exist a set $\tilde{A}'$ such that $\Delta(\tilde{A}', \tilde{A}_{\rho_{\mathsf{out}}}) \le \sigma_{\mathsf{out}}$ and $\mathbf{V}_{\mathsf{out}}^{\mathsf{dc}}(\mathbb{x}, \rho_{\mathsf{out}}, \mathbf{\Pi}_{\mathbb{i}}[Q_{\mathbb{i}}], \tilde{A}') = 1$. Since $\delta_{\mathrm{in}} \le \sigma_{\mathsf{out}}$, $\tilde{A}_{\rho_{\mathsf{out}}}$ is far enough from any true claim that the proximity soundness property of $\mathbf{V}_{\mathrm{in}}$ applies:
$$\Pr_{\rho_{\mathrm{in}}}\left[\, \mathbf{V}_{\mathrm{in}}^{\tilde{A}_{\rho_{\mathsf{out}}},\tilde{\Pi}_{\mathrm{in}}[\rho_{\mathsf{out}}]}(\mathbf{V}_{\mathsf{out}}^{\mathsf{dc}}(\mathbb{x}, \rho_{\mathsf{out}}, \mathbf{\Pi}_{\mathbb{i}}[Q_{\mathbb{i}}], \cdot); \rho_{\mathrm{in}}) = 1 \,\right] \le \beta_{\mathrm{in}} \ \ .$$

Hence, letting p be the probability that ρ_{out} induces a choice of queries whose answers $\tilde{A}_{\rho_{\mathsf{out}}}$ are such that Item 1 occurs, we have that $\mathbf{V}_{\mathsf{PCP}}$ accepts with probability at most $p + (1-p)\cdot\beta_{\mathrm{in}}$. By assumption, the probability that $\mathbf{V}_{\mathsf{PCP}}$ accepts is greater than $\kappa_{\mathsf{out}} + (1-\kappa_{\mathsf{out}})\cdot\beta_{\mathrm{in}}$. From this we can infer that $p > \kappa_{\mathsf{out}}$. By the prover-robust decodability of the outer (index-decodable) PCP, we deduce that there exits a witness $\mathbb{w}$ such that:

$$(\mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_1), \ldots, \mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_{\mathsf{k}}), \mathbb{x}, \mathbb{w}) = (\mathbf{D}_{\mathsf{out}}(\tilde{\pi}_1), \ldots, \mathbf{D}_{\mathsf{out}}(\tilde{\pi}_{\mathsf{k}}), \mathbb{x}, \mathbb{w}) \in R \ \ .$$

Efficiency. We analyze the efficiency parameters of the resulting PCP.

- *Alphabet.* The new PCP involves the alphabet of the outer index-decodable PCP, which has size λ_{out}, and the alphabet of the inner PCP of proximity, which has size λ_{in}. One can use the same alphabet to write both, in which case its size would be $\max\{\lambda_{\mathsf{out}}, \lambda_{\mathrm{in}}\}$.
- *Indexer proof length.* $\mathbf{I}_{\mathsf{PCP}}$ outputs the same indexer proofs as $\mathbf{I}_{\mathsf{out}}$, which are of length $\mathsf{l}_{\mathsf{out},\mathbf{I}}$.
- *Prover proof length.* $\mathbf{P}_{\mathsf{PCP}}$ sends the proof (of length $\mathsf{l}_{\mathsf{out},\mathbf{P}}$) of the index-decodable PCP and also, for every $\rho_{\mathsf{out}} \in \{0,1\}^{\mathsf{r}_{\mathsf{out}}}$, sends a proof (of length l_{in}) for the inner PCP for randomness ρ_{out}. Hence the total proof length is $\mathsf{l}_{\mathsf{out},\mathbf{P}} + 2^{\mathsf{r}_{\mathsf{out}}} \cdot \mathsf{l}_{\mathrm{in}}$.
- *Query complexity.* $\mathbf{V}_{\mathsf{PCP}}$ makes as many queries as $\mathbf{V}_{\mathrm{in}}$, which is q_{in}.
- *Randomness complexity.* $\mathbf{V}_{\mathsf{PCP}}$ samples randomness for $\mathbf{V}_{\mathsf{out}}$ and $\mathbf{V}_{\mathrm{in}}$, using $\mathsf{r}_{\mathsf{out}} + \mathsf{r}_{\mathrm{in}}$ random bits.
- *Indexer running time.* $\mathbf{I}_{\mathsf{PCP}}$ runs runs $\mathbf{I}_{\mathsf{out}}$, and so its running time is $\mathsf{it}_{\mathsf{out}}$.
- *Prover running time.* $\mathbf{P}_{\mathsf{PCP}}$ runs $\mathbf{P}_{\mathsf{out}}$ once and $\mathbf{V}_{\mathsf{out}}^{\mathsf{qry}}, \mathbf{P}_{\mathrm{in}}$ a total of $2^{\mathsf{r}_{\mathsf{out}}}$ times, and so its running time is $\mathsf{pt}_{\mathsf{out}} + 2^{\mathsf{r}_{\mathsf{out}}} \cdot (\mathsf{vt}_{\mathsf{out}} + \mathsf{pt}_{\mathrm{in}})$.
- *Verifier running time.* $\mathbf{V}_{\mathsf{PCP}}$ runs $\mathbf{V}_{\mathsf{out}}^{\mathsf{qry}}$ to compute its query locations and runs $\mathbf{V}_{\mathrm{in}}$ to decide, thereby running in time at most $\mathsf{vt}_{\mathsf{out}} + \mathsf{vt}_{\mathrm{in}}$.
- *Decoder running time.* $\mathbf{D}_{\mathsf{PCP}}$ equals $\mathbf{D}_{\mathsf{out}}$, and so its running time is $\mathsf{dt}_{\mathsf{out}}$.

□

6.2 Robustification

We now show how to get a prover-robust index-decodable PCP with a binary alphabet and large number of queries to the prover proof. This is later reduced by using proof composition.

Theorem 6.4. *Let $R = \{(\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}], \mathbb{x}, \mathbb{w})\}$ be a multi-indexed relation whose membership can be verified by a boolean circuit C. Then R a non-adaptive prover-robust index-decodable PCP with the following parameters:*

Prover-robust Index-Decodable PCP for $(\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}], \mathbb{x}, \mathbb{w}) \in R$	
Indexer proof length (per proof)	$O(\lvert\mathbb{i}[i]\rvert)$
Proof length	$\mathrm{poly}(\lvert C\rvert)$
Alphabet size	2
Queries to indexer proof	$O(1)$
Queries to prover proof	$O(\mathsf{k})$
Randomness	$O(\log \lvert C\rvert)$
Prover-robustness	$\Omega(1)$
Decodability bound	$O(1)$
Indexer running time	$\tilde{O}(\lvert\mathbb{i}[i]\rvert)$
Prover running time	$\mathrm{poly}(\lvert C\rvert)$
Verifier running time	$\tilde{O}(\lvert C\rvert)$
Decoder running time	$\tilde{O}(\lvert\mathbb{i}[i]\rvert)$

We first describe the construction, and then prove Theorem 6.4.

Construction 6.5. Let $\mathsf{PCP} = (\mathbf{I}_{\mathsf{PCP}}, \mathbf{P}_{\mathsf{PCP}}, (\mathbf{V}^{\mathsf{qry}}_{\mathsf{PCP}}, \mathbf{V}^{\mathsf{dc}}_{\mathsf{PCP}}), \mathbf{D}_{\mathsf{PCP}})$ be a non-adaptive index-decodable PCP for a relation R and $\mathsf{ECC} = (\mathbf{Enc}, \mathbf{Dec})$ be a $(r, \delta_{\mathsf{ECC}})$-code with $r(k) = c \cdot k$ for constant c. Let $\mathsf{l}_{\mathsf{PCP},\mathbf{P}}$ and $\Sigma_{\mathbf{P}}$ be the prover proof length and alphabet respectively.

- $\mathbf{I}(\mathbb{i}[i])$: Output $\pi_i := \mathbf{I}_{\mathsf{PCP}}(\mathbb{i}[i])$.
- $\mathbf{P}(\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}], \mathbb{x}, \mathbb{w})$: Compute $\Pi' := \mathbf{P}_{\mathsf{PCP}}(\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}], \mathbb{x}, \mathbb{w})$ and output $\Pi := \mathbf{Enc}(\Pi', \mathsf{l}_{\mathsf{PCP},\mathbf{P}})$.
- $\mathbf{V}^{\tilde{\pi}_1, \ldots, \tilde{\pi}_{\mathsf{k}}, \tilde{\Pi}}(\mathbb{x})$:
  1. Sample randomness ρ and generate $(Q_{\mathbb{i}}, Q_*) \leftarrow \mathbf{V}^{\mathsf{qry}}_{\mathsf{PCP}}(\mathbb{x}, \rho)$ the queries the PCP verifier makes given instance $\mathbb{x}$ and randomness ρ. $Q_{\mathbb{i}}$ are the queries made to the indexer proofs and Q_* are the queries made to the prover proof.
  2. Let $A := \{\tilde{\Pi}[q] \mid q \in Q_*\}$ and $A_{\mathsf{PCP}} := \{\mathbf{Dec}(a) \mid a \in A\}$. Let $\tilde{\mathbf{\Pi}}_{\mathbb{i}} = (\tilde{\pi}_1, \ldots, \tilde{\pi}_{\mathsf{k}})$. Accept if and only if $\mathbf{V}^{\mathsf{dc}}_{\mathsf{PCP}}(\mathbb{x}, \rho, \tilde{\mathbf{\Pi}}_{\mathbb{i}}[Q_{\mathbb{i}}], A_{\mathsf{PCP}}) = 1$.
- $\mathbf{D}(\tilde{\pi}_i)$: Output $\tilde{\mathbb{i}}[i] := \mathbf{D}_{\mathsf{PCP}}(\mathbb{i}[i])$.

Proof of Theorem 6.4. We use Construction 6.5 where the index-decodable PCP used is the one of Theorem 5.1. We use an error correcting code with parameters as in Theorem 3.4. We first argue completeness, then decodability and, finally, we analyze the other complexity measures of the resulting PCP.

