Electronic Colloquium on Computational Complexity, Report No. 127 (2019)

Local Proofs Approaching the Witness Length

Noga Ron-Zewi* Ron D. Rothblum†

September 19, 2019

Abstract

Interactive oracle proofs (IOPs) are a hybrid between interactive proofs and PCPs. In an IOP the prover is allowed to interact with a verifier (like in an interactive proof) by sending relatively long messages to the verifier, who in turn is only allowed to query a few of the bits that were sent (like in a PCP).

In this work we construct, for a large class of NP relations, IOPs in which the communication complexity approaches the witness length. More precisely, for any NP relation for which membership can be decided in polynomial-time and bounded polynomial space (e.g., SAT, Hamiltonicity, Clique, Vertex-Cover, etc.) and for any constant $\gamma > 0$, we construct an IOP with communication complexity $(1+\gamma) \cdot n$, where n is the original witness length. The number of rounds as well as the number of queries made by the IOP verifier are constant.

This result improves over prior works on short IOPs/PCPs in two ways. First, the communication complexity in these short IOPs is proportional to the complexity of *verifying* the NP witness, which can be polynomially larger than the witness size. Second, even ignoring the difference between witness length and non-deterministic verification time, prior works incur (at the very least) a large constant multiplicative overhead to the communication complexity.

In particular, as a special case, we also obtain an IOP for CircuitSAT with rate approaching 1: the communication complexity is $(1+\gamma) \cdot t$, for circuits of size t and any constant $\gamma > 0$. This improves upon the prior state-of-the-art work of Ben Sasson *et. al* (ICALP, 2017) who construct an IOP for CircuitSAT with rate that is a small (unspecified) constant bounded away from 0.

Our proof leverages recent constructions of high-rate locally testable tensor codes. In particular, we bypass the barrier imposed by the low rate of *multiplication codes* (e.g., Reed-Solomon, Reed-Muller or AG codes) - a core component in all known short PCP/IOP constructions.

*Department of Computer Science, University of Haifa. Email: noga@cs.haifa.ac.il. Supported by the Milgrom family grant for Collaboration between the Technion and University of Haifa.

†Department of Computer Science, Technion. Email: rothblum@cs.technion.ac.il. Supported in part by the Milgrom family grant for Collaboration between the Technion and University of Haifa, by the Israeli Science Foundation (Grant No. 1262/18), and the Technion Hiroshi Fujiwara cyber security research center and Israel cyber directorate.

ISSN 1433-8092

Contents

1 Introduction

The celebrated PCP Theorem, established in the early 90's [BFLS91, FGL+96, AS98, ALM+98], shows that it is possible to encode any NP witness in such a way that the veracity of the witness can be verified by reading only a constant number of bits from the encoding. This foundational result has had a transformative effect on TCS with diverse applications in cryptography and complexity theory.

A basic and natural question that has drawn a great amount of interest is what is the minimal overhead in encoding that is needed to allow for such local checking of an NP witness. While the original PCP theorem only guarantees a polynomial overhead, a beautiful line of work has culminated in remarkably short PCPs. More precisely, the works of Ben Sasson and Sudan [BS08] combined with that of Dinur [Din07] yield PCPs of length $t \cdot \text{polylog}(t)$ for any time t non-deterministic computation.

A central open question in the area is whether such poly-logarithmic overhead is indeed necessary, or can one construct truly linear length PCPs. Actually, since the question is highly dependent on the computational model (since transitions between standard computational models usually involve at the very least a logarithmic overhead), a cleaner formalization of this question is the following:

Does the NP*-complete language*[1] CircuitSAT
have a PCP *whose length is linear in the given circuit?*

Beyond its intrinsic interest, this question also has implications to the construction of efficient proof-systems that build on PCPs [Kil92, Mic00, BCS16] as well as to the field of hardness of approximation [Din16, MR17, App17]. The state-of-the-art is a result of [BKK+16] who construct a linear length PCP for CircuitSAT albeit with only n^{ε} query complexity, for any desired constant $\varepsilon > 0$. This result falls well short of the desired goal of *constant* query complexity (and has the additional drawback that the verification is non-uniform).

Motivated by the goal of constructing such short PCPs and their applications to the construction of efficient proof-systems, Ben Sasson *et. al* [BCS16] recently proposed a natural generalization of PCPs called *interactive oracle proofs*, or IOPs for short.[2] An IOP is an interactive protocol (similarly to an interactive proof), but at each round the prover can send a *long* message from which the verifier is allowed to read only a few bits (i.e., PCP-style access to the prover messages). Ben Sasson *et. al* [BCG+17] later constructed an IOP for CircuitSAT in which the communication complexity (which we will also refer to as *proof length*) is $O(t)$, thereby demonstrating that local proofs with constant overhead exist, if one allows for interaction.

A recent exciting sequence of works [BBHR18, BCR+19, BBHR19, BGKS19] has leveraged the efficiency of IOPs to construct highly efficient succinct argument schemes which are now at the core of leading *practical* implementations [BCR+19].

[1] Recall that the language CircuitSAT consists of the set of all satisfiable Boolean circuits, and that there is an $O(t \cdot \log(t))$-time reduction from $\text{NTIME}(t)$ to CircuitSAT [PF79].

[2] The same notion was proposed independently by Reingold *et. al* [RRR16] (who used the term *probabilistically checkable interactive proof*) but for a different motivation - facilitating the construction of doubly-efficient interactive proofs.

1.1 Our Results

In this work we construct IOPs with nearly optimal proof length. As our first main result, which actually follows from a more general statement that will be elaborated on in Section 1.1.1, we construct an IOP for CircuitSAT in which the proof length is $(1+\gamma)\cdot t$, for circuits of size t and for any constant $\gamma > 0$.

Theorem 1 (Informally Stated, see Corollary 3.3 and Remark 1.2)**.** *For every constants $\gamma, \varepsilon > 0$,* CircuitSAT *has a constant-round and constant-query* IOP *in which the proof-length is $(1+\gamma)\cdot t$, where t is the circuit size. The* IOP *has perfect completeness and soundness error ε. The verifier runs in time $\tilde{O}(t)$ and the prover runs in time* $\mathrm{poly}(t)$ *(given the satisfying assignment).*

Theorem 1 should be contrasted with the main result of [BCG+17], who give an IOP for CircuitSAT with proof length $c \cdot t$, for a constant $c \geq 1$ that is left unspecified.[3] We remark that while the proof length in Theorem 1 is shorter than that in [BCG+17], the round complexity is larger (but still constant).

Remark 1.1. *The* IOP *in Theorem 1 can be extended to some* sub-constant *values of $\gamma = \gamma(t) > 0$ and $\varepsilon = \varepsilon(t) > 0$ (simultaneously) at the cost of increasing the query complexity to* $\mathrm{poly}\left(\frac{1}{\gamma}, \log(1/\varepsilon)\right)$.[4] *This leads to an* IOP *with communication rate $1 - o(1)$ and negligible soundness error, but with poly-logarithmic query complexity.*

1.1.1 Approaching the Witness Length

When considering a generic NP relation $\mathcal{R}$, the result of Theorem 1, the main result in [BCG+17], and essentially all short PCP constructions, only yield constructions in which the communication scales with the *verification complexity* of $\mathcal{R}$, rather than with the length of the original NP witness. For example, while 3-Colorability of a (connected) graph $G = (V, E)$ has an NP witness of size $\Theta(|V|)$, the IOP constructed in [BCG+17] has length $\Theta(|E|)$, which can be quadratically larger than $|V|$.

This limitation actually turns out to be inherent for PCPs. Fortnow and Santhanam [FS11] showed that constructing PCPs whose length is a fixed polynomial of the witness length is impossible, unless $\mathsf{NP} \subseteq \mathsf{co\text{-}NP/poly}$.

Interestingly however, Kalai and Raz [KR08] show that the picture changes drastically if one allows for interaction. In particular, their work yields IOPs with proof length that is polynomial in the witness size for a large class of NP relations, with poly-logarithmic query complexity.[5] We improve upon the result of [KR08] by constructing IOPs, also for a large class of NP relations, in which the proof length is $(1+\varepsilon)\cdot n$, where n is the length of the original NP witness, rather than $\mathrm{poly}(n)$ as in [KR08], and with constant query complexity.

[3] We estimate that c is at least $3 \cdot 6^3 = 648$, meaning that the rate is ≤ 0.002. This is since the [BCG+17] IOP includes three codewords, where each codeword is a tensor (of dimension ≥ 3) of an AG code. The AG code has rate $1 - \frac{1}{q-1}$ using an alphabet of size q^2 (for any prime power q). However, since they encode *binary* messages, the effective rate is $(1 - \frac{1}{q-1})/\log(q^2)$, which achieves its maximum (over prime powers) at $q = 4$.

[4] The fact that the soundness error can be reduced is non-trivial: straightforward error reduction by repetition can significantly increase the proof length.

[5] Kalai and Raz [KR08] consider a restricted model (in fact a predecessor) of IOPs called *interactive* PCPs. Their result, combined with followup works [GKR15, RRR16], yields IOPs with length that is polynomial in the witness size for all NP relations that can be verified in either bounded depth, or bounded space. Jumping ahead, we remark that our results can also be interpreted as interactive PCPs.

The class of relations that we can support consists of all NP relations that can be verified in polynomial-time and bounded polynomial space.[6] This class includes many natural NP problems, including in particular: SAT, Hamiltonicity, TSP and all 21 of Karp's original NP-complete problems [Kar72].[7]

Theorem 2 (Informally Stated, see Corollary 3.3)**.** *Let $\mathcal{R}$ be an* NP *relation which can be verified in polynomial-time and bounded polynomial space (i.e., space n^{ξ} for some sufficiently small constant $\xi > 0$). Then for any constants $\gamma, \varepsilon > 0$, the relation $\mathcal{R}$ has a constant-round and constant-query* IOP *with proof-length $(1+\gamma) \cdot n$, where n is the original witness length. The* IOP *has perfect completeness and soundness error ε. The verifier runs in quasi-linear time and the prover runs in polynomial-time (given the* NP *witness).*

Similarly to Theorem 1, we can extend Theorem 2 to some sub-constant values of $\gamma = \gamma(n) > 0$ and $\varepsilon = \varepsilon(n) > 0$, leading to communication rate $1 - o(1)$ and negligible soundness error, but at the cost of mildly super-constant query complexity.

We remark that the communication complexity in Theorem 2 is very close to optimal, under reasonable complexity theoretic conjectures. Indeed, the works of Goldreich and Håstad [GH98] and Goldreich, Vadhan and Wigderson [GVW02] show that NP complete languages are unlikely to have interactive proofs (let alone IOPs) in which the communication complexity is sublinear in the witness size.

We also mention that the limitation in Theorem 2 to relations computable in bounded polynomial space is inherited from the work of [RRR16], on which we build. In this work we have not made an attempt to optimize the constant ξ. However, we believe that ξ could potentially be any constant smaller than $1/2$ (i.e., leading to a space bound of $n^{0.499}$). Whether or not the space bound can be improved or altogether eliminated is an interesting open problem

Remark 1.2. *It is worth emphasizing that* every *language in* NP *has a corresponding* NP *relation which can be verified in small space. Indeed, this follows directly from the Cook-Levin theorem. However, the Cook-Levin transformation incurs a polynomial blowup to the witness size (corresponding to the non-deterministic verification time of the relation). The point of Theorem 2 is that for* NP *relations which* a priori *can be verified in small space, we do not need to pay any additional overhead.*

Still, by applying Theorem 2 to CircuitSAT *using the* NP *relation arising from the Cook-Levin theorem (in which the witness includes, in addition to the satisfying assignment, the values of all of the gates in the circuit), we obtain an* IOP *for* CircuitSAT *with proof length $(1+\varepsilon) \cdot t$, where t is the circuit size. Thus, Theorem 1 follows as an immediate corollary of Theorem 2.*

[6] We emphasize that machine verifying the relation is allowed read-many access to the witness (in contrast to the much more restricted complexity class NL in which the verifier has read-once access to the witness). For further discussion, see [Gol08, Section 5.3.1].

[7] It is straightforward to see that all of Karp's problems can be verified in bounded space except for two problems: *feedback vertex set* and *feedback arc set*. In these two problems, given a graph G one needs to decide whether there is a small set of vertices (resp., edges) whose removal makes the graph acyclic. While we are unaware of a *deterministic* bounded space verification procedure for these relations, it is straightforward to show that these two problems can be verified by a *randomized* bounded space algorithm. Theorem 2 can be extended to *randomized* bounded space by using Nisan's pseudorandom generator [Nis92] and thus we can handle also these two problems.

1.1.2 Interactive oracle proofs of proximity with sublinear proof length

Loosely speaking, proofs of proximity are proof-systems in which the verifier runs in *sublinear* time. Since the verifier cannot even read its own input, we only require that she rejects inputs that are *far* from the language. Various models of proofs of proximity have been considered in the literature, depending on the type of communication with the prover. In particular, PCP-style access [BGH+06, DR06], interactive proofs [EKR04, RVW13], non-interactive proofs [GR18], as well as an IOP variant [BCS16, RRR16].

As a technical step toward proving Theorem 2, we also construct short *interactive oracle proofs of proximity* (IOPP). In an IOPP, the verifier is given oracle access to an implicit input w and is allowed to communicate with an all powerful but untrusted prover (who sees all of w). In each round of the interaction the prover sends a long message and the verifier can choose to read a few of the bits of the message. At the end of the interaction the verifier should accept if w belongs to the language and should reject (with high probability) if w is *far* from the language (no matter what the prover does).

Theorem 3 (Informally Stated, see Corollary 3.6)**.** *Let $\mathcal{L}$ be a language computable in* $\mathrm{poly}(n)$ *time and $n^{o(1)}$ space. Then, for any constant $\beta, \gamma > 0$ there exists an* IOPP *for $\mathcal{L}$ with communication complexity $\gamma \cdot n$ and constant query and round complexities. The verifier's running time is n^{β}, and the prover's running time is* $\mathrm{poly}(n)$.

We remark that the communication complexity in Theorem 3 is strictly less than the input length n. At first glance this may seem quite surprising as, by the aforementioned works of [GH98, GVW02], we do not expect IOPs for NP with communication that is shorter than the witness length. Indeed, the key difference which enables such short communication is the fact that Theorem 3 is for *deterministic* languages.

We remark that Theorem 3 is optimal in several ways. First, the work of Kalai and Rothblum [KR15] implies that there exists a language in $\mathcal{L}^* \in \mathsf{Logspace}$ such that any IOP for $\mathcal{L}^*$ with communication complexity $o(n)$ must have query complexity $\omega(1)$, under a strong but plausible cryptographic assumption (i.e., exponentially hard pseudorandom generators computable in logarithmic space). Second, a bound on the space complexity of $\mathcal{L}$ is inherent under the widely believed assumption that P is not contained in $\mathsf{SPACE}(\tilde{O}(n))$ (c.f., [Gol18, Theorem 1.4]).

Remark 1.3. *Interestingly, since the proof length is strictly less than the input size, Theorem 3 is non-trivial even if we ignore the fact the verifier only reads a small part of the proof. Thus, the result can also be viewed as an* interactive proof of proximity *(*IPP*) [EKR04, RVW13]. As a point of comparison, note that [RVW13, RRR16] also construct* IPP*s with sublinear communication for bounded-space computations, but in a different parameter regime: the result in [RVW13, RRR16] has much smaller communication complexity (e.g., they support communication $O(\sqrt{n})$) but on the other hand it does not support constant query complexity as in Theorem 3.*

1.2 Techniques

Next we give an overview of the proof of Theorem 2 (from which Theorem 1 follows, see Remark 1.2). As a warmup, we focus here on a high level overview of a short IOP for 3SAT (i.e., rather than all bounded space relations) and with only some non-trivial query complexity (i.e., $O(\sqrt{n})$ rather than constant). Later, in Section 1.2.1, we discuss the additional steps needed to generalize to any bounded-space relation and with constant query complexity.

Let $\phi : \{0,1\}^n \to \{0,1\}$ be a 3CNF formula and let $\gamma > 0$. We construct an IOP for proving that ϕ is satisfiable with communication complexity $(1+\gamma) \cdot n$ and query complexity $O(\sqrt{n})$. To construct the IOP, our starting point is the following rough outline shared by most PCP and IOP constructions:

1. The prover provides an error-corrected encoding of the computation (either as part of the PCP or as the first message of the IOP).

2. The error-correcting code is chosen so that there is way for the verifier to check that the given alleged codeword is actually a valid codeword. For example, this can be done if the above code is *locally testable* and *(relaxed) locally correctable*,[8] or by providing an auxiliary proof that the codeword is close to being valid (as in [BS08]).

3. Lastly, the PCP/IOP is designed to have a mechanism for ensuring that the message encoded within the codeword - an alleged computation - is indeed a valid and accepting computation.

Our construction also shares this basic schema but departs significantly in the details. Let us focus first on Step 1. The first main difference here is that in our construction we only provide an encoding of the NP witness (i.e., the satisfying assignment in the case of 3SAT) rather than the entire computation (which includes also the values of all the clauses in ϕ). Intuitively, this makes our job harder when handling Step 3. The second main difference, on which we elaborate next, is that the code that we use in Step 1 is a high rate code which is not a *multiplication code*.

Multiplication Codes and How to Avoid[9] Them. Loosely speaking, a multiplication code [Mei13, Mei14] is a linear error correcting code C, over a finite field $\mathbb{F}$, so that the set $\{c_1 \star c_2 : c_1, c_2 \in C\}$, where $c_1 \star c_2$ denotes entry-wise multiplication (over $\mathbb{F}$), is itself a code (i.e., it has large relative distance). The archtypical example of a multiplication code is the Reed-Solomon code (since the product of two sufficiently low degree polynomials is a low degree polynomial). As elegantly articulated in the works of Meir [Mei13, Mei14], the multiplication property is leveraged in PCP and IOP constructions to facilitate checking of *non-linear* relations that arise in the computation (e.g., verifying correctness of AND gates).

Unfortunately, all known multiplications codes (e.g., the Reed-Solomon [RS60] code, the Reed-Muller code [Mul54, Ree54], low rate tensor codes [Mei13, Mei14] or AG codes [Sti06, BKK+16]) have rate less than 1/2. Since we are aiming for rate close to 1, we cannot afford to encode the entire computation using such codes.

Instead, we encode the satisfying assignment w using a high rate binary code $C : \{0,1\}^n \to \{0,1\}^{(1+\gamma/2)\cdot n}$ with constant relative distance. Beyond having high rate and constant relative distance, we will also need for C to be (1) locally testable, (2) (relaxed) locally correctable, and (3) support a certain "sumcheck-like property", elaborated on below.

It turns out that the tensor product of a high-rate binary code of constant relative distance satisfies all the aforementioned properties. Given a linear code $C : \{0,1\}^k \to \{0,1\}^n$, consider the

[8] Informally, a code is said to be locally testable if, given a string w, it is possible to determine whether w is a codeword of C, or far from all codewords in C, by reading only a small part of w. A code is said to be locally correctable if, given a codeword c that has been partially corrupted by some errors, it is possible to recover any coordinate of c by reading only a small part of the corrupted version of c. Finally, in relaxed local corecation, the local corrector is additionally allowed to abort whenever a corruption is detected.

[9] Actually, our construction *does* use multiplication codes (specifically the Reed-Muller or Low Degree Extension code). What we avoid is to explicitly send an encoding of the computation (or witness) via a multiplication code.

(two-dimensional) *tensor product code* $C \otimes C : \{0,1\}^{k \times k} \to \{0,1\}^{n \times n}$; we will define the tensor product formally in Definition 2.4, but for now, we will view the codewords of $C \otimes C$ as $n \times n$ matrices with the constraints that the rows and columns are all codewords of the base code C.

It is well-known that the tensor product operation squares the rate and the relative distance of the base code. In particular, if the base code has high-rate and constant relative distance, then so does its tensor product. Moreover, a recent line of work [BS06, BV15, Vid15, KMRS17, GRR18] has established the local testability and relaxed local correctability of high-rate tensor codes.[10] Thus, given a satisfying assignment $w \in \{0,1\}^n$ for ϕ, the prover in our IOP first sends $C(w)$ (which has length $(1 + \gamma/2) \cdot n$). Next, addressing Step 2 in the outline, we observe that C is indeed locally testable and relaxed locally correctable with query complexity $O(\sqrt{n})$.

Thus, we are left with the question of how to implement Step 3, which is the key technical challenge. This is where we use the "sumcheck-like" property of our code C. To explain our approach we first need to take a brief detour into a classical method for constructing PCPs (with rate approaching 0) due to [BFL91, BFLS91]. Our presentation follows [Sud00].

A Quick Recap of Classical PCP Techniques. Imagine momentarily that the prover can provide an encoding of the satisfying assignment w under the *low degree extension code* (a variant of the Reed Muller code). This code is very "PCP-friendly" but has very poor rate.

In more detail, let $\mathbb{F}$ be a finite field of size $|\mathbb{F}| \gg \log(n)$ and let $\hat{w} : \mathbb{F}^{\log(n)} \to \mathbb{F}$ be the unique multilinear polynomial such that for every $i \in \{0,1\}^{\log(n)}$ it holds that $\hat{w}(i) = w_i$, where we identify $\{0,1\}^{\log(n)}$ with $[n]$ in the natural way. The existence of such a (unique) polynomial, referred to as the *multilinear extension of* w, basically follows from interpolation, see Section 2.4 for details. Observe that the truth table of $\hat{w}$ has *super-polynomial* length and so we cannot afford for the prover to send $\hat{w}$. Nevertheless, let us assume for now that the verifier also has oracle access to $\hat{w}$.

Consider the polynomial $P : (\mathbb{F}^{\log(n)})^3 \times \mathbb{F}^3 \to \mathbb{F}$ of total degree $O(\log n)$ defined as:

$$P(i_1, i_2, i_3, b_1, b_2, b_3) = \hat{I}_\phi(i_1, i_2, i_3, b_1, b_2, b_3) \cdot \Big(\hat{w}(i_1) - b_1\Big) \cdot \Big(\hat{w}(i_2) - b_2\Big) \cdot \Big(\hat{w}(i_3) - b_3\Big), \qquad (1)$$

where $\hat{I}_\phi$ is a multilinear extension of a Boolean function I_ϕ that on input $(i_1, i_2, i_3, b_1, b_2, b_3) \in \{0,1\}^{3\log(n)+3}$ outputs 1 if the clause $(x_{i_1} = b_1) \vee (x_{i_2} = b_2) \vee (x_{i_3} = b_3)$ appears in ϕ and otherwise outputs 0. The significance of P is that it has the following easy-to-verify property: P is identically 0 in the hypercube $\{0,1\}^{3\log(n)+3}$ if and only if $\hat{w}$ corresponds to a satisfying assignment for ϕ.

Thus, we only need to check that indeed $P|_{\{0,1\}^{3\log(n)+3}} \equiv 0$. This can be done using a variant of the classical (interactive) sumcheck protocol [LFKN92].[11] More specifically, and without getting into the details, there exists a constant round interactive proof for checking whether a given low degree polynomial $Q : \mathbb{F}^m \to \mathbb{F}$ is identically 0 in $\{0,1\}^m$. Most importantly, the verifier in the protocol only needs oracle access to Q and moreover, only makes a single query to Q.

It is very instructive to think of the above protocol as an *interactive reduction* - a central notion in our work. Generally speaking, in an interactive reduction a (computationally) complex claim about an input w is reduced, by interaction with a prover, to a much simpler claim about w. Completeness means that if the original claim was true then the honest prover will make the verifier generate a

[10] Local testability requires the dimension of the tensor product to be at least 3. For simplicity we ignore this fact in this high-level overview.

[11] Since we aim for a constant-round IOP, we can use a constant-round variant of sumcheck with communication $O(\sqrt{n})$.

true (and simpler) claim, whereas soundness means that if the original claim was false, then no matter what the prover does, with high probability, either the verifier rejects or it generates a false claim. Since the claim has been simplified, intuitively, progress has been made.

We emphasize that in an interactive reduction the verifier doesn't get any form of access to the input w - it merely reduces the complexity of claims about w, without ever seeing it. For example, the above protocol (for checking whether the polynomial Q vanishes on the hypercube) can be viewed as an interactive reduction from a claim about 2^m values of Q to a claim about a single value.

Using this reduction, applied to the polynomial P, the prover and verifier can reduce the satisfiability of the formula ϕ to a claim about a single point of the polynomial P. The verifier can now directly check this claim, via Eq. (1), by making three queries to $\hat{w}$.[12]

Taking a step back, what we started off with was a claim that w is a satisfying assignment for ϕ and we ended up with claims about three particular values of $\hat{w}$. Thus, we can view this entire process itself as an interactive reduction from the claim that the assignment w satisfies the formula ϕ to claims about three specific points of its low degree extension $\hat{w}$. We emphasize that in this interactive reduction, which we denote by Π_{reduce}, the verifier only needs to get ϕ and doesn't need any form of access to w.

Back to Our IOP Construction. If we could afford for the prover to send $\hat{w}$ then at this point we would be done - after sending $\hat{w}$, the prover and verifier run Π_{reduce} and then the verifier checks the three claims about $\hat{w}$ by reading the three corresponding points from the prover's first message (using also the local testability and correctability of the low degree extension).

Alas, in our actual IOP the prover can only send a high-rate encoding $C(w)$ and cannot afford to send $\hat{w}$. Still, we can use Π_{reduce} to our advantage. In particular, after the prover sends $C(w)$, the two parties run Π_{reduce}. The reduction generates claims about three values of $\hat{w}$. We are now faced with a (potentially) simpler task. Given oracle access to $C(w)$, we merely need to check the three claims about $\hat{w}$. For simplicity let us focus on one of these three claims, that is, a claim of the form $\hat{w}(z) = b$ (where $z \in \mathbb{F}^{\log(n)}$ and $b \in \mathbb{F}$).

It is natural to wonder at this point whether checking that $\hat{w}(z) = b$, given oracle access to $C(w)$, is any simpler than checking that $\phi(w) = 1$. We would like to argue that the answer is affirmative. In particular, since the low degree extension is a *linear* code (over the field $\mathbb{F}$), the claim $\hat{w}(z) = b$ is *linear* - i.e., it can be rephrased as a claim of the form $\langle \lambda_z, w \rangle = b$, for some $\lambda_z \in \mathbb{F}^n$ (that depends only on $z \in \mathbb{F}^{\log(n)}$).

Thus, we need a procedure for checking a linear claim about w, given oracle access to $C(w)$. Observing that C is a tensor code, a natural approach is to use the classical sumcheck protocol. Recall that the sumcheck protocol is an interactive proof for computing $\sum_{i \in [n]} w_i$, given oracle access to $C(w)$. While the protocol was originally designed specifically for the low degree extension code, it was later abstracted by Meir [Mei13], who showed that it can be applied to any tensor code.

The discussion so far comes close to resolving our problem. The difficulty that remains is that we would like to check that $\langle \lambda_z, w \rangle = b$ whereas the sumcheck protocol supports claims of the form $\langle \mathbf{1}, w \rangle = b$, where $\mathbf{1}$ is the all 1's vector. That is, the sumcheck protocol seems limited to the particular linear claim in which all coefficients are equal to 1. Unfortunately, the linear claim in question (corresponding to the vector λ_z) does not have this form.

