Electronic Colloquium on Computational Complexity, Revision 2 of Report No. 127 (2019)

Local Proofs Approaching the Witness Length*

Noga Ron-Zewi† Ron D. Rothblum‡

September 30, 2023

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). Efficient IOPs are currently at the core of leading practical implementations of highly efficient proof-systems.

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 with 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 communication complexity $(1+\gamma)\cdot t$, 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 communication length $c \cdot t$ for a large (unspecified) constant $c \geq 1$.

Our proof leverages the local testability and (relaxed) local correctability of high-rate tensor codes, as well as their support of a sumcheck-like procedure. In particular, we bypass the barrier imposed by the low rate of *multiplication codes* (e.g., Reed-Solomon, Reed-Muller or AG codes) - a key building block of all known short PCP/IOP constructions.

*This is an updated and revised version of an extended abstract that has appeared at FOCS 2020 (see [RR19] for a full version). For simplicity and clarity of presentation, we have decided to omit some of the results from [RR19].

†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, by ISF grant 735/20, and by the European Union (ERC, ECCC, 101076663). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.

‡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 (Grants No.f 1262/18 and 2137/19), 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 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, BCG+17, ARW17]. 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 in 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 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, BGKS20] 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.* [RRR21] (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 communication complexity. 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 communication complexity is $(1+\gamma)\cdot t$, for circuits of size t and for any constant $\gamma > 0$.

Theorem 1 (Informally Stated, see Theorem 3.1 and Remark 1.2)**.** *For any constant $\gamma, \varepsilon > 0$,* CircuitSAT *has a constant-round and constant-query* IOP *in which the communication complexity 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 communication complexity $c \cdot t$, for a constant $c \geq 1$ that is left unspecified.[3] While the communication complexity in Theorem 1 is shorter than that in [BCG+17], the round complexity is slightly larger (Indeed, while we have not attempted to optimize the round complexity, a naive implementation seems to require 6 rounds, whereas [BCG+17] has only 3 rounds).

The proof of Theorem 1 relies on techniques developed in Meir's [Mei13] code-based proof of the IP = PSPACE Theorem, see Section 1.2.

Remark 1.1. *The fact that the soundness error in Theorem 1 is an* arbitrarily small *constant is non-trivial, since naive repetition increases the communication complexity. Also, we note that the* IOP *in Theorem 1 can be extended to* sub-constant *values of $\gamma = \gamma(t) > 0$ and $\varepsilon = \varepsilon(t) > 0$ (simultaneously) at the cost of increasing the query complexity to* $\mathrm{poly}\left(\frac{1}{\gamma}, \log(1/\varepsilon)\right)$ *(and no overhead to the round complexity).*[4] *This leads to an* IOP *with communication complexity $(1+o(1))\cdot t$ and negligible soundness error, but with poly-logarithmic query complexity. For simplicity and clarity the details are omitted from the current version, see the previous version [RR19] for details.*

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. Essentially, this is because PCPs — in order to facilitate local checking — typically encode the entire computation (rather than just the witness) via a suitable error correcting code.

For example, consider checking the 3-Colorability of a (connected) graph $G = (V, E)$. Using any of the known PCPs or IOPs in the literature would give proof length or communication complexity $\Omega(|E|)$, which can be quadratically larger than the length of the natural NP witness (which has length $O(|V|)$). Alternatively, checking the satisfiability of a given CNF formula ϕ with m clauses and n variables is only known to have PCPs and IOPs of proof length or communication complexity $\Omega(m+n)$, but has a standard NP witness of length n (which again can be significantly smaller than $m+n$).

[3] We estimate that c is at least $3 \cdot 6^3 = 648$. This is since the [BCG+17] IOP includes three codewords, each of which 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 communication complexity.

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 NP $\subseteq$ co-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 communication complexity 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 communication complexity is $(1+\gamma)\cdot n$, where n is the length of the original NP witness, rather than poly(n) as in [KR08], and with constant query and round complexities.

The class of relations that we can support consists of all NP relations that can be verified in polynomial-time with bounded polynomial space.[6] In particular, we obtain IOPs with communication approaching the witness length for many natural NP problems, such as: SAT, Hamiltonicity, TSP, all 21 of Karp's original NP-complete problems[7] [Kar75], etc.