Completeness. Fix $(\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}], \mathbb{x}, \mathbb{w}) \in R$. Let $\pi_1, \ldots, \pi_{\mathsf{k}}$ and Π be the proofs generated by the indexer and the prover respectively. Since the prover is honest, $\mathbf{P}_{\mathsf{PCP}}(\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}], \mathbb{x}, \mathbb{w}) = \mathbf{Dec}(\Pi, \mathsf{l}_{\mathsf{PCP},\mathbf{P}})$. Hence, letting $\Pi' := \mathbf{P}_{\mathsf{PCP}}(\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}], \mathbb{x}, \mathbb{w})$ and $\tilde{\mathbf{\Pi}}_{\mathbb{i}} = (\tilde{\pi}_1, \ldots, \tilde{\pi}_{\mathsf{k}})$, we have:

$$\Pr\left[\, \mathbf{V}^{\pi_1, \ldots, \pi_{\mathsf{k}}, \Pi}(\mathbb{x}) = 1 \,\right] = \Pr_{\rho}\left[\mathbf{V}^{\mathsf{dc}}_{\mathsf{PCP}}(\mathbb{x}, \rho, \tilde{\mathbf{\Pi}}_{\mathbb{i}}[Q_{\mathbb{i}}], A_{\mathsf{PCP}}) = 1 \;\middle|\; \begin{array}{c} (Q_{\mathbb{i}}, Q_*) \leftarrow \mathbf{V}^{\mathsf{qry}}_{\mathsf{PCP}}(\mathbb{x}, \rho) \\ A_{\mathsf{PCP}} := \{\, \mathbf{Dec}(\Pi[q]) \mid q \in Q_* \,\} \end{array} \right]$$

$$
\begin{aligned}
&= \Pr_{\rho}\left[\mathbf{V}^{\mathsf{dc}}_{\mathsf{PCP}}(\mathbb{x}, \rho, \tilde{\mathbf{\Pi}}_{\mathbb{i}}[Q_{\mathbb{i}}], A_{\mathsf{PCP}}) = 1 \;\middle|\; \begin{array}{c} (Q_{\mathbb{i}}, Q_*) \leftarrow \mathbf{V}^{\mathsf{qry}}_{\mathsf{PCP}}(\mathbb{x}, \rho) \\ A_{\mathsf{PCP}} := \{\, \Pi'[q] \mid q \in Q_* \,\} \end{array} \right] \\
&= \Pr\left[\mathbf{V}^{\pi_1,\ldots,\pi_{\mathsf{k}},\Pi'}_{\mathsf{PCP}}(\mathbb{x}) = 1 \right] \\
&= 1 \ .
\end{aligned}
$$

Prover-robust decodability. Fix an instance $\mathbb{x}$ and proofs $\tilde{\mathbf{\Pi}}_{\mathbb{i}} = (\tilde{\pi}_1, \ldots, \tilde{\pi}_{\mathsf{k}})$ and $\tilde{\Pi}$. Denote by $\kappa_{\mathsf{PCP}} = O(1)$, and $\mathsf{q}_{\mathsf{PCP}} = O(1)$ the decodability bound and number of queries to the prover proof respectively. Let $\delta_{\mathsf{ECC}} = \Omega(1)$ be the distance of the error correcting code such that $\frac{\delta_{\mathsf{ECC}}}{\mathsf{q}_{\mathsf{PCP}}} = \Omega(1)$. Denote by $\mathbf{V}^{\mathsf{qry}}$ and $\mathbf{V}^{\mathsf{dc}}$ the query generation and decision predicate of $\mathbf{V}$ respectively. Suppose that

$$
\Pr_{\rho}\left[\exists\, A' \text{ s.t. } \mathbf{V}^{\mathsf{dc}}(\mathbb{x}, \rho, \tilde{\mathbf{\Pi}}_{\mathbb{i}}[Q_{\mathbb{i}}], A') = 1 \ \wedge\ \Delta(A', A) \le \frac{\delta_{\mathsf{ECC}}}{\mathsf{q}_{\mathsf{PCP}}} \;\middle|\; \begin{array}{c} (Q_{\mathbb{i}}, Q_*) \leftarrow \mathbf{V}^{\mathsf{qry}}(\mathbb{x}, \rho) \\ A := \{\, \tilde{\Pi}[q] \mid q \in Q_* \,\} \end{array} \right] > \kappa_{\mathsf{PCP}} \ .
$$

We show that this implies that there exists $\mathbb{w}$ such that:

$$
(\mathbf{D}(\tilde{\pi}_1), \ldots, \mathbf{D}(\tilde{\pi}_{\mathsf{k}}), \mathbb{x}, \mathbb{w}) \in R \ .
$$

Notice that

$$
\Pr_{\rho}\left[\exists\, A' \text{ s.t. } \mathbf{V}^{\mathsf{dc}}(\mathbb{x}, \rho, \tilde{\mathbf{\Pi}}_{\mathbb{i}}[Q_{\mathbb{i}}], A') = 1 \ \wedge\ \Delta(A', A) \le \frac{\delta_{\mathsf{ECC}}}{\mathsf{q}_{\mathsf{PCP}}} \;\middle|\; \begin{array}{c} (Q_{\mathbb{i}}, Q_*) \leftarrow \mathbf{V}^{\mathsf{qry}}(\mathbb{x}, \rho) \\ A := \{\, \tilde{\Pi}[q] \mid q \in Q_* \,\} \end{array} \right] \ ,
$$

is equal to

$$
\Pr_{\rho}\left[\exists\, A' \text{ s.t. } \mathbf{V}^{\mathsf{dc}}_{\mathsf{PCP}}\left(\mathbb{x}, \rho, \tilde{\mathbf{\Pi}}_{\mathbb{i}}[Q_{\mathbb{i}}], \{\mathbf{Dec}(a) \mid a \in A'\}\right) = 1 \ \wedge\ \Delta(A', A) \le \frac{\delta_{\mathsf{ECC}}}{\mathsf{q}_{\mathsf{PCP}}} \;\middle|\; \begin{array}{c} (Q_{\mathbb{i}}, Q_*) \leftarrow \mathbf{V}^{\mathsf{qry}}_{\mathsf{PCP}}(\mathbb{x}, \rho) \\ A := \{\, \tilde{\Pi}[q] \mid q \in Q_* \,\} \end{array} \right] \ .
$$

Fix randomness ρ. Let $(Q_{\mathbb{i}}, Q_*) \leftarrow \mathbf{V}^{\mathsf{qry}}_{\mathsf{PCP}}(\mathbb{x}, \rho)$ and $A = \{\, \mathbf{\Pi}'[q] \mid q \in Q_* \,\}$. Suppose that A' is the set closest to A such that $\mathbf{V}^{\mathsf{dc}}(\mathbb{x}, \rho, \tilde{\mathbf{\Pi}}_{\mathbb{i}}[Q_{\mathbb{i}}], A') = 1$ and that $\Delta(A', A) \le \frac{\delta_{\mathsf{ECC}}}{\mathsf{q}_{\mathsf{PCP}}}$. By a simple counting argument it must be that for every i, $\Delta(A'[i], A[i]) \le \delta_{\mathsf{ECC}}$, and so $\mathbf{Dec}(A[i]) = \mathbf{Dec}(A'[i])$. Moreover, since $A[i] = \tilde{\Pi}[Q_*[i]]$, it follows that $\mathbf{Dec}(A'[i]) = \mathbf{Dec}(\tilde{\Pi}[Q[_*i]])$. Hence, considering the decoded proof $\tilde{\Pi}' = \mathbf{Dec}(\tilde{\Pi}, \mathsf{l}_{\mathsf{PCP},\mathbf{P}})$, we have:

$$
\begin{aligned}
&\Pr_{\rho}\left[\exists\, A' \text{ s.t. } \mathbf{V}^{\mathsf{dc}}_{\mathsf{PCP}}\left(\mathbb{x}, \rho, \tilde{\mathbf{\Pi}}_{\mathbb{i}}[Q_{\mathbb{i}}], \{\mathbf{Dec}(a) \mid a \in A'\}\right) = 1 \ \wedge\ \Delta(A', A) \le \frac{\delta_{\mathsf{ECC}}}{\mathsf{q}_{\mathsf{PCP}}} \;\middle|\; \begin{array}{c} (Q_{\mathbb{i}}, Q_*) \leftarrow \mathbf{V}^{\mathsf{qry}}(\mathbb{x}, \rho) \\ A := \{\, \tilde{\Pi}[q] \mid q \in Q_* \,\} \end{array} \right] \\
&\le \Pr_{\rho}\left[\mathbf{V}^{\mathsf{dc}}_{\mathsf{PCP}}(\mathbb{x}, \rho, \tilde{\mathbf{\Pi}}_{\mathbb{i}}[Q_{\mathbb{i}}], A_{\mathsf{PCP}}) = 1 \;\middle|\; \begin{array}{c} (Q_{\mathbb{i}}, Q_*) \leftarrow \mathbf{V}^{\mathsf{qry}}(\mathbb{x}, \rho) \\ A_{\mathsf{PCP}} = \{\, \tilde{\Pi}'[q] \mid q \in Q_* \,\} \end{array} \right] \ ,
\end{aligned}
$$

and so

$$
\Pr\left[\mathbf{V}^{\tilde{\pi}_1,\ldots,\tilde{\pi}_{\mathsf{k}},\tilde{\Pi}'}_{\mathsf{PCP}}(\mathbb{x}) \right] = \Pr_{\rho}\left[\mathbf{V}^{\mathsf{dc}}_{\mathsf{PCP}}(\mathbb{x}, \rho, \tilde{\mathbf{\Pi}}_{\mathbb{i}}[Q_{\mathbb{i}}], A_{\mathsf{PCP}}) = 1 \;\middle|\; \begin{array}{c} (Q_{\mathbb{i}}, Q_*) \leftarrow \mathbf{V}^{\mathsf{qry}}_{\mathsf{PCP}}(\mathbb{x}, \rho) \\ A_{\mathsf{PCP}} = \{\, \tilde{\mathbf{\Pi}}_*[q] \mid q \in Q \,\} \end{array} \right] > \kappa_{\mathsf{PCP}} \ .
$$

By decodability of the index-decodable PCP, it follows that there exists a witness $\mathbb{w}$ such that

$$
(\mathbf{D}(\tilde{\pi}_1), \ldots, \mathbf{D}(\tilde{\pi}_{\mathsf{k}}), \mathbb{x}, \mathbb{w}) = (\mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_1), \ldots, \mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_{\mathsf{k}}), \mathbb{x}, \mathbb{w}) \in R \ .
$$

Efficiency. We analyze the efficiency parameters of the resulting PCP.