[12] Here we also use the fact that the verifier can compute $\hat{I}_\phi$ by itself in $\tilde{O}(n)$ time, see Section 2.4 for details.

Before proceeding, we remark that if C were a *multiplication* code then we could have easily handled this difficulty be applying the sumcheck protocol to the codeword $C(w) \star C(\lambda_z)$.

Sumcheck for Rank 1 Tensor Coefficients. While we do not know how to extend the sumcheck protocol to handle linear claims with arbitrary coefficients, we show that it is possible to extend it to handle a particular form of coefficient structure. Luckily, the vector λ_z has a suitable form.

More specifically, we show how to extend the sumcheck protocol to computing linear claims of the form $\langle \lambda, w \rangle$ for any $\lambda \in \mathbb{F}^n$ which corresponds to a rank 1 matrix of dimension $\sqrt{n} \times \sqrt{n}$ (or more generally to any rank 1 tensor). That is, we assume that there exist $\lambda^{(1)}, \lambda^{(2)} \in \mathbb{F}^{\sqrt{n}}$ such that $\lambda = \lambda^{(1)} \otimes \lambda^{(2)}$ (where $\otimes$ denotes the tensor product, and we view λ simultaneously as a vector in $\mathbb{F}^n$ and as a matrix in $\mathbb{F}^{\sqrt{n}} \times \mathbb{F}^{\sqrt{n}}$ in the natural way). The fact that λ_z has this structure follows from the fact that the low degree extension is itself a tensor code.

Thus, we would like to use the sumcheck protocol to compute $\langle \lambda^{(1)} \otimes \lambda^{(2)}, w \rangle$. Let $C_0 : \{0,1\}^{n_0} \to \{0,1\}^{n_0'}$ be a systematic linear code such that $C = C_0 \otimes C_0$. Thus, $n_0 = \sqrt{n}$ and $n_0' = \sqrt{(1+\gamma/2) \cdot n}$. Let $c = C(w)$. We view c as an $n_0' \times n_0'$ dimensional matrix in the natural way, and denote its (i,j)-th entry by $c_{i,j}$.

In our sumcheck variant, the (honest) prover sends the message $\pi \in \mathbb{F}^{n_0'}$ which is defined as $\pi_i = \sum_{j \in [n_0]} \lambda_j^{(2)} \cdot c_{i,j}$, for every $i \in [n_0']$. In other words, π is computed by taking a linear combination of the first n_0 columns of c, with coefficients corresponding to $\lambda^{(2)}$.

At first glance it may seem as though π is a codeword of C_0. This is actually not true since C_0 is linear over the field $\mathbb{GF}(2)$ whereas we are using coefficients in a different (and larger) field $\mathbb{F}$. Nevertheless, if we choose $\mathbb{F}$ to be an extension field of $\mathbb{GF}(2)$, then with some elementary algebraic manipulations, we can show that π can be decomposed into $\log_2(|\mathbb{F}|)$ codewords of C_0.

The verifier, given a string $\tilde{\pi}$ which is allegedly equal to π, first checks that $\tilde{\pi}$ indeed consists of the aforementioned $\log(|\mathbb{F}|)$ codewords and rejects otherwise. Since both π and $\tilde{\pi}$ are composed of codewords, this test ensures us that if they differ then they must differ on a constant number of coordinates.

The verifier then chooses a random $i^* \in [n_0']$ and checks that $\tilde{\pi}_{i^*} = \sum_{j \in [n_0]} \lambda_j^{(2)} \cdot c_{i^*,j}$. Assuming that the prover sent $\tilde{\pi} \neq \pi$, with constant probability over the choice of i^* it holds that $\tilde{\pi}_{i^*} \neq \pi_{i^*} = \sum_{j \in [n_0]} \lambda_j^{(2)} \cdot c_{i^*,j}$ and so the verifier rejects. This probability can be amplified by choosing a suitably large constant number of random i^*'s. Note that verifier only reads a constant number of columns from c and so the number of queries to c is $O(n_0') = O(\sqrt{n})$.

Since we can now assume that the prover actually sent π, the verifier can simply output $\sum_{i \in [n_0]} \lambda_i^{(1)} \cdot \pi_i = \sum_{i,j \in [n_0]} \lambda_i^{(1)} \cdot \lambda_j^{(2)} \cdot c_{i,j} = \langle \lambda^{(1)} \otimes \lambda^{(2)}, w \rangle$ as desired.

This concludes the description of the warmup. Observe that the communication complexity is $(1+\gamma/2) \cdot n + \tilde{O}(\sqrt{n}) \leq (1+\gamma) \cdot n$ and the query complexity is $O(\sqrt{n})$ as promised.

A Brief Digest: Interactive Code Switching. The key idea in the above proof is that, using interaction, we are able to "switch" between different tensor codes during the protocol. More specifically, we implicitly use a (low-rate) multiplication code to check correctness of the computation, but are able to use a high-rate tensor code for actually encoding the witness. The key facilitator for switching between these codes is the power of the sumcheck protocol.

1.2.1 Additional Steps From Warmup to Theorem 2

Constant Query Complexity. The approach outlined so far only yields an IOP with $O(\sqrt{n})$ query complexity. To obtain a result with *constant* query complexity we follow the usual route - query reduction via composition [AS98]. In more detail, rather than actually performing the queries we will ask the prover to provide a constant query PCP (or more accurately a PCP of proximity, or PCPP) that the verifier would have accepted had it read these queries. Since the PCPP is applied to an input of length $O(\sqrt{n})$ (or more accurately to a computation of size $\tilde{O}(\sqrt{n})$) we can afford to use existing PCPP constructions (e.g., those with poly-logarithmic overhead [BS08, Din07]). A similar type of IOP composition was used also in [BCG+17] and as in their work, we utilize interaction to pay only linearly in the randomness complexity of the IOPP (rather than exponentially as in standard PCP composition).

Actually making this idea go through is somewhat more technically invloved. For example, we need to ensure that our base IOP (with $O(\sqrt{n})$ query complexity) is *robust* (i.e., the verifier cannot be made to accept even if the prover is allowed to flip a constant fraction of its answers a posteriori). We defer the details to the technical sections.

Extending to General Bounded-Space Computations. When trying to extend our approach past 3SAT, we observe that the main property that we used is that 3SAT has an interactive reduction to a linear claim about the witness, where the linear claim has a rank 1 tensor structure.

Using the doubly-efficient interactive proofs of Reingold *et. al* [RRR16] we show that a similar statement holds for any NP relation computable in polynomial-time and bounded polynomial-space. This basically follows from the fact that the verifier in [RRR16] runs in sublinear time given access to the low degree extension of its input. Plugging in the [RRR16] protocol instead of Π_{reduce} lets us obtain a high-rate IOP for any non-deterministic bounded space computation, thereby proving Theorem 2.

1.3 Open problems

Hardness of approximation. One of the key applications of PCPs is to the field of *hardness of approximation* (see, e.g., [FGL+96, Hås01]). A fascinating open question, pointed out by [BCG+17, ARW17], is whether IOPs may have similar implications. In particular, it is tempting to try to use high-rate IOPs to establish GapETH (i.e., that GapSAT requires exponential-time) based on a better understood hardness assumption. This approach could also potentially yield new hardness of approximation results in fine-grained complexity [ARW17, Rub18].

Shorter PCPs. As mentioned above, the work of [FS11] shows that SAT does not have a PCP with length that is a fixed polynomial in the witness size (let alone with rate approaching 1), unless NP $\subseteq$ co-NP/poly. This still leaves open the possibility that an interesting sub-class of NP relations has such short proofs. Somewhat along this vein, a recent work of Ben Eliezer *et. al* [BEFLR19] constructs a PCPP with length $n \cdot (\log n)^{o(1)}$ but for a very specific problem.

While we have constructed nearly optimal IOPs for CircuitSAT, the question of whether CircuitSAT has a linear length constant-query PCP remains wide open. A major difficulty in trying to adapt our approach (as well as previous approaches) is that we apply an *interactive* version of the sumcheck protocol that has sub-linear communication. In contrast, the only way in which we know how to

make the sumcheck protocol non-interactive (i.e., by specifying the answers of all possible queries of the verifier) leads to super-linear proof length.

Lastly, while [BKK+16] construct a linear-length PCP for CircuitSAT with non-trivial query complexity, we are curious whether CircuitSAT has a PCP with rate approaching 1 with *any* non-trivial query complexity.

Extending to general bounded-depth computations. A interesting open problem is to extend our result to NP relations computable in *bounded depth*. A natural approach to doing so would be to replace the [RRR16] protocol in our proof, with the protocol of Goldwasser *et. al* [GKR15] (a doubly-efficient interactive proof for bounded depth computations). We believe that this would indeed yield a high rate IOP, but the resulting query complexity would be super-constant (due to the super-constant number of rounds in the [GKR15] protocol).

Interactive PCPs. Our results can also be interpreted as *interactive* PCP*s* (IPCP) [KR08], a more restricted model than IOP in which the prover first sends a single long message to which the verifier has oracle access (like PCPs), followed by short interactive protocol with sublinear communication during which the verifier reads the prover's messages in full (like in interactive proofs).

Specifically, Theorem 2 (see Theorem 3.1) gives an IPCP in which the first message has length $(1+\gamma)\cdot n$ and constant query complexity, and the rest of the communication is of length n^{β}, for arbitrarily small constants $\gamma, \beta > 0$. Optimizing over communication complexity of the IP part gives an IPCP in which the first message has length $(1+2^{-(\log m)^{1-\varepsilon_0}})\cdot n$ and query complexity $2^{(\log m)^{1-\varepsilon_0}}$, and the rest of communication is of length $2^{(\log m)^{1-\varepsilon_0}}$, where m is the input length, n is the witness length, and $\varepsilon_0 > 0$ is a small absolute constant (see Corollary 3.4). This should be contrasted with the main result of [KR08] that gives IPCP in which the first message has length $\mathrm{poly}(n)$ with constant query complexity (in fact, one query suffices), and the rest of the communication has length $\mathrm{poly}\log n$.

The bound of $2^{(\log m)^{1-\varepsilon_0}}$ on communication of IP part is inherited from the work of [RRR16], as well as the comunication required for performing 'sumcheck' for high-rate tensor codes (see discussion in Section 1.2). An interesting open problem is whether one can adapt the variant of sumcheck we use to work also for the high-rate locally testable and relaxed locally correctable codes of [KMRS17, GRR18] which can potentially improve the communication of the IP part to $(\log n)^{O(\log\log n)}$ (when replacing [RRR16] with [GKR15]).

1.4 Organization

We start with preliminaries in Section 2. Our main results are stated formally in Section 3. In Section 4 we introduce the main definitions and lemmas that will be used in the proof. These lemmas are proved in Sections 5 to 7. In Section 8 we use the results obtained in the previous section to construct our IOPP and in Section 9 we construct our IOP.

2 Preliminaries

The relative distance between strings $x, y \in \Sigma^n$, over a finite alphabet Σ, is the fraction of coordinates $i \in [n]$ on which x and y differ, and is denoted by $\mathrm{dist}(x,y) := |\{i \in [n] : x_i \neq y_i\}|/n$. The relative distance of $x \in \Sigma^n$ from a non-empty set $S \subseteq \Sigma^n$ is $\mathrm{dist}(x,S) = \min_{y\in S}\mathrm{dist}(x,y)$.

2.1 Interactive oracle proofs

Throughout this work for an NP relation $\mathcal{R}$, we use n to denote the length of the NP *witness*. We also define $\mathcal{R}(x) = \{w : (x, w) \in \mathcal{R}\}$.

Definition 2.1 (Interactive oracle proof (IOP))**.** *A (public coin)* interactive oracle proof *for an* NP *relation $\mathcal{R}$ is a pair $(\mathcal{P}, \mathcal{V})$ of probabilistic algorithms that satisfy the following requirements.*

- **Input:** *$\mathcal{P}$ receives a pair (x, w) as an input, and $\mathcal{V}$ receives x as an input.*
- **Communication phase:** *$\mathcal{P}$ and $\mathcal{V}$ interact for $\ell = \ell(n)$ rounds with total communication $\mathrm{cc} = \mathrm{cc}(n)$, where $\mathcal{V}$'s messages only depend on the explicit input x and $\mathcal{V}$'s randomness.*
- **Query phase:** *At the end of the interaction, $\mathcal{V}$ makes $q = q(n)$ (non-adaptive) queries to the transcript, and applies a predicate $\phi : \{0,1\}^{q(n)} \to \{\texttt{accept}, \texttt{reject}\}$, where the location of the queries and the predicate ϕ only depends on the explicit input x and $\mathcal{V}$'s randomness.*
- **Completeness:** *If $(x, w) \in \mathcal{R}$, then when $\mathcal{V}$ interacts with $\mathcal{P}$, it accepts with probability* 1*.*
- **Soundness:** *If $\mathcal{R}(x) = \emptyset$, then for any prover strategy $\mathcal{P}^*$, when $\mathcal{V}$ interacts with $\mathcal{P}^*$, it accepts with probability at most $\varepsilon = \varepsilon(n)$.*

We call ℓ, cc, q, and ε the round complexity, communication complexity, query complexity, and soundness error of the interactive oracle proof, respectively. We say that the interactive oracle proof is $(\alpha(n), \varepsilon(n)$-robust if the soundness requirement is replaced with the stronger requirement that the answers to the queries made by $\mathcal{V}$ are $\alpha(n)$-close to $\phi^{-1}(\texttt{accept})$ with probability at most $\varepsilon(n)$.

Interactive Oracle Proofs of Proximity (IOPP). We next define IOPPs which are a variant of IOPs in which the verifier only gets oracle access to the input and is required to reject inputs that are *far* from the language.

Following the PCPP literature, we will consider pair languages which are sets of the form $\mathcal{L} \subseteq \{(x, w) \in \{0,1\}^* \times \{0,1\}^*\}$. We refer to x (which will be given explicitly to the verifier) as the explicit input and to w (to which the verifier only has oracle access) as the implicit input. For $x \in \{0,1\}^*$, we use the notation $\mathcal{L}_x := \{w \colon (x, w) \in \mathcal{L}\}$. We will sometimes consider pair languages in which the explicit input is empty. For simplicity we refer to such pair languages simply as languages.

Definition 2.2 (Interactive oracle proof of proximity (IOPP))**.** *A public-coin* interactive oracle proof of proximity *for the pair language $\mathcal{L}$ is a pair $(\mathcal{P}, \mathcal{V})$ of probabilistic algorithms that satisfy the following requirements.*

- **Input:** *$\mathcal{P}$ receives a pair (x, w) as an input, and $\mathcal{V}$ receives x as an input and also gets oracle access to w, where $|w| = n$.*
- **Communication phase:** *$\mathcal{P}$ and $\mathcal{V}$ interact for $\ell = \ell(n)$ rounds with total communication $\mathrm{cc}(n)$, where $\mathcal{V}$'s messages only depend on the explicit input x and $\mathcal{V}$'s randomness.*
- **Query phase:** *At the end of the interaction, $\mathcal{V}$ makes $q = q(n)$ (non-adaptive) queries to the implicit input w and to the transcript, and applies a predicate $\phi : \{0,1\}^{q(n)} \to \{\texttt{accept}, \texttt{reject}\}$, where the location of the queries and the predicate ϕ only depend on the explicit input x and $\mathcal{V}$'s randomness.*

- **Completeness:** *If $(x, w) \in \mathcal{L}$, then when $\mathcal{V}$ interacts with $\mathcal{P}$, it accepts with probability 1.*
- **Soundness:** *If w is $\delta(n)$-far from $\mathcal{L}_x$, then for any prover strategy $\mathcal{P}^*$, when $\mathcal{V}$ interacts with $\mathcal{P}^*$, it accepts with probability at most $\varepsilon = \varepsilon(n)$.*

We refer to δ as the proximity parameter of the IOPP. Similarly to IOPs, we refer to ℓ, cc, q, and ε as the round complexity, communication complexity, query complexity, and soundness error of the IOPP, respectively. As before, we say that the interactive oracle proof of proximity is $(\alpha(n), \varepsilon(n))$-robust if the soundness requirement is replaced with the stronger requirement that the answers to the queries made by $\mathcal{V}$ (both to the implicit input and the transcript) are $\alpha(n)$-close to $\phi^{-1}(\texttt{accept})$ with probability at most $\varepsilon(n)$.

2.2 Error-correcting codes

Let Σ be a finite alphabet, and k, n be positive integers (the message length and the block length, respectively). An (error-correcting) code is an injective map $C : \Sigma^k \to \Sigma^n$. The elements in the domain of C are called messages, and the elements in the image of C are called codewords. We say that C is systematic if the message is a prefix of the corresponding codeword, i.e., for every $x \in \Sigma^k$ there exists $z \in \Sigma^{n-k}$ such that $C(x) = (x, z)$.

The rate of a code $C : \Sigma^k \to \Sigma^n$ is the ratio $\rho := \frac{k}{n}$. The relative distance $\mathrm{dist}(C)$ of C is the maximum $\delta > 0$ such that for every pair of distinct messages $x, y \in \Sigma^k$ it holds that $\mathrm{dist}(C(x), C(y)) \geq \delta$.

If $\Sigma = \mathbb{F}$ for some finite field $\mathbb{F}$, and C is a linear map between the vector spaces $\mathbb{F}^k$ and $\mathbb{F}^n$ then we say that C is linear. The generating matrix of a linear code $C : \mathbb{F}^k \to \mathbb{F}^n$ is a matrix $G \in \mathbb{F}^{n \times k}$ such that $C(x) = G \cdot x$ for any $x \in \mathbb{F}^k$. A parity-check matrix for C is an $(n-k) \times n$ matrix H over $\mathbb{F}$ such that $\mathrm{Ker}(H) = \mathrm{Im}(C)$. If C is linear then its relative distance is equal to the minimal Hamming weight of any non-zero codeword.

Code Ensembles. In this work we will typically want error-correcting codes that are defined for an infinite sequence of message lengths $\mathcal{I} \subseteq \mathbb{N}$. Thus, a code ensemble $C = \{C_k : (\Sigma_k)^k \to (\Sigma_k)^n\}_{k \in \mathcal{I}}$ is a countable collection of error correcting codes, one for each input length $k \in \mathcal{I}$. We say that C has rate $\rho = \rho(k) \in (0, 1)$ and relative distance $\delta = \delta(k) \in (0, 1)$ if for any sufficiently large $k \in \mathcal{I}$, the code C_k has rate $\rho(k)$ and relative distance $\delta(k)$.

We say that C is T-time encodable if given $x \in (\Sigma_k)^k$, the codeword $C_k(x)$ can be generated in time $T(k)$. The code is linear-time encodable (resp., quasi-linear time encodable) if it is encodable in time $O(k)$ (resp., $\tilde{O}(k)$).

High-Rate Codes. A main ingredient in our constructions is the efficiently encodable binary code due to Justesen [Jus72]. Actually we use a somewhat non-standard setting of Justesen's code. First, we are interested in very high rate. Second, we need a construction for *every* (sufficiently large) message length (rather than just for an infinite sequence of message lengths). Thus, we provide a full proof in Appendix A.

Theorem 2.3 (High-rate Justesen code)**.** *For any $\gamma = \gamma(k) \in \left(\frac{100 \log k}{k}, 1\right)$ there exists a systematic code ensemble $C = \{C_k : \{0,1\}^k \to \{0,1\}^n\}_{k \in \mathbb{N}}$ of binary linear codes of rate at least $1 - \gamma$ and relative distance $\Omega(\gamma^3)$. Moreover,*

- *C_k is encodable in time $k \cdot \text{polylog}(k)$.*
- *Any individual coordinate in the encoding can be generated in time $k \cdot \text{polylog}(k)$ and space $\text{polylog}(k)$.*

2.2.1 Tensor codes

A main ingredient in our constructions is the tensor product operation, defined as follows (see, e.g., [Sud01, DSW06]).

Definition 2.4 (Tensor codes)**.** *Let $C : \mathbb{F}^k \to \mathbb{F}^n$, $C' : \mathbb{F}^{k'} \to \mathbb{F}^{n'}$ be linear codes, and let $G \in \mathbb{F}^{n \times k}$ and $G' \in \mathbb{F}^{n' \times k'}$ be generating matrices for C, C' respectively. Then the* tensor code *$C \otimes C' : \mathbb{F}^{k \times k'} \to \mathbb{F}^{n \times n'}$ is defined as $(C \otimes C')(M) = G \cdot M \cdot (G')^T$.*

Note that the codewords of $C \otimes C'$ are $n \times n'$ binary matrices whose columns belong to the code C and whose rows belong to the code C'. The following effects of the tensor product operation on the classical parameters of the code are well known.

Fact 2.5. *Suppose that $C : \mathbb{F}^k \to \mathbb{F}^n$, $C' : \mathbb{F}^{k'} \to \mathbb{F}^{n'}$ are linear codes of rates ρ, ρ' and relative distances δ, δ' respectively. Then, the tensor code $C \otimes C'$ has rate $\rho \cdot \rho'$ and relative distance $\delta \cdot \delta'$.*

For a linear code C, let $C^{\otimes 1} := C$ and $C^{\otimes t} := C \otimes C^{\otimes (t-1)}$, for any $t \geq 2$. The codewords of $C^{\otimes t} : \mathbb{F}^{k^t} \to \mathbb{F}^{n^t}$ can be viewed as t-dimensional cubes, satisfying that their projection on any axis-parallel line is a codeword of C. Moreover, by the above fact, if C has rate ρ and relative distance δ then $C^{\otimes t}$ has rate ρ^t and relative distance δ^t.

Finally, for $\lambda^{(1)}, \lambda^{(2)}, \ldots, \lambda^{(t)} \in \mathbb{F}^n$ we let $\lambda^{(1)} \otimes \lambda^{(2)} \otimes \cdots \otimes \lambda^{(t)} \in \mathbb{F}^{n^t}$ denote the vector satisfying that $(\lambda^{(1)} \otimes \lambda^{(2)} \otimes \cdots \otimes \lambda^{(t)})_{i_1, i_2, \ldots, i_t} = (\lambda^{(1)})_{i_1} \cdot (\lambda^{(2)})_{i_2} \cdots (\lambda^{(t)})_{i_t}$ for any $i_1, i_2, \ldots, i_t \in [n]$.

2.2.2 Relaxed Locally Correctable Codes

Locally correctable codes are error-correcting codes in which it is possible to recover individual sybmols from a noisy codeword by reading only a few of its coordinates. We consider here a relaxation, due to [GRR18] (extending the work of [BGH+06]), that allows the corrector to reject by outputting a special abort symbol.

Definition 2.6 (Relaxed Locally Correctable Codes (RLCCs))**.** *Let $C : \Sigma^k \to \Sigma^n$ be an error correcting code. We say that C is* relaxed locally correctable wrt correcting radius *$\delta_{\mathsf{dec}} \in (0, 1]$,* query complexity *$q$* and soundness error *$\varepsilon$ (*RLCC*) if there exists a randomized oracle machine M, called the corrector, which gets as input oracle access to $w \in \Sigma^n$ and explicit access to an index $i \in [n]$, makes at most q queries to the oracle and satisfies the following two conditions.*

1. *If $w \in \mathcal{C}$, then $M^w(i) = w_i$ with probability 1.*
2. *If w is δ_{dec}-close to some codeword $c \in C$, then with probability at least $1 - \varepsilon$ it holds that $M^w(i)$ either outputs c_i or a special abort symbol $\perp$.*

We say that C is (α, ε) robust if it satisfies the following more stringent requirement:

1. *If w is δ_{dec}-close to some codeword $c \in C$, then with probability $1 - \varepsilon$ the view of the corrector is α-far from any view that would make it output a value in $\Sigma \backslash \{c_i\}$.*

We extend the definition to a *code ensemble* in the natural way. A code ensemble $C = \{C^{(k)} : (\Sigma_k)^k \to (\Sigma_k)^n\}$ is a relaxed locally correctable code with query complexity $q = q(k)$ and soundness error $\varepsilon = \varepsilon(k)$ if there exists a constant $\delta_{\mathsf{dec}} \in (0, 1]$ such that for every k the code $C^{(k)}$ is relaxed locally correctable wrt correcting radius δ_{dec} with query complexity $q(k)$ and soundness error $\varepsilon(k)$. The code ensemble C is (α, ε)-robust for $\alpha = \alpha(k)$ and $\varepsilon = \varepsilon(k)$ if for every k the code C_k is $(\alpha(k), \varepsilon(k))$-robust.

Relaxed locally decodable codes (RLDC) [BGH+06] are defined similarly to RLCCs except that machine M, referred to as the decoder, needs to recover the message bits, rather than the codewords bits. Observe that any *systematic* RLCC is also automatically an RLDC.

2.3 Constructible Finite Fields

We will use finite fields extensively throughout this work. For sake of efficient implementation of the field operations, we need the field to be *constructible*.

Definition 2.7. *We say that an ensemble of finite fields $\mathbb{F} = (\mathbb{F}_n)_{n\in\mathbb{N}}$ is* constructible *if elements in $\mathbb{F}_n$ can be represented by $O(\log(|\mathbb{F}_n|))$ bits and field operations (i.e., addition, subtraction, multiplication, inversion and sampling random elements) can all be performed in $\mathrm{polylog}(|\mathbb{F}_n|)$ time given this representation.*

Lemma 2.8 (see [Sho88])**.** *For every $S = S(n) \geq 1$, there exists a constructible field ensemble of characteristic 2 and size $O(S)$.*

2.4 Low Degree Extension

Let $\mathbb{F}$ be a finite field, $H \subseteq \mathbb{F}$ an additive subgroup of $\mathbb{F}$ and let $m \in \mathbb{N}$. Let $\hat{I} : \mathbb{F}^m \times \mathbb{F}^m \to \mathbb{F}$ be the individual degree $|H| - 1$ polynomial defined as:

$$\hat{I}(x, z) \stackrel{\text{def}}{=} \prod_{i\in[m]} \prod_{h\in H\setminus\{0\}} \frac{z_i - x_i + h}{h}. \tag{2}$$

Fact 2.9. *For every $h, h' \in H^m$ it holds that $\hat{I}(h, h') = 1$ if $h = h'$ and $\hat{I}(h, h') = 0$ otherwise.*

Proof. For $h = h'$ this follows by inspection. For $h \neq h'$, observe that $h - h' \in H\setminus\{0\}$ (since H is an additive group) and so the product includes a zero term (corresponding to $h - h'$). □

Proposition 2.10. *For every function $\phi : H^m \to \mathbb{F}$, there exists a unique extension of ϕ into an individual degree $|H| - 1$ polynomial $\hat{\phi} : \mathbb{F}^m \to \mathbb{F}$, which agrees with ϕ on H^m (i.e., $\hat{\phi}|_{H^m} \equiv \phi$).*

The polynomial $\hat{\phi}$ is called the *low degree extension* of ϕ (with respect to $\mathbb{F}$, H and m).