Theorem 2 (Informally Stated, see Theorem 3.1)**.** *Let $\mathcal{R}$ be an* NP *relation which can be verified in polynomial-time with bounded polynomial space (i.e., space n^{ξ} for some sufficiently small constant $\xi > 0$). Then for any constant $\gamma, \varepsilon > 0$, the relation $\mathcal{R}$ has a constant-round and constant-query* IOP *with communication complexity $(1+\gamma)\cdot n$, 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).*

Remark 1.2. *It is worth emphasizing that* every *language in* NP *has a corresponding* NP *relation which can be verified in polynomial time with 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 polynomial time with small space, we do not need to pay this additional (potentially large) 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 communication complexity $(1+\gamma)\cdot t$, where t is the circuit size. Thus, Theorem 1 follows as an immediate corollary of Theorem 2.*[8]

[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, RRR21], yields IOPs with communication complexity 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 (see Remark 1.3).

[6] We emphasize that the 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] For all but 2 of Karp's 21 problems, it is straightforward to see that their natural NP relation can be verified in polynomial time with bounded polynomial space. The two exceptions are: *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. The natural NP witness for these problems is a specification of the set of vertices (resp., edges) to be removed. While we are unaware of a *deterministic* bounded space algorithm for checking whether a (directed) graph is a-cyclic, it is straightforward to show that this problem is in *co-*$\mathcal{NL}$. Using an unpublished extension [RRR17] of the [RRR21] protocol to $\mathcal{NL}$ (and therefore also *co-*$\mathcal{NL}$ [Imm88, Sze87]) one can obtain short IOPs also for these two problems.

[8] We note that we cannot use only the satisfying assignment as the witness, since there are circuits whose evaluation require a linear space in the number of variables.

Similarly to Theorem 1, we can extend Theorem 2 to sub-constant values of $\gamma = \gamma(n) > 0$ and $\varepsilon = \varepsilon(n) > 0$, at the cost of increasing the query complexity to $\mathrm{poly}(\frac{1}{\gamma}, \log(1/\varepsilon))$ (and no overhead to the round complexity), leading to communication complexity $(1+o(1))\cdot n$ and negligible soundness error, but with poly-logarithmic query complexity (omitted from the current version, see the previous version [RR19] for details).

The communication complexity in Theorem 2 is very close to optimal, under the following plausible complexity theoretic conjecture. Loosely speaking, the *randomized strong exponential time hypothesis* (RSETH) states that SAT is not contained in $\mathsf{BPTIME}\big(2^{(1-\gamma)\cdot n}\big)$ for any constant $\gamma > 0$ (where n here is the number of variables in the formula). The work of Goldreich and Håstad [GH98] shows that only languages in $\mathsf{BPTIME}(2^b \cdot \mathrm{poly}(n))$ have constant-round public-coin interactive proofs (let alone IOPs) in which the prover sends at most b bits. Put together, these two facts imply that SAT does not have a (constant-round and public-coin) IOP in which the prover sends less than $(1-\gamma)\cdot n$ bits, unless RSETH is false. For a more precise statement and further details, see Appendix A.

We also mention that the limitation in Theorem 2 to relations computable in space n^ξ (for a sufficiently small constant $\xi > 0$) is inherited from the work of [RRR21], 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 $\frac{1}{2}$ (i.e., leading to a space bound of $n^{0.499}$). On the other hand, a recent follow-up work of Nassar and Rothblum [NR22] gives evidence that some space bound is inherent.

Remark 1.3. *Our results can also be interpreted as* interactive PCPs *(*IPCP*) [KR08], a more restricted model than* IOP*s in which the prover first sends a single long message to which the verifier has oracle access (like a* PCP*), followed by short interactive protocol with sublinear communication during which the verifier reads the prover's messages in full (like an interactive proof).*

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 $\gamma, \beta > 0$. Alternatively, optimizing the communication complexity, we can obtain 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 the communication has 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 (omitted from the current version, see the previous version [RR19]). These results 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* $\mathrm{polylog}(n)$.

1.1.2 Interactive oracle proofs of proximity with sublinear communication complexity

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, RRR21].

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, as well as a few bits of w. 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 Theorem 3.2)**.** *Let $\mathcal{L}$ be a language that can be decided in polynomial-time with bounded-polynomial space. Then, for any constant $\gamma, \beta > 0$, the language $\mathcal{L}$ has a constant-round and constant-query* IOPP *with communication complexity $\gamma \cdot n$. The verifier runs in time n^{β}, and the prover runs in time* $\mathrm{poly}(n)$.