- *Alphabet.* The alphabet used by the system is binary as the error correcting code returns strings of bits.

- *Indexer proof length.* The indexer proof is of length $O(|\mathbb{i}[i]|)$.

- *Prover proof length.* The prover wraps an error-correcting code with constant rate around its proofs. Therefore the proof length is preserved up to constant factors and is $\mathrm{poly}(|C|)$.

- *Query complexity.* Queries to the indexer proofs remain unchanged. For each of the $O(1)$ queries made to the prover proof of the original PCP, the verifier queries $O(\mathsf{k})$ bits. Hence it makes $O(\mathsf{k})$ queries to the prover proof.

- *Randomness complexity.* The verifier uses the same number of random bits as the original verifier, $O(\log |C|)$.

- *Indexer running time.* The indexer simply runs $\mathbf{I}_{\mathsf{PCP}}$ and so has running time $\tilde{O}(|\mathbb{i}[i]|)$.

- *Prover running time.* The prover runs $\mathbf{P}_{\mathsf{PCP}}$ and encodes every k-bit symbol of the proof in time that is quasi-polynomial in k. Hence it runs in time $\mathrm{poly}(|C|)$.

- *Verifier running time.* The verifier runs the original verifier in time $\tilde{O}(|C|)$, and the efficient decoding procedure of the error correcting code to decode $O(1)$ symbols of length k bits. This takes time $\tilde{O}(\mathsf{k}) = \tilde{O}(|C|)$. Thus, the verifier runs in time $\tilde{O}(|C|)$.

- *Decoder running time.* The decoder simply runs the original decoder and so runs in time $\tilde{O}(|\mathbb{i}[i]|)$.

- *Adaptivity.* The original index-decodable is non-adaptive, and all that this transformation does is to error-correct queries to the prover proof. Thus the queries are still independent of the proof, and the resulting PCP is non-adaptive.

□

7 Transforming IPs into IOPs

We show how to use index-decodable PCPs to transform public-coin IPs into IOPs. We then combine this with the index-decodable PCP from Section 6 to obtain our main theorem. Unless otherwise stated, all of the interactive proofs in this section are assumed to have no decision randomness.

Theorem 7.1 (restatement of Theorem 1)**.** *Let* $\mathsf{IP} = (\mathbf{P}_{\mathsf{IP}}, \mathbf{V}_{\mathsf{IP}})$ *be a public-coin IP for a relation* $R = \{(\mathbb{x}, \mathbb{w})\}$*. Then there exists a public-coin IOP* $\mathsf{IOP} = (\mathbf{P}_{\mathsf{IOP}}, \mathbf{V}_{\mathsf{IOP}})$ *for* R *with the parameters below.*

IP $(\mathbf{P}_{\mathsf{IP}}, \mathbf{V}_{\mathsf{IP}})$ for R	
Rounds	k_{IP}
Prover-to-verifier communication	l_{IP}
Total randomness	r_{IP}
Soundness error	$O(1)$
Verifier running time	$\mathsf{vt}_{\mathsf{IP}}$

$\longrightarrow$

IOP $(\mathbf{P}_{\mathsf{IOP}}, \mathbf{V}_{\mathsf{IOP}})$ for R	
Rounds	k_{IP}
Proof length	$\mathrm{poly}(\lvert\mathbb{x}\rvert, \mathsf{l}_{\mathsf{IP}})$
Queries per round	$O(1)$
Interaction randomness	$\mathrm{poly}(\lvert\mathbb{x}\rvert, \mathsf{r}_{\mathsf{IP}})$
Decision randomness	$O(\log \lvert\mathbb{x}\rvert)$
Soundness error	$O(1)$
Verifier running time	$\mathrm{poly}(\mathsf{vt}_{\mathsf{IP}})$

Proof. We begin by using the round reduction technique of [BM88] to reduce the number of rounds of the protocol to $\mathsf{k}_{\mathsf{IP}}/2$ (assuming that $\mathsf{k}_{\mathsf{IOP}} \geq 2$, as otherwise Drucker's result can be applied). Then, we modify the IP using Theorem 7.2 to have the verifier read little of its own randomness. This yields a k_{IP}-round IP whose parameters are polynomially related to the original proof, that has $O(\log \lvert\mathbb{x}\rvert)$ bits of decision randomness, and in which the verifier randomness query complexity is $O(1)$ (per-round). We then plug in the resulting IP into Theorem 7.7 using the index-decodable PCP of Theorem 6.1, noting Remark 7.8 to get the final result. □

7.1 Local access to randomness

In this section we show that any interactive proof can be transformed into an interactive proof in which the verifier reads $O(1)$ bits from the randomness generated by it during interaction with the prover:

Theorem 7.2. *Let* $\mathsf{IP} = (\mathbf{P}_{\mathsf{IP}}, \mathbf{V}_{\mathsf{IP}})$ *be a public-coin interactive proof system for a relation* R *with (per round) randomness complexity* r_{IP}*, communication complexity* l_{IP}*. Then,* R *has a public-coin interactive proof system* $\mathsf{IP}' = (\mathbf{P}'_{\mathsf{IP}}, \mathbf{V}'_{\mathsf{IP}})$ *with the parameters indicated below.*

IP	
Rounds	k_{IP}
Prover-to-verifier communication	l_{IP}
Randomness (per round)	r_{IP}
Soundness error	$O(1)$
Verifier running time	$\mathsf{vt}_{\mathsf{IP}}$

$\longrightarrow$

IP'	
Rounds	$2\mathsf{k}_{\mathsf{IP}}$
Prover-to-verifier communication	$\mathrm{poly}(\mathsf{l}_{\mathsf{IP}})$
Interaction randomness (per round)	$\mathrm{poly}(\lvert\mathbb{x}\rvert + \mathsf{r}_{\mathsf{IP}})$
Interaction randomness queries (per round)	$O(1)$
Decision randomness	$O(\log \lvert\mathbb{x}\rvert + \log \mathsf{k}_{\mathsf{IP}})$
Soundness error	$O(1)$
Verifier running time	$\mathrm{poly}(\mathsf{vt}_{\mathsf{IP}})$

Moreover, the verifier is non-adaptive with respect to its queries to its interaction randomness.

Proof. On input $\mathbb{x}$, with parameters $n_z, n_s \in \mathbb{N}$ the protocol $(\mathbf{P}'_{\mathsf{IP}}, \mathbf{V}'_{\mathsf{IP}})$ works as follows, given an extractor $\mathsf{Ext}\colon \{0,1\}^{n_z} \times \{0,1\}^{n_s} \to \{0,1\}^{\mathsf{r}_{\mathsf{IP}}}$ with error $\varepsilon_{\mathsf{Ext}}$.

1. Augment IP such that it has round-by-round soundness error $\beta_{\mathsf{IOP},\mathrm{rbr}} \leq 1/4(|\mathbb{x}| + \mathsf{k}_{\mathsf{IP}}^2)$, and per round randomness complexity $\mathsf{r}'_{\mathsf{IP}} = \mathrm{poly}(|\mathbb{x}| + \mathsf{k}_{\mathsf{IP}})$. This can be achieved by $O(\mathsf{k}_{\mathsf{IP}} \cdot \log(|\mathbb{x}| + \mathsf{k}_{\mathsf{IP}}))$ parallel repetitions (see Fact 3.3).

2. For $j = 1, \ldots, \mathsf{k}_{\mathsf{IP}}$:

   (a) $\mathbf{V}'_{\mathsf{IP}}(\mathbb{x}, (z'_1, s'_1, a_1), \ldots, (z'_{j-1}, s'_{j-1}, a_{j-1}))$: Send to the prover a random string $z_j \leftarrow \{0,1\}^{n_z}$.

   (b) $\mathbf{P}'_{\mathsf{IP}}(\mathbb{x}, \mathbb{w}, z_1, s_1, \ldots, z_j)$: Respond with $z'_j \in \{0,1\}^{n_z}$ where (honestly) $z'_j := z_j$.

   (c) $\mathbf{V}'_{\mathsf{IP}}(\mathbb{x}, (z'_1, s'_1, a_1), \ldots, (z'_{j-1}, s'_{j-1}, a_{j-1}), z'_j)$: Send to the prover a random seed $s_j \in \{0,1\}^{n_s}$.

   (d) $\mathbf{P}'_{\mathsf{IP}}(\mathbb{x}, \mathbb{w}, z_1, s_1, \ldots, z_j, s_j)$:

      i. Compute $\rho_j := \mathsf{Ext}(z'_j, s_j)$.

      ii. Compute $a_j \leftarrow \mathbf{P}_{\mathsf{IP}}(\mathbb{x}, \mathbb{w}, \rho_1, \ldots, \rho_j)$.

      iii. Send (s_j, a_j) to the verifier.

3. $\mathbf{V}'^{z_1, s_1, \ldots, z_{\mathsf{k}_{\mathsf{IP}}}, s_{\mathsf{k}_{\mathsf{IP}}}}_{\mathsf{IP}}(\mathbb{x}, (z'_1, s'_1, a_1), \ldots, (z'_{\mathsf{k}_{\mathsf{IP}}}, s'_{\mathsf{k}_{\mathsf{IP}}}, a_{\mathsf{k}_{\mathsf{IP}}}))$:

   (a) Sample a random $m_z \leftarrow [n_z]$ and check that, for every $j \in [\mathsf{k}_{\mathsf{IP}}]$, $z_j[m_z] = z'_j[m_z]$.

   (b) Sample a random $m_s \leftarrow [n_s]$ and check that, for every $j \in [\mathsf{k}_{\mathsf{IP}}]$, $s_j[m_s] = s'_j[m_s]$.

   (c) For every $j \in [\mathsf{k}_{\mathsf{IP}}]$, compute $\rho_j := \mathsf{Ext}(z'_j, s'_j)$.

   (d) Accept if and only if $\mathbf{V}_{\mathsf{IP}}(\mathbb{x}, \rho_1, a_1, \ldots, \rho_{\mathsf{k}_{\mathsf{IP}}}, a_{\mathsf{k}_{\mathsf{IP}}}) = 1$.