Proof of Proposition 2.10. Consider the polynomial $\hat{\phi}(x) = \sum_{h\in H^m} \phi(h) \cdot \hat{I}(x, h)$. The fact that $\hat{\phi}$ and ϕ agree on H^m follows from Fact 2.9. Uniqueness follows from the fact that any two individual degree $H - 1$ polynomials that agree on H^m, agree everywhere. □

2.5 The Sumcheck Protocol

We use (a slight variant of) the celebrated sumcheck protocol of [LFKN92] for verifying the sum of a low-degree polynomial P over all inputs in a subcube $\mathbb{H}^m \subseteq \mathbb{F}^m$. This variant allows for a trade-off between the number of rounds and the total amount of communication.

Lemma 2.11 (The Sumcheck Protocol)**.** *Fix a constructible finite field ensemble $\mathbb{H} = (\mathbb{H}_n)_{n\in\mathbb{N}}$, a constructible finite field ensemble $\mathbb{F} = (\mathbb{F}_n)_{n\in\mathbb{N}}$ (i.e., $\mathbb{F}_n \supseteq \mathbb{H}_n$ is an extension field of $\mathbb{H}_n$) such that $|\mathbb{H}|_n \geq \log(|\mathbb{F}_n|)$.*

For every $m = m(n) \geq 1$ and round parameter $r = r(n) \in [m]$, where r divides m, there exists a r-round public-coin interactive protocol as follows.

The verifier $\mathcal{V}_{\mathsf{sumchk}}$ gets as input $n \in \mathbb{N}$. The prover $\mathcal{P}_{\mathsf{sumchk}}$ gets as input a polynomial $P\colon \mathbb{F}^m \to \mathbb{F}$ of individual degree $t = t(n) \in [|\mathbb{H}|, |\mathbb{F}|-1]$. At the end of the interaction $\mathcal{V}_{\mathsf{sumchk}}$ either rejects or outputs a point $z \in \mathbb{F}^m$, which depends only on $\mathcal{V}_{\mathsf{sumchk}}$'s coin tosses, and a value $v \in \mathbb{F}$ such that:

- **Completeness:** *If $\sum_{z\in\mathbb{H}^m} P(z) = 0$, then $(\mathcal{P}_{\mathsf{sumchk}}(P,n), \mathcal{V}_{\mathsf{sumchk}}(n))$ outputs (z,v) such that $P(z) = v$.*
- **Soundness:** *If $\sum_{z\in\mathbb{H}^m} P(z) \neq 0$, for every cheating prover $\tilde{P}_{\mathsf{sumchk}}$, with probability $(1 - \frac{t\cdot m}{|\mathbb{F}|})$, either $\mathcal{V}_{\mathsf{sumchk}}$ rejects or $P(z) \neq v$.*
- **Complexity:** *The total communication is $\left(r \cdot (t+1)^{m/r} \cdot \log|\mathbb{F}| + m \cdot \log|\mathbb{F}|\right)$. The prover runs in time $\left(|\mathbb{H}|^m \cdot t^{O(m/r)} \cdot \mathrm{poly}(m, \log|\mathbb{F}|)\right)$ and the verifier runs in time $\left(t^{O(m/r)} \cdot \mathrm{poly}(m, \log|\mathbb{F}|)\right)$.*

3 Formal statement of our results

We first state our IOP results in Section 3.1. Then, in Section 3.2, we state our IOPP results.

3.1 IOP Results

Our main result is an IOP for every NP relation that can be verified in bounded space. The general statement is given in Theorem 3.1 below, which gives tradeoffs depending on parameters β (which offers a tradeoff between number of rounds and total communication) and γ (which offers a tradeoff between the rate and the query, communication and verification complexities). Since the statement of Theorem 3.1 is somewhat involved, it may be useful for the first reading to skip directly to Corollary 3.3 which considers a particularly interesting setting of the parameters.

In the following we say that a function $\alpha = \alpha(n) \in (0,1)$ is nice if it is computable in time polylog(n) and $\alpha(\Theta(n)) = \Theta(\alpha(n))$ (e.g., $\alpha(n) = 1/\log(n)$ is nice).

Theorem 3.1 (IOP for NP)**.** *Let $\mathcal{L} \in$ NP with corresponding relation $\mathcal{R}_{\mathcal{L}}$ in which the instances have length m and witnesses have length n, where n and m are polynomially related, and such that $\mathcal{R}_{\mathcal{L}}$ can be decided in time $\mathrm{poly}(n)$ and space $s \geq \log(n)$. Also, we assume that $m \geq n$ (i.e., instances are not shorter than their corresponding witnesses).*[13]

Let $\gamma = \gamma(m) \in (0,1)$ and $\beta = \beta(m) \in (0,1)$ be nice functions such that $\mathrm{poly}(1/\beta) \leq \log(n)$ and $\gamma \geq m^{-O(\beta)}$. Then, there exists a $\beta^{-O(1/\beta)}$-round IOP for $\mathcal{L}$ with soundness error $1/2$. The query

[13] This requirement can be handled by simply padding the input with 0's if necessary. This increases the input size by at most n.

complexity is $(\gamma\beta)^{-O(1/\beta)}$ *and the communication consists of a first (deterministic) message sent by the prover of length* $(1+\gamma)\cdot n$ *bits followed by* $\text{poly}\left(m^{\beta}, (\gamma\beta)^{-1/\beta}, s\right)$ *additional communication. The* IOP *verifier runs in time* $\tilde{O}(m) + \text{poly}\left(m^{\beta}, (\gamma\beta)^{-1/\beta}, s\right)$ *and the* IOP *prover runs in time* $\text{poly}(m)$.

Remark 3.2. *The soundness error in Theorem 3.1 can be reduced by parallel repetition, while observing that since the first prover message is deterministic, it does not to be repeated (note that in a typical setting of parameters the first prover message in Theorem 3.1 is by far the largest part of the communication).*

A particularly interesting setting of parameters is when $\gamma > 0$ is an arbitrarily small constant, and $\beta > 0$ is a sufficiently small constant. In this regime we obtain the following corollary from Theorem 3.1.

Corollary 3.3. *There exists a fixed constant* $\xi > 0$ *such that the following holds. Let* $\mathcal{L} \in \mathsf{NP}$ *be as in Theorem 3.1 with* $s = s(n) \leq n^{\xi}$. *Then, for any constant* $\gamma, \varepsilon > 0$ *there exists an* IOP *for* $\mathcal{L}$ *with communication complexity* $(1+\gamma)\cdot n$, *constant query complexity, constant round complexity, and soundness error* ε. *The verifier runs in time* $\tilde{O}(m)$ *and the prover runs in time* $\text{poly}(m)$.

We also state another corollary of Theorem 3.1, focusing on minimizing the communication complexity following the first prover's message. Specifically, letting $\beta(m) = \frac{1}{(\log m)^{\Theta(\varepsilon_0)}}$ and $\gamma(m) = 2^{-(\log m)^{1-\Theta(\varepsilon_0)}}$ we obtain the following.

Corollary 3.4. *There exists an absolute constant* $\varepsilon_0 > 0$ *such that the following holds. Let* $\mathcal{L} \in \mathsf{NP}$ *be as in Theorem 3.1 with* $s = s(n) \leq 2^{(\log m)^{1-\varepsilon_0}}$. *Then, there exists a* $2^{(\log m)^{\varepsilon_0}}$*-round* IOP *for* $\mathcal{L}$ *with soundness error* $1/2$. *The query complexity is* $2^{(\log m)^{1-\varepsilon_0}}$, *and the communication consists of a first (deterministic) message sent by the prover of length* $(1 + 2^{-(\log m)^{1-\varepsilon_0}})\cdot n$, *followed by* $2^{(\log m)^{1-\varepsilon_0}}$ *additional communication. The* IOP *verifier runs in time* $\tilde{O}(m)$ *and the* IOP *prover runs in time* $\text{poly}(m)$.

3.2 IOPP Results

We next state our IOPP results. Our main result is an IOPP for bounded space computations in which the communication complexity is slightly less than n.

Theorem 3.5 (IOPP for bounded-space computations)**.** *Let* $\mathcal{L}$ *be a language computable in time* $\text{poly}(n)$ *with space* $s = s(n) \geq \log n$. *Then for every* $\delta = \delta(n) \in (0,1)$, $\beta = \beta(n) \in (0,1)$ *and* $\gamma = \gamma(n) \in (0,1)$ *such that* $\text{poly}(1/\beta) \leq \log(n)$ *and* $\gamma = \gamma(n) \geq \frac{200\cdot 4^{1/\beta}\cdot \log n}{n^{\beta/2}}$ *the following holds.*

There exists a $\beta^{-O(1/\beta)}$*-round* IOPP *for* $\mathcal{L}$ *with respect to proximity parameter* δ, *and with soundness error* $1/2$. *The query complexity is* $\text{poly}\left((\gamma\beta)^{-1/\beta}, 1/\delta\right)$, *and the communication consists of a first (deterministic) message sent by the prover of length* $\gamma\cdot n$ *bits followed by* $\text{poly}\left(n^{\beta}, (\gamma\beta)^{-1/\beta}, 1/\delta, s\right)$ *additional communication. The* IOP *verifier runs in time* $\text{poly}\left(n^{\beta}, (\gamma\beta)^{-1/\beta}, 1/\delta, s\right)$, *and the* IOP *prover runs in time* $\text{poly}(n)$.

As in the IOP case, letting $\delta, \gamma > 0$ be arbitrary constants and $\beta > 0$ be a sufficiently small constant, we obtain the following constant query IOPP.

Corollary 3.6. *For any* $\beta > 0$ *there exists a constant* $\varepsilon_0 > 0$ *such that the following holds. Let* $\mathcal{L}$ *be a language computable in time* $\text{poly}(n)$ *and space* n^{ε_0}. *Then, for any constants* $\delta, \gamma, \varepsilon > 0$ *there*

exists an IOPP *for* $\mathcal{L}$*, wrt proximity parameter* δ*, with communication complexity* $\gamma \cdot n$*, constant query complexity, constant round complexity, and soundness error* ε*. The verifier runs in time* n^{β}*, and the prover runs in time* $\mathrm{poly}(n)$*.*

4 Main ingredients

In this section we introduce the main definitions and claims that will be used to establish our results.

4.1 Notation and central definitions

In what follows, let $\mathcal{I} \subseteq \mathbb{N}$ be an infinite sequence of integers, and let $\mathbb{F} = \{\mathbb{F}_n\}_{n \in \mathcal{I}}$ be a constructible ensemble (see Definition 2.7) of finite fields of characteristic 2. Let $\mathcal{S} = \{S_n\}_{n \in \mathcal{I}}$ be a sequence of sets, where $\mathcal{S}_n \subseteq (\mathbb{F}_n)^n$ for any $n \in \mathcal{I}$ (i.e., S_n is a subset of the n-dimensional vector space over the field $\mathbb{F}_n$). We further assume that for any $n \in \mathcal{I}$, any vector $\lambda \in S_n$ can be represented by a short string $\ll\lambda\gg$ of length $a(n)$. The actual details of this representation depend on the specifics choice of the collection $\mathcal{S}$, which will be fixed later. Finally, we say that a pair language $\mathcal{L}$ has implicit length sequence $\mathcal{I}$ if for any $(x, w) \in \mathcal{L}$ it holds that $|w| \in \mathcal{I}$.

Definition 4.1 ($\mathcal{S}$-linear IOP reduction)**.** *An* $\mathcal{S}$-linear IOP reduction *for a pair language* $\mathcal{L}$ *with implicit length sequence* $\mathcal{I}$ *is defined similarly to an* IOP *except that rather than accepting or rejecting, at the end of the protocol the verifier either rejects or outputs a pair* $(\ll\lambda\gg, b) \in \{0,1\}^{a(n)} \times \mathbb{F}$ *such that the following holds:*

- **Completeness:** *If* $(x, w) \in \mathcal{L}$*, then when* $\mathcal{V}$ *interacts with* $\mathcal{P}$*, the output of* $\mathcal{V}$ *is a pair* $(\ll\lambda\gg, b) \in \{0,1\}^{a(n)} \times \mathbb{F}$ *so that* $\lambda \in S_n$ *and* $\langle \lambda, w \rangle = b$ *with probability* 1*, where* $n = |w|$*.*
- **Soundness:** *If* x *and* w *are such that* $w \notin \mathcal{L}_x$*, then for any prover strategy* $\mathcal{P}^*$*, when* $\mathcal{V}$ *interacts with* $\mathcal{P}^*$*, the probability that* $\mathcal{V}$ *outputs a pair* $(\ll\lambda\gg, b) \in \{0,1\}^{a(n)} \times \mathbb{F}$ *such that* $\lambda \in S_n$ *and* $\langle \lambda, w \rangle = b$ *is at most* $\varepsilon(n)$*.*

We say that the IOP *reduction is* robust *if it further satisfies the following requirement:*

- **Robustness:** *The* IOP *is* $(\alpha(n), \varepsilon(n))$-robust *if for every* $w \notin \mathcal{L}_x$ *and prover strategy* $\mathcal{P}^*$*, when* $\mathcal{V}$ *interacts with* $\mathcal{P}^*$*, with probability* $1 - \varepsilon$*, there exists* $\lambda \in S_n$ *and* $b \in \mathbb{F}$ *such that* $\langle \lambda, w \rangle \neq b$ *and for any view* v' *that is* α*-close to the view* v *of the verifier in the protocol, the verifier either rejects or outputs* $(\ll\lambda\gg, b)$*.*

Thus, the output of the verifier in an IOP reduction (assuming that it does not reject) is a concise representation of a vector $\lambda \in \mathbb{F}^n$ and a scalar $b \in \mathbb{F}$, where the condition $\langle \lambda, w \rangle = b$ is interpreted as accepting and $\langle \lambda, w \rangle \neq b$ is interpreted as rejecting.

Remark 4.2. *The robustness property in Definition 4.1 is somewhat subtle. It basically says that (with high probability) any view that is close to the actual view of the verifier either leads to rejection or to a* unique λ *and* b *(that do not depend on the specific neighboring view) such that* $\langle \lambda, w \rangle \neq b$*.*

Also, observe that robustness is a stronger requirement than soundness in the sense that (α, ε)*-robustness implies* ε*-soundness for* any $\alpha > 0$*.*

Our construction will rely on codes for which it is possible to apply a "sumcheck-like" procedure for linear functions corresponding to the set $\mathcal{S}$.

Definition 4.3 ($\mathcal{S}$-sumcheckable code)**.** *A code ensemble $C = \{C_n : \{0,1\}^n \to \{0,1\}^{n'}\}_{n\in\mathcal{I}}$ is an* $\mathcal{S}$-sumcheckable code *if there exists an* IOPP *for the language*

$$\mathcal{L}_{\mathcal{S}} := \left\{ \Big(\big(\ll\lambda\gg, b\big), w \Big) \in \big(S_n \times \mathbb{F}_n\big) \times \{0,1\}^{n'} : \exists x \in \{0,1\}^n \text{ s.t. } w = C_n(x) \text{ and } \langle \lambda, x\rangle = b \right\},$$

where $(\ll\lambda\gg, b)$ is the explicit input and w is the implicit input.

We further require that in the corresponding IOPP*, $\mathcal{V}$'s messages and query locations only depend on $\mathcal{V}$'s randomness (and not on the explicit input $(\ll\lambda\gg, b)$).*

To construct our final IOPs, we shall also require that our sumcheckable code is locally decomposable. Intuitively, this definition requires that individual bits in the encoding $C(m_1 \circ m_2)$ can be reconstructed given oracle access to $C(m_1)$ and $C(m_2)$. More formally:

Definition 4.4 (Locally decomposable code)**.** *Let $C : \{0,1\}^n \to \{0,1\}^{n'}$ be a code, and let ℓ be such that ℓ divides n. We say that C is* locally ℓ-decomposable *if there exist a* base code *$C_0 : \{0,1\}^{n/\ell} \to \{0,1\}^*$, and an oracle machine A such that the following holds. For every $m^{(1)}, \ldots, m^{(\ell)} \in \{0,1\}^{n/\ell}$, given oracle access to the codewords $C_0(m^{(1)}), \ldots, C_0(m^{(\ell)})$ and explicit access to an index $i \in [n']$, the machine A makes at most $O(1)$ queries to each oracle, and outputs the i-th coordinate of $C(m^{(1)} \circ \cdots \circ m^{(\ell)})$.*

We extend the definition of local decomposability to *code ensembles* in the natural way. Namely, let $C = \{C_n : \{0,1\}^n \to \{0,1\}^{n'}\}_{n\in\mathcal{I}}$ be a code ensemble, and let $\ell = \ell(n) \geq 1$ be such that $\ell(n)$ divides n for any $n \in \mathcal{I}$. We say that C is locally ℓ-decomposable if there exists a code ensemble $C_0 := \{(C_0)_{n/\ell} : \{0,1\}^{n/\ell} \to \{0,1\}^*\}_{n\in\mathcal{I}}$ such that for any $n \in \mathcal{I}$, the code C_n is locally $\ell(n)$-decomposable with respect to the base code $(C_0)_{n/\ell}$, and moreover the corresponding oracle machine can be implemented by a time $\ell \cdot \mathrm{polylog}(n)$-time Turing machine.

4.2 Main claims

We next show how to combine together a $\mathcal{S}$-linear IOP reduction for a language $\mathcal{L}$, together with a $\mathcal{S}$-sumcheckable code, to obtain an IOPP for $\mathcal{L}$.

Lemma 4.5 (Code switching)**.** *Suppose that the following exist:*

- *An $\mathcal{S}$-linear* IOP *reduction for a pair language $\mathcal{L}$ with implicit length sequence $\mathcal{I}$ of communication complexity* cc*, query complexity q, round complexity ℓ, and soundness error ε.*
- *A systematic $\mathcal{S}$-sumcheckable code $C = \{C_n : \{0,1\}^n \to \{0,1\}^{n'}\}_{n\in\mathcal{I}}$ of rate $1-\gamma$ whose corresponding* IOPP *has communication complexity* cc′*, query complexity q', round complexity ℓ', proximity parameter $\delta'(n') < \frac{\mathrm{dist}(C_n)}{2}$, and soundness error ε'.*

Then, there exists an IOPP *for $\mathcal{L}$ with communication complexity $\frac{\gamma(n)}{1-\gamma(n)} \cdot n + \mathrm{cc}(n) + \mathrm{cc}'(n')$, query complexity $q(n) + q'(n')$, round complexity $\ell(n) + \ell'(n')$, proximity parameter $\frac{\delta'(n')}{1-\gamma(n)}$, and soundness error $\varepsilon(n) + \varepsilon'(n')$.*

Moreover,

- *The communication phase consists of a single prover message which is the non-systematic part of $C_n(w)$ (i.e., of length $\frac{\gamma(n)}{1-\gamma(n)} \cdot n$), followed by a two-way communication of length $\mathrm{cc}(n) + \mathrm{cc}'(n')$.*
- *If the verifier in the $\mathcal{S}$-linear* IOP *reduction and the $\mathcal{S}$-sumcheckable code* IOPP *have running times T and T', respectively, then the verifier in the resulting* IOP *has running time $T(n) + T'(n')$.*
- *If the $\mathcal{S}$-linear* IOP *reduction and the $\mathcal{S}$-sumcheckable code* IOPP *are (α, ε)-robust and (α', ε')-robust, respectively, then the resulting* IOPP *is $\left(\min\left\{\frac{\alpha(n)}{2}, \frac{\alpha'(n')}{2}\right\}, \varepsilon(n) + \varepsilon'(n')\right)$-robust with query complexity $2 \cdot \max\{q(n), q'(n')\}$ and verifier running time $2 \cdot \max\{T(n), T'(n')\}$.*
- *If the prover in the $\mathcal{S}$-linear* IOP *reduction and the $\mathcal{S}$-sumcheckable code* IOPP *have running times T and T', respectively, and the $\mathcal{S}$-sumcheckable code has encoding time T_{enc}, then the prover in the resulting* IOP *has running time $T_{enc}(n) + T(n) + T'(n')$.*

We prove Lemma 4.5 in Section 5.

We next define a particular subset ensemble $\mathcal{S}$ that we will focus on throughout this work. Loosely speaking this subset, parameterized by an integer t, consists of all vectors which can be interpreted as rank 1 tensors of dimension t. For example, when $t = 2$ the set consists of all $\sqrt{n} \times \sqrt{n}$ matrices of rank 1 (viewed as vectors in $\mathbb{F}^n$ in the natural way).

Let $t = t(n) \geq 2$ be an integer valued function. For any $n \in \mathbb{N}$ we let $\bar{n} := \left(\lceil n^{1/(2t(n))} \rceil\right)^{2t(n)}$ be the minimal integer no smaller than n that is a power of $2t(n)$.[14] We mention that for values of t that we will consider, it holds that $\bar{n}$ is extremely close to n: i.e., $\bar{n} \leq (1 + o(1)) \cdot n$ (see Eq. (6) below). We define the set of indices as $\mathcal{I} = \{\bar{n} \mid n \in \mathbb{N}\}$, and the subset ensemble as $\mathcal{S}_{\otimes t} = \left\{S^{(\bar{n})}_{\otimes t}\right\}_{\bar{n} \in \mathcal{I}}$ where

$$S^{(\bar{n})}_{\otimes t} = \left\{\lambda^{(1)} \otimes \cdots \otimes \lambda^{(t)} \,:\, \lambda^{(1)}, \ldots, \lambda^{(t)} \in (\mathbb{F}_{\bar{n}})^{\bar{n}^{1/t}}\right\}$$

for any $\bar{n} \in \mathcal{I}$. Having defined the set $\mathcal{S}_{\otimes t}$ we next define a concise representation for its elements. The choice of representation is the natural one: for any $\lambda = \lambda^{(1)} \otimes \cdots \otimes \lambda^{(t)} \in S^{(\bar{n})}_{\otimes t}$, we let $\ll\lambda\gg = (\lambda^{(1)}, \ldots, \lambda^{(t)})$. Note that the length of this representation is $a(\bar{n}) = t \cdot \bar{n}^{1/t} \cdot \log_2(|\mathbb{F}|)$.

The following lemma establishes the existence of an $\mathcal{S}_{\otimes t}$-linear IOP reduction for a large class of languages. The lemma is stated relative to a parameter β which offers a tradeoff between the number of rounds and total amount of communication. For sake of simplicity, it may be useful for the reader to focus first on the setting where β is a small constant (e.g., $\beta = 0.1$).

Lemma 4.6 ($\mathcal{S}_{\otimes t}$-linear IOP reduction)**.** *Let $\mathcal{L}$ be a language with implicit length sequence $\mathcal{I}$, that can be decided in time $\mathrm{poly}(n)$ and space $s = s(n) \geq \log(n)$. Then, for any $t = t(n) \in [\log n]$, $\beta = \beta(n) \in (0, 1/2)$ with $\mathrm{poly}(1/\beta) \leq \log(n)$, and constructible field ensemble $\mathbb{F} = (\mathbb{F}_n)_{n \in \mathbb{N}}$ with $|\mathbb{F}_n| \geq \beta^{-O(1/\beta)} \cdot \mathrm{polylog}(n)$, the following holds.*

There exists an $\mathcal{S}_{\otimes t}$-linear IOP *reduction for $\mathcal{L}$ with communication and query complexity $n^{\beta} \cdot \beta^{-O(1/\beta)} \cdot \mathrm{poly}(s) \cdot \mathrm{polylog}(|\mathbb{F}|)$, round complexity $\beta^{-O(1/\beta)}$, and soundness error $\mathrm{poly}(1/|\mathbb{F}|)$.*

Moreover,

[14] Jumping ahead, we mention that the reason that we want $\bar{n}$ to be a power of $2t$, rather than just t, is that it will sometimes be more convenient for us to use tensors of even dimension.

- *The* IOP *reduction is* $\left(\beta^{O(1/\beta)}, \text{poly}(1/|\mathbb{F}|)\right)$*-robust.*
- *The verifier's running time is* $\text{poly}\left(n^{\beta} \cdot \beta^{-1/\beta}, s, \log(|\mathbb{F}|)\right)$.
- *The prover's running time is* $\text{poly}(n, \log(|\mathbb{F}|))$.

We prove Lemma 4.6 in Section 6.

Lastly, the following lemma states the existence of an $\mathcal{S}_{\otimes t}$-sumcheckable code with an efficient IOPP.

Lemma 4.7 ($\mathcal{S}_{\otimes t}$-sumcheckable code)**.** *The following holds for any* $\delta = \delta(n) > 0$, $t = t(n) \in \{2, \ldots, (\log n)/2\}$, *and* $\gamma = \gamma(n) \in \left(\frac{200t \log n}{n^{1/(2t)}}, 1\right)$.

There exists a systematic linear $\mathcal{S}_{\otimes t}$*-sumcheckable code* $C = \{C_{\bar{n}} : \{0,1\}^{\bar{n}} \to \{0,1\}^{n'}\}_{\bar{n} \in \mathcal{I}}$, *where* $C_{\bar{n}}$ *has rate at least* $1 - \gamma$ *and relative distance* $(\gamma/t)^{O(t)}$, *and the corresponding* IOPP *has communication complexity* $n^{1/t} \cdot (t/\gamma)^{O(t)} \cdot \log(|\mathbb{F}|)$, *query complexity* $n^{1/t} \cdot (t/\gamma)^{O(t)} \cdot \frac{1}{\delta}$, *round complexity* t, *and soundness error* $1/2$ *wrt proximity parameter* δ.

Moreover,

- *The* IOPP *is* $\left((\gamma/t)^{O(t)} \cdot \delta, 1 - (\gamma/t)^{O(t)} \cdot \delta\right)$*-robust with query complexity* $n^{1/t} \cdot (t/\gamma)^{O(t)} \cdot \frac{1}{\delta} \cdot \log|\mathbb{F}|$.
- *The verifier's running time is* $n^{O(1/t)} \cdot (t/\gamma)^{O(t)} \cdot \frac{1}{\delta} \cdot \text{polylog}(|\mathbb{F}|)$.
- *The prover's running time is* $n' \cdot (t/\gamma)^{O(t)} \cdot \text{polylog}(|\mathbb{F}|)$.
- $C_{\bar{n}}$ *is encodable in time* $n \cdot \text{polylog}(n)$, *and for any integer* $d = d(n) \leq 2t(n) - 1$, *the code* $C_{\bar{n}}$ *is locally* $\bar{n}^{1-d/(2t)}$*-decomposable with running time* $n^{1-d/(2t)} \cdot \text{polylog}(n)$ *with respect to a base code* C_0 *of rate at least* $1 - \gamma$.

We prove Lemma 4.7 in Section 7.

5 Code switching – proof of Lemma 4.5

Let Π be the $\mathcal{S}$-linear IOP reduction, let $\{C_n\}_n$ be the $\mathcal{S}$-sumcheckable code, and let Π' be the corresponding IOPP (for $\mathcal{L}_{\mathcal{S}}$). The composed IOPP $\tilde{\Pi}$ is given in Fig. 1.

It can be verified that the communication complexity, query complexity, round complexity, verifier running time, and prover running time are all as claimed. Next we show that completeness, soundness, and robustness properties also hold.