We remark that the communication complexity in Theorem 3 is actually less than even the input length n. At first glance this may seem quite surprising as, by the aforementioned work [GH98] (see also [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 $\mathcal{L}^* \in \mathsf{Logspace}$ 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 cryptographic 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 1.4. *Interestingly, since the communication complexity 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, RRR21, RR20] also construct* IPP*s with sublinear communication for bounded-space computations, but in a different parameter regime: the results in [RVW13, RRR21, RR20] have a much smaller communication complexity (e.g., $O(\sqrt{n})$) but on the other hand do 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., $\tilde{O}(\sqrt{n})$ 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 $\tilde{O}(\sqrt{n})$ (note that the witness here is the satisfying assignment, and so the witness length is 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*,[9] 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[10] them. Loosely speaking, a multiplication code [Mei13, Mei14] is a linear[11] error correcting code C, over a finite field $\mathbb{F}$, so that the vector space $\text{span}\{c_1 \star c_2 : c_1, c_2 \in C\}$, where $c_1 \star c_2$ denotes entry-wise multiplication (over $\mathbb{F}$), is itself a code (i.e., it has large relative distance[12]). 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 code [RS60], the Reed-Muller code [Mul54, Ree54], low rate tensor codes [Mei13, Mei14] or AG codes [Sti06, BKK+16]) have rate[13] less than $\frac{1}{2}$, and this is inherent [Ran13]. Since we are aiming for rate close to 1, we cannot afford to encode the entire computation using such a code.

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 (which is not a multiplication code). 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 $C_0 : \{0,1\}^{n_0} \to \{0,1\}^{n_0'}$, consider the (two-dimensional) *tensor product code* $C_0 \otimes C_0 : \{0,1\}^{n_0 \times n_0} \to \{0,1\}^{n_0' \times n_0'}$; we will define the tensor product formally in Definition 2.11, but for now, we will view the codewords of $C_0 \otimes C_0$ as $n_0' \times n_0'$ matrices with the constraints that the rows and columns are all codewords of the base code C_0.

[9] 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 correction, the local corrector is additionally allowed to abort whenever a corruption is detected.

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

[11] A code $C : \mathbb{F}^n \to \mathbb{F}^{n'}$ is *linear* if it is a linear transformation over $\mathbb{F}$.

[12] The *(minimal) relative distance* of C is the minimal *relative Hamming distance* of every two (distinct) codewords, i.e., the minimum fraction of coordinates on which any pair of codewords differ.

[13] The *rate* of a code $C : \Sigma^n \to \Sigma^{n'}$ is defined as n/n', and it measures the amount of redundancy in encoding.

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.[14] Hence, we take $C : \{0,1\}^n \to \{0,1\}^{(1+\gamma/2)\cdot n}$ to be a tensor product of some high-rate base code C_0. Given a satisfying assignment $w \in \{0,1\}^n$ for ϕ, the prover in our IOP first sends $C(w)$ (which has length $(1+\frac{\gamma}{2})\cdot n$). 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 take a brief detour into the classical method for constructing PCPs 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 2.3 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].[15] More specifically, and without getting into the details, there exists a constant round interactive proof with sublinear communication complexity 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 true (and simpler) claim, whereas soundness means that if the original claim was false, then no

[14] Local testability actually requires the dimension of the tensor product to be at least 3 [Val05, GM12]. For simplicity we ignore this fact in this high-level overview.

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

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 $\hat{w}$.[16]

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. Before proceeding, we remark

[16] Here we also use the fact that the verifier can compute $\hat{I}_\phi$ by itself in $\tilde{O}(n)$ time, see Section 2.3 for details.

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, following [Mei13], 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[17] linear code such that $C = C_0 \otimes C_0$. Thus, $n_0 = \sqrt{n}$ and $n_0' = \sqrt{(1 + \frac{\gamma}{2}) \cdot n}$. 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 $\pi(j) = \sum_{i \in [n_0]} \lambda_1(i) \cdot c(i,j)$, for every $j \in [n_0']$. In other words, π is computed by taking a linear combination of the first n_0 rows of c, with coefficients corresponding to λ_1.