The parameters n_z, n_s and $\varepsilon_{\mathsf{Ext}}$ will be fixed during the analysis. Let $\delta > 0$ be a small constant that will specified later.

Completeness. Fix $(\mathbb{x}, \mathbb{w}) \in R$. For every j, let $z_j, s_j, z'_j, s'_j, \rho_j$ and a_j be the strings specified in a random execution of the protocol. Since the prover is honest, we have that $z'_j = z_j$ and $s'_j = s_j$, and so the verifier's tests in Item 3a and Item 3b pass with probability 1. Moreover, since the original interactive proof has perfect completeness, and for every j, $a_j := \mathbf{P}_{\mathsf{IP}}(\mathbb{x}, \rho_1, \ldots, \rho_j)$, we have that (always) $\mathbf{V}_{\mathsf{IP}}(\mathbb{x}, \mathbb{w}\rho_1, a_1, \ldots, \rho_{\mathsf{k}_{\mathsf{IP}}}, a_{\mathsf{k}_{\mathsf{IP}}}) = 1$. Therefore, the new IP verifier accepts with probability 1.

Soundness. Fix $\mathbb{x} \notin L(R)$ and a malicious prover $\tilde{\mathbf{P}}_{\mathsf{IP}}$. Let E be the event over the verifier's random coins, $(z_1, s_1, \ldots, z_{\mathsf{k}_{\mathsf{IP}}}, s_{\mathsf{k}_{\mathsf{IP}}})$, that there exists some $j \in [\mathsf{k}_{\mathsf{IP}}]$ such that at least one of the following is true: (i) $\Delta(z'_j, z_j) \geq \delta$ or; (ii) $\Delta(s'_j, s_j) \geq \delta$, where $z'_j := \tilde{\mathbf{P}}_{\mathsf{IP}}(z_1, s_1, \ldots, z_{j-1}, s_{j-1}, z_j)$ and $s'_j := \tilde{\mathbf{P}}_{\mathsf{IP}}(z_1, s_1, \ldots, z_{j-1}, s_{j-1}, z_j, s_j)$. We first show that if E is true with probability 1/2 (i.e., with probability 1/2, $\tilde{\mathbf{P}}_{\mathsf{IP}}$ gives some z'_j or s'_j which is far from the matching string sent by the verifier), then the verifier rejects with constant probability.

Claim 7.3. *Suppose that*

$$\Pr_{(z_1, s_1, \ldots, z_{\mathsf{k}_{\mathsf{IP}}}, s_{\mathsf{k}_{\mathsf{IP}}})}[\, (z_1, s_1, \ldots, z_{\mathsf{k}_{\mathsf{IP}}}, s_{\mathsf{k}_{\mathsf{IP}}}) \in \mathsf{E} \,] \geq 1/2 \ .$$

Then $\mathbf{V}'_{\mathsf{IP}}$ *accepts with probability at most* $1 - \delta/2$ *when interacting with* $\tilde{\mathbf{P}}_{\mathsf{IP}}$.

Proof. For every choice of verifier randomness $(z_1, s_1, \ldots, z_{\mathsf{k}_{\mathsf{IP}}}, s_{\mathsf{k}_{\mathsf{IP}}}) \in \mathsf{E}$, there exists some round j in which either $\Delta(z_j, z'_j) \geq \delta$ or $\Delta(s_j, s'_j) \geq \delta$. As a result, one of the tests made by $\mathbf{V}'_{\mathsf{IP}}$ in Item 3b and Item 3a, causes the verifier to reject with probability at least δ. The verifier rejects if both $(z_1, s_1, \ldots, z_{\mathsf{k}_{\mathsf{IP}}}, s_{\mathsf{k}_{\mathsf{IP}}}) \in \mathsf{E}$ and the test fails, and so rejects with probability at least $1/2 \cdot \delta = \delta/2$. □

We now show that if E does not happen with probability 1/2, then the prover's messages z'_j have high min-entropy.

Claim 7.4. *Suppose that*

$$\Pr_{(z_1,s_1,\ldots,z_{\mathsf{k}_{\mathsf{IP}}},s_{\mathsf{k}_{\mathsf{IP}}})} \left[\, (z_1, s_1, \ldots, z_{\mathsf{k}_{\mathsf{IP}}}, s_{\mathsf{k}_{\mathsf{IP}}}) \in \mathsf{E} \,\right] < 1/2 \enspace .$$

Then for every j, $H_{\mathsf{min}}(Z'_j \mid \neg\mathsf{E}) \geq 0.5 n_z$, where Z'_j is the random variable describing the output z'_j of $\tilde{\mathbf{P}}_{\mathsf{IP}}$ in a random interaction with $\mathbf{V}_{\mathsf{IP}}$ on input $\mathbb{x}$.

Proof. Fix a round number j, and some string z^*_j. We have that

$$\begin{aligned}
\Pr\left[\, Z'_j = z^*_j \mid \neg\mathsf{E} \,\right] &= \Pr\left[\, Z'_j = z^*_j \ \wedge\ \neg\mathsf{E} \,\right] / \Pr\left[\, \neg\mathsf{E} \,\right] \\
&\leq 2 \cdot \Pr_{z_j}\left[\, \Delta(z^*_j, z_j) < 1/4 \,\right] && (1) \\
&= 2 \cdot |\{\, x' \in \{0,1\}^{n_z} : \Delta(x, x') \leq \delta \,\}| / 2^{n_z} \\
&\leq 2^{-n_z + n_z H(\delta) + 1} && (2) \\
&< 2^{-0.5 n_z} \enspace . && (3)
\end{aligned}$$

Above, Equation (1) is due to the fact that whenever $(z_1, s_1, \ldots, z_{\mathsf{k}_{\mathsf{IP}}}, s_{\mathsf{k}_{\mathsf{IP}}}) \notin \mathsf{E}$, we have by definition that the output z'_j of $\tilde{\mathbf{P}}_{\mathsf{IP}}$, given $(z_1, s_1, \ldots, z_{\mathsf{k}_{\mathsf{IP}}}, s_{\mathsf{k}_{\mathsf{IP}}})$, has Hamming distance at less than δ from z_j. Equation (2) true due to Fact 3.10 and Equation (3) is true for a small enough constant δ. Then, we get that

$$H_{\mathsf{min}}(Z'_j \mid \neg\mathsf{E}) = \min_{z^*_j} - \log \Pr[Z'_j = z^*_j \mid \neg\mathsf{E}] > 0.5 n_z \enspace .$$

□

Claim 7.5. *Let* state *be the state function of the original interactive proof. Then for every transcript* tr *where the verifier is about to make its j-th move such that* $\mathsf{state}(\mathbb{x}, \mathsf{tr}) = 0$*:*

$$\Pr[\, \mathsf{state}(\mathbb{x}, \mathsf{tr} || \rho_j) = 1 \mid \neg\mathsf{E} \,] \leq (\beta_{\mathsf{IOP},\mathrm{rbr}} + \varepsilon_{\mathsf{Ext}}) \cdot 2^{n_s \cdot H(\delta)} \enspace ,$$

where ρ_j is drawn as in the protocol description.

Proof. Fix some j and a transcript as in the claim statement. In the following, for convenience, we do not write the condition on $\neg\mathsf{E}$ but all of our random variables have this added conditioning. By Claim 7.4, we have that z'_j has min-entropy at least $0.5 n_z$. Thus, by definition of the extractor,

$$|\Pr[\, \mathsf{state}(\mathbb{x}, \mathsf{tr} || \mathsf{Ext}(z'_j, U_{n_s}) = 1 \,] - \Pr[\, \mathsf{state}(\mathbb{x}, \mathsf{tr} || U_{\mathsf{r}_{\mathsf{IP}}}) = 1 \,]| \leq \varepsilon_{\mathsf{Ext}} \enspace ,$$

where U_{n_s} and $U_{\mathsf{r}_{\mathsf{IP}}}$ are the uniform distributions over bit strings of length n_s and r_{IP} respectively. Furthermore, by round-by-round soundness of the original interactive proof, we have that

$$\Pr[\, \mathsf{state}(\mathbb{x}, \mathsf{tr} || U_{\mathsf{r}_{\mathsf{IP}}}) = 1 \,] < \beta_{\mathsf{IOP},\mathrm{rbr}} \enspace .$$

Therefore the fraction of seeds that cause the state function to change from 0 to 1 is at most $\varepsilon_{\mathsf{Ext}} + \beta_{\mathsf{IOP},\mathrm{rbr}}$. Recall that in the protocol, $\rho_j := \mathsf{Ext}(z'_j, s'_j)$, i.e., the seed of the extractor is s'_j rather than a uniformly random seed. Since we have that $\neg\mathsf{E}$, we know that the message s'_j chosen by the prover has $\Delta(s'_j, s_j) < \delta$. We say that a seed s_j is *bad* if there exists some s_j with $\Delta(s'_j, s_j) < \delta$ such that $\mathsf{state}(\mathbb{x}, \mathsf{tr}||\mathsf{Ext}(z'_j, s'_j)) = 1$. Every point s'_j that inhibits changing of the state function has a ball of size $2^{n_s \cdot H(\delta)}$ of random seeds that have distance at most δ from it (see Fact 3.10). The total probability of landing on a bad seed is at most the probability that a random seed s_j falls within one of these balls. Therefore the probability that s_j bad is at most $(\varepsilon_{\mathsf{Ext}} + \beta_{\mathsf{IOP},\mathrm{rbr}}) \cdot 2^{n_s \cdot H(\delta)}$. □

We set the extractor error to be $\varepsilon_{\mathsf{Ext}} = \beta_{\mathsf{IOP},\mathrm{rbr}}$, on a source with min entropy $0.5n_z$. We set $n_z = 4\mathsf{r}'_{\mathsf{IP}}$ such that the source has min entropy $2\mathsf{r}'_{\mathsf{IP}}$, and thus we extract $\mathsf{r}'_{\mathsf{IP}}$ random bits with error $\varepsilon_{\mathsf{Ext}}$. The seed length is $n_s = O(\log 1/\varepsilon_{\mathsf{Ext}}) = O(\log(1/\beta_{\mathsf{IOP},\mathrm{rbr}}))$.