Completeness. Suppose that $(x, w) \in \mathcal{L}$. Then, by the completeness of Π, with probability 1, the output of f is $(\ll\lambda\gg, b) \in \{0,1\}^{a(n)} \times \mathbb{F}_n$ satisfying $\langle \lambda, w \rangle = b$. Consequently, $((\ll\lambda\gg, b), (w, z)) = ((\ll\lambda\gg, b), C_n(w)) \in \mathcal{L}_{\mathcal{S}}$, and the protocol Π' will accept with probability 1.

The composed protocol $\tilde{\Pi}$:

Explicit Input: x.
Implicit Input: $w \in \{0,1\}^n$.

1. $\mathcal{P}$ sends the non-systematic part z of $C_n(w)$.
2. $\mathcal{P}$ and $\mathcal{V}$ run the $\mathcal{S}$-linear IOP reduction Π on explicit input x and implicit input w. If the verifier of Π rejects then $\mathcal{V}$ rejects. Otherwise, the protocol Π outputs a pair $(\ll\lambda\gg, b) \in \{0,1\}^{a(n)} \times \mathbb{F}_n$.
3. $\mathcal{P}$ and $\mathcal{V}$ run the IOPP Π' for the code C_n (wrt $\mathcal{S}$) on explicit input $(\ll\lambda\gg, b)$ and implicit input (w, z). The verifier $\mathcal{V}$ accepts if and only if the verifier of Π' accepts.

Figure 1: IOPP for $\mathcal{L}$

Soundness. We first show that soundness holds since it is simpler to establish although it will also follow as a special case of the robustness analysis below (see Remark 4.2).

Let x and w such that w is $\frac{\delta'}{1-\gamma}$-far from $\mathcal{L}_x$. Fix a cheating prover strategy $\mathcal{P}^*$. We assume without loss of generality that $\mathcal{P}^*$ is deterministic and denote the first message that it sends (on input (x, w)) by z.

Suppose first that (w, z) is δ'-far from the code C_n. In this case, we also have that (w, z) is δ'-far from $(\mathcal{L}_{\mathcal{S}})_{(\ll\lambda\gg,b)}$, and so the protocol Π' will reject with probability at least $1 - \varepsilon'$, in which case the protocol $\tilde{\Pi}$ will also reject. Hence we may assume that (w, z) is δ'-close to some codeword $C_n(w')$.

Next observe that our assumption that $\mathrm{dist}((w, z), C_n(w')) \leq \delta'$ implies that $\mathrm{dist}(w, w') \leq \frac{\delta' \cdot n'}{n} = \frac{\delta'}{1-\gamma}$. By assumption that w is $\frac{\delta'}{1-\gamma}$-far from $\mathcal{L}_x$, this implies in turn that $w' \notin \mathcal{L}_x$. Consequently, with probability at least $1 - \varepsilon$ the protocol Π will either reject (in which case the protocol $\tilde{\Pi}$ will also reject), or output $(\ll\lambda\gg, b)$ satisfying $\langle \lambda, w' \rangle \neq b$.

We claim that in the latter case (w, z) is δ'-far from $(\mathcal{L}_{\mathcal{S}})_{(\ll\lambda\gg,b)}$, and so the protocol Π' will reject with probability at least $1 - \varepsilon'$, in which case the protocol $\tilde{\Pi}$ will also reject. To see this, suppose in contradiction that (w, z) is δ'-close to some codeword $C_n(w'')$ with $\langle \lambda, w'' \rangle = b$. Then we have that $C_n(w')$ and $C_n(w'')$ are distinct codewords (since $\langle \lambda, w' \rangle \neq b$ but $\langle \lambda, w'' \rangle = b$) that are both of distance at most δ' from (w, z). By triangle inequality and our assumption that $\delta' < \frac{\mathrm{dist}(C_n)}{2}$ this implies in turn that $\mathrm{dist}(C_n(w'), C_n(w'')) \leq 2\delta' < \mathrm{dist}(C_n)$, a contradiction.

Robustness. In what follows we assume that the queries made by $\mathcal{V}$ as part of the protocols Π and Π' are of equal length. This can be achieved by duplicating $\frac{q}{q'}$ times the view of $\mathcal{V}$ in Π' if $q > q'$, and duplicating $\frac{q'}{q}$ times the view of $\mathcal{V}$ in Π otherwise, and accepting if and only if all duplicated views are consistent with each other and the original protocol $\tilde{\Pi}$ accepts. Note that this transformation increases the total number of queries and verifier running time to $2 \cdot \max\{q, q'\}$ and $2 \cdot \max\{T, T'\}$, respectively.

As in the soundness analysis, let x and w such that w is $\frac{\delta'}{1-\gamma}$-far from $\mathcal{L}_x$. Fix a (deterministic) cheating prover strategy $\mathcal{P}^*$ and denote its first message by z.

Consider first the case that (w, z) is δ'-far from the code C_n. As before, this implies in turn that (w, z) is δ'-far from $(\mathcal{L}_{\mathcal{S}})_{(\ll\lambda\gg,b)}$. By the robustness of Π', the queries made in the sub-protocol Π'

(to both the transcript and w) will be α'-far from making Π' accept with probability at least $1-\varepsilon'$. Since these account for half of the queries of the composed protocol, we get that the queries in the protocol $\tilde{\Pi}$ will be $\frac{\alpha'}{2}$-far from an accepting view. Hence in what follows we may assume that (w,z) is δ'-close to some codeword $C_n(w')$. Moreover, as we showed in the soundness analysis, it must be the case that $w' \notin \mathcal{L}_x$.

By the robustness of Π, with probability at least $1-\varepsilon$, there exist $\lambda \in \mathbb{F}^n$ and $b \in \mathbb{F}$ such that $\langle \lambda, w' \rangle \neq b$ and any view that is α-close to the view of the verifier of Π makes it either reject or output $(\ll\lambda\gg, b)$. Fix the coins of $\mathcal{V}$ only for Step 2 of $\tilde{\Pi}$ such that the foregoing event occurs and consider the corresponding λ and b (guaranteed by the fact that the event occurs). Note that this fixes half of the view of the verifier (corresponding to queries from Step 2). Denote this part of the view by view_Π. We emphasize that at this point the coins for the second half of the protocol (i.e., Step 3) have not been fixed.

Let view'_Π be α-close to view_Π.

Claim 5.1. *With probability $1-\varepsilon'$ over the remaining coins of $\mathcal{V}$ (i.e., those from Step 3), for every view $\mathsf{view}'_{\Pi'}$ that is α'-close to the partial view of $\mathcal{V}$ corresponding to Step 3, it holds that $\mathcal{V}$ rejects given the view $(\mathsf{view}'_\Pi, \mathsf{view}'_{\Pi'})$.*

Proof. Since view'_Π is α-close to view_Π, given view'_Π the protocol Π either rejects or outputs $(\ll\lambda\gg, b)$ (which have already been fixed). We next consider these two possibilities.

First, if Π rejects then also $\mathcal{V}$ rejects (regardless of the partial view for Step 3) and we are done. Thus, let us assume that Π outputs $(\ll\lambda\gg, b)$ given the view view'_Π.

Since $C_n(w')$ is δ'-close to (w,z) and $\langle \lambda, w' \rangle \neq b$, as was done in the soundness analysis, we can show that (w,z) is δ'-far from $(\mathcal{L}_\mathcal{S})_{(\ll\lambda\gg,b)}$. Thus, by the robustness of Π', with probability $1-\varepsilon'$ (over the coins in Step 3), the queries made at Step 3 (to Π' and (w,z)) are α'-far from making Π' accept. Thus, with probability $1-\varepsilon'$, for every view $\mathsf{view}'_{\Pi'}$ that is α'-close to the partial view of $\mathcal{V}$ corresponding to Step 3 of $\tilde{\Pi}$, the verifier $\mathcal{V}$ rejects given the view $(\mathsf{view}'_\Pi, \mathsf{view}'_{\Pi'})$. □

Since (by assumption) the two parts of the verifier's view have equal weights, using Claim 5.1, we have that with probability $1-\varepsilon-\varepsilon'$, any view that is $\min\left\{\frac{\alpha}{2}, \frac{\alpha'}{2}\right\}$-close to the actual view of $\mathcal{V}$ makes it reject.

6 Tensor IOPP reduction – proof of Lemma 4.6

We prove Lemma 4.6 based on a result of Reingold *et. al* [RRR16].

Theorem 6.1 (Doubly Efficient Interactive Proofs for Bounded Space [RRR16, see Corollary 9 and Theorem 10])**.** *Let $\mathcal{L}$ be a language computable in time $\mathrm{poly}(n)$ and space $s = s(n) \geq \log(n)$. Then, for every $\beta \in (0, 1/2)$ such that $\mathrm{poly}(1/\beta) \leq \log(n)$, the language $\mathcal{L}$ has a $\beta^{-O(1/\beta)}$-round public-coin interactive proof with perfect completeness and soundness error $\frac{1}{2}$. The communication complexity is $n^\beta \cdot \mathrm{poly}(s)$. The prover runs in time $\mathrm{poly}(n)$ and the verifier runs in time $\tilde{O}(n) + n^\beta \cdot \mathrm{poly}(s)$.*

Furthermore, the verifier runs in time $n^\beta \cdot \mathrm{poly}(s)$ given only oracle access to the low degree extension $\hat{x}$ of the input $x \in \{0,1\}^n$, where the low degree extension is relative to $\mathbb{F}$, $\mathbb{H}$ and dimension $\log_{|\mathbb{H}|}(n)$, where $\mathbb{H} = (\mathbb{H}_n)_{n\in\mathbb{N}}$ and $\mathbb{F} = (\mathbb{F}_n)_{n\in\mathbb{N}}$ are constructible field ensembles such that $\mathbb{F}$ is an extension field of $\mathbb{H}$, where $|\mathbb{H}| = \log(n) \cdot (1/\beta)^{O(1/\beta)}$ and $|\mathbb{F}| = \mathrm{poly}(|\mathbb{H}|)$.

Throughout this section we refer to the protocol of Theorem 6.1 as the RRR protocol. At a high level, we would like to prove Lemma 4.6 by simply running the RRR protocol wrt to $\mathcal{L}$. At first glance this seems problematic since the RRR verifier needs to get w as its input, whereas our reduction does not have any access whatsoever to w.

To overcome this difficulty we rely on the furthermore clause of Theorem 6.1, which states that all that the RRR verifier needs is oracle access to the low degree extension of w. Since the prover can provide (alleged) values for the queries that the verifier would like to make, all that we need to ensure is that the prover's claims about the low degree extension of w are correct. By relying on the fact that the low degree extension is a tensor code, it is possible to show that the verifier simply generates linear claims, where the coefficients of these linear claims have a rank 1 tensor structure, as required by Lemma 4.6.

There are two main issues that we encounter when trying to implement this high-level approach. First, the verifer in Lemma 4.6 should output a *single* claim about the low degree extension of w, whereas the RRR verifier may make many queries to $\hat{w}$ (which corresponds to generating *many* linear claims). Second, the furthermore clause of Lemma 4.6 requires robust soundness which is not directly provided by Theorem 6.1.

We handle these two issues in the following subsections. First, in Section 6.1 we show how to generically reduce the number of queries to the low degree extension (via interaction). Then, in Section 6.2 we show how to obtain robustness essentially for free, as long as one does not care about the query complexity. Lastly, in Section 6.3 we combine the above to derive Lemma 4.6.

6.1 Interactive Query Reduction for Low-Degree Extension

In this section we show how, using interaction, one can reduce the number of queries to the low degree extension encoding.

The "traditional" way to (interactively) reduce the number of queries k to the low degree extension code (originating in [KR08, GKR15]) is to take a low degree curve passing through all the k points (as well as some additional points), ask the prover for the evaluation on *all* points on the curve and check the correctness on a single *random* point. To make this approach work, one needs to use a large field $\mathbb{F}$ of cardinality $|\mathbb{F}| \geq k$. We would like to avoid the use of such a large field and so we do not follow this approach.

A natural generalization that has been considered in the literature (see, e.g., [RRR16]) is to use a high dimensional (low degree) manifold rather than a (univariate) curve. Unfortunately, this idea seems to introduce a quadratic dependence on k (to both communication and verification time), which again we would like to avoid. Instead, we take a new approach based on the sumcheck protocol, which is described next.

Lemma 6.2. *Let $\mathbb{F} = (\mathbb{F}_n)_{n\in\mathbb{N}}$ and $\mathbb{H} = (\mathbb{H}_n)_{n\in\mathbb{N}}$ be constructible field ensembles such that $\mathbb{F}_n$ is an extension field of $\mathbb{H}_n$ for every $n \in \mathbb{N}$.*

Let $(\mathcal{P}, \mathcal{V})$ be a public-coin interactive proof for a language $\mathcal{L}$ with soundness error ε, in which $\mathcal{V}$ receives oracle access to the low degree extension $\hat{x} : \mathbb{F}^m \to \mathbb{F}$ of the input $x \in \{0,1\}^n$, wrt $\mathbb{F}$, $\mathbb{H}$ and dimension $m = \log_{|\mathbb{H}|}(n)$. Then, there exists a public-coin interactive proof $(\mathcal{P}', \mathcal{V}')$ with soundness error $\varepsilon + O(m \cdot |\mathbb{H}|/|\mathbb{F}|)$, in which $\mathcal{V}'$ similarly receives oracle access to $\hat{x}$ but only makes a single query.

For every parameter $r \in [m]$, we can implement the foregoing interactive proof with an additional r rounds of communication, additive overhead of $r \cdot |\mathbb{H}|^{\lceil m/r \rceil} \cdot \mathrm{polylog}(|\mathbb{F}|) + m \cdot \mathrm{polylog}(|\mathbb{F}|)$ to the

communication complexity and verification time and $\mathrm{poly}(|\mathbb{F}|^m)$ *prover time overhead.*

Proof. Let $x \in \{0,1\}^n$ be an input to the protocol. Since $|\mathbb{H}|^m = n$ we can identify $\mathbb{H}^m$ with $[n]$ in some canonical way. Recall that the low degree extension of x wrt $\mathbb{F}$, $\mathbb{H}$, and m is the (unique) individual degree $|\mathbb{H}| - 1$ polynomial $\hat{x} : \mathbb{F}^m \to \mathbb{F}$ that agrees with x on $\mathbb{H}^m$ (see Section 2.4).

We start with an intuitive description and then proceed to a formal description of the protocol. The prover $\mathcal{P}'$ and verifier $\mathcal{V}'$ exactly emulate the communication phase of $(\mathcal{P}, \mathcal{V})$. Then, during the query phase, the verifier $\mathcal{V}$ needs to make some $\leq k$ queries to $\hat{x}$. Let $\{q_1, \ldots, q_k\} \subseteq \mathbb{F}^t$ be the locations that $\mathcal{V}$ wants to query (i.e., $\mathcal{V}$ wants to obtain the values $\hat{x}(q_j)$ for all $j \in [k]$). The prover $\mathcal{P}'$ sends to the verifier $\mathcal{V}'$ a list of alleged values for all these points. That is, values $(\nu_1, \ldots, \nu_k)$ which the prover claims are equal to $(\hat{x}(q_1), \ldots, \hat{x}(q_k))$.

Given $(\nu_1, \ldots, \nu_k)$, the verifier $\mathcal{V}'$ checks that $\mathcal{V}$ accepts given these answers (and immediately rejects otherwise). Thus, $\mathcal{V}'$ only needs to verify the claim

$$\forall j \in [k],\ \nu_j = \hat{x}(q_j). \tag{3}$$

To do so, $\mathcal{V}'$ selects at random $r_1, \ldots, r_k \in \mathbb{F}$. Observe that if Eq. (3) holds, then $\sum_j r_j \nu_j = \sum_j r_j \hat{x}(q_j)$. Otherwise however (i.e., if Eq. (3) does not hold) then, with probability $1 - 1/|\mathbb{F}|$, it holds that $\sum_j r_j \nu_j \neq \sum_j r_j \hat{x}(q_j)$. In other words, after choosing $r_1, \ldots, r_k$, it suffices for $\mathcal{V}'$ to check the identity

$$\sum_j r_j \nu_j = \sum_j r_j \hat{x}(q_j). \tag{4}$$

By definition of the low degree extension (see Proposition 2.10), the RHS of Eq. (4) can be expressed as $\sum_j r_j \sum_{h \in \mathbb{H}^m} \hat{x}(h) \cdot \hat{I}(h, q_j)$ (see Section 2.4 for the definition of $\hat{I}$ and more background on the low degree extension). Define $Q(\lambda) = \hat{x}(\lambda) \cdot \left(\sum_j r_j \cdot \hat{I}(\lambda, q_j) \right)$. Observe that Q is an individual degree $2(|\mathbb{H}| - 1)$ polynomial and that Eq. (4) is equivalent to checking that $\sum_{h \in \mathbb{H}^m} Q(h)$ is equal to some fixed value. We can perform this test using the sumcheck protocol (see Lemma 2.11) which requires only a single query to Q (which itself can be emulated using a single query to $\hat{x}$). We proceed to a formal description of the protocol.

1. Input for $\mathcal{P}'$: $x \in \{0,1\}^n$.
2. Oracle Input for $\mathcal{V}'$: $\hat{x} : \mathbb{F}^m \to \mathbb{F}$ the low degree extension of x.
3. $\mathcal{P}'$ and $\mathcal{V}'$ emulate the communication phase of $(\mathcal{P}, \mathcal{V})$. At the end of this phase either $\mathcal{V}$ rejects (in which case $\mathcal{V}'$ immediately rejects), or it generates a list of queries $q_1, \ldots, q_k \in \mathbb{F}^m$ to $\hat{x}$.
4. $\mathcal{V}'$ sends $q_1, \ldots, q_k$ to $\mathcal{P}'$ which responds with values $\nu_1, \ldots, \nu_k$ such that $\nu_j = \hat{x}(q_j)$.
5. Upon receiving the values $\tilde{\nu}_1, \ldots, \tilde{\nu}_k$ (where, allegedly, $\tilde{\nu}_j = \nu_j$) the verifier $\mathcal{V}'$ first checks that $\mathcal{V}$ accepts given the answer sequence $(\tilde{\nu}_1, \ldots, \tilde{\nu}_k)$. If not then $\mathcal{V}'$ immediately rejects.
6. $\mathcal{V}'$ chooses at random $r_1, \ldots, r_k \in \mathbb{F}$ and sends them to $\mathcal{P}'$.
7. Define $Q(\lambda) = \hat{x}(\lambda) \cdot \left(\sum_j r_j \cdot \hat{I}(\lambda, q_j) \right)$. The prover $\mathcal{P}'$ and verifier $\mathcal{V}'$ run the sumcheck protocol (see Lemma 2.11) wrt the claim $\sum_{h \in \mathbb{H}^m} Q(h) = \sum_j r_j \nu_j$. The sumcheck verifier outputs a claim of the form $Q(z) = \eta$ for some $z \in \mathbb{F}^m$ and $\eta \in \mathbb{F}$.
8. The verifier $\mathcal{V}'$ checks that $\eta = \hat{x}(z) \cdot \sum_j r_j \cdot \hat{I}(z, q_j)$, using a single query to $\hat{x}$.

Completeness. Suppose that $x \in \mathcal{L}$. By completeness of $(\mathcal{P}, \mathcal{V})$, in Item 3 $\mathcal{V}'$ generates queries $q_1, \ldots, q_k \in \mathbb{F}^m$ such that $\mathcal{V}$ accepts given the answer sequence $\{\nu_j = \hat{x}(q_j)\}_j$. Observe that

$$\begin{aligned}\sum_{h \in \mathbb{H}^m} Q(h) &= \sum_{h \in \mathbb{H}^m} \hat{x}(h) \cdot \left(\sum_j r_j \cdot \hat{I}(h, q_j) \right) \\ &= \sum_j r_j \sum_{h \in \mathbb{H}^m} \hat{x}(h) \cdot \hat{I}(h, q_j) \\ &= \sum_j r_j \hat{x}(q_j) \\ &= \sum_j r_j \nu_j\end{aligned}$$

and so by the completeness of Lemma 2.11, the sumcheck verifier outputs a claim (z, η) such that indeed $Q(z) = \eta$. Thus, the verifier accepts when checking that $\eta = \hat{x}(z) \cdot \sum_j r_j \cdot \hat{I}(z, q_j)$.

Soundness. Suppose that $x \notin \mathcal{L}$. By the soundness of $(\mathcal{P}, \mathcal{V})$, with probability at least ε (over the coins of $\mathcal{V}$) it holds that if $\mathcal{P}'$ sends $\tilde{\nu}_j = \nu_j$ for all $j \in [k]$, then $\mathcal{V}'$ will reject. Assume that the latter holds. Then, to avoid having $\mathcal{V}'$ immediately reject, $\mathcal{P}'$ must sends $\tilde{\nu}_{j^*} \neq \nu_{j^*}$ for some $j^* \in [k]$. Thus, with probability $1 - 1/|\mathbb{F}|$ over the choice of $r_1, \ldots, r_k$, it holds that

$$\sum_j r_j \tilde{\nu}_j \neq \sum_j r_j \nu_j = \sum_j r_j \hat{x}(q_j) = \sum_j r_j \sum_{h \in \mathbb{H}^m} \hat{x}(h) \cdot \hat{I}(h, q_j) = \sum_{h \in \mathbb{H}^m} Q(h).$$

Therefore, the sumcheck protocol is invoked on a *false* claim and with probability $1 - O(m \cdot |\mathbb{H}|/|\mathbb{F}|)$, the sumcheck verifier either rejects or outputs $z \in \mathbb{F}^m$ and $\eta \in \mathbb{F}$ such that $Q(z) \neq \eta$. Assuming the latter, $\mathcal{V}'$ rejects when checking that $\eta = \hat{x}(z) \cdot \sum_j r_j \cdot \hat{I}(z, q_j) = Q(z)$. Thus, overall, the verifier accepts with probability at most $\varepsilon + O(m \cdot |\mathbb{H}|/|\mathbb{F}|)$. □

6.2 Trivial Robustification of IOP Reductions

In this section we show how to transform any constant-round IOP reduction, in which the verifier reads the entire communication transcript (as in an interactive proof), into a *robust* IOP reduction. The transformation is trivial - we simply have the prover encode each of its messages before sending them (using a code with constant relative distance) and have the verifier check that it indeed receives encoded messages. We show that due to the encoding, and since the protocol only involves a constant ℓ number of messages, even in retrospect, making $O(1/\ell)$ changes to the transcript cannot help the prover.

Lemma 6.3. *Suppose that $(\mathcal{P}, \mathcal{V})$ is an ℓ-round $\mathcal{S}$-linear IOP reduction for a pair language $\mathcal{L}$ with communication complexity* cc*, query complexity* cc *(i.e., the verifier reads the entire communication transcript), soundness error ε, verifier running time T_V and prover running time T_P.*

Then, $\mathcal{L}$ has an ℓ-round $\mathcal{S}$-linear IOP reduction $(\mathcal{P}', \mathcal{V}')$ with robustness $(O(1/\ell), \varepsilon)$, communication complexity $O(\text{cc})$, query complexity $O(\text{cc})$, verifier running time $T_V + \tilde{O}(\text{cc})$ and prover running time $T_P + \tilde{O}(\text{cc})$.

Proof. Let $C : \{0,1\}^* \to \{0,1\}^*$ be the error correcting code from Theorem 2.3 and observe that C has constant rate, constant relative distance $\mathrm{dist}(C)$, and that both encoding, checking membership in the code, and decoding valid codewords, can be done in quasi-linear time.

We construct the robust IOP reduction $(\mathcal{P}', \mathcal{V}')$ by modifying $(\mathcal{P}, \mathcal{V})$ as follows. In each round $i \in [\ell]$, when $\mathcal{P}$ needs to send a message α_i, the prover $\mathcal{P}'$ instead sends the message $\hat{\alpha}_i = C(\alpha_i)$. In its query phase, the verifier $\mathcal{V}'$ reads the *entire* (encoded) transcript. The verifier $\mathcal{V}'$ checks that all messages $(\hat{\alpha}_1, \ldots, \hat{\alpha}_\ell)$ sent from $\mathcal{P}'$ are valid encodings under C, and rejects otherwise. Denote the corresponding underlying messages by $\alpha_1, \ldots, \alpha_\ell$ (i.e., $\hat{\alpha}_i = C(\alpha_i)$). Finally, $\mathcal{V}'$ accepts if and only if $\mathcal{V}$ accepts the transcript $(\alpha_1, \ldots, \alpha_\ell)$.

Completeness of the protocol $(\mathcal{P}', \mathcal{V}')$ follows directly from the completeness of $(\mathcal{P}, \mathcal{V})$. For robustness (which also implies soundness), let x and w such that $w \notin \mathcal{L}_x$ and fix a cheating prover $\mathcal{P}^*$. We need to show that with probability $1-\varepsilon$, there exists $\lambda \in \mathbb{F}^n$ and $b \in \mathbb{F}$ such that $\langle \lambda, w \rangle \neq b$ and for any view v' that is $\frac{\mathrm{dist}(C)}{2\ell}$-close to the view v of the verifier in the protocol, the verifier either rejects or outputs $(\ll\lambda\gg, b)$. Recall that in our protocol, the view of the verifier consists of the entire transcript sent by $\mathcal{P}^*$.

Denote the message sent by $\mathcal{P}^*$ by $(\tilde{\alpha}_1, \ldots, \tilde{\alpha}_\ell)$. For every $i \in [\ell]$, let $\hat{\alpha}_i$ be the closest codeword to $\tilde{\alpha}_i$ (breaking ties in some arbitrary fixed way). We refer to $(\hat{\alpha}_1, \ldots, \hat{\alpha}_\ell)$ as the *error-corrected* transcript.

Claim 6.4. *With probability $1 - \varepsilon$, it holds that the transcript $(\hat{\alpha}_1, \ldots, \hat{\alpha}_\ell)$ makes $\mathcal{V}'$ reject or outputs $(\ll\lambda\gg, b)$ such that $\langle \lambda, w \rangle \neq b$.*

Proof. Suppose otherwise. We can construct a cheating prover for $\mathcal{V}$ that emulates $\mathcal{P}^*$ while sending a decoding of its messages. Claim 6.4 now follows from the soundness of $\mathcal{V}$. □

Fix an execution of the protocol $(\mathcal{P}^*, \mathcal{V}')$ (i.e., fix the random coins of the verifier) conditioned on the event that the corresponding error-corrected transcript $(\hat{\alpha}_1, \ldots, \hat{\alpha}_\ell)$ makes $\mathcal{V}'$ either reject or output $(\ll\lambda\gg, b)$ such that $\langle \lambda, w \rangle \neq b$. By Claim 6.4, the event that we have conditioned on happens with probability $1-\varepsilon$. Denote the messages actually sent by $\mathcal{P}^*$ in the foregoing execution by $(\tilde{\alpha}_1, \ldots, \tilde{\alpha}_\ell)$ and notice that these messages are now fixed (since we have fixed the coins of the verifier).