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 fraction of coordinates. The verifier then chooses a random $j_0 \in [n_0']$ and checks that $\tilde{\pi}(j_0) = \sum_{i \in [n_0]} \lambda_1(i) \cdot c(i, j_0)$ by reading the j_0-th column of c. Assuming that the prover sent $\tilde{\pi} \neq \pi$, with constant probability over the choice of j_0 it holds that $\tilde{\pi}(j_0) \neq \pi(j_0) = \sum_{i \in [n_0]} \lambda_1(i) \cdot c(i, j_0)$ and so the verifier rejects. Since we can now assume that the prover actually sent π, the verifier can simply output $\sum_{j \in [n_0]} \lambda_2(j) \cdot \pi(j) = \sum_{i,j \in [n_0]} \lambda_1(i) \cdot \lambda_2(j) \cdot c(i,j) = \langle \lambda_1 \otimes \lambda_2, w \rangle$ as desired.

This concludes the description of the warmup. Observe that the communication complexity is $(1 + \frac{\gamma}{2}) \cdot n + \tilde{O}(\sqrt{n}) \leq (1 + \gamma) \cdot n$ and the query complexity is $\tilde{O}(\sqrt{n})$ as promised. In fact, a variant of this protocol only requires the verifier to make a *single* query to c (and additional $\tilde{O}(\sqrt{n})$ queries to the prover's messages).

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.

[17] We say that C is *systematic* if the message appears as the beginning of every codeword.

This approach is inspired by Meir's [Mei13] combinatorial proof of the IP = PSPACE theorem. In Meir's protocol, which is an abstraction of the [GKR15] protocol, claims about larger tensor codewords are reduced to claims about smaller tensor codewords via 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 $\tilde{O}(\sqrt{n})$ 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 precisely a PCP of proximity (PCPP), that the verifier would have accepted had it read these queries. Since the PCPP is applied to an input of length $\tilde{O}(\sqrt{n})$ we can afford to use existing constant-query PCPP constructions (e.g., those with poly-logarithmic overhead [BS08, Din07, Mie09]). A similar type of IOP composition was used also in [BCG+17] and as in their work, we utilize interaction to pay only polynomially in the randomness complexity of the so-called "outer" IOP (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 $\tilde{O}(\sqrt{n})$ 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 also need the outer IOP verifier to run in sublinear time. 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.* [RRR21] we show that a similar statement holds for any NP relation computable in polynomial-time with bounded polynomial-space. This basically follows from the fact that the verifier in [RRR21] runs in sublinear time given access to the low degree extension of its input. Plugging in the [RRR21] protocol instead of Π_{reduce} lets us obtain a high-rate IOP for any non-deterministic bounded space computation, thereby proving Theorem 2.

Remark 1.5. *One can replace the [RRR21] protocol with the doubly efficient interactive proof-system of Goldwasser et al. [GKR15] to obtain an* IOP *approaching the witness length also for* NP *relations that can be verified in small* depth *(rather than small space). However, the resulting* IOP *only has poly-logarithmic query complexity due to the poly-logarithmic number of rounds in [GKR15].*

1.3 Subsequent work and open problems

Prover Efficiency. A key bottleneck in leading modern implementations of succinct argument systems[18] is the overhead incurred by the prover. For IOP based arguments, the communication complexity is a lower bound on the IOP prover efficiency. Since we have managed to minimize the

[18] An argument system is a proof system that is sound against computationally bounded cheating provers.

communication complexity, it is tempting to ask whether it is possible to achieve *linear* overhead for the IOP prover.

In a sequence of follow up works [BCG20, BCL22, GLS+21, RR22], which all build on our code-switching technique, this goal has indeed been realized. In particular [RR22] constructs IOPs in which the prover can be implemented by a strictly linear-size Boolean circuit.

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 proof length approaching the witness length), 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.* [BFLR20] constructs a PCPP with length $n \cdot (\log n)^{o(1)}$ but for a very specific problem. In particular, 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.

A starting point may be to try to obtain PCPs for CircuitSAT of proof length approaching t with *any* non-trivial query complexity (we note that while [BKK+16] constructed a PCP for CircuitSAT of length $O(t)$ with non-trivial query complexity, the hidden constant in the $O(\cdot)$ notation is very large). A positive answer to this question would have to bypass the use of multiplication codes in a fundamentally different way than in our protocol (which capitalizes on interaction). Moreover, we note that for the case of linear length locally testable and correctable codes, a key for reducing the query complexity was indeed to first construct such codes of rate approaching 1 and non-trivial sublinear query complexity [BV15, Vid15, KSY14, KMRS17]. (And the case of obtaining linear length locally-testable codes with constant query complexity was recently settled [DEL+22, PK22].)