If E happens with probability less than 1/2 then we have that:

$$\begin{aligned}
\Pr[\langle \mathbf{V}_{\mathsf{IP}}(\mathbb{x}), \tilde{\mathbf{P}}_{\mathsf{IOP}} \rangle = 1] &\leq \Pr[\mathsf{E}] + \Pr[\ \exists j : \mathsf{state}(\mathbb{x}, \mathsf{tr}||\rho_j) = 1 \mid \neg\mathsf{E}\ ] && (4)\\
&\leq 1/2 + \mathsf{k}_{\mathsf{IP}} \cdot (\beta_{\mathsf{IOP},\mathrm{rbr}} + \varepsilon_{\mathsf{Ext}}) \cdot 2^{n_s \cdot H(\delta)} && (5)\\
&\leq 1/2 + \mathsf{k}_{\mathsf{IP}} \cdot 2\beta_{\mathsf{IOP},\mathrm{rbr}} \cdot 2^{O(\log(1/\beta_{\mathsf{IOP},\mathrm{rbr}}))\cdot H(\delta)} && (6)\\
&\leq 1/2 + \mathsf{k}_{\mathsf{IP}} \cdot \sqrt{\beta_{\mathsf{IOP},\mathrm{rbr}}} < 9/10\ . && (7)
\end{aligned}$$

Equation (4) follows from the fact that the verifier $\mathbf{V}'_{\mathsf{IP}}$ accepts only if $\mathbf{V}_{\mathsf{IP}}$ accepts given $\mathbb{x}$, prover messages $a_1, \ldots, a_{\mathsf{k}_{\mathsf{IP}}}$ and verifier randomness $\rho_1, \ldots . \rho_{\mathsf{k}_{\mathsf{IP}}}$. By the round-by-round soundness of the original IP, since $\mathsf{state}(\mathbb{x}) = 0$ (which follows from the fact that $\mathbb{x} \notin L$), in order for the verifier to accept, it must be that the value of the state function changed from 0 to 1 in some round. Equation (5) is true by applying the union bound and Claim 7.5. We have Equation (6) by noting that we set $\varepsilon_{\mathsf{Ext}} = \beta_{\mathsf{IOP},\mathrm{rbr}}$ and $n_s = O(\log(1/\beta_{\mathsf{IOP},\mathrm{rbr}}))$. Finally, Equation (7) holds for a small enough constant $\delta > 0$.

If the probability that E happens is greater than 1/2, then by Claim 7.3 the verifier rejects with constant probability $1 - \delta/2$ (with the same setting of δ as before).

Thus we have that in both options for the probability that E occurs the verifier rejects with constant probability.

Complexity measures. We analyze the efficiency parameters of the PCP:

- *Prover-to-verifier communication.* We first amplify the protocol, giving polynomial overhead to all messages. In addition to the original prover messages, the prover also sends z'_j and s'_j. This adds at most polynomial overhead to the prover-to-verifier communication complexity.
- *Query complexity to randomness.* The verifier queries each s_j and z_j in $O(1)$ locations.
- *Randomness complexity.* $\mathbf{V}'_{\mathsf{PCP}}$ generates $n_z + n_s = \mathrm{poly}(\mathsf{r}_{\mathsf{IP}}, |\mathbb{x}|)$ bits in every round.
- *Decision randomness.* $\mathbf{V}'_{\mathsf{PCP}}$ uses $\log n_z + \log n_s = O(\log |\mathbb{x}| + \log \mathsf{k}_{\mathsf{IP}})$ bits of decision randomness.
- *Verifier running time.* $\mathbf{V}'_{\mathsf{IP}}$ runs the original IP verifier for polynomially many repetitions, generates a few random strings and runs the extractor. Its running time is therefore polynomially related to the running time of $\mathbf{V}_{\mathsf{IP}}$.
- *Adaptivity.* $\mathbf{V}'_{\mathsf{PCP}}$ makes non-adaptive queries to its interaction randomness.

□

7.2 Local access to prover messages

Definition 7.6. *Given a k_{IP}-round public-coin IP $\mathsf{IP} = (\mathbf{P}_{\mathsf{IP}}, \mathbf{V}_{\mathsf{IP}})$, define the multi-indexed relation*

$$\Psi(\mathbf{V}_{\mathsf{IP}}) := \{(a_1, \ldots, a_{\mathsf{k}_{\mathsf{IP}}}, (\mathbb{x}, \rho_1, \ldots, \rho_{\mathsf{k}_{\mathsf{IP}}}), \bot) \mid \mathbf{V}_{\mathsf{IP}}(\mathbb{x}, \rho_1, a_1, \ldots, \rho_{\mathsf{k}_{\mathsf{IP}}}, a_{\mathsf{k}_{\mathsf{IP}}}) = 1\} \ .$$

Here $\mathbb{x}$ corresponds to the common input instance to the IP prover and IP verifier, $\rho_1, \ldots, \rho_{\mathsf{k}_{\mathsf{IP}}}$ correspond to verifier (random) messages, and $a_1, \ldots, a_{\mathsf{k}_{\mathsf{IP}}}$ correspond to prover messages.

Theorem 7.7. *Suppose that:*
- $\mathsf{IP} = (\mathbf{P}_{\mathsf{IP}}, \mathbf{V}_{\mathsf{IP}})$ *is a public-coin IP for a relation R; and*
- $\mathsf{PCP} = (\mathbf{I}_{\mathsf{PCP}}, \mathbf{P}_{\mathsf{PCP}}, \mathbf{V}_{\mathsf{PCP}}, \mathbf{D}_{\mathsf{PCP}})$ *is an index-decodable PCP for the multi-indexed relation $\Psi(\mathbf{V}_{\mathsf{IP}})$.*

Then Construction 7.9 is a $(\mathsf{k}_{\mathsf{IP}} + 1)$-round public-coin IOP for R with the parameters below.

IP $(\mathbf{P}_{\mathsf{IP}}, \mathbf{V}_{\mathsf{IP}})$ for R	
Rounds	k_{IP}
Prover-to-verifier communication	l_{IP}
Interaction randomness	$\mathsf{r}_{\mathsf{IP},\mathsf{int}}$
Decision randomness	$\mathsf{r}_{\mathsf{IP},\mathsf{dc}}$
Soundness error	β_{IP}
Verifier running time	$\mathsf{vt}_{\mathsf{IP}}$

$+$

Index-Decodable PCP for $\Psi(\mathbf{V}_{\mathsf{IP}})$	
Indexer proof length	$\mathsf{l}_{\mathsf{PCP},\mathbf{I}}$
Proof length	$\mathsf{l}_{\mathsf{PCP},\mathbf{P}}$
Queries per proof	$\mathsf{q}_{\mathsf{PCP}}$
Randomness	$\mathsf{r}_{\mathsf{PCP}}$
Decodability bound	κ_{PCP}
Verifier running time	$\mathsf{vt}_{\mathsf{PCP}}$

$\longrightarrow$

IOP $(\mathbf{P}_{\mathsf{IOP}}, \mathbf{V}_{\mathsf{IOP}})$ for R	
Rounds	k_{IP}
Proof length	$\mathsf{k}_{\mathsf{IP}} \cdot \mathsf{l}_{\mathsf{PCP},\mathbf{I}} + \mathsf{l}_{\mathsf{PCP},\mathbf{P}} \cdot 2^{\mathsf{r}_{\mathsf{IP},\mathsf{dc}}}$
Queries per round	$\mathsf{q}_{\mathsf{PCP}}$
Total round randomness	$\mathsf{r}_{\mathsf{IP},\mathsf{int}}$
Decision randomness	$\mathsf{r}_{\mathsf{PCP}} + \mathsf{r}_{\mathsf{IP},\mathsf{dc}}$
Soundness error	$\beta_{\mathsf{IP}} + \kappa_{\mathsf{PCP}}$
Verifier running time	$\mathsf{vt}_{\mathsf{PCP}}$

Moreover, if $\mathbf{D}_{\mathsf{PCP}}$ is efficient then the transformation maintains computational soundness (if IP has computational soundness error β_{IP}, then IOP has computational soundness error $\beta_{\mathsf{IP}} + \kappa_{\mathsf{PCP}}$).

Remark 7.8. The transformation in Theorem 7.7 can be modified to preserve the verifier's randomness query complexity if the verifier is non-adaptive with respect to the queries it makes to its interaction randomness. Suppose that the verifier reads q bits from its own messages. Then we define a multi-indexed relation that consists of tuples:

$$\Big(\mathbb{i}[1], \ldots, \mathbb{i}[\mathsf{k}], \mathbb{x}, \mathbb{w}\Big) = \Big(a_1, \ldots, a_{\mathsf{k}}, (\mathbb{x}, b_1, \ldots, b_{\mathsf{q}}, \rho_{\mathsf{dc}}), \bot\Big)$$

such that given decision randomness ρ_{dc} the IP verifier $\mathbf{V}_{\mathsf{IP}}$ accepts given instance $\mathbb{x}$, decision randomness ρ_{dc}, prover messages $(a_1, \ldots, a_{\mathsf{k}})$, and $(b_1, \ldots, b_{\mathsf{q}})$ as answers to queries to its own interaction randomness. Given a multi-indexed PCP for this relation, the IP to IOP transformation is identical to the one in Construction 7.9, except that at the end, after the verifier chooses decision randomness, it also queries its own randomness to get bits $b_1, \ldots, b_{\mathsf{q}}$, and these replace $\rho_1, \ldots, \rho_{\mathsf{k}_{\mathsf{IP}}}$ as explicit inputs to the index-decodable PCP verifier.

We now prove Theorem 7.7; we describe the construction and then analyze it.

Construction 7.9. The IOP verifier $\mathbf{V}_{\mathsf{IOP}}$ receives an instance $\mathbb{x}$ and the (honest) IOP prover $\mathbf{P}_{\mathsf{IOP}}$ receives $\mathbb{x}$ and a witness $\mathbb{w}$. They interact as follows.