Consider first the case that there exists some $i \in [\ell]$ such that $\tilde{\alpha}_i$ is $\mathrm{dist}(C)/2$ far from $\hat{\alpha}_i$. In such a case, all transcripts that are $\frac{\mathrm{dist}(C)}{2\ell}$-close to the actual transcript $(\tilde{\alpha}_1, \ldots, \tilde{\alpha}_\ell)$ will lead $\mathcal{V}'$ to reject (since $\mathcal{V}$ checks the i-th message is a codeword). Thus, the robustness property is trivially satisfied (wrt an arbitrary choice of $\lambda \in \mathbb{F}^n$ and $b \in \mathbb{F}$ for which $\langle \lambda, w \rangle \neq b$). Therefore, we may assume that each $\tilde{\alpha}_i$ is $\mathrm{dist}(C)/2$-close to $\hat{\alpha}_i$.

Observe that due to the distance of the code C, for any sequence of messages $(\tilde{\alpha}'_1, \ldots, \tilde{\alpha}'_\ell)$ that is $\frac{\mathrm{dist}(C)}{2\ell}$-close to $(\tilde{\alpha}_1, \ldots, \tilde{\alpha}_\ell)$, other than $(\hat{\alpha}_1, \ldots, \hat{\alpha}_\ell)$, there exists some $i \in [n]$ such that $\tilde{\alpha}'_i$ is not a codeword and so $\mathcal{V}'$ rejects given $(\tilde{\alpha}'_1, \ldots, \tilde{\alpha}'_\ell)$.

Recall that we have already conditioned on the event that $\mathcal{V}'$ either rejects given $(\hat{\alpha}_1, \ldots, \hat{\alpha}_\ell)$ or outputs $(\ll\lambda\gg, b)$ such that $\langle \lambda, w \rangle \neq b$. Let us first handle the former case (i.e., given $(\hat{\alpha}_1, \ldots, \hat{\alpha}_\ell)$ the verifier $\mathcal{V}'$ rejects). In such a case we can now conclude that for *every* transcript that is $\frac{\mathrm{dist}(C)}{2\ell}$-close to $(\tilde{\alpha}_1, \ldots, \tilde{\alpha}_\ell)$ the verifier $\mathcal{V}$ rejects and again the robustness property is trivially satisfied (wrt an arbitrary choice of $\lambda \in \mathbb{F}^n$ and $b \in \mathbb{F}$ for which $\langle \lambda, w \rangle \neq b$).

Thus, we are left with the case that given $(\hat{\alpha}_1, \ldots, \hat{\alpha}_\ell)$ the verifier $\mathcal{V}'$ outputs $(\ll\lambda\gg, b)$ such that $\langle \lambda, w \rangle \neq b$. In such case the robustness property also holds, wrt the the specific λ and b specified

by $\mathcal{V}'$ on input $(\hat{\alpha}_1, \ldots, \hat{\alpha}_\ell)$. This concludes the proof of Lemma 6.3.

□

6.3 Proof of Lemma 4.6

Let $\mathcal{L}$ be a language with implicit length sequence $\mathcal{I}$ that can be computed in time $\text{poly}(n)$ and space $s = s(n) \geq \log(n)$. Let $t = t(n)$, $\beta = \beta(n)$ and $\varepsilon = \varepsilon(n)$ as in the theorem's statement.

Let $\mathbb{H} = (\mathbb{H}_n)_{n\in\mathbb{N}}$ and $\mathbb{F} = (\mathbb{F}_n)_{n\in\mathbb{N}}$ be binary constructible field ensembles such that $\mathbb{F}_n$ is an extension field of $\mathbb{H}_n$ for every $n \in \mathbb{N}$, where $|\mathbb{H}| = \log(n) \cdot (1/\beta)^{O(1/\beta)}$ and $|\mathbb{F}| \geq \text{poly}(|\mathbb{H}|)$. Let $d = \log_{|\mathbb{H}|}(n)$. We choose $\mathbb{H}$ so that (1) d is an integer, and (2) t divides d. Note that $|\mathbb{F}| \geq |\mathbb{H}|^2$ (since $\mathbb{F}$ is a field extension) and that $d \leq \log(n)$.

By Theorem 6.1, there exists a constant-round public-coin interactive proof $(\mathcal{P}, \mathcal{V})$ for $\mathcal{L}$ with soundness error $1/2$, communication complexity $n^\beta \cdot \text{poly}(s, m)$ and prover run time $\text{poly}(n)$. By the furthermore part of Theorem 6.1, the verifier $\mathcal{V}$ runs in time $n^\beta \cdot \text{poly}(s, m)$ given oracle access to the low degree extension of the input w (wrt $\mathbb{F}$, $\mathbb{H}$ and d). We repeat this protocol $O(\log(|\mathbb{F}|))$ times (in parallel) to reduce the soundness error to $1/|\mathbb{F}|$.

At this point we apply the transformation of Lemma 6.2 to $(\mathcal{P}', \mathcal{V}')$ to obtain an IOP with soundness error $1/|\mathbb{F}| + O(d \cdot |\mathbb{H}|/|\mathbb{F}|) \leq \text{poly}(1/|\mathbb{F}|)$ in which the verifier only needs to make a single query to the low degree extension of w. Since the prover can provide an (alleged) value for this query, we can view the verifier as simply outputting a single claim about the low degree extension of w.

Let $(z, b) \in \mathbb{F}^d \times \mathbb{F}$ be the claim generated by the verifier. That is, the verifier would like to check that $\hat{w}(z) = b$ (where $\hat{w} : \mathbb{F}^d \to \mathbb{F}$ is the low degree extension of w). Using Proposition 2.10, this precisely corresponds to $\sum_{h\in\mathbb{H}^d} w(h) \cdot \hat{I}(z, h) = b$, a linear claim about w. In more detail, consider a vector $\lambda \in \mathbb{F}^n$ defined as $\lambda_h = \hat{I}(z, h)$ for every $h \in \mathbb{H}^d$ (recall that we associate the set $[n]$ with $\mathbb{H}^d$). Then the claim that the verifier generates is that $\langle \lambda, w \rangle = b$. Observe further that since $\hat{I}(z, h) = \prod_{i=1}^{d} \hat{I}(z_i, h_i)$ it holds that $\lambda \in S_{\otimes d}$.

Fact 6.5. *If $\lambda \in S_{\otimes d}$ and t divides d, then $\lambda \in S_{\otimes t}$.*

Proof. Since $\lambda \in S_{\otimes d}$, there exist $\lambda_1, \ldots, \lambda_d \in \mathbb{F}^{n^{1/d}}$ such that $\lambda = \lambda_1 \otimes \cdots \otimes \lambda_d$. For every $i \in [t]$, let $\lambda'_i = \lambda_{(d/t)\cdot(i-1)+1} \otimes \cdots \otimes \lambda_{(d/t)\cdot i} \in \mathbb{F}^{n^{1/t}}$. Observe that $\lambda'_1 \otimes \cdots \otimes \lambda'_t = \lambda_1 \otimes \cdots \otimes \lambda_d = \lambda$. Thus, $\lambda \in S_{\otimes t}$. □

Thus, the single query generated by the verifier does indeed lie in $\mathcal{S}_{\otimes t}$ as required. This concludes all but the furthermore part of Lemma 4.6. For the furthermore part, since the IOP contructed so far is a constant-round IOP, using Lemma 6.3 we can transform it into a robust IOP as desired.

7 $\mathcal{S}_{\otimes t}$-sumcheckable code – proof of Lemma 4.7

The sumcheckable code that we construct is the t-dimensional tensor $C^{\otimes t}$ of a high-rate binary code C. We show that it is sumcheckable by constructing the required IOPP, while utilizing known properties of tensor codes such as the sumcheck protocol [LFKN92, Mei13], local testing [Vid15], and a relaxed local correcting procedure [GRR18].

We start with the following lemma, which provides a sumcheck protocol for tensor codes that is very similar to our desired IOPP, except that we assume that the verifier is given oracle access

to a genuine codeword. To simulate this access when the implicit input is arbitrary we shall later combine this protocol with local testing (to reject inputs that are far from the code) and relaxed local correction (to decode the input for the nearest codeword) procedures for tensor codes.

Lemma 7.1 (Sumcheck for tensor codes)**.** *Let $C = \{C^{(n)} : \{0,1\}^k \to \{0,1\}^n\}_{n\in\mathbb{N}}$ be a systematic linear code ensemble of relative distance δ, and let $\mathbb{F}$ be a finite field of characteristic 2. Then for any integer $t \geq 2$ and $\varepsilon > 0$, there exist $d = O(\log(t/\varepsilon)/\delta)$ and a protocol $(\mathcal{P}, \mathcal{V})$ satisfying the following properties.*

- **Prover $\mathcal{P}$'s Input:** *$(\lambda^{(1)}, \ldots, \lambda^{(t)}, b) \in (\mathbb{F}^k)^t \times \mathbb{F}$ and a codeword $c = C^{\otimes t}(x) \in \{0,1\}^{n^t}$.*
- **Verifier $\mathcal{V}$'s Input:** *the same $(\lambda^{(1)}, \ldots, \lambda^{(t)}, b)$ and oracle access to c.*
- **Communication phase:** *$\mathcal{P}$ and $\mathcal{V}$ interact for t rounds with total communication $O(n \cdot d^{t+1} \cdot \log|\mathbb{F}|)$, where $\mathcal{V}$'s messages only depend on $\mathcal{V}$'s randomness.*
- **Query phase:** *At the end of the interaction, $\mathcal{V}$ makes d^t (non-adaptive) queries to c, reads the entire transcript, and applies a Boolean predicate ϕ to the values that it read (from c and the transcript) which instructs it whether to accepts or reject. We further require that the location of the queries only depends on $\mathcal{V}$'s randomness, and that the predicate ϕ only depends on the explicit input $(\lambda^{(1)}, \ldots, \lambda^{(t)}, b)$ and $\mathcal{V}$'s randomness.*
- **Completeness:** *If $\langle \lambda^{(1)} \otimes \cdots \otimes \lambda^{(t)}, x\rangle = b$, then when $\mathcal{V}$ interacts with $\mathcal{P}$, it accepts with probability 1.*
- **Soundness:** *If $\langle \lambda^{(1)} \otimes \cdots \otimes \lambda^{(t)}, x\rangle \neq b$, then for any prover strategy $\mathcal{P}^*$, when $\mathcal{V}$ interacts with $\mathcal{P}^*$, it accepts with probability at most ε.*

Moreover,

- *The verifier's running time is $d^{t+1} \cdot \mathrm{poly}(n, \log|\mathbb{F}|)$, given a parity-check matrix for C.*
- *The prover's running time is $n^t \cdot t \cdot d^t \cdot \mathrm{polylog}(\mathbb{F})$.*
- *If $\langle \lambda^{(1)} \otimes \cdots \otimes \lambda^{(t)}, x\rangle \neq b$, then for any prover strategy $\mathcal{P}^*$, when $\mathcal{V}$ interacts with $\mathcal{P}^*$, the transcript is $\Omega(d^{-t})$-close to an accepting view with probability at most ε.*

We prove Lemma 7.1 in Section 7.2. The next lemma from [Vid15] gives a local testing procedure for tensor codes.

Lemma 7.2 (Local testing of tensor codes, [Vid15], Theorem 3.1)**.** *Let $\{C^{(n)} : \{0,1\}^k \to \{0,1\}^n\}_{n\in\mathbb{N}}$ be a linear code of relative distance δ. Then for any integer $t \geq 3$ and $\varepsilon > 0$ there exists a randomized algorithm A satisfying the following properties.*

- **Input:** *A gets oracle access to a string $w \in \{0,1\}^{n^t}$ and a parameter $\alpha \in (0,1)$.*
- **Query complexity:** *A makes $n^2 \cdot \delta^{-O(t)} \cdot \alpha^{-1} \cdot \log(1/\varepsilon)$ (non-adaptive) queries to the oracle w.*
- **Completeness:** *If w is a codeword of $C^{\otimes t}$, then A accepts with probability 1.*
- **Soundness:** *If w is α-far $C^{\otimes t}$, then A accepts with probability at most ε.*

Moreover,

- *If w is α-far from $C^{\otimes t}$, then, in expectation, A's view is $(\delta^{O(t)} \cdot \alpha)$-far from an accepting view.*
- *The running time of A is $n^{O(1)} \cdot \delta^{-O(t)} \cdot \alpha^{-1} \cdot \log(1/\varepsilon)$, given a parity-check matrix for C.*

Remark 7.3. *We remark on the following differences from [Vid15, Theorem 3.1]:*

1. *Viderman [Vid15, Theorem 3.1] only states the rejection probability of the n-query test that chooses a random axis parallel line (according to some specified distribution) and checks that the projection of w to the line is a codeword of C. However, the proof shows that the n^2-query test that chooses a random axis parallel plane (according to some specified distribution) and checks that the projection of w to the plane is a codeword of $C^{\otimes 2}$ is robust, in the sense that the average view of the tester is $(\delta^{O(t)} \cdot \alpha)$-far from $C^{\otimes 2}$.*

   Finally, note that the above implies in turn that the tester rejects with probability at least $(\delta^{O(t)} \cdot \alpha)$, and the rejection probability can be amplified to $1-\varepsilon$ by repeating the tester $\delta^{-O(t)} \cdot \alpha^{-1} \cdot \log(1/\varepsilon)$ times and accepting if and only if all invocations accept. Note that this increases the query complexity and running time by a multiplicative factor of $\delta^{-O(t)} \cdot \alpha^{-1} \cdot \log(1/\varepsilon)$.

2. *Running time is not stated explicitly in [Vid15]. However, inspection shows that a random plane from the aforementioned distribution can be sampled in time $O(t \cdot \log n)$. Moreover, checking whether a given string is a codeword of $C^{\otimes 2}$ can be implemented in time $\mathrm{poly}(n)$ given a parity-check matrix for C.*

Finally, we use the following relaxed local correction procedure for tensor codes from [GRR18]. As [GRR18] (cf., [GRR17, Lemma 5.5]) only deals with the two-dimensional (i.e., $t = 2$) case, and since we also require an additional robustness property, we provide a full proof in Section 7.3.

Lemma 7.4 (Relaxed local correction of tensor codes)**.** *Let $C = \{C^{(n)} : \{0,1\}^k \to \{0,1\}^n\}_{n \in \mathbb{N}}$ be a linear code ensemble of relative distance δ. Then for any integer $t \geq 2$ and $\varepsilon > 0$ there exists a randomized algorithm A satisfying the following properties.*

- **Input:** *A takes as input a coordinate $i \in [n^t]$, and also gets oracle access to a string $w \in \{0,1\}^{n^t}$.*
- **Query complexity:** *A makes $n \cdot \delta^{-O(t)} \cdot \log(1/\varepsilon)$ (non-adaptive) queries to the oracle w.*
- **Completeness:** *If w is a codeword of $C^{\otimes t}$, then A outputs c_i with probability 1.*
- **Soundness:** *If w is $\left(\frac{\delta}{4}\right)^t$-close to a codeword $c \in C^{\otimes t}$, then, with probability at least $1-\varepsilon$, the output of A is either c_i or $\perp$.*

Moreover,

- *If w is $\left(\frac{\delta}{4}\right)^t$-close to a codeword c of $C^{\otimes t}$, then with probability at least $1-\varepsilon$, the view of A is $\delta^{-O(t)} \cdot \log^{-1}(1/\varepsilon)$-far from any view that would cause it to output $1-c_i$.*
- *The running time of A is $n^{O(1)} \cdot \delta^{-O(t)} \cdot \log(1/\varepsilon)$, given a parity-check matrix for C.*

Lastly, we shall also use the following local decomposability property (cf., Definition 4.4) of tensor codes. We prove the following lemma in Section 7.4.

Lemma 7.5 (Local decomposability of tensor codes)**.** *Let $C : \{0,1\}^k \to \{0,1\}^n$ be a linear code, and let $t > d \geq 1$. Then $C^{\otimes t}$ is locally k^{t-d}-decomposable with respect to the base code $C^{\otimes d}$. Moreover, if each coordinate of C can be encoded in time T, then the running time of the oracle machine A is at most $O(T \cdot k^{t-d-1})$.*

Section Organization. In Section 7.1 we show how to use Lemmas 7.1, 7.2, 7.4 and 7.5 to prove Lemma 4.7. Then, in Sections 7.2 to 7.4 we prove Lemmas 7.1, 7.4 and 7.5, respectively.

7.1 Proof of Lemma 4.7

Recall that in Section 4 we defined $\mathcal{I} = \{\bar{n} \mid n \in \mathbb{N}\}$, where $\bar{n} := \left(\lceil n^{1/(2t(n))} \rceil\right)^{2t(n)}$ and defined $S_{\otimes t}$ as the set of all t-dimensional rank 1 tensors. Namely, $\mathcal{S}_{\otimes t} = \left\{S^{(\bar{n})}_{\otimes t}\right\}_{\bar{n}\in\mathcal{I}}$ where

$$S^{(\bar{n})}_{\otimes t} = \left\{\lambda^{(1)} \otimes \cdots \otimes \lambda^{(t)} : \lambda^{(1)}, \ldots, \lambda^{(t)} \in (\mathbb{F}_{\bar{n}})^{\bar{n}^{1/t}}\right\}$$

.

Fix $\bar{n} \in \mathcal{I}$, and let $t = t(n)$, $\gamma = \gamma(n)$, $\mathbb{F} = \mathbb{F}_n$, and $S = S^{(n)}_{\otimes t}$. We proceed to describe the code $C = C_{\bar{n}}$.

The code C. Let R be the systematic binary linear code guaranteed by Theorem 2.3 of message length $\bar{n}^{1/(2t)}$ (noting that $\bar{n}^{1/(2t)}$ is an integer for any $\bar{n} \in \mathcal{I}$), rate at least $1 - \frac{\gamma}{2t}$, and relative distance $\left(\frac{\gamma}{t}\right)^{O(1)}$ (noting that $\frac{\gamma}{2t} \geq \frac{100 \log \bar{n}}{\bar{n}^{1/(2t)}}$ by our constraints on γ). Let $C = R^{\otimes 2t}$ be the $(2t)$-wise tensor product of R. Then, C has message length $\bar{n}$, rate at least $(1 - \frac{\gamma}{2t})^{2t} \geq 1 - \gamma$, and relative distance $\left(\frac{\gamma}{t}\right)^{O(t)}$. Since R is systematic, also the code C is systematic.

Let $d = d(n) \in [2t(n) - 1]$. By Lemma 7.5, C is locally $\bar{n}^{1-d/(2t)}$-decomposable with running time $\bar{n}^{1-d/(2t)} \cdot \mathrm{polylog}(n)$ with respect to the base code $R^{\otimes 2d}$ which has rate least $1-\gamma$. Moreover, $C_{\bar{n}}$ is encodable in time $\bar{n} \cdot \mathrm{polylog}(n)$.

Next we describe the IOPP $\tilde{\Pi}$ corresponding to C.

The IOPP $\tilde{\Pi}$. Let Π be the sumcheck protocol given by Lemma 7.1 for with respect to the base code $R^{\otimes 2}$, dimension t, and soundness error $\frac{1}{4}$ with $d = (t/\gamma)^{O(1)}$. Let A be the local tester given by Lemma 7.2 for the base code R, dimension $2t$, proximity parameter $\min\{\delta, \frac{\mathrm{dist}(C)}{4^t}\}$ and failure probability $1/2$. Lastly, let A' be the relaxed local corrector given by Lemma 7.4 for the base code $R^{\otimes 2}$, dimension t, and failure probability $\frac{1}{4d^t} = (\gamma/t)^{O(t)}$.

On explicit input $\lambda^{(1)}, \ldots, \lambda^{(t)} \in \mathbb{F}^{\bar{n}^{1/t}}$ and $b \in \mathbb{F}$, and implicit input $w \in \{0,1\}^{n'}$, $\mathcal{P}$ and $\mathcal{V}$ interact according to the protocol Π, where in the query phase, $\mathcal{V}$ first runs the local tester A on w, and then replaces each query of Π to c_i by applying the relaxed local corrector A' on input i with oracle access to w. If either A or A' reject in any of the invocations, then $\mathcal{V}$ rejects. Otherwise, $\mathcal{V}$ answers according to the protocol Π.

It can be readily verified that the communication complexity, query complexity, round complexity, verifier running time, and prover running time are all as claimed. Completeness is also straighfroward. Next we show soundness and robustness.

Soundness. For soundness, let $w \in \{0,1\}^{\bar{n}}$ be δ-far from any codeword $C(x)$ satisfying $\langle \lambda, x \rangle = b$,

Suppose first that w is $\min\{\delta, \frac{\mathrm{dist}(C)}{4^t}\}$-far from the code C. Then the local tester A rejects with probability at least $1/2$, and so we are done. Hence we may assume that w is $\min\{\delta, \frac{\mathrm{dist}(C)}{4^t}\}$-close to some codeword $C(x)$. By our assumption, this implies that $\langle \lambda, x \rangle \neq b$, where $\lambda = \lambda^{(1)} \otimes \cdots \otimes \lambda^{(t)}$.

With probability at least $3/4$, the relaxed local corrector either rejects (in which case $\mathcal{V}$ also rejects) or answer all d^t queries of Π with the corresponding values of $C(x)$. Conditioned on this, the protocol Π accepts with probability at most $\frac{1}{4}$, so we get an overall rejection probability of $\frac{1}{2}$.

Robustness. Let cc denote the transcript length of Π, and let q (resp., q') denote the total number of queries made by A (resp., A') during the protocol $\tilde{\Pi}$. In what follows we assume that $\mathrm{cc} = q = q'$ which can be achieved by duplicating the view of the verifier in any of the parts, if necessary, and accepting if and only if all duplicated views are consistent with each other and the original protocol accepts. Note that this increases the asymptotic query complexity by a multiplicative factor of $\log(|\mathbb{F}|)$, while maintaining the same asymptotic verifier running time.

Suppose next that w is δ-far from any codeword $C(x)$ satisfying $\langle \lambda, x \rangle = b$. Let us assume first that w is $\min\{\delta, \frac{\mathrm{dist}(C)}{4^t}\}$-far from any codeword of C. By the robustness of the local tester A, the average view of A is $(\gamma/t)^{O(t)} \cdot \delta$-far from an accepting view, and so with probability at least $(\gamma/t)^{O(t)} \cdot \delta$, A's view is $(\gamma/t)^{O(t)} \cdot \delta$-far from an accepting view. Since these account for $1/3$ of the queries of $\mathcal{V}$ in the protocol $\tilde{\Pi}$, we get that the queries in the protocol $\tilde{\Pi}$ will be $(\gamma/t)^{O(t)} \cdot \delta$-far from an accepting view with probability at least $(\gamma/t)^{O(t)} \cdot \delta$.

Hence, as in the soundness analysis, we may assume that w is $\frac{\mathrm{dist}(C)}{4^t}$-close to some codeword $C(x)$ with $\langle \lambda, x \rangle \neq b$. Let v' be the view of A' corresponding to all $d^t = (\gamma/t)^{O(t)}$ queries of the protocol Π. By the robustness of A', with probability at least $3/4$, any view that is $(\gamma/t)^{O(t)}$-close to v makes it either reject or output the correct values of $C(x)$ to all of the queries of Π. Assume that the foregoing event occurs, and note that conditioning on this event does not change the distribution of the transcript Π. Let v' be $(\gamma/t)^{O(t)}$-close to v.

Claim 7.6. *With probability* $3/4$ *over the randomness of* Π*, for every transcript* τ' *that is* $(\gamma/t)^{O(t)}$*-close to the transcript* τ *of* Π*, it holds that* $\mathcal{V}$ *rejects given the view* (v', τ')*.*

Proof. Since v' is $(\gamma/t)^{O(t)}$-close to v, given v' the relaxed local corrector A' either rejects or outputs the correct values of $C(x)$ to all queries of Π. If A' rejects then also $\mathcal{V}$ rejects and we are done. Thus, let us assume that A' outputs the correct values of $C(x)$ given the view v'. Since $\langle \lambda, x \rangle \neq b$, by the robustness of Π, with probability $\frac{3}{4}$ (over the coins of Π), the transcript of Π is $(\gamma/t)^{O(t)}$-far from making Π accept. Thus, with probability $\frac{3}{4}$, for every view τ' that is $(\gamma/t)^{O(t)}$-close to the transcript of Π, the verifier $\mathcal{V}$ rejects given the view (v', τ'). □

Since (by assumption) A''s view and the transcript of Π have equal weight, and constitute each $1/3$ of the queries of $\mathcal{V}$ in the protocol $\tilde{\Pi}$, using Claim 7.6, we have that with probability $(\gamma/t)^{O(t)} \cdot \delta$, any view that is $(\gamma/t)^{O(t)} \cdot \delta$-close to the actual view of $\mathcal{V}$ makes it reject.

7.2 Sumcheck for tensor codes – proof of Lemma 7.1

The proof of Lemma 7.1 is based on the ubiquitous sumcheck protocol [LFKN92], or rather its extension to tensor codes [Mei13]. Our construction differs in two ways from the standard sumcheck protocol. First, in contrast to the typical usage, here we do not merely compute a sum of the

coordinates of the message, but allow certain linear combinations (corresponding to a rank 1 tensor). Second, we allow the coefficient to lie in an extension field of the field over which our code is defined.

Fix a basis A for $\mathbb{F}$ over the binary field $\mathbb{F}_2$, where we view $\mathbb{F}$ as a $\log_2(|\mathbb{F}|)$ dimensional vector space over $\mathbb{F}_2$ in the natural way (e.g., A can be the standard basis). Every element in $\mathbb{F}$ can be expressed as a linear combination of the basis elements. Extending this fact to vectors, any $y \in \mathbb{F}^n$ can be expressed as $y = \sum_{a \in A} a \cdot y|_a$, where $y|_a \in (\mathbb{F}_2)^n$ for any $a \in A$.

For convenience, we will present the sumcheck protocol where the communication and query phases are interleaved, however it can be checked that the verifier queries and checks could be deferred to the end of the protocol.

Our description of the sumcheck protocol is recursive. On each call, $\mathcal{P}$ and $\mathcal{V}$ are given as input an iteration number $i \in [t+1]$, a string $r \in [n]^{i-1}$, and a value $\hat{b} \in \mathbb{F}$, where the protocol is initiated at the beginning to $i = 1$, $r = \perp$, and $\hat{b} = b$.

In iteration $i \neq t+1$, the prover sends a certain $\mathbb{F}$-linear combination of axis-parallel lines of the codeword c in direction i, where the linear combination is determined by $\lambda^{(1)}, \ldots, \lambda^{(t)}$ and the string r. The verifier then checks that the resulting linear combination $w \in \mathbb{F}^n$ is a codeword of C when projected on any basis element $a \in A$, and that $\langle \lambda^{(i)}, w \rangle = \hat{b}$. If any of the checks fail then $\mathcal{V}$ rejects. Otherwise, the verifier sends a uniform random index $\sigma \in [n]$, and $\mathcal{V}$ and $\mathcal{P}$ continue to iteration $i+1$ with $r = (r, \sigma) \in [n]^i$ and $\hat{b} = w_\sigma \in \mathbb{F}$ (this latter step is repeated d times to amplify the rejection probability). Finally, on iteration $i = t+1$, the verifier checks that the value of c on point $r \in [n]^t$ equals $\hat{b}$, and rejects otherwise. If all checks pass then the verifier eventually accepts.