1. For every round $i \in [\mathsf{k}_{\mathsf{IP}}]$:
   (a) $\mathbf{V}_{\mathsf{IOP}}$ sends a uniformly random string ρ_i as sampled by $\mathbf{V}_{\mathsf{IP}}$;
   (b) $\mathbf{P}_{\mathsf{IOP}}$ computes $a_i \leftarrow \mathbf{P}_{\mathsf{IP}}(\mathbb{x}, \mathbb{w}, \rho_1, \ldots, \rho_i)$ and sends $\pi_i := \mathbf{I}_{\mathsf{PCP}}(a_i)$.
2. $\mathbf{P}_{\mathsf{IOP}}$ sends, for every $\rho_{\mathsf{dc}} \in \{0,1\}^{\mathsf{r}_{\mathsf{IP,dc}}}$, $\Pi_{\rho_{\mathsf{dc}}} := \mathbf{P}_{\mathsf{PCP}}(a_1, \ldots, a_{\mathsf{k}_{\mathsf{IP}}}, (\mathbb{x}, \rho_1, \ldots, \rho_{\mathsf{k}_{\mathsf{IP}}}, \rho_{\mathsf{dc}}), \bot)$.
3. $\mathbf{V}_{\mathsf{IOP}}$ (given oracle access to $\tilde{\pi}_1, \ldots, \tilde{\pi}_{\mathsf{k}_{\mathsf{IP}}}$ and $\tilde{\Pi} = \{\tilde{\Pi}_{\rho_{\mathsf{dc}}}\}_{\rho_{\mathsf{dc}}}$) samples decision randomness ρ_{dc} and PCP randomness ρ_{PCP} and checks that
$$\mathbf{V}_{\mathsf{PCP}}^{\tilde{\pi}_1, \ldots, \tilde{\pi}_{\mathsf{k}_{\mathsf{IP}}}, \tilde{\Pi}_{\rho_{\mathsf{dc}}}}((\mathbb{x}, \rho_1, \ldots, \rho_{\mathsf{k}_{\mathsf{IP}}}, \rho_{\mathsf{dc}}); \rho_{\mathsf{PCP}}) = 1 \ .$$

Proof of Theorem 7.7. First we argue completeness, then argue soundness, and finally analyze efficiency measures.

Completeness. Fix $(\mathbb{x}, \mathbb{w}) \in R$. The strings $a_1, \ldots, a_{\mathsf{k}_{\mathsf{IP}}}$ are computed by running the honest IP prover $\mathbf{P}_{\mathsf{IP}}$ given $(\mathbb{x}, \mathbb{w})$. By the (perfect) completeness of the IP, $\mathbf{V}_{\mathsf{IP}}(\mathbb{x}, \rho_1, a_1, \ldots, \rho_{\mathsf{k}_{\mathsf{IP}}}, a_{\mathsf{k}_{\mathsf{IP}}}; \rho_{\mathsf{dc}}) = 1$ with probability 1 over $\mathbf{V}_{\mathsf{IP}}$'s randomness $\rho_1, \ldots, \rho_{\mathsf{k}_{\mathsf{IP}}}, \rho_{\mathsf{dc}}$. Hence, $(a_1, \ldots, a_{\mathsf{k}_{\mathsf{IP}}}, (\mathbb{x}, \rho_1, \ldots, \rho_{\mathsf{k}_{\mathsf{IP}}}, \rho_{\mathsf{dc}}), \bot) \in \Psi(\mathbf{V}_{\mathsf{IP}})$ with probability 1 over $\rho_1, \ldots, \rho_{\mathsf{k}_{\mathsf{IP}}}, \rho_{\mathsf{dc}}$. Moreover, by the (perfect) completeness of the index-decodable PCP, $\mathbf{V}_{\mathsf{PCP}}$ accepts with probability 1 (over ρ_{PCP}) when given access to the indexer proofs π_i obtained from the indexes a_i via $\mathbf{I}_{\mathsf{PCP}}$ and the prover proof $\Pi_{\rho_{\mathsf{dc}}}$ output by the PCP prover $\mathbf{P}_{\mathsf{PCP}}$. We conclude that $\mathbf{V}_{\mathsf{IOP}}$ accepts with probability 1, as desired.

Soundness. Fix $\mathbb{x} \notin L(R)$ ($L(R)$ is the language implied by the relation R) and let $\tilde{\mathbf{P}}_{\mathsf{IOP}}$ be a malicious IOP prover. In the following, we let $\boldsymbol{\rho} = (\rho_1, \ldots, \rho_{\mathsf{k}_{\mathsf{IP}}}, \rho_{\mathsf{dc}})$ denote a list of k_{IP} verifier messages and $\boldsymbol{\rho}_i = (\rho_1, \ldots, \rho_i)$ a prefix of length i. Let E be the event over the verifier's coins $\boldsymbol{\rho}$ that
$$(\mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_1), \ldots, \mathbf{D}_{\mathsf{PCP}}(\mathsf{k}_{\mathsf{IP}}, \tilde{\pi}_{\mathsf{k}_{\mathsf{IP}}}), (\mathbb{x}, \boldsymbol{\rho}), \bot) \in \Psi(\mathbf{V}_{\mathsf{IP}}) \ ,$$
where $\tilde{\pi}_i := \tilde{\mathbf{P}}_{\mathsf{IOP}}(\boldsymbol{\rho}_i)$ for any $i \in \{1, \ldots, \mathsf{k}_{\mathsf{IP}} - 1\}$ and $(\tilde{\pi}_{\mathsf{k}_{\mathsf{IP}}}, \tilde{\Pi}) := \tilde{\mathbf{P}}_{\mathsf{IOP}}(\boldsymbol{\rho}_{\mathsf{k}_{\mathsf{IP}}})$ where $\tilde{\Pi} = \{\tilde{\Pi}_{\rho_{\mathsf{dc}}}\}_{\rho_{\mathsf{dc}} \in [\mathsf{r}_{\mathsf{IP,dc}}]}$. By the definition of $\Psi(\mathbf{V}_{\mathsf{IP}})$, $(\rho_1, \ldots, \rho_{\mathsf{k}_{\mathsf{IP}}}, \rho_{\mathsf{dc}}) \in E$ if and only if the proofs $\tilde{\pi}_1, \ldots, \tilde{\pi}_{\mathsf{k}_{\mathsf{IP}}}$ can be decoded into messages that make the IP verifier accept:
$$\mathbf{V}_{\mathsf{IP}}(\mathbb{x}, \rho_1, \mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_1), \ldots, \rho_{\mathsf{k}_{\mathsf{IP}}}, \mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_{\mathsf{k}_{\mathsf{IP}}}); \rho_{\mathsf{dc}}) = 1 \ .$$

Using the following claims, by the law of total probability we conclude that $\mathbf{V}_{\mathsf{IOP}}$ accepts with probability at most $\kappa_{\mathsf{PCP}} + \beta_{\mathsf{IP}}$ as desired.

Claim 7.10. *We have that:*
$$\Pr_{\boldsymbol{\rho}, \rho_{\mathsf{PCP}}}\left[\mathbf{V}_{\mathsf{IOP}}^{\tilde{\pi}_1, \ldots, \tilde{\pi}_{\mathsf{k}_{\mathsf{IP}}}, \tilde{\Pi}}(\mathbb{x}, \boldsymbol{\rho}; \rho_{\mathsf{PCP}}) = 1 \ \wedge \ E \ \middle| \ \begin{array}{c} \tilde{\pi}_1 \leftarrow \tilde{\mathbf{P}}_{\mathsf{IOP}}(\boldsymbol{\rho}_1) \\ \vdots \\ \tilde{\pi}_{\mathsf{k}_{\mathsf{IP}}-1} \leftarrow \tilde{\mathbf{P}}_{\mathsf{IOP}}(\boldsymbol{\rho}_{\mathsf{k}_{\mathsf{IP}}-1}) \\ (\tilde{\pi}_{\mathsf{k}_{\mathsf{IP}}}, \tilde{\Pi}) \leftarrow \tilde{\mathbf{P}}_{\mathsf{IOP}}(\boldsymbol{\rho}_{\mathsf{k}_{\mathsf{IP}}}) \end{array} \right] \le \beta_{\mathsf{IP}} \ .$$

Proof. Consider the malicious IP prover $\tilde{\mathbf{P}}_{\mathsf{IP}}$ that simulates $\tilde{\mathbf{P}}_{\mathsf{IOP}}$ by passing it the verifier's messages, decoding the proof that $\tilde{\mathbf{P}}_{\mathsf{IOP}}$ sends in return, and sending the decoded message to the IP verifier. More formally, in round $1 \le i \le \mathsf{k}_{\mathsf{IP}}$, letting $\rho_1, \ldots, \rho_i$ be the verifier messages up to this point, $\tilde{\mathbf{P}}_{\mathsf{IP}}$ computes $\pi_i := \tilde{\mathbf{P}}_{\mathsf{IOP}}(\rho_1, \ldots, \rho_i)$ and sends $a_i := \mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_i)$ as its message to the IP verifier (in round k_{IP}, $\tilde{\mathbf{P}}_{\mathsf{IOP}}(\rho_1, \ldots, \rho_{\mathsf{k}_{\mathsf{IP}}})$ outputs in addition to $\tilde{\pi}_{\mathsf{k}_{\mathsf{IP}}}$ also a proof $\tilde{\Pi}$, but this proof is ignored by $\tilde{\mathbf{P}}_{\mathsf{IP}}$).