The sumcheck protocol is given in Fig. 2.

Sumcheck$(i, r, \hat{b})$, where $i \in [t+1]$, $r \in [n]^{i-1}$ and $\hat{b} \in \mathbb{F}$ (initialized to $i \leftarrow 1$, $r \leftarrow \perp$, and $\hat{b} \leftarrow b$)

1. If $i = t+1$, $\mathcal{V}$ queries $c(r_1, \ldots, r_t)$ and checks that it equals $\hat{b}$; If not, $\mathcal{V}$ rejects and otherwise it accepts.
2. Otherwise (i.e., $i \in [t]$):
   (a) $\mathcal{P}$ sends $w \in \mathbb{F}^n$ whose ℓ-th coordinate equals
   $$w_\ell = \sum_{\ell_{i+1}, \ldots, \ell_t \in [k]} (\lambda^{(i+1)})_{\ell_{i+1}} \cdots (\lambda^{(t)})_{\ell_t} \cdot c(r_1, \ldots, r_{i-1}, \ell, \ell_{i+1}, \ldots, \ell_t),$$
   for every $\ell \in [n]$.
   (b) $\mathcal{V}$ checks that for any $a \in A$, $w|_a$ is a codeword of C; If not, $\mathcal{V}$ rejects.
   (c) $\mathcal{V}$ checks that $\sum_{\ell \in [k]} (\lambda^{(i)})_\ell \cdot w_\ell = \hat{b}$; If not, $\mathcal{V}$ rejects.
   (d) $\mathcal{V}$ sends uniform random independent $\sigma_1, \ldots, \sigma_d \in [n]$.
   (e) For $j = 1, \ldots, d$: $\mathcal{P}$ and $\mathcal{V}$ run **Sumcheck**$(i+1, (r, \sigma_j), w_{\sigma_j})$. Accept if all invocations accepted and otherwise reject.

Figure 2: Sumcheck for Tensor Codes

First note that we run the above recursive call with $i \neq t+1$ for $1 + d + \cdots + d^{t-1} \leq d^t$ times, and with $i = t+1$ for d^t times. For any call corresponding to $i \neq t+1$, the communication complexity is $n \cdot \log |\mathbb{F}| + d \cdot \log n$, verifier running time is $n^{O(1)} \cdot \text{polylog}(|\mathbb{F}|) + d \cdot \log n$, and prover running time

is at most $n^t \cdot t \cdot \text{polylog}(\mathbb{F})$, leading to a total communication complexity of $O(n \cdot d^{t+1} \cdot \log|\mathbb{F}|)$, total verifier running time $n^{O(1)} \cdot d^{t+1} \cdot \text{polylog}(|\mathbb{F}|)$, and total prover running time $n^t \cdot t \cdot d^t \cdot \text{polylog}(|\mathbb{F}|)$. The verifier queries a single bit of c on each of the calls corresponding to $i = t+1$, leading to total query complexity of d^t. Finally, for any $i \in [t]$, we invoke all calls corresponding to i in parallel, leading to t total rounds. Next we show completeness, soundness, and robustness.

For $i \in [t]$ and $r \in [n]^{i-1}$, let $w^{(i,r)} \in \mathbb{F}^n$ denote the message sent by the honest prover $\mathcal{P}$ on a call corresponding to i, r. That is,

$$(w^{(i,r)})_\ell = \sum_{\ell_{i+1},\ldots,\ell_t \in [k]} (\lambda^{(i+1)})_{\ell_{i+1}} \cdots (\lambda^{(t)})_{\ell_t} \cdot c(r_1, \ldots, r_{i-1}, \ell, \ell_{i+1}, \ldots, \ell_t)$$

for any $\ell = 1, \ldots, n$. Note that the definition extends naturally to $i = t+1$ and $r \in [n]^t$.

Then for any $i \in [t-1]$, $r \in [n]^{i-1}$, and $\sigma \in [n]$ we have that

$$\sum_{\ell \in [k]} (\lambda^{(i+1)})_\ell \cdot (w^{(i+1,(r,\sigma))})_\ell = \sum_{\ell_{i+1},\ldots,\ell_t \in [k]} (\lambda^{(i+1)})_{\ell_{i+1}} \cdots (\lambda^{(t)})_{\ell_t} \cdot c(r_1, \ldots, r_{i-1}, \sigma, \ell_{i+1}, \ldots, \ell_t) = (w^{(i,r)})_\sigma.$$

Moreover, if we let $\lambda^{(t+1)} := (1, 0^{k-1}) \in \{0,1\}^k$, then for any $r \in [n]^{t-1}$ and $\sigma \in [n]$ we have that

$$\sum_{\ell \in [k]} (\lambda^{(t+1)})_\ell \cdot (w^{(t+1,(r,\sigma))})_\ell = c(r_1, \ldots, r_{t-1}, \sigma) = (w^{(t,r)})_\sigma.$$

We conclude that

$$\sum_{\ell \in [k]} (\lambda^{(i+1)})_\ell \cdot (w^{(i+1,(r,\sigma))})_\ell = (w^{(i,r)})_\sigma \tag{5}$$

for any $i \in [t]$, $r \in [n]^{i-1}$, and $\sigma \in [n]$.

Completeness. Suppose that $\langle \lambda^{(1)} \otimes \cdots \otimes \lambda^{(t)}, x \rangle = b$. On each call corresponding to $i \in [t]$ and $r \in [n]^{i-1}$ we have that $w^{(i,r)}$ is a linear combination of codewords in C with coefficients in $\mathbb{F}$. Thus for any $a \in A$ we have that $w^{(i,r)}|_a$ is a linear combination of codewords of C with coefficient in $\mathbb{F}_2$, which by linearity is a codeword of C. So $w^{(r,i)}$ will pass the test in Step 2b.

Next observe that since the code C is systematic, the assumption that $\langle \lambda^{(1)} \otimes \cdots \otimes \lambda^{(t)}, x \rangle = b$ implies that $\sum_{\ell_1,\ldots,\ell_t \in [k]} (\lambda^{(1)})_{\ell_1} \cdots (\lambda^{(t)})_{\ell_t} \cdot c(\ell_1, \ldots, \ell_t) = b$, which is equivalent in turn to $\sum_{\ell \in [k]} (\lambda^{(1)})_\ell \cdot (w^{(1,\perp)})_\ell = b$. Moreover, for any call corresponding to $i \in \{2, \ldots, t\}$, $r \in [n]^{i-1}$, and the particular value of $\hat{b} \in \mathbb{F}$ in the i-th iteration, we have that $\sum_{\ell \in [k]} (\lambda^{(i)})_\ell \cdot (w^{(i,r)})_\ell = \hat{b}$ by Eq. (5). So for any $i \in [t]$, $w^{(i,r)}$ will pass the test in Step 2c. Finally, note that for $i = t+1$ we have that $(w^{(t,r)})_\sigma = \sum_{\ell \in [k]} (\lambda^{(t+1)})_\ell \cdot (w^{(t+1,r)})_\ell = c(r_1, \ldots, r_t)$, and so $\mathcal{V}$ will not reject in Step 1.

We conclude that all tests will pass with probability 1, and so $\mathcal{V}$ accepts with probability 1.

Soundness. Suppose that $\langle \lambda^{(1)} \otimes \cdots \otimes \lambda^{(t)}, x \rangle \neq b$, and let $\mathcal{P}^*$ be a prover strategy. We may assume that on each call corresponding to $i \in [t]$, $r \in [n]^{i-1}$, and $\hat{b} \in \mathbb{F}$, the cheating prover $\mathcal{P}^*$ sends $y \in \mathbb{F}^n$ such that $y|_a$ is a codeword of C for any $a \in A$, and $\sum_{\ell \in [k]} (\lambda^{(i)})_\ell \cdot y_\ell = \hat{b}$, since otherwise $\mathcal{V}$ rejects.

Claim 7.7. *Let $y \in \mathbb{F}^n$ be a message sent by $\mathcal{P}^*$ on some call corresponding to $i \in [t]$, $r \in [n]^{i-1}$, and $\hat{b} \in \mathbb{F}$. Suppose that $\sum_{\ell \in [k]} (\lambda^{(i)})_\ell \cdot (w^{(i,r)})_\ell \neq \hat{b}$. Then with probability at least δ over the choice of a uniform random $\sigma \in [n]$ it holds that $\sum_{\ell \in [k]} (\lambda^{(i+1)})_\ell \cdot (w^{(i+1,(r,\sigma))})_\ell \neq y_\sigma$.*

Proof. By assumption we have that $\sum_{\ell \in [k]} (\lambda^{(i)})_\ell \cdot y_\ell = b$ while $\sum_{\ell \in [k]} (\lambda^{(i)})_\ell \cdot (w^{(i,r)})_\ell \neq \hat{b}$, and so $y \neq w^{(i,r)}$. In particular, there exists a basis element $a \in A$ for which $y|_a \neq w^{(i,r)}|_a$. Moreover, by assumption both $y|_a$ and $w^{(i,r)}|_a$ are codewords of C, and so $\mathrm{dist}(y|_a, w^{(i,r)}|_a) \geq \delta$ which implies in turn that $\mathrm{dist}(y, w^{(i,r)}) \geq \delta$. Thus, with probability at least δ over the choice of a uniform random $\sigma \in [n]$ it holds that $y_\sigma \neq w^{(i,r)}_\sigma$. By Eq. (5) this implies in turn that $\sum_{\ell \in [k]} (\lambda^{(i+1)})_\ell \cdot w^{(i+1,(r,\sigma))} = (w^{(i,r)})_\sigma \neq y_\sigma$. □

Next observe that since the code C is systematic, the assumption that $\langle \lambda^{(1)} \otimes \cdots \otimes \lambda^{(t)}, x \rangle \neq b$ is equivalent to $\sum_{\ell_1,\ldots,\ell_t \in [k]} (\lambda^{(1)})_{\ell_1} \cdots (\lambda^{(t)})_{\ell_t} \cdot c(\ell_1, \ldots, \ell_t) \neq b$, which is equivalent in turn to $\sum_{\ell \in [k]} (\lambda^{(1)})_\ell \cdot (w^{(1,\perp)})_\ell \neq b$. By Claim 7.7, this implies in turn that with probability at least $1 - (1-\delta)^d \geq 1 - \frac{\varepsilon}{t}$ there will be a call corresponding to $i = 2$, $r \in [n]$, and $\hat{b} \in \mathbb{F}$ for which $\sum_{\ell \in [k]} (\lambda^{(2)})_\ell \cdot (w^{(2,r)})_\ell \neq \hat{b}$. Continuing this way, by the union bound, there will be a call corresponding to $i = t+1$, $r \in [n]^t$, and $\hat{b} \in \mathbb{F}$ for which $\sum_{\ell \in [k]} (\lambda^{(t+1)})_\ell \cdot (w^{(t+1,r)})_\ell \neq \hat{b}$. Finally, note that the left hand side of this inequality equals $c(r_1, \ldots, r_t)$, and consequently in this case the verifier will reject at Step 1.

Robustness. We show robustness we slightly modify the protocol as follows. In Step 2a the prover sends an encoding of w (viewed as a bit string of length $n \cdot \log |\mathbb{F}|$) via an explicit linear code C' of constant rate and constant relative distance δ' (see, e.g., Theorem 2.3). Upon receiving a message z from the verifier, the prover first checks that z is a codeword of C'. If not, it rejects. Otherwise, it decodes z, and proceeds to Step 2b with the decoded message. Note that this only increases the communication complexity by a constant factor and the verification time by a poly-logarithmic factor.

Next suppose that $\langle \lambda^{(1)} \otimes \cdots \otimes \lambda^{(t)}, x \rangle \neq b$, and let $\mathcal{P}^*$ be a prover strategy. We may assume that on each call corresponding to $i \in [t]$, $r \in [n]^{i-1}$, and $\hat{b} \in \mathbb{F}$, the cheating prover $\mathcal{P}^*$ sends $z \in \mathbb{F}^n$ that is $\frac{\delta'}{2}$-close to some codeword $C'(y')$. Indeed, otherwise, any z' that is $\frac{\delta'}{2}$-close to z will cause $\mathcal{V}$ to reject, and since the total number of calls corresponding to $i \neq t+1$ is at most d^t, this will imply in turn that the transcript is at least $\frac{\delta'}{2 \cdot d^t}$-far from an accepting view.

Next we note that under the above assumption any z' that is $\frac{\delta'}{2}$-close to z is either not a codeword of C' or equals $C'(y')$. Indeed, suppose in contradiction that $z' = C'(y'')$ for some $y'' \neq y'$. Then by the triangle inequality $z = C'(y')$ is δ'-close to $z' = C'(y'')$, which by distance properties of C' implies in turn that $y'' = y'$.

We conclude that there is only a single transcript τ' that is $\frac{\delta'}{2 \cdot d^t}$-close to the original transcript τ and contains only valid encodings of C'. Finally, note that any transcript that contains a non-valid encoding of C' will cause the verifier to reject. Moreover, by the soundness property established above, with probability at least $1 - \varepsilon$ the transcript τ' will cause the verifier to reject.

7.3 Relaxed local correction of tensor codes – proof of Lemma 7.4

Lemma 7.4 can be deduced from the following lemma by induction.

Lemma 7.8. *Suppose that $C : \{0,1\}^k \to \{0,1\}^n$ and $C' : \{0,1\}^{k'} \to \{0,1\}^{n'}$ are relaxed locally correctable linear codes (see Definition 2.6) with query complexity q, q', decoding radius δ_R, δ'_R, and failure probability ε, $\frac{\delta_R}{4}$, respectively. Then the tensor product $C \otimes C'$ is relaxed locally correctable with query complexity $O(q' \cdot \log(1/\varepsilon)/\delta_R + q)$, decoding radius $\frac{\delta_R \cdot \delta'_R}{2}$, and failure probability ε.*

Moreover,

- *If the relaxed local corrector for C, C' are (α, ε), $(\alpha', \frac{\delta_R}{4})$-robust, respectively, then the relaxed local corrector for $C \otimes C'$ is $(\tilde{\alpha}, \varepsilon)$-robust for*

$$\tilde{\alpha} = \min\left\{\frac{\alpha}{2},\ \Omega\left(\frac{\alpha' \cdot \delta_R}{\log(1/\varepsilon)}\right)\right\}.$$

- *If the relaxed local corrector for C, C' have running times T, T', respectively, then the relaxed local corrector for $C \otimes C'$ has running time $O(T' \cdot \log(1/\varepsilon)/\delta_R + T)$.*

Before we prove the above lemma we show how it implies Lemma 7.4.

Proof of Lemma 7.4. Apply Lemma 7.8 iteratively for $i = 1, \ldots, t-1$ with C' being the code $C^{\otimes i}$ constructed so far, and C being a new copy of C, noting that C is trivially relaxed locally correctable with query complexity n, decoding radius $\frac{\delta}{2}$, and failure probability 0 using the relaxed local corrector that reads the whole string w and outputs w_i if w is a codeword of C and $\perp$ otherwise.

Then on each step the decoding radius decreases by a multiplicative factor of $\delta/4$. Moreover, one each step other than the last one we may obtain failure probability $\delta/8$ by increasing the query complexity by a multiplicative factor of $O(\log(1/\delta)/\delta)$, while on the last step we may obtain failure probability ε by increasing the query complexity by a multiplicative factor of $O(\log(1/\varepsilon)/\delta)$. This leads to the claimed query complexity, decoding radius, and failure probability.

As to robustness, note first that the trivial relaxed local corrector for C is $(\delta/4, 0)$-robust since if w is $\delta/2$-close to a codeword c of C, then any w' that is $\delta/4$-close to w is either a non-codeword or equals to c. Next observe that in each step other than the last one the robustness is multiplied by a factor of $\Omega(\delta \cdot \log^{-1}(1/\delta))$, while in the last step it is multiplied by a factor of $\Omega(\delta \cdot \log^{-1}(1/\varepsilon))$, leading to the claimed robustness. Finally, note that the trivial relaxed local corrector for C can be implemented in time $n^{O(1)}$ given a parity-check matrix for C, and so we also obtain a total running time of $n^{O(1)} \cdot \delta^{-O(t)} \cdot \log(1/\varepsilon)$. □

We proceed to the proof of Lemma 7.8.

Proof of Lemma 7.8. The relaxed local corrector $\tilde{A}$ for $C \otimes C'$ is given in Fig. 3 below.

It can be verified that the query complexity and running time are as claimed. completeness is also clear. Next we show soundness and robustness.

Soundness. Suppose that $w \in \{0,1\}^{n \times n'}$ is $\frac{\delta_R \cdot \delta'_R}{2}$-close to a codeword $c \in \{0,1\}^{n \times n'}$ of $C \otimes C'$.

If the j-th column of w is δ_R-close to the j-th column of c, then in Step 2 the relaxed local corrector A will output either $c_{i,j}$ or $\perp$ with probability at least $1 - \varepsilon$. Since the only possible output in Step 1 is $\perp$, in this case $\tilde{A}$ will also output either $c_{i,j}$ or $\perp$ with probability at least $1 - \varepsilon$. Hence we may assume that the j-th column of w is δ_R-far from the j-th column of c.

Next observe that by averaging, for at least $(1 - \frac{\delta_R}{2})$-fraction of the rows i' it holds that the i'-th row of w is δ'_R-close to the i'-th row of c. Moreover, by assumption that the j-th column of w is

The composed corrector $\tilde{\mathbf{A}}$

$\triangleright$ $\tilde{A}$ receives oracle access to a matrix $w \in \{0,1\}^{n \times n'}$ and as explicit input $i \in [n]$ and $j \in [n']$.

1. Repeat for $O(\log(1/\varepsilon)/\delta_R)$ times:
   - Pick a random row $i' \in [n]$.
   - Run the relaxed local corrector A' for C' on input j with oracle access to the i'-row $w|_{\{i'\} \times [n']}$. If the result is not equal to $w_{i',j}$ then output $\perp$ and abort.
2. Run the relaxed local corrector A for C on input i with oracle access to the j-th column $w|_{[n] \times \{j\}}$, and return the value output by A.

Figure 3: Relaxed Local Corrector for $C \otimes C'$

δ_R-far from the j-th column of c, we also have that $w_{i',j} \neq c_{i',j}$ for at least δ_R-fraction of the indices $i' \in [n]$. We conclude that for at least $\frac{\delta_R}{2}$-fraction of the indices $i' \in [n]$, it holds that *both* the i'-th row of w is δ'_R-close to the i'-th row of c, and $w_{i',j} \neq c_{i',j}$. Let $I \subseteq [n]$ denote the subset of these indices, and note that $|I| \geq \frac{\delta_R}{2} \cdot n$.

Now $\tilde{A}$ picks $i' \in I$ in Step 1 with probability at least $\frac{\delta_R}{2}$, and conditioned on this, the relaxed local corrector A' will output in Step 1 either $\perp$ or $c_{i',j} \neq w_{i',j}$ with probability at least $1 - \frac{\delta_R}{4}$, which will cause $\tilde{A}$ to output $\perp$. Finally, repeating this process for $O(\log(1/\varepsilon)/\delta_R)$ times, we obtain that $\tilde{A}$ will output $\perp$ with probability at least $1 - \varepsilon$.

Robustness. In what follows assume that in Step 1 the relaxed local corrector $\tilde{A}$ queries $w_{i',j}$ for q' times and outputs $\perp$ if the duplicated views are non-identical. Additionally, assume that total number of queries in Step 1 equals number of queries in Step 2 which can be achieved by duplicating the views and outputting $\perp$ whenever the views are inconsistent with each other. Note that this does not change the asymptotic query complexity and running time.

Next suppose that $w \in \{0,1\}^{n \times n'}$ is $\frac{\delta_R \cdot \delta'_R}{2}$-close to a codeword $c \in \{0,1\}^{n \times n'}$ of $C \otimes C'$, and let v be $\tilde{A}$'s view. We will show that with probability at least $1 - \varepsilon$ any view v' that is obtained from v by changing at most $\tilde{\alpha}$-fraction of coordinates outputs either $c_{i,j}$ or $\perp$.

As in the soundness analysis, suppose first that the j-th column of w is δ_R-close to the j-th column of c. Then by robustness of A, with probability at least $1 - \varepsilon$, any view that is α-close to A's view in Step 2 will output either $c_{i,j}$ or $\perp$. Since the only possible output in Step 1 is $\perp$, by our assumption that the queries in Step 2 constitute half the queries of $\tilde{A}$, we conclude that with probability at least $1 - \varepsilon$, any view that is obtained from v by changing at most $\frac{\alpha}{2}$-fraction of the coordinates will cause $\tilde{A}$ to output either $c_{i,j}$ or $\perp$.

Hence we may assume that the j-th column of w is δ_R-far from the j-th column of c. As in the soundness analysis, this implies the existence of a subset $I \subseteq [n]$, $|I| \geq \frac{\delta_R}{2} \cdot n$, such that for any $i' \in I$ it holds that the i'-th row of w is δ'_R-close to the i'-th row of c, and additionally, $w_{i',j} \neq c_{i',j}$. So $\tilde{A}$ picks $i' \in I$ in Step 1 with probability at least $\frac{\delta_R}{2}$.

Conditioned on the above, with probability at least $1 - \frac{\delta_R}{4}$, any view that is α'-close to A''s view in Step 1 will output either $\perp$ or $c_{i',j} \neq w_{i',j}$. Recalling that we duplicated $w_{i',j}$ for q' times, we conclude that with probability at least $\frac{\delta_R}{4}$, any view that is $\frac{\alpha'}{2}$-close to the view on Step 1 will caluse

$\tilde{A}$ to output $\perp$. Finally, repeating this process for $O(\log(1/\varepsilon)/\delta_R)$ times, and recalling that the total number of queries in Step 1 account for half of the queries of $\tilde{A}$, we obtain that with probability at least $1-\varepsilon$, any view that is obtained from v by changing at most $\Omega(\alpha' \cdot \delta_R \cdot \log^{-1}(1/\varepsilon))$-fraction of the coordinates will cause $\tilde{A}$ to output $\perp$. □

7.4 Local decomposability for tensor codes – proof of Lemma 7.5

Observe that $C^{\otimes t} = R \otimes R'$, where $R := C^{\otimes d}$ and $R' := C^{\otimes(t-d)}$. Thus, we can view a codeword $c \in \{0,1\}^{n^t}$ of $C^{\otimes t}$ as an $n^d \times n^{t-d}$ binary matrix whose columns belong to R and whose rows belong to R'. We may also view a message $x \in \{0,1\}^{k^t}$ as an $k^d \times k^{t-d}$ binary matrix.

For $r \in [k^{t-d}]$, we let $x^{(r)} \in \{0,1\}^{k^d}$ denote the r-th column of x, and let $y^{(r)} := R(x^{(r)})$. Suppose we wish to encode a coordinate $(i,j) \in [n^d] \times [n^{t-d}]$ of $C^{\otimes t} = R \otimes R'$. Then we have that the (i,j) coordinate of $R \otimes R'$ equals the j-th coordinate of $R'\left((y^{(1)})_i, \ldots, (y^{(k^d)})_i\right)$, and so this coordinate can be retrieved by making at most 1 query to each $y^{(r)} = C^{\otimes d}(x^{(r)})$, for every $r \in [k^{t-d}]$.

Finally, note that the running time of the oracle machine is the time required for decoding a single coordinate of $R' = C^{\otimes t-d}$ which is at most $O(T \cdot k^{t-d-1})$ by the claim below.

Claim 7.9. *Suppose that each coordinate of C is encodable in time T. Then each coordinate of $C^{\otimes t}$ is encodable in time $O(T \cdot k^{t-1})$.*

Proof. We prove the claim by induction on t. The claim clearly holds when $t = 1$. For $t > 1$, the above proof with $d = t-1$ shows that any coordinate of $C^{\otimes t}$ can be computed by computing at most k coordinates of $C^{\otimes t-1}$ and a single coordinate of C. Denoting the encoding time for dimension i by E_i we have that

$$E_i = k \cdot E_{i-1} + T = \cdots = k^j \cdot E_{i-j} + T \cdot \left(1 + k + k^2 + \cdots + k^{j-1}\right) = T \cdot \left(k^{i-1} + \frac{k^{i-1}-1}{k-1}\right)$$

Thus, $E_t = O(T \cdot k^{t-1})$ and the claim follows. □

8 IOPP for bounded space computation – proof of Theorem 3.5

In this section, building on results established in the previous sections, we construct an IOPP for bounded space languages, thereby proving Theorem 3.5. Actually, we will prove a more general (and somewhat technical) lemma, from which Lemma 8.1 immediately follows. The technical details in Lemma 8.1 will be useful for us in Section 9 when we construct a (short) IOP for NP.

Lemma 8.1. *Let $\mathcal{L}$ be a language computable in time $\mathrm{poly}(n)$ with space $s = s(n) \geq \log n$. Then for every $\beta = \beta(n) \in (0,1)$ and $\gamma = \gamma(n) \in (0,1)$ such that $\mathrm{poly}(1/\beta) \leq \log(n)$ and $\gamma = \gamma(n) \geq \frac{200 \cdot 4^{1/\beta} \cdot \log n}{n^{\beta/2}}$ the following holds.*

There exists an IOPP for $\mathcal{L}$ with respect to proximity parameter $\delta > 0$ with communication complexity $\gamma \cdot n + \mathrm{poly}\left(n^\beta, (\gamma\beta)^{-1/\beta}, 1/\delta, s\right)$, query complexity $\mathrm{poly}\left((\gamma\beta)^{-1/\beta}, 1/\delta\right)$, round complexity $\beta^{-O(1/\beta)}$, and soundness error $1/2$.

Moreover,

- *The verifier's running time is $\mathrm{poly}\left(n^\beta, (\gamma\beta)^{-1/\beta}, 1/\delta, s\right)$.*

- *The prover's running time is* $\mathrm{poly}(n)$.
- *There exists a systematic code* $C = \{C_n : \{0,1\}^n \to \{0,1\}^{n'}\}_{n\in\mathbb{N}}$ *of rate at least* $\frac{1}{1+\gamma}$ *such that the communication phase consists of a single prover's message of length* $\gamma \cdot n$ *which is the non-systematic part of* C_n*, followed by a two-way communication of length* $\mathrm{poly}\left(n^\beta, (\gamma\beta)^{-1/\beta}, 1/\delta, s\right)$.

  For any integer $d \le 2 \cdot \lceil 1/\beta \rceil$*, the code* C_n *is locally* $(\bar{n})^{1-2\beta d}$*-decomposable with respect to a base code* C_0 *of rate at least* $1-\gamma$ *that is encodable in quasi-linear time.*

We prove Lemma 8.1 in two stages. First, using the results established in Sections 4 to 7, we construct a short *robust* IOPP with sublinear query complexity. The next step is a composition procedure for IOPPs that combines an outer IOPP of small communication complexity but relatively large query complexity (such as the one that we constructed in the first stage) with an inner IOPP of small query complexity but relatively large communication complexity (here we can use an "off-the-shelf" PCPP or IOPP) to obtain the best of both worlds: an IOPP with small communication and query complexities. In particular, the following lemma extends the composition lemma from [BCG+17] which combines an outer (robust) PCPP with an inner IOPP.