Let $\varepsilon_{\mathsf{IP}}$ be the probability that the IP verifier $\mathbf{V}_{\mathsf{IP}}$ accepts when interacting with $\tilde{\mathbf{P}}_{\mathsf{IP}}$:

$$\varepsilon_{\mathsf{IP}} = \Pr_{\rho_1,\ldots,\rho_{\mathsf{k}_{\mathsf{IP}}},\rho_{\mathsf{dc}}}\left[\mathbf{V}_{\mathsf{IP}}(\rho_1, a_1, \ldots, \rho_{\mathsf{k}_{\mathsf{IP}}}, a_{\mathsf{k}_{\mathsf{IP}}}; \rho_{\mathsf{dc}}) = 1 \;\middle|\; \begin{array}{r} a_1 \leftarrow \tilde{\mathbf{P}}_{\mathsf{IP}}(\rho_1) \\ \vdots \\ a_{\mathsf{k}_{\mathsf{IP}}} \leftarrow \tilde{\mathbf{P}}_{\mathsf{IP}}(\rho_1, \ldots, \rho_{\mathsf{k}_{\mathsf{IP}}}) \end{array} \right] \;.$$

By definition, whenever $\boldsymbol{\rho} \in E$ the IP verifier accepts given messages $(a_1, \ldots, a_{\mathsf{k}_{\mathsf{IP}}})$ decoded from the proofs that the prover sent given instance $\mathbb{x}$ and verifier messages $\boldsymbol{\rho} = (\rho_1, \ldots, \rho_{\mathsf{k}_{\mathsf{IP}}})$ (i.e., $a_i := \mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_i)$ where $\tilde{\pi}_i := \tilde{\mathbf{P}}_{\mathsf{IOP}}(\rho_1, \ldots, \rho_i)$). This is precisely how the malicious IP prover computes its own messages. Hence, for every instance $\mathbb{x}$ and verifier messages $\boldsymbol{\rho}$ for which simultaneously the IOP verifier accepts and also $\boldsymbol{\rho} \in E$, the malicious IP prover $\tilde{\mathbf{P}}_{\mathsf{IP}}$ makes the IP verifier accept. By soundness of the IP, the verifier accepts $\mathbb{x} \notin L(R)$ with probability at most β_{IP} over its random messages regardless of what the malicious IP prover does. Thus we conclude that:

$$\Pr_{\boldsymbol{\rho},\rho_{\mathsf{PCP}}}\left[\mathbf{V}_{\mathsf{IOP}}^{\tilde{\pi}_1,\ldots,\tilde{\pi}_{\mathsf{k}_{\mathsf{IP}}},\tilde{\Pi}}(\mathbb{x}, \boldsymbol{\rho}; \rho_{\mathsf{PCP}}) = 1 \;\wedge\; E \;\middle|\; \begin{array}{r} \tilde{\pi}_1 \leftarrow \tilde{\mathbf{P}}_{\mathsf{IOP}}(\boldsymbol{\rho}_1) \\ \vdots \\ \tilde{\pi}_{\mathsf{k}_{\mathsf{IP}}-1} \leftarrow \tilde{\mathbf{P}}_{\mathsf{IOP}}(\boldsymbol{\rho}_{\mathsf{k}_{\mathsf{IP}}-1}) \\ (\tilde{\pi}_{\mathsf{k}_{\mathsf{IP}}}, \tilde{\Pi}) \leftarrow \tilde{\mathbf{P}}_{\mathsf{IOP}}(\boldsymbol{\rho}_{\mathsf{k}_{\mathsf{IP}}}) \end{array} \right] \leq \varepsilon_{\mathrm{IP}} < \beta_{\mathsf{IP}} \;.$$

□

Claim 7.11. *We have that:*

$$\Pr_{\boldsymbol{\rho},\rho_{\mathsf{PCP}}}\left[\mathbf{V}_{\mathsf{IOP}}^{\tilde{\pi}_1,\ldots,\tilde{\pi}_{\mathsf{k}_{\mathsf{IP}}},\tilde{\Pi}}(\mathbb{x}, \boldsymbol{\rho}; \rho_{\mathsf{PCP}}) = 1 \;\wedge\; \neg E \;\middle|\; \begin{array}{r} \tilde{\pi}_1 \leftarrow \tilde{\mathbf{P}}_{\mathsf{IOP}}(\boldsymbol{\rho}_1) \\ \vdots \\ \tilde{\pi}_{\mathsf{k}_{\mathsf{IP}}-1} \leftarrow \tilde{\mathbf{P}}_{\mathsf{IOP}}(\boldsymbol{\rho}_{\mathsf{k}_{\mathsf{IP}}-1}) \\ (\tilde{\pi}_{\mathsf{k}_{\mathsf{IP}}}, \tilde{\Pi}) \leftarrow \tilde{\mathbf{P}}_{\mathsf{IOP}}(\boldsymbol{\rho}_{\mathsf{k}_{\mathsf{IP}}}) \end{array} \right] \leq \kappa_{\mathsf{PCP}}(|\mathbb{x}| + \mathsf{r}_{\mathsf{IP,int}} + \mathsf{r}_{\mathsf{IP,dc}}) \;.$$

Proof. Assume towards contradiction that the claim does not hold. There must exist $\boldsymbol{\rho} \notin E$ such that

$$\Pr_{\rho_{\mathsf{PCP}}}\left[\mathbf{V}_{\mathsf{IOP}}^{\tilde{\pi}_1,\ldots,\tilde{\pi}_k,\tilde{\Pi}}(\mathbb{x}, \boldsymbol{\rho}; \rho_{\mathsf{PCP}}) = 1 \;\middle|\; \begin{array}{r} \tilde{\pi}_1 \leftarrow \tilde{\mathbf{P}}_{\mathsf{IOP}}(\boldsymbol{\rho}_1) \\ \vdots \\ \tilde{\pi}_{\mathsf{k}_{\mathsf{IP}}-1} \leftarrow \tilde{\mathbf{P}}_{\mathsf{IOP}}(\boldsymbol{\rho}_{\mathsf{k}_{\mathsf{IP}}-1}) \\ (\tilde{\pi}_{\mathsf{k}_{\mathsf{IP}}}, \tilde{\Pi}) \leftarrow \tilde{\mathbf{P}}_{\mathsf{IOP}}(\boldsymbol{\rho}_{\mathsf{k}_{\mathsf{IP}}}) \end{array} \right] > \kappa_{\mathsf{PCP}}(|\mathbb{x}| + \mathsf{r}_{\mathsf{IP,int}} + \mathsf{r}_{\mathsf{IP,dc}}) \;.$$

The IOP verifier accepts if and only if the underlying PCP verifier accepts. This means that the PCP verifier accepts with probability greater than $\kappa_{\mathsf{PCP}}(|\mathbb{x}| + \mathsf{r}_{\mathsf{IP,int}} + \mathsf{r}_{\mathsf{IP,dc}})$ (the knowledge bound of the PCP). Thus, by decodability of the index-decodable PCP, we get that:

$$(\mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_1), \ldots, \mathbf{D}_{\mathsf{PCP}}(\tilde{\pi}_{\mathsf{k}_{\mathsf{IP}}}), (\mathbb{x}, \boldsymbol{\rho}), \bot) \in \Psi(\mathbf{V}_{\mathsf{IP}}) \;.$$

This contradicts the assumption that $\boldsymbol{\rho} \notin E$. □

Notice that the prover $\tilde{\mathbf{P}}_{\mathsf{IP}}$ above simply runs the malicious IOP prover $\tilde{\mathbf{P}}_{\mathsf{IOP}}$ and the decoder $\mathbf{D}_{\mathsf{PCP}}$. If both $\tilde{\mathbf{P}}_{\mathsf{IOP}}$ and $\mathbf{D}_{\mathsf{PCP}}$ run in polynomial time, then so does $\tilde{\mathbf{P}}_{\mathsf{IP}}$. Hence if this is the case, and the IP is *computationally* sound (sound against efficient adversaries) then so is the resulting IOP.

Complexity measures. The number of rounds is k_{IP}. The IOP verifier uses $\mathsf{r}_{\mathsf{IP},\mathsf{int}}$ during interaction and $\mathsf{r}_{\mathsf{PCP}} + \mathsf{r}_{\mathsf{IP},\mathsf{dc}}$ random bits in the decision phase. The IOP verifier makes $\mathsf{q}_{\mathsf{PCP}}$ queries to its oracles when running the PCP verifier. The IOP verifier's running time is $\mathsf{vt}_{\mathsf{PCP}}$ since it runs the PCP verifier. The IOP prover generates k_{IP} indexer proofs each of length $\mathsf{l}_{\mathsf{PCP},\mathbf{I}}$, and $2^{\mathsf{r}_{\mathsf{dc}}}$ prover proofs of length $\mathsf{l}_{\mathsf{PCP},\mathbf{P}}$.

□

Acknowledgments

Gal Arnon is supported in part by a grant from the Israel Science Foundation (no. 2686/20) and by the Simons Foundation Collaboration on the Theory of Algorithmic Fairness. Alessandro Chiesa is funded by the Ethereum Foundation. Eylon Yogev is funded by the ISF grants 484/18, 1789/19, Len Blavatnik and the Blavatnik Foundation, and The Blavatnik Interdisciplinary Cyber Research Center at Tel Aviv University.

References

[ALMSS98] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. “Proof verification and the hardness of approximation problems”. In: *Journal of the ACM* 45.3 (1998). Preliminary version in FOCS ’92., pp. 501–555.

[AS98] Sanjeev Arora and Shmuel Safra. “Probabilistic checking of proofs: a new characterization of NP”. In: *Journal of the ACM* 45.1 (1998). Preliminary version in FOCS ’92., pp. 70–122.

[BBHR18] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. “Fast Reed–Solomon Interactive Oracle Proofs of Proximity”. In: *Proceedings of the 45th International Colloquium on Automata, Languages and Programming.* ICALP ’18. 2018, 14:1–14:17.

[BBHR19] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. “Scalable Zero Knowledge with No Trusted Setup”. In: *Proceedings of the 39th Annual International Cryptology Conference.* CRYPTO ’19. 2019, pp. 733–764.

[BCG20] Jonathan Bootle, Alessandro Chiesa, and Jens Groth. “Linear-Time Arguments with Sublinear Verification from Tensor Codes”. In: *Proceedings of the 18th Theory of Cryptography Conference.* TCC ’20. 2020, pp. 19–46.

[BCGGHJ17] Jonathan Bootle, Andrea Cerulli, Essam Ghadafi, Jens Groth, Mohammad Hajiabadi, and Sune K. Jakobsen. “Linear-Time Zero-Knowledge Proofs for Arithmetic Circuit Satisfiability”. In: *Proceedings of the 23rd International Conference on the Theory and Applications of Cryptology and Information Security.* ASIACRYPT ’17. 2017, pp. 336–365.

[BCGGRS19] Eli Ben-Sasson, Alessandro Chiesa, Lior Goldberg, Tom Gur, Michael Riabzev, and Nicholas Spooner. “Linear-Size Constant-Query IOPs for Delegating Computation”. In: *Proceedings of the 17th Theory of Cryptography Conference.* TCC ’19. 2019, pp. 494–521.