Lemma 8.2 (IOPP query reduction)**.** *Suppose that the following exist:*

- **(Outer IOPP:)** *An* IOPP *for a language* $\mathcal{L}$ *wrt proximity parameter* $\delta > 0$*, with communication complexity* cc*, query complexity* q*, round complexity* ℓ*,* (α, ε)*-robustness, verifier randomness complexity* r*, and predicate set* Φ*.*
- **(Inner IOPP:)** *An* IOPP *for the language* $\mathcal{L}_\Phi := \{(\phi, w) \mid \phi \in \Phi,\ \phi(w) = \texttt{accept}\}$ *(where* ϕ *is an explicit input and* w *is the implicit input) wrt proximity parameter* δ'*, with communication complexity* cc′*, query complexity* q'*, round complexity* ℓ'*, and soundness error* ε'*, where* $\delta'(q(n)) = \alpha(n)$*.*

Then, there exists an IOPP *for* $\mathcal{L}$ *wrt proximity parameter* $\delta > 0$ *with communication complexity* $\mathrm{cc}(n) + \mathrm{cc}'(q(n)) + r(n)$*, query complexity* $q'(q(n))$*, round complexity* $\ell(n) + \ell'(q(n))$ *and soundness error* $\varepsilon(n) + \varepsilon'(q(n))$*.*

Moreover,

- If the verifiers in the outer and inner IOPPs have running times T and T', respectively, then the verifier in the resulting IOPP has running time $T(n) + T'(q(n))$.
- If the provers in the outer and inner IOPPs have running times T and T', respectively, then the prover in the resulting IOPP has running time $T(n) + T'(q(n))$.

We prove Lemma 8.2 in Section 8.1. As the outer IOPP we use the IOPP given by the combination of Lemmas 4.5 to 4.7.

Lemma 8.3 (Outer IOPP)**.** *Let* $\mathcal{L}$ *be a language computable in time* $\mathrm{poly}(n)$ *and space* $s = s(n) \ge \log n$*. Then there exists an absolute constant* $\varepsilon_0 > 0$ *such that the following holds for any* $\delta = \delta(n) > 0$*,* $\beta = \beta(n) \in (\frac{1}{(\log n)^{\varepsilon_0}}, \frac{1}{2})$*, and* $\gamma = \gamma(n) \in (\frac{200 \cdot 4^{1/\beta} \cdot \log n}{n^{\beta/2}}, 1)$*.*

There exists an IOPP *for* $\mathcal{L}$ *of communication complexity* $\gamma \cdot n + \mathrm{poly}\left(n^\beta, (\gamma\beta)^{-1/\beta}, 1/\delta, s\right)$*, query complexity and verifier running time* $\mathrm{poly}\left(n^\beta, (\gamma\beta)^{-1/\beta}, 1/\delta, s\right)$*, round complexity* $\beta^{-O(1/\beta)}$*, proximity parameter* δ*, and* $\left((\gamma\beta)^{O(1/\beta)} \cdot \delta, 1 - (\gamma\beta)^{O(1/\beta)} \cdot \delta\right)$*-robustness.*

Moreover,

- *Prover's running time is* $\mathrm{poly}(n)$.
- *There exists a systematic code* $\mathcal{C} := \{C_n : \{0,1\}^n \to \{0,1\}^{n'}\}_{n\in\mathbb{N}}$ *of rate at least* $\frac{1}{1+\gamma}$ *such that the communication phase consists of a single prover's message of length* $\gamma \cdot n$ *which is the non-systematic part of* C_n*, followed by a two-way communication of length* $\mathrm{poly}\left(n^{\beta}, (\gamma\beta)^{-1/\beta}, 1/\delta, s\right)$.
- *For any integer* $d \leq 2 \cdot \lceil 1/\beta \rceil$*, the code* C_n *is locally* $(\bar{n})^{1-2\beta d}$*-decomposable with running time* $n^{1-2\beta d} \cdot \mathrm{polylog}(n)$ *with respect to a base code* C_0 *of rate at least* $1-\gamma$.

We prove Lemma 8.3 in Section 8.2.

As the inner IOPP we use an off the shelf PCPP (recall that a PCPP is simply a 1-round IOPP). For convenience we use the PCPP of Mie [Mie09, Theorem 1] although we note that slightly less efficient PCPPs, such as those constructed in [BGH+05] suffice for our purposes.

Theorem 8.4 (Inner IOPP)**.** *Let* $\mathcal{L}$ *be a pair language computable in time* $\mathrm{poly}(n)$*. Then for any* $\delta = \delta(n) > 0$ *and* $\varepsilon = \varepsilon(n) > 0$ *there exists an* IOPP *for* $\mathcal{L}$ *wrt proximity parameter* $\delta > 0$*, with communication complexity* $\mathrm{poly}(n)$*, query complexity* $O(\log(1/\varepsilon) \cdot \log(1/\delta)/\delta)$*, round complexity* 1 *and soundness error* ε*.*

Moreover, the prover's running time is $\mathrm{poly}(n)$*, and the verifier's running time is* $\mathrm{poly}(n) \cdot \log(1/\varepsilon) \cdot \log(1/\delta)/\delta$*.*

Remark 8.5. *We remark that [Mie09, Theorem 1] does not explicitly state the prover's running time but it can be readily verified that the prover can be implemented in polynomial-time. We also note that follow up works obtain prover running time that is* quasi-linear *(rather than merely polynomial) in the original computation [BCGT13].*

The main result of [Mie09] only talks about constant proximity parameter and soundness error. In Section 8.3 we show how to extend this result to arbitrary $\varepsilon = \varepsilon(n)$ and $\delta = \delta(n)$. Next we prove Theorem 3.5, based on Lemmas 8.2 and 8.3 and Theorem 8.4.

Proof of Lemma 8.1. By Lemma 8.3, there exists an IOPP for $\mathcal{L}$ with communication complexity $\gamma \cdot n + n^{\beta} \cdot (\gamma\beta)^{-O(1/\beta)} \cdot \frac{1}{\delta} \cdot s^2$, query complexity $n^{\beta} \cdot (\gamma\beta)^{-O(1/\beta)} \cdot \frac{1}{\delta} \cdot s^2$, round complexity $\beta^{-O(1/\beta)}$, proximity parameter δ, verifier running time $n^{O(\beta)} \cdot (\gamma\beta)^{-O(1/\beta)} \cdot \frac{1}{\delta} \cdot s^2$, and $\left((\gamma\beta)^{O(1/\beta)} \cdot \delta, 1 - (\gamma\beta)^{O(1/\beta)} \cdot \delta\right)$-robustness. Moreover, the communication phase consists of a single prover message which is the non-systematic part $C_{\bar{n}}(w, 0^{\bar{n}-n})$ of length $\gamma \cdot n$, followed by a two-way communication of length $n^{\beta} \cdot (\gamma\beta)^{-O(1/\beta)} \cdot \frac{1}{\delta} \cdot s^2$.

Let Φ be the predicate set for the above outer IOPP, let $\mathcal{L}_{\Phi} := \{(\phi, w) \mid \phi \in \Phi,\ \phi(w) = \texttt{accept}\}$, and note that $\mathcal{L}_{\Phi}$ is computable in time $n^{O(\beta)} \cdot (\gamma\beta)^{-O(1/\beta)} \cdot \frac{1}{\delta} \cdot s^2$. By Theorem 8.4, there exists an IOPP for $\mathcal{L}_{\Phi}$ of communication complexity $n^{O(\beta)} \cdot (\gamma\beta)^{-O(1/\beta)} \cdot \delta^{-O(1)} \cdot s^{O(1)}$, query complexity $(\gamma\beta)^{-O(1/\beta)} \cdot \delta^{-O(1)}$, round complexity 1, proximity $(\gamma\beta)^{O(1/\beta)} \cdot \delta$, and soundness error $(\gamma\beta)^{O(1/\beta)} \cdot \delta$.

Lemma 8.2 then implies the existence of an IOPP for $\mathcal{L}$ of communication complexity $\gamma n + n^{O(\beta)} \cdot (\gamma\beta)^{-O(1/\beta)} \cdot \delta^{-O(1)} \cdot s^{O(1)}$, query complexity $(\gamma\beta)^{-O(1/\beta)} \cdot \delta^{-O(1)}$, round complexity $(\gamma\beta)^{-O(1/\beta)} \cdot \frac{1}{\delta}$, proximity parameter δ, and soundness error $1 - (\gamma\beta)^{O(1/\beta)} \cdot \delta$. Moreover, the communication phase consists of a single prover message which is the non-systematic part $C_{\bar{n}}(w, 0^{\bar{n}-n})$ of length $\gamma \cdot n$, followed by a two-way communication of length $n^{O(\beta)} \cdot (\gamma\beta)^{-O(1/\beta)} \cdot \delta^{-O(1)} \cdot s^{O(1)}$.

Finally, one can decrease the soundness error to $1/2$ by repeating the protocol for $(\gamma\beta)^{-O(1/\beta)} \cdot \frac{1}{\delta}$ times, and accepting if and only if all invocations accept. Note that since the first prover message is deterministic, one does not need to repeat this message. We conclude that there exists an IOPP for $\mathcal{L}$ of communication complexity $\gamma \cdot n + n^{O(\beta)} \cdot (\gamma\beta)^{-O(1/\beta)} \cdot \delta^{-O(1)} \cdot s^{O(1)}$, query complexity $(\gamma\beta)^{-O(1/\beta)} \cdot \delta^{-O(1)}$, round complexity $(\gamma\beta)^{-O(1/\beta)} \cdot \delta^{-O(1)}$, proximity parameter δ, and soundness error $1/2$. Moreover, the communication phase consists of a single prover message which is the non-systematic part of $C_{\bar{n}}(w, 0^{\bar{n}-n})$ of length $\gamma \cdot n$, followed by a two-way communication of length $n^{O(\beta)} \cdot (\gamma\beta)^{-O(1/\beta)} \cdot \delta^{-O(1)} \cdot s^{O(1)}$.

□

8.1 Query reduction – proof of Lemma 8.2

The composed protocol $\tilde{\Pi}$ is given in Fig. 4

The composed protocol $\tilde{\Pi}$:

Explicit input: x

Implicit input: $w \in \{0,1\}^n$

Communication phase:

1. $\mathcal{P}$ and $\mathcal{V}$ interact for $\ell(n)$ rounds with total communication $\mathrm{cc}(n)$ according to the outer IOPP Π for $\mathcal{L}$ on explicit input x and implicit input w.
2. $\mathcal{V}$ sends its randomness string of length $r(n)$, and $\mathcal{P}$ and $\mathcal{V}$ compute the corresponding query set I of size $q(n)$ and predicate ϕ.
3. $\mathcal{P}$ and $\mathcal{V}$ interact for $\ell'(q(n))$ rounds with total communication $\mathrm{cc}'(q(n))$ according to the promised inner IOPP Π' for $\mathcal{L}_\Phi$ on explicit input ϕ, viewing the restriction of w and the transcript of Π to the query set I as the implicit input to Π'.

Query phase: $\mathcal{V}$ makes $q'(q(n))$ queries to the transcripts of Π' and Π and to the implicit input w according to the protocol Π' and accepts if and only if the protocol Π' accepts.

Figure 4: Query Reduced Protocol $\tilde{\Pi}$

It can be verified that communication complexity, query complexity, round complexity, and verifier and prover running time are all as claimed. Next we show completeness and soundness.

Completeness. Suppose that $(x, w) \in \mathcal{L}$. Then with probability 1, the output of ϕ on query set I is accept. Consequently, the protocol Π' will accept with probability 1.

Soundness. Suppose that w is $\delta(n)$-far from $\mathcal{L}_x$, and let $\mathcal{P}^*$ be a prover strategy. By the robustness property of Π, with probability at least $1 - \varepsilon(n)$, the restriction of the transcript of Π and the implicit input w to the query set I is $\alpha(n)$-far from $\phi^{-1}(\texttt{accept})$, where $\delta'(q(n)) = \alpha(n)$. Consequently, the protocol Π' will reject with probability at least $1 - \varepsilon'(q(n))$. By the union bound, we conclude that the protocol $\tilde{\Pi}$ will reject with probability at least $1 - \varepsilon(n) - \varepsilon'(q(n))$.

8.2 Outer IOPP – proof of Lemma 8.3

Let $t = t(n) := \lceil 1/\beta(n) \rceil$, and let $\bar{\mathcal{L}}$ be a language with implicit length sequence $\mathcal{I} = \{\bar{n} \mid n \in \mathbb{N}\}$ (recalling that $\bar{n} = \left(\lceil n^{1/(2t(n))} \rceil\right)^{2t(n)}$), obtained by padding any implicit input belonging to $\mathcal{L}$ of length n with $\bar{n} - n$ zeros. Note that

$$\begin{aligned}
\frac{n}{\bar{n}} &\geq \frac{n}{\left(n^{1/(2t)} + 1\right)^{2t}} \\
&= \frac{n}{\sum_{i=0}^{2t} \binom{2t}{i} n^{i/(2t)}} \\
&\geq \frac{n}{n + n^{(2t-1)/(2t)} \cdot \sum_{i=0}^{2t} \binom{2t}{i}} \\
&= \frac{1}{1 + 4^t \cdot n^{-1/(2t)}} \\
&\geq \frac{1}{1+\gamma/2}, \qquad (6)
\end{aligned}$$

where the last inequality follows by assumption that $\gamma \geq \frac{2 \cdot 4^{1/\beta}}{n^{\beta/2}}$.

By Lemma 4.6, there exists an $\mathcal{S}_{\otimes t}$-linear IOP reduction for $\bar{\mathcal{L}}$, over a constructible field ensemble of size poly $\left(\log(n), (\gamma\beta)^{-1/\beta}, 1/\delta\right)$ of characteristic 2, with communication, query complexity, and verifier running time poly $\left(n^\beta, (\gamma\beta)^{-1/\beta}, 1/\delta, s\right)$, round complexity $\beta^{-O(1/\beta)}$, prover's running time poly(n), and $(\beta^{O(1/\beta)}, (\gamma\beta)^{O(1/\beta)} \cdot \delta)$-robustness.

By Lemma 4.7, there exists a systematic $\mathcal{S}_{\otimes t}$-sumcheckable code $C = \{C_{\bar{n}} : \{0,1\}^{\bar{n}} \to \{0,1\}^{n'}\}_{\bar{n} \in \mathcal{I}}$, where $C_{\bar{n}}$ has rate at least $\frac{1}{1+\gamma/2}$ and relative distance $(\gamma\beta)^{O(1/\beta)}$, and the corresponding IOPP has communication complexity, query complexity, and verifier running time poly $\left(n^\beta, (\gamma\beta)^{-1/\beta}, 1/\delta\right)$, round complexity $\lceil 1/\beta \rceil$, proximity parameter $\min\left\{\delta \cdot \frac{1}{1+\gamma/2}, \frac{\mathrm{dist}(C_n)}{2}\right\}$, prover's running time poly(n), and $\left((\gamma\beta)^{O(1/\beta)} \cdot \delta, 1 - (\gamma\beta)^{O(1/\beta)} \cdot \delta\right)$-robustness.

Lemma 4.5 implies in turn the existence of an IOPP for $\bar{\mathcal{L}}$ with communication complexity $\frac{\gamma}{2} \cdot \bar{n} +$ poly $\left(n^\beta, (\gamma\beta)^{-1/\beta}, 1/\delta, s\right)$, query complexity and verifier running time poly $\left(n^\beta, (\gamma\beta)^{-1/\beta}, 1/\delta, s\right)$, round complexity $\beta^{-O(1/\beta)}$, proximity parameter δ, prover's running time poly(n), and robustness $\left((\gamma\beta)^{O(1/\beta)} \cdot \delta, 1 - (\gamma\beta)^{O(1/\beta)} \cdot \delta\right)$.

Moreover, the communication phase consists of a single prover message which is the non-systematic part of $C_{\bar{n}}(w)$ of length $\frac{\gamma}{2} \cdot \bar{n}$, followed by a two-way communication of length poly $\left(n^\beta, (\gamma\beta)^{-1/\beta}, 1/\delta, s\right)$. Moreover, for any integer $d \leq 2 \cdot \lceil 1/\beta \rceil$, the code $C_{\bar{n}}$ is locally $(\bar{n})^{1-2\beta d}$-dcomposable with running time $n^{1-2\beta d} \cdot \mathrm{polylog}(n)$ with respect to a base code C_0 of rate at least $1 - \gamma$.

Finally, passing to $\mathcal{L}$ (without the padding), and using Eq. (6) we obtain that $\mathcal{L}$ also has an IOPP with the same parameters as above, where the first prover message is the non-systematic part of $C_{\bar{n}}(w, 0^{\bar{n}-n})$ of length $\gamma \cdot n$.

8.3 Inner IOPP – proof of Theorem 8.4

We use the following lemma that shows how to transform an IOPP of high (i.e., constant) proximity into a IOPP of arbitrarily low proximity.

Lemma 8.6. *Let $\mathcal{L}$ be a pair language, and let $\delta = \delta(n) > 0$. Suppose that the following exist.*

- *A relaxed locally decodable code* $\mathcal{C} = \{C_n : \{0,1\}^n \to \{0,1\}^{n'}\}_{n\in\mathbb{N}}$ *with relative distance* δ'*, decoding radius* $\frac{\delta'}{2}$*, query complexity* q'*, and failure probability* $\delta/2$*.*
- *An* IOPP *for* $\mathcal{L}' := \{(x, C(w)) : (x, w) \in \mathcal{L}\}$ *with communication complexity* cc*, query complexity* q*, round complexity* ℓ*, proximity parameter* $\frac{\delta'}{2}$*, and soundness error* ε*.*

Then, there exists an IOPP *for* $\mathcal{L}$ *with communication complexity* $n' + \mathrm{cc}(n')$*, query complexity* $q(n') + q'(n) \cdot O(\log(1/\varepsilon)/\delta)$*, round complexity* ℓ*, proximity parameter* δ*, and soundness error* ε*. Moreover,*

- *If the verifier in the* IOPP *for* $\mathcal{L}'$ *has running time* T'*, and the relaxed local decoder for* $\mathcal{C}$ *has running time time* T_{rlcc}*, then the verifier in the resulting* IOPP *has running time* $T'(n') + O(\log(1/\varepsilon)/\delta) \cdot T_{rlcc}(n)$*.*
- *If the prover in the* IOPP *for* $\mathcal{L}'$ *has running time* T'*, and* $\mathcal{C}$ *has encoding time* T_{enc}*, then the prover in the resulting* IOPP *has running time* $T'(n') + T_{enc}(n)$*.*

Proof. The prover in the IOPP Π for $\mathcal{L}$ first sends the string $C(w)$. The prover and the verifier then interact according to the IOPP Π' for $\mathcal{L}'$, wrt to the explicit input x and implicit input $C(w)$. Let z denote the first prover's message that is allegedly equal to $C(w)$.

In the query phase, the verifier first runs the check of Π', wrt to the explicit input x and the implicit input z. If Π' rejects, then the verifier rejects. Otherwise, the verifier repeats the following procedure $O(\log(1/\varepsilon)/\delta)$ times: the verifier picks a uniform random $i \in [n]$, and applies the relaxed local decoder on input coordinate i with oracle access to z. If in any of the invocations the output of the relaxed local decoder is different than w_i (which it obtains by making a query to the implicit input w) then the verifier rejects. Otherwise, the verifier accepts.

It can be verified that the communication complexity, query complexity, and verifier and prover running times are all as claimed. Completeness is also clear. Next we show soundness.

Suppose that w is δ-far from $\mathcal{L}_x$. If z is $\frac{\delta'}{2}$-far from any codeword of C then Π' rejects with probability at least $1 - \varepsilon$. Hence we may assume that z is $\frac{\delta'}{2}$-close to some codeword $C(w')$. Next suppose that $w' \notin \mathcal{L}_x$. Then, since the code C has relative distance at least δ', we have that $C(w')$ is δ'-far from $\mathcal{L}'_x$. By the triangle inequality, this implies in turn that z is $\frac{\delta'}{2}$-far from $\mathcal{L}'_x$, and so Π' will once more reject with probability at least $1-\varepsilon$. Thus, we we may also assume that $C(w') \in \mathcal{L}'_x$.

By definition, if $C(w') \in \mathcal{L}'_x$ then $w' \in \mathcal{L}_x$. But as we have assumed that w is δ-far from $\mathcal{L}_x$, the above implies in turn that $\mathrm{dist}(w, w') \geq \delta$. Thus, with probability at least δ, the verifier will pick $i \in [n]$ on which w and w' differ, and moreover, the relaxed local decoder will output w'_i with probability at least $1 - \delta/2$. We conclude that the verifier rejects with probability at least $\delta/2$ on each of the invocations. Finally, repeating this procedure for $\log(1/\varepsilon)/\delta$ times gives rejection probability at least $1 - \varepsilon$. □

We now proceed to the proof of Theorem 8.4, based on the above lemma.

Proof of Theorem 8.4. Theorem 1.5 in [BGH+06] gives an explicit linear relaxed locally decodable code $C = \{C_n : \{0,1\}^n \to \{0,1\}^{\mathrm{poly}(n)}\}_{n\in\mathbb{N}}$ of some constant relative distance $\delta' > 0$, constant decoding radius $\alpha' > 0$, constant query complexity, failure probability $1/3$, and running time $\mathrm{poly}(n)$. By repeating the relaxed local decoder for $O(\log(1/\delta))$ times, and outputting some value other than $\perp$ if and only if all invocations output this value, we may assume that the relaxed local

decoder has failure probability at most $\delta/2$, at the cost of increasing the query complexity and running time by a multiplicative factor of $O(\log(1/\delta))$.

For any language in P and for any constant $\delta_0, \varepsilon_0 > 0$, Mie [Mie09] gives an IOPP of communication complexity $\mathrm{poly}(n)$, constant query complexity, round complexity 1, proximity parameter δ_0, soundness error ε_0, and verifier and prover running time $\mathrm{poly}(n)$.

Note that as $\mathcal{L}$ is computable in time $\mathrm{poly}(n)$, and as $\mathcal{C}$ is an explicit linear code, $\mathcal{L}'$ is also computable in time $\mathrm{poly}(n)$. Consequently, there exists an IOPP for $\mathcal{L}'$ of communication complexity $\mathrm{poly}(n)$, constant query complexity, round complexity 1, proximity parameter $\min\{\delta', \alpha'\}$, soundness error $1/2$, and verifier and prover running time $\mathrm{poly}(n)$. Note that the soundness error can be decreased to ε by repeating the verifier check for $O(\log(1/\varepsilon))$ times and accepting if and only if all invocations accept, at the cost of increasing the query complexity and verifier running time by a multiplicative factor of $O(\log(1/\varepsilon))$.

Lemma 8.6 then gives an IOPP for $\mathcal{L}$ of communication complexity $\mathrm{poly}(n)$, query complexity $O(\log(1/\varepsilon) \cdot \log(1/\delta)/\delta)$, round complexity 1, proximity parameter δ, soundness error ε, prover running time $\mathrm{poly}(n)$, and verifier running time $\mathrm{poly}(n) \cdot \log(1/\varepsilon) \cdot \log(1/\delta)/\delta$. □

9 IOP for NP – proof of Theorem 3.1

The proof of Theorem 3.1 relies on the following lemma that, loosely speaking, shows how to transform an IOPP for *deterministic* languages into an IOP for *non-deterministic* languages. This transformation is analogous to the known transformation from PCPPs for P to PCPs for NP [BGH+06, DR06]. The main difference however is that, since we cannot afford even a constant blowup in communication, we have to be extremely careful when composing and in particular use some specific properties of the underlying IOPP.

Lemma 9.1. *Let $\mathcal{L} \in \mathsf{NP}$ with corresponding relation $\mathcal{R}_{\mathcal{L}}$, in which the instances have length m and witnesses have length n, where n and m are polynomially related and $m \geq n$ (i.e., witnesses are shorter than their corresponding instances). Let $E = \{E^{(m)} : \{0,1\}^m \to \{0,1\}^{m'}\}_{m \in \mathbb{N}}$ be a code ensemble with relative distance $\delta > 0$ and quasi-linear time encoding. Let $C_0 = \{C_0^{(n)} : \{0,1\}^n \to \{0,1\}^{n'}\}_{n \in \mathbb{N}}$ be a code ensemble with rate $1 - \eta$ and quasi-linear time encoding. Let $\ell = \lceil \frac{n+m'}{n} \rceil$ and suppose that $C = \{C^{(n \cdot \ell)} : \{0,1\}^{n \cdot \ell} \to \{0,1\}^*\}_{n \in \mathbb{N}}$ is a systematic locally ℓ-decomposable code wrt the base code C_0.*

Let $\mathcal{L}' = \{(E(x), w) : (x, w) \in \mathcal{R}_{\mathcal{L}}\}$ (note that we view the tuple $(E(x), w)$ as the implicit input to $\mathcal{L}'$ and there is no explicit input) and suppose that $\mathcal{L}'$ has a q-query r-round IOPP, with respect to proximity parameter $\delta/2$ and soundness error ε with the following properties:

- *The first message in the IOPP for $\mathcal{L}'$ is the non-systematic part of $C(E(x), w)$.*
- *The rest of the communication in the IOPP is of length cc.*
- *The IOPP verifier runs in time T_V and the IOPP prover runs in time T_P.*

Then, $\mathcal{L}$ has an $r(m' + n)$-round IOP with soundness error $\varepsilon(m' + n)$. The query complexity is $O(q(m' + n))$ and the communication consists of a first message sent by the prover of length $n' = \frac{1}{1-\eta} \cdot n$ bits followed by $\mathrm{cc}(m' + n)$ additional communication. The IOP verifier runs in time $\tilde{O}(m) + T_V(m' + n)$ and the IOP prover runs in time $\tilde{O}(m) + T_P(m' + n)$.

We first prove Lemma 9.1 in Section 9.1 and then show how to use it to obtain Theorem 3.1 in Section 9.2.

9.1 Proof of Lemma 9.1

Recall that the IOP verifier gets as input x and the IOP prover gets x and a witness w. First, the prover sends w to the verifier. Then, the two parties run the IOPP for $\mathcal{L}'$ wrt to the input $(E(x), w)$ (we once again emphasize that $(E(x), w)$ is an implicit input to the IOPP and there is no explicit input).[15] The IOP verifier emulates queries to $E(x)$ by generating $E(x)$ by itself (recall that the verifier has explicit access to x and so can compute $E(x)$ by itself), and redirects queries to w to the initial message sent by the prover. For reasons that will become apparent soon, we assume without loss of generality that $|E(x)| = m'$ is a multiple of n (this can be achieved by padding with 0's since we only assumed that E has constant relative distance and quasi-linear time encoding).