[BCGRS17] Eli Ben-Sasson, Alessandro Chiesa, Ariel Gabizon, Michael Riabzev, and Nicholas Spooner. “Interactive Oracle Proofs with Constant Rate and Query Complexity”. In: *Proceedings of the 44th International Colloquium on Automata, Languages and Programming.* ICALP ’17. 2017, 40:1–40:15.

[BCGV16] Eli Ben-Sasson, Alessandro Chiesa, Ariel Gabizon, and Madars Virza. “Quasilinear-Size Zero Knowledge from Linear-Algebraic PCPs”. In: *Proceedings of the 13th Theory of Cryptography Conference.* TCC ’16-A. 2016, pp. 33–64.

[BCL20] Jonathan Bootle, Alessandro Chiesa, and Siqi Liu. *Zero-Knowledge Succinct Arguments with a Linear-Time Prover.* Cryptology ePrint Archive, Report 2020/1527. 2020.

[BCRSVW19] Eli Ben-Sasson, Alessandro Chiesa, Michael Riabzev, Nicholas Spooner, Madars Virza, and Nicholas P. Ward. “Aurora: Transparent Succinct Arguments for R1CS”. In: *Proceedings of the 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques.* EUROCRYPT ’19. Full version available at `https://eprint.iacr.org/2018/828`. 2019, pp. 103–128.

[BCS16] Eli Ben-Sasson, Alessandro Chiesa, and Nicholas Spooner. “Interactive Oracle Proofs”. In: *Proceedings of the 14th Theory of Cryptography Conference.* TCC ’16-B. 2016, pp. 31–60.

[BFLS91] László Babai, Lance Fortnow, Leonid A. Levin, and Mario Szegedy. "Checking computations in polylogarithmic time". In: *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing*. STOC '91. 1991, pp. 21–32.

[BGG90] Mihir Bellare, Oded Goldreich, and Shafi Goldwasser. "Randomness in Interactive Proofs". In: *Proceedings of the 31st Annual Symposium on Foundations of Computer Science*. FOCS '90. 1990, pp. 563–572.

[BGHSV05] Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil Vadhan. "Short PCPs Verifiable in Polylogarithmic Time". In: *Proceedings of the 20th Annual IEEE Conference on Computational Complexity*. CCC '05. 2005, pp. 120–134.

[BGHSV06] Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil P. Vadhan. "Robust PCPs of Proximity, Shorter PCPs, and Applications to Coding". In: *SIAM Journal on Computing* 36.4 (2006), pp. 889–974.

[BGKS19] Eli Ben-Sasson, Lior Goldberg, Swastik Kopparty, and Shubhangi Saraf. *DEEP-FRI: Sampling Outside the Box Improves Soundness*. ECCC TR19-044. 2019.

[BM88] László Babai and Shlomo Moran. "Arthur-Merlin Games: A Randomized Proof System, and a Hierarchy of Complexity Classes". In: *Journal of Computer and System Sciences* 36.2 (1988), pp. 254–276.

[BN21] Sarah Bordage and Jade Nardi. *Interactive Oracle Proofs of Proximity to Algebraic Geometry Codes*. ArXiv cs/2011.04295. 2021.

[BS08] Eli Ben-Sasson and Madhu Sudan. "Short PCPs with Polylog Query Complexity". In: *SIAM Journal on Computing* 38.2 (2008). Preliminary version appeared in STOC '05., pp. 551–607.

[Bab85] László Babai. "Trading group theory for randomness". In: *Proceedings of the 17th Annual ACM Symposium on Theory of Computing*. STOC '85. 1985, pp. 421–429.

[Ben+17] Eli Ben-Sasson et al. "Computational integrity with a public random string from quasi-linear PCPs". In: *Proceedings of the 36th Annual International Conference on Theory and Application of Cryptographic Techniques*. EUROCRYPT '17. 2017, pp. 551–579.

[CCHLRR18] Ran Canetti, Yilei Chen, Justin Holmgren, Alex Lombardi, Guy N. Rothblum, and Ron D. Rothblum. *Fiat–Shamir From Simpler Assumptions*. Cryptology ePrint Archive, Report 2018/1004. 2018.

[CFLS95] Anne Condon, Joan Feigenbaum, Carsten Lund, and Peter W. Shor. "Probabilistically Checkable Debate Systems and Nonapproximability of PSPACE-Hard Functions". In: *Chicago Journal of Theoretical Computer Science* 1995 (1995).

[CFLS97] Anne Condon, Joan Feigenbaum, Carsten Lund, and Peter W. Shor. "Random Debaters and the Hardness of Approximating Stochastic Functions". In: *SIAM Journal on Computing* 26.2 (1997), pp. 369–400.

[CHMMVW20] Alessandro Chiesa, Yuncong Hu, Mary Maller, Pratyush Mishra, Noah Vesely, and Nicholas Ward. "Marlin: Preprocessing zkSNARKs with Universal and Updatable SRS". In: *Proceedings of the 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques*. EUROCRYPT '20. 2020.

[COS20] Alessandro Chiesa, Dev Ojha, and Nicholas Spooner. "Fractal: Post-Quantum and Transparent Recursive Proofs from Holography". In: *Proceedings of the 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques*. EUROCRYPT '20. 2020, pp. 769–793.

[DH13] Irit Dinur and Prahladh Harsha. "Composition of Low-Error 2-Query PCPs Using Decodable PCPs". In: *SIAM Journal on Computing* 42.6 (2013). Preliminary version appeared in Property Testing '10., pp. 2452–2486.

[DR04] Irit Dinur and Omer Reingold. “Assignment Testers: Towards a Combinatorial Proof of the PCP Theorem”. In: *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*. FOCS ’04. 2004, pp. 155–164.

[Din07] Irit Dinur. “The PCP theorem by gap amplification”. In: *Journal of the ACM* 54.3 (2007), p. 12.

[Dru11a] Andrew Drucker. “A PCP Characterization of AM”. In: *Proceedings of the 38th International Colloquium on Automata, Languages and Programming*. ICALP ’11. 2011, pp. 581–592.

[Dru11b] Andrew Drucker. “Efficient Probabilistically Checkable Debates”. In: *Proceedings of the 15th International Workshop on Approximation, Randomization, and Combinatorial Optimization*. RANDOM ’11. 2011, pp. 519–529.

[FGLSS91] Uriel Feige, Shafi Goldwasser, László Lovász, Shmuel Safra, and Mario Szegedy. “Approximating clique is almost NP-complete (preliminary version)”. In: *Proceedings of the 32nd Annual Symposium on Foundations of Computer Science*. SFCS ’91. 1991, pp. 2–12.

[FGLSS96] Uriel Feige, Shafi Goldwasser, Laszlo Lovász, Shmuel Safra, and Mario Szegedy. “Interactive proofs and the hardness of approximating cliques”. In: *Journal of the ACM* 43.2 (1996). Preliminary version in FOCS ’91., pp. 268–292.

[FGMSZ89] Martin Fürer, Oded Goldreich, Yishay Mansour, Michael Sipser, and Stathis Zachos. “On Completeness and Soundness in Interactive Proof Systems”. In: *Advances in Computing Research* 5 (1989), pp. 429–442.

[GI05] Venkatesan Guruswami and Piotr Indyk. “Linear-time encodable/decodable codes with near-optimal rate”. In: *IEEE Transactions on Information Theory* 51.10 (2005). Preliminary version appeared in STOC ’03., pp. 3393–3400.

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. “The knowledge complexity of interactive proof systems”. In: *SIAM Journal on Computing* 18.1 (1989). Preliminary version appeared in STOC ’85., pp. 186–208.

[GMW91] Oded Goldreich, Silvio Micali, and Avi Wigderson. “Proofs that Yield Nothing But Their Validity or All Languages in NP Have Zero-Knowledge Proof Systems”. In: *Journal of the ACM* 38.3 (1991). Preliminary version appeared in FOCS ’86., pp. 691–729.

[GS86] Shafi Goldwasser and Michael Sipser. “Private Coins versus Public Coins in Interactive Proof Systems”. In: *Proceedings of the 18th Annual ACM Symposium on Theory of Computing*. STOC ’86. 1986, pp. 59–68.

[GUV09] Venkatesan Guruswami, Christopher Umans, and Salil P. Vadhan. “Unbalanced expanders and randomness extractors from Parvaresh–Vardy codes”. In: *Journal of the ACM* 56.4 (2009), 20:1–20:34.

[GVW02] Oded Goldreich, Salil Vadhan, and Avi Wigderson. “On interactive proofs with a laconic prover”. In: *Computational Complexity* 11.1/2 (2002), pp. 1–53.

[HRT07] Ishay Haviv, Oded Regev, and Amnon Ta-Shma. “On the Hardness of Satisfiability with Bounded Occurrences in the Polynomial-Time Hierarchy”. In: *Theory of Computing* 3.1 (2007), pp. 45–60.

[IW14] Yuval Ishai and Mor Weiss. “Probabilistically Checkable Proofs of Proximity with Zero-Knowledge”. In: *Proceedings of the 11th Theory of Cryptography Conference*. TCC ’14. 2014, pp. 121–145.

[KR08] Yael Kalai and Ran Raz. “Interactive PCP”. In: *Proceedings of the 35th International Colloquium on Automata, Languages and Programming*. ICALP ’08. 2008, pp. 536–547.

[LFKN92] Carsten Lund, Lance Fortnow, Howard J. Karloff, and Noam Nisan. “Algebraic Methods for Interactive Proof Systems”. In: *Journal of the ACM* 39.4 (1992), pp. 859–868.

[Mie09] Thilo Mie. “Short PCPPs verifiable in polylogarithmic time with O(1) queries”. In: *Annals of Mathematics and Artificial Intelligence* 56 (3 2009), pp. 313–338.

[RR20] Noga Ron-Zewi and Ron Rothblum. “Local Proofs Approaching the Witness Length”. In: *Proceedings of the 61st Annual IEEE Symposium on Foundations of Computer Science*. FOCS ’20. 2020, pp. 846–857.

[RRR16] Omer Reingold, Ron Rothblum, and Guy Rothblum. “Constant-Round Interactive Proofs for Delegating Computation”. In: *Proceedings of the 48th ACM Symposium on the Theory of Computing*. STOC ’16. 2016, pp. 49–62.

[Sha92] Adi Shamir. “IP = PSPACE”. In: *Journal of the ACM* 39.4 (1992), pp. 869–877.