Observe that if $(x, w) \in \mathcal{R}_{\mathcal{L}}$ then $(E(x), w) \in \mathcal{L}'$ and by the completeness of the IOPP, the IOP verifier accepts.

For soundness, fix $x \notin \mathcal{L}$ and a cheating prover strategy P^*. We assume without loss of generality that P^* is deterministic and denote its first message by w^* (an alleged encoding of a witness). Observe that since $x \notin L$ and $|x| \geq |w^*|$, the pair $(E(x), w^*)$ is at least $\delta/2$ far from $\mathcal{L}'$. Thus, by soundness of the IOPP, the verifier accepts with probability at most ε.

To obtain the desired efficiency we slightly modify the above protocol (while making certain that this change does not effect completeness nor soundness). Recall that, by assumption, the first message in the IOPP is the non-systematic part of $C(E(x), w))$ where C is a systematic locally decomposable code wrt the base code ensemble C_0. Therefore, as described so far, the first message of our IOP prover consists of w and of the non-systematic part of $C(E(x), w)$.

To obtain a more efficient protocol, rather than sending this (relatively long) message, our IOP prover only sends $C_0(w)$ as its first message (of length $\frac{1}{1-\gamma} \cdot n$) and then continues the interaction as before. Since the code C is locally decomposable, queries to $C(E(x), w)$ (whether to the systematic part or not) can be emulated by making at most $O(q)$ queries to $C_0(w)$ (to which the verifier has oracle access since it was sent by the prover) and to $C_0((E(x))_1), \ldots, C_0((E(x))_{\ell-1})$, where $E(x)_i$ is the i-th block of $E(x)$ of length n and $\ell = \lceil \frac{n+m'}{n} \rceil$. Note that that verifier can generate $C_0\big((E(x))_1\big), \ldots, C_0\big((E(x))_{\ell-1}\big)$ by itself since it has x. The stated efficiency follows from the rate of C_0 and the quasi-linear time encoding of both E and C_0.

9.2 IOPs for NP: Proof of Theorem 3.1

Let $\mathcal{L} \in \mathsf{NP}$ with corresponding relation $\mathcal{R}_{\mathcal{L}}$ in which the instances have length m and witnesses have length n, where n and m are polynomially related, and such that $\mathcal{R}_L$ can be decided in time $\mathrm{poly}(n)$ and space $s \geq \log(n)$. Also, we assume without loss of generality that $m \geq n$ (by padding the input with 0's if necessary), while noting that this can increase our input size by at most n.

Let $E = \{E^{(m)} : \{0,1\}^m \to \{0,1\}^{m'}\}_{m \in \mathbb{N}}$ be the code ensemble from Theorem 2.3 with constant rate and constant relative distance $\delta > 0$ (note that we do not require E to have rate close to 1).

[15] Note that we cannot use x as an explicit input to the IOPP. Syntactically, this is because the IOPP that we eventually use is for languages that do not have an explicit input. More fundamentally however, while we could have extended that IOPP to support an explicit input $x \in \{0,1\}^m$, this would introduce a $\mathrm{poly}(m)$ dependence in the communication and verification time that we would like to avoid.

Observe that E also has quasi-linear time encoding and that deciding membership in E can be done in polynomial-time and poly-logarithmic space.

Let $\beta \in (0,1)$ and $\gamma \in (0,1)$ be nice functions as in the statement of Theorem 3.1. Note that $\beta(n+m') = \beta(\Theta(m)) \geq \Theta(\beta(m))$. Thus, $\text{poly}(1/\beta(n+m')) = \text{poly}(1/\beta(m)) \leq \log(m)$ and $\gamma(n+m') = \Theta(\gamma(m)) \geq m^{-O(\beta(m))} \geq (n+m')^{-O(\beta(n+m')}$. Choosing the constant in the big-Oh appropriately, this means that $\gamma(n+m') \geq \frac{200 \cdot 4^{1/\beta(n+m')} \cdot \log(n+m')}{(n+m')^{\beta(n+m')/2}}$, for sufficiently large n. In particular this implies that β and γ satisfy the requirements of Lemma 8.1.

Let C and C_0 be the code ensembles from Lemma 8.1 wrt the above choice of β and γ'. Observe that (1) C_0 has rate $\frac{1}{1+\gamma}$ and is encodable in quasi-linear time, and (2) that C is a systematic locally $\lceil \frac{n+m'}{n} \rceil$-decomposable code wrt the base code C_0.

Consider the language $\mathcal{L}' = \{(E(x), w) : (x, w) \in \mathcal{R}_{\mathcal{L}}\}$ and note that inputs to $\mathcal{L}'$ have length $n + m' = \Theta(m)$. Observe that membership in $\mathcal{L}'$ can be decided in polynomial-time and space $s' = \max s, \text{polylog}(n)$.

Thus, by Lemma 8.1, there exists an IOPP for $\mathcal{L}'$ with respect to proximity parameter $\delta/2$ with query complexity poly $\left((\gamma\beta)^{-1/\beta}, 1/\delta\right)$, round complexity $\beta^{-O(1/\beta)}$, and soundness error $1/2$. Moreover,

- The verifier's running time is poly $\left(m^{\beta}, (\gamma\beta)^{-1/\beta}, 1/\delta, s\right)$.
- The prover's running time is poly(m).
- The communication phase consists of a single prover's message of length $\gamma \cdot (n+m')$ which is the non-systematic part of C_n, followed by a two-way communication of length poly $\left(m)^{\beta}, (\gamma\beta)^{-1/\beta}, 1/\delta, s\right)$.

Now, by Lemma 9.1, we obtain that $\mathcal{L}$ has a $\beta^{-O(1/\beta)}$-round IOP with soundness error $1/2$. The query complexity is poly $\left((\gamma\beta)^{-1/\beta}, 1/\delta\right)$ and the communication consists of a first message sent by the prover of length $\frac{1}{1-\gamma} \cdot n$ bits followed by poly $\left(m^{\beta}, (\gamma\beta)^{-1/\beta}, 1/\delta, s\right)$ additional communication. The IOP verifier runs in time $\tilde{O}(m) + \text{poly}\left(m^{\beta}, (\gamma\beta)^{-1/\beta}, 1/\delta, s\right)$ and the IOP prover runs in time poly(m). The theorem now follows by observing that δ is a constant.

Acknowledgement

We thank Swastik Kopparty for useful discussions.

References

[ALM+98] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *J. ACM*, 45(3):501–555, 1998.

[App17] Benny Applebaum. Exponentially-hard gap-csp and local PRG via local hardcore functions. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 836–847, 2017.

[ARW17] Amir Abboud, Aviad Rubinstein, and R. Ryan Williams. Distributed PCP theorems for hardness of approximation in P. In *58th IEEE Annual Symposium on Foundations*

of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017, pages 25–36, 2017.

[AS98] Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: A new characterization of NP. *J. ACM*, 45(1):70–122, 1998.

[BBHR18] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. Fast reed-solomon interactive oracle proofs of proximity. In *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, pages 14:1–14:17, 2018.

[BBHR19] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. Scalable zero knowledge with no trusted setup. In *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part III*, pages 701–732, 2019.

[BCG+17] Eli Ben-Sasson, Alessandro Chiesa, Ariel Gabizon, Michael Riabzev, and Nicholas Spooner. Interactive oracle proofs with constant rate and query complexity. In *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, pages 40:1–40:15, 2017.

[BCGT13] Eli Ben-Sasson, Alessandro Chiesa, Daniel Genkin, and Eran Tromer. On the concrete efficiency of probabilistically-checkable proofs. In *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 585–594, 2013.

[BCR+19] Eli Ben-Sasson, Alessandro Chiesa, Michael Riabzev, Nicholas Spooner, Madars Virza, and Nicholas P. Ward. Aurora: Transparent succinct arguments for R1CS. In *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part I*, pages 103–128, 2019.

[BCS16] Eli Ben-Sasson, Alessandro Chiesa, and Nicholas Spooner. Interactive oracle proofs. In *Theory of Cryptography - 14th International Conference, TCC 2016-B, Beijing, China, October 31 - November 3, 2016, Proceedings, Part II*, pages 31–60, 2016.

[BEFLR19] Omri Ben-Eliezer, Eldar Fischer, Amit Levi, and Ron D. Rothblum. Hard properties with (very) short pcpps and their applications. 2019.

[BFL91] László Babai, Lance Fortnow, and Carsten Lund. Non-deterministic exponential time has two-prover interactive protocols. *Computational Complexity*, 1:3–40, 1991.

[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, May 5-8, 1991, New Orleans, Louisiana, USA*, pages 21–31, 1991.

[BGH+05] Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil P. Vadhan. Short pcps verifiable in polylogarithmic time. In *20th Annual IEEE Conference on Computational Complexity (CCC 2005), 11-15 June 2005, San Jose, CA, USA*, pages 120–134, 2005.

[BGH+06] Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil P. Vadhan. Robust PCPs of proximity, shorter PCPs, and applications to coding. *SIAM J. Comput*, 36(4):889–974, 2006.

[BGKS19] Eli Ben-Sasson, Lior Goldberg, Swastik Kopparty, and Shubhangi Saraf. DEEP-FRI: sampling outside the box improves soundness. *Electronic Colloquium on Computational Complexity (ECCC)*, 26:44, 2019.

[BKK+16] Eli Ben-Sasson, Yohay Kaplan, Swastik Kopparty, Or Meir, and Henning Stichtenoth. Constant rate PCPs for circuit-sat with sublinear query complexity. *J. ACM*, 63(4):32:1–32:57, 2016.

[BS06] Eli Ben-Sasson and Madhu Sudan. Robust locally testable codes and products of codes. *Random Structures and Algorithms*, 28(4):387–402, 2006.

[BS08] Eli Ben-Sasson and Madhu Sudan. Short PCPs with polylog query complexity. *SIAM J. Comput.*, 38(2):551–607, 2008.

[BV15] Eli Ben-Sasson and Michael Viderman. Composition of semi-LTCs by two-wise tensor products. *Computational Complexity*, 24(3):601–643, 2015.

[Din07] Irit Dinur. The PCP theorem by gap amplification. *J. ACM*, 54(3):12, 2007.

[Din16] Irit Dinur. Mildly exponential reduction from gap 3sat to polynomial-gap label-cover. *Electronic Colloquium on Computational Complexity (ECCC)*, 23:128, 2016.

[DR06] Irit Dinur and Omer Reingold. Assignment testers: Towards a combinatorial proof of the PCP theorem. *SIAM J. Comput.*, 36(4):975–1024, 2006.

[DSW06] Irit Dinur, Madhu Sudan, and Avi Wigderson. Robust local testability of tensor products of LDPC codes. In *proceedings of the 9th International Workshop on Randomization and Computation (RANDOM)*, pages 304–315. Springer, 2006.

[EKR04] Funda Ergün, Ravi Kumar, and Ronitt Rubinfeld. Fast approximate probabilistically checkable proofs. *Inf. Comput.*, 189(2):135–159, 2004.

[FGL+96] Uriel Feige, Shafi Goldwasser, László Lovász, Shmuel Safra, and Mario Szegedy. Interactive proofs and the hardness of approximating cliques. *J. ACM*, 43(2):268–292, 1996.

[FS11] Lance Fortnow and Rahul Santhanam. Infeasibility of instance compression and succinct PCPs for NP. *J. Comput. Syst. Sci.*, 77(1):91–106, 2011.

[GH98] Oded Goldreich and Johan Håstad. On the complexity of interactive proofs with bounded communication. *Inf. Process. Lett.*, 67(4):205–214, 1998.

[GKR15] Shafi Goldwasser, Yael Tauman Kalai, and Guy N. Rothblum. Delegating computation: Interactive proofs for muggles. *J. ACM*, 62(4):27:1–27:64, 2015.

[Gol08] Oded Goldreich. *Computational complexity - a conceptual perspective.* Cambridge University Press, 2008.

[Gol18] Oded Goldreich. On doubly-efficient interactive proof systems. *Foundations and Trends in Theoretical Computer Science*, 13(3):158–246, 2018.

[GR18] Tom Gur and Ron D. Rothblum. Non-interactive proofs of proximity. *Computational Complexity*, 27(1):99–207, 2018.

[GRR17] Tom Gur, Govind Ramnarayan, and Ron Rothblum. Relaxed locally correctable codes. *Electronic Colloquium on Computational Complexity (ECCC)*, 24:143, 2017.

[GRR18] Tom Gur, Govind Ramnarayan, and Ron D. Rothblum. Relaxed locally correctable codes. In *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*, pages 27:1–27:11, 2018.

[GVW02] Oded Goldreich, Salil P. Vadhan, and Avi Wigderson. On interactive proofs with a laconic prover. *Computational Complexity*, 11(1-2):1–53, 2002.

[Hås01] Johan Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, 2001.

[Jus72] Jørn Justesen. Class of constructive asymptotically good algebraic codes. *IEEE Trans. Information Theory*, 18(5):652–656, 1972.

[Kar72] Richard M. Karp. Reducibility among combinatorial problems. In *Proceedings of a symposium on the Complexity of Computer Computations, held March 20-22, 1972, at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York, USA*, pages 85–103, 1972.

[Kil92] Joe Kilian. A note on efficient zero-knowledge proofs and arguments (extended abstract). In *Proceedings of the 24th Annual ACM Symposium on Theory of Computing, May 4-6, 1992, Victoria, British Columbia, Canada*, pages 723–732, 1992.

[KMRS17] Swastik Kopparty, Or Meir, Noga Ron-Zewi, and Shubhangi Saraf. High-rate locally correctable and locally testable codes with sub-polynomial query complexity. *J. ACM*, 64(2):11:1–11:42, 2017.

[KR08] Yael Tauman Kalai and Ran Raz. Interactive PCP. In *Automata, Languages and Programming, 35th International Colloquium, ICALP 2008, Reykjavik, Iceland, July 7-11, 2008, Proceedings, Part II - Track B: Logic, Semantics, and Theory of Programming & Track C: Security and Cryptography Foundations*, pages 536–547, 2008.

[KR15] Yael Tauman Kalai and Ron D. Rothblum. Arguments of proximity - [extended abstract]. In *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part II*, pages 422–442, 2015.

[LFKN92] Carsten Lund, Lance Fortnow, Howard J. Karloff, and Noam Nisan. Algebraic methods for interactive proof systems. *J. ACM*, 39(4):859–868, 1992.

[Mei13] Or Meir. IP = PSPACE using error-correcting codes. *SIAM J. Comput.*, 42(1):380–403, 2013.

[Mei14] Or Meir. Combinatorial PCPs with efficient verifiers. *Computational Complexity*, 23(3):355–478, 2014.

[Mic00] Silvio Micali. Computationally sound proofs. *SIAM J. Comput.*, 30(4):1253–1298, 2000.

[Mie09] Thilo Mie. Short PCPPs verifiable in polylogarithmic time with O (1) queries. *Ann. Math. Artif. Intell*, 56(3-4):313–338, 2009.

[MR17] Pasin Manurangsi and Prasad Raghavendra. A birthday repetition theorem and complexity of approximating dense csps. In *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, pages 78:1–78:15, 2017.

[Mul54] David E. Muller. Application of boolean algebra to switching circuit design and to error detection. *Trans. I.R.E. Prof. Group on Electronic Computers*, 3(3):6–12, 1954.

[Nis92] Noam Nisan. Pseudorandom generators for space-bounded computation. *Combinatorica*, 12(4):449–461, 1992.

[PF79] Nicholas Pippenger and Michael J. Fischer. Relations among complexity measures. *J. ACM*, 26(2):361–381, 1979.

[Ree54] Irving S. Reed. A class of multiple-error-correcting codes and the decoding scheme. *Trans. of the IRE Professional Group on Information Theory (TIT)*, 4:38–49, 1954.

[RRR16] Omer Reingold, Guy N. Rothblum, and Ron D. Rothblum. Constant-round interactive proofs for delegating computation. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 49–62, 2016.

[RS60] Irving S. Reed and Gustave Solomon. Polynomial codes over certain finite fields. *SIAM Journal of the Society for Industrial and Applied Mathematics*, 8(2):300–304, 1960.

[Rub18] Aviad Rubinstein. Hardness of approximate nearest neighbor search. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 1260–1268, 2018.

[RVW13] Guy N. Rothblum, Salil P. Vadhan, and Avi Wigderson. Interactive proofs of proximity: delegating computation in sublinear time. In *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 793–802, 2013.

[Sho88] Victor Shoup. New algorithms for finding irreducible polynomials over finite fields. In *29th Annual Symposium on Foundations of Computer Science, White Plains, New York, USA, 24-26 October 1988*, pages 283–290, 1988.

[Sti06] Henning Stichtenoth. Transitive and self-dual codes attaining the tsfasman-vla/spl breve/dut$80-zink bound. *IEEE Trans. Information Theory*, 52(5):2218–2224, 2006.

[Sud00] Madhu Sudan. Probabilistically checkable proofs - lecture notes, 2000. Available at http://madhu.seas.harvard.edu/MIT/pcp/pcp.ps.

[Sud01] Madhu Sudan. Algorithmic introduction to coding theory (lecture notes), 2001.

[Vid15] Michael Viderman. A combination of testability and decodability by tensor products. *Random Structures and Algorithms*, 46(3):572–598, 2015.

A A high-rate variant of Justesen's code

In this section we prove Theorem 2.3, based on Justesen's [Jus72] code. The construction uses the well-known family of Reed-Solomon codes.

Fact A.1 (Reed-Solomon codes, [RS60])**.** *For any constructible finite field $\mathbb{F}$ and integers $k \leq n \leq |\mathbb{F}|$, there exists a systematic linear code $C : \mathbb{F}^k \to \mathbb{F}^n$ of rate k/n and relative distance at least $1 - k/n$. Moreover,*

- *C_k is encodable in time $|\mathbb{F}| \cdot \text{polylog}(|\mathbb{F}|)$.*
- *Any coordinate of C is encodable in time $k \cdot \text{polylog}(k, |\mathbb{F}|)$ with space $\text{polylog}(k, |\mathbb{F}|)$.*

The fact that Reed Solomon codes can be encoded in quasi-linear time is based on the Fast Fourier Transform, whereas the fact that each coordinate in the encoding can be generated in small space follows from exponentiation via repeated squaring.

Fix a sufficiently large $k \in \mathbb{N}$, and let $\gamma = \gamma(k)$. In what follows we construct the code $C := C_k : \{0,1\}^k \to \{0,1\}^n$. The code C is obtained by encoding the symbols of a Reed-Solomon code C' given by the above fact with the Wozencraft Ensemble. We start by setting the parameters for the Reed-Solomon code C'.

The code C'. Let $m = \lfloor \log k \rfloor$, and let $\mathbb{F}$ be a finite field of size 2^m. Let $C' : \mathbb{F}^{k'} \to \mathbb{F}^{n'}$ be the code guaranteed by Fact A.1 of message length $k' = \lceil \frac{k}{m} \rceil$ and codeword length $n' = \lfloor k' \cdot (1 + \gamma/3) \rfloor$, noting that $k' \leq n'$, and

$$\begin{aligned} n' &\leq k' \cdot \left(1 + \frac{\gamma}{3}\right) \\ &\leq \left(\frac{k}{m} + 1\right) \cdot \left(1 + \frac{\gamma}{3}\right) \\ &\leq \left(\frac{k}{\log k - 1} + 1\right) \cdot \left(1 + \frac{\gamma}{3}\right) \\ &\leq |\mathbb{F}|, \end{aligned}$$

where the last inequality holds for sufficiently large k.

Then, C' has relative distance at least

$$\begin{aligned} 1 - \frac{k'}{n'} &\geq 1 - \frac{k'}{k' \cdot (1+\gamma/3) - 1} \\ &\geq \frac{\gamma/3 - 1/k'}{1+\gamma/3} \\ &\geq \frac{\gamma/3 - m/k}{1+\gamma/3} \\ &\geq \frac{\gamma/3 - \log k/k}{1+\gamma/3} \\ &\geq \frac{\gamma}{6}, \end{aligned}$$

where the last inequality holds for sufficiently large k, using the assumption that $\gamma \geq \frac{100 \log k}{k}$. Moreover, C' is encodable in time $k \cdot \mathrm{polylog}(k)$, and any coordinate of C' is encodable in time $k \cdot \mathrm{polylog}(k)$ with space $\mathrm{polylog}(k)$.

The code C. Let $\mathbb{F} = \{\alpha_1, \alpha_2, \cdots, \alpha_{2^m}\}$, let $s = \left\lfloor \frac{\gamma}{3} \cdot m \right\rfloor$, and for $\alpha \in \mathbb{F}$, let $\sigma(\alpha) \in \{0,1\}^s$ denote the projection of α (viewed as a length m bit string over the binary base field in the natural way) to the first s bits. For $i = 1, \ldots, 2^m$, define $C^{(i)} : \mathbb{F} \to \{0,1\}^{m+s}$ as $C^{(i)}(x) = (x, \sigma(\alpha_i \cdot x))$.

Given a message $x \in \{0,1\}^k$, let $x' \in \mathbb{F}^{k'}$ be the string obtained from x by viewing any consecutive m coordinates as an element of $\mathbb{F}$ in the natural way (and appending a couple of zeros to x if necessary). Let $y = C'(x') \in \mathbb{F}^{n'}$, and recalling that $n' \leq |\mathbb{F}|$, let

$$C(x) = \left(C^{(1)}(y_1), C^{(2)}(y_2), \ldots, C^{(n')}(y_{n'}) \right).$$

It can be verified that C is a binary systematic (up to reordering of coordinates) linear code, that it is encodable in time $k \cdot \mathrm{polylog}(k)$, and that any coordinate of C is encodable in time $k \cdot \mathrm{polylog}(k)$ with space $\mathrm{polylog}(k)$. Next we analyze the rate and distance of C.

Rate of C. The codeword length of C is

$$\begin{aligned} n := n' \cdot (m+s) &\leq \left(\frac{k}{m} + 1 \right) \cdot \left(1 + \frac{\gamma}{3} \right) \cdot m \cdot \left(1 + \frac{\gamma}{3} \right) \\ &\leq k \cdot \left(\left(1 + \frac{\gamma}{3} \right)^2 + \frac{2 \log k}{k} \right) \\ &\leq k \cdot (1+\gamma), \end{aligned}$$

where the last inequality follows by assumption that $\gamma \geq \frac{100 \log k}{k}$. We conclude that C has rate $k/n \geq 1/(1+\gamma) \geq 1 - \gamma$.

Distance of C: We divide into cases according to the value of γ.

Suppose first that $\gamma \leq \frac{100 \cdot \log \log k}{\log k}$. Since C' has relative distance at least $\frac{\gamma}{6}$, and since all the inner codes $C^{(i)}$ are injective, we have that any pair of distinct codewords of C differ on at least $\frac{\gamma}{6} \cdot n'$ coordinates. We conclude that C' has relative distance at least

$$\frac{(\gamma/6) \cdot n'}{n' \cdot (m+s)} \geq \frac{\gamma/6}{\log k \cdot (1+\gamma/3)} \geq \frac{\gamma}{100 \cdot \log k} \geq \Omega(\gamma^3),$$

where the last inequality holds for sufficiently large k using the assumption that $\gamma \leq \frac{100 \cdot \log\log k}{\log k}$.

Next assume that $\gamma \geq \frac{100 \cdot \log\log k}{\log k}$. In this case, we use the following claim to bound the relative distance of C. In what follows let $H : [0,1] \to [0,1]$ denote the binary entropy function given by $H(x) = -x\log(x) - (1-x)\log(1-x)$.

Claim A.2. *For any $\varepsilon > 0$, for at least $(1 - 2^{-\varepsilon(m+s)}) \cdot 2^m$ indices $i \in \{1, \ldots, 2^m\}$ it holds that the code $C^{(i)}$ has relative distance at least $H^{-1}\left(\frac{s}{m+s} - \varepsilon\right)$.*

Proof. Observe that each vector $z \in \{0,1\}^{m+s}$ can appear in the image of at most 2^{m-s} different codes $C^{(i)}$. The number of vectors $z \in \{0,1\}^{m+s}$ of weight less than $H^{-1}\left(\frac{s}{m+s} - \varepsilon\right)$ is at most $2^{(s/(m+s)-\varepsilon)\cdot(m+s)}$. Therefore, the number of codes $C^{(i)}$ containing some codeword $z \in \{0,1\}^{m+s}$ of weight less than $H^{-1}\left(\frac{s}{m+s} - \varepsilon\right)$ is at most

$$2^{(s/(m+s)-\varepsilon)\cdot(m+s)} \cdot 2^{m-s} = 2^m \cdot 2^{-\varepsilon(m+s)}.$$

□

Since C' has relative distance least $\frac{\gamma}{6}$, for any codeword c' of C', at least $\frac{\gamma}{6}$-fraction of the coordinates are non-zero. By Claim A.2, with $\varepsilon = \frac{s}{2(m+s)}$, we conclude that for at most $2^{m-s/2}$ coordinates $i \in [n']$, the code $C^{(i)}$ has relative distance less than $H^{-1}\left(\frac{s}{m+s} - \varepsilon\right)$, or equivalently, for at most $\frac{2^{m-s/2}}{n'}$-fraction of the coordinates $i \in [n']$ it holds that the code $C^{(i)}$ has relative distance less than $H^{-1}\left(\frac{s}{m+s} - \varepsilon\right)$.

We conclude that for at least $\left(\frac{\gamma}{6} - \frac{2^{m-s/2}}{n'}\right)$-fraction of the coordinates $i \in [n']$ it holds that $c'_i \neq 0$, and additionally, $C^{(i)}$ has relative distance at least $H^{-1}\left(\frac{s}{2(m+s)}\right)$. Consequently, C has relative distance at least

$$\left(\frac{\gamma}{6} - \frac{2^{m-s/2}}{n'}\right) \cdot H^{-1}\left(\frac{s}{2(m+s)}\right). \tag{7}$$

Using the assumption that $\gamma \geq \frac{100 \cdot \log\log k}{\log k}$ (implying that $\frac{100\log(100/\gamma)}{\gamma} \leq \log k$), we can bound the first factor in Eq. (7) by

$$\frac{\gamma}{6} - \frac{2^{m-s/2}}{n'} \geq \frac{\gamma}{6} - \frac{2^{(1-\gamma/6)\cdot\log k+1}}{k/\log k} = \frac{\gamma}{6} - 2^{-(\gamma\cdot\log k/6 - \log\log k - 1)} \geq \frac{\gamma}{6} - 2^{-\gamma\cdot\log k/12} \geq \frac{\gamma}{12}. \tag{8}$$

We can also bound the second factor in Eq. (7) by

$$H^{-1}\left(\frac{s}{2(m+s)}\right) \geq H^{-1}\left(\frac{\gamma/3 - 1/(\log k - 1)}{2 + 2\cdot(\gamma/3 - 1/(\log k - 1))}\right) = \Omega(\gamma^2), \tag{9}$$

where the last inequality follows for sufficiently large k, using the assumption that $\gamma \geq \frac{100 \cdot \log\log k}{\log k}$.

Using Eqs. (7) to (9), we conclude that also in this case C has relative distance at least $\Omega(\gamma^3)$.

ECCC ISSN 1433-8092
https://eccc.weizmann.ac.il