Hardness of approximation. A major application of PCPs is showing *hardness of approximation* for central optimization problems (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, a major barrier in applying the traditional PCP methodology for obtaining hardness of approximation results in *fine-grained complexity* is the overhead in the length of existing PCPs. For example, to show that GapSAT is $2^{(1-\varepsilon)\cdot n}$ hard based on SETH, one would need a PCP with overhead $1+\varepsilon'$. (In particular, note that a PCP of length, say, $100n$ would only roughly yield hardness of $2^{n/100}$.) Interestingly, recent breakthrough results bypassed this barrier by relying on interactive proof machinery [ARW17, Rub18, CGL+19]. It is natural to ask whether short IOPs such as those constructed in this work can be used to establish new or improved hardness of approximation results. We remark that (not necessarily short) IOPs have been used in the recent works of Arnon *et al.* [ACY22b, ACY22a] to derive hardness of approximation results.

1.4 Organization

We start with preliminaries in Section 2. In Section 3 we state our main results and also introduce the main definitions and lemmas that will be used in the proof. These lemmas are proved in Sections 4 to 6. In Section 7 we use the results obtained in the previous sections to construct our IOPP and in Section 8 we construct our IOP.

2 Preliminaries

We will often view a string $x \in \Sigma^n$, over an alphabet Σ, as a function $x : [n] \to \Sigma$. In particular, the i-th entry of x is denoted $x(i)$. 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 $\mathrm{dist}(x, S) = \min_{y \in S} \mathrm{dist}(x, y)$.

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 2.1. *We say that an ensemble of finite fields $\mathcal{F} = (\mathbb{F}_n)_{n \in \mathbb{N}}$ 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 2.2 (see [Sho88])**.** *For every $S = S(n) \geq 1$, there exists a constructible field ensemble $\mathcal{F} = (\mathbb{F}_n)_n$, where each $\mathbb{F}_n$ has scharacteristic 2 and size $O(S(n))$.*

2.1 Interactive oracle proofs

We next define the notion of *interactive oracle proof*, due to [BCS16, RRR21]. We will give a general definition that applies to *promise problems*, which will allow us to model settings in which the input has some particular structure (e.g., is encoded under an error-correcting code). As we will often be interested in verifiers that run in *sub-linear time*, we will also view the input as being separated into two parts $x = (x_{\mathrm{exp}}, x_{\mathrm{imp}})$, where the verifier explicitly reads the explicit input x_{exp}, but only has *oracle access* to the implicit input x_{imp} (we will sometimes consider languages in which either the explicit or the implicit inputs are empty). For simplicity, we restrict our attention to the public-coin setting which means that all of the verifier's messages simply consist of uniformly random coins.

Definition 2.3 (Interactive oracle proof (IOP))**.** *An ℓ-round (public coin) interactive oracle proof (IOP) with soundness error ε for a promise problem $(\mathsf{YES}, \mathsf{NO})$ is a pair $(\mathcal{P}, \mathcal{V})$ of probabilistic algorithms satisfying the following properties.*

- **Prover $\mathcal{P}$'s Input:** *$\mathcal{P}$ receives as input $x = (x_{\mathrm{exp}}, x_{\mathrm{imp}})$.*
- **Verifier $\mathcal{V}$'s Input:** *$\mathcal{V}$ receives as input x_{exp}, and is also given oracle access to x_{imp}.*
- **Communication phase:** *$\mathcal{P}$ and $\mathcal{V}$ interact for ℓ rounds, where $\mathcal{V}$'s messages are uniformly random strings $R_1, \ldots, R_\ell$.*
- **Query phase:** *At the end of the interaction, $\mathcal{V}$ makes (non-adaptive) queries to the implicit input x_{imp} and $\mathcal{P}$'s messages, where the location of the queries only depends on the explicit input x_{exp} and $\mathcal{V}$'s randomness string $R = (R_1, \ldots, R_\ell)$, and decides whether to accept or reject based on the value of these queries, the explicit input x_{exp}, and the randomness string R.*
- **Completeness:** *If $x \in \mathsf{YES}$, then when $\mathcal{V}$ interacts with $\mathcal{P}$, it accepts with probability 1.*

- **Soundness:** *If $x \in \mathsf{NO}$, then for any prover strategy $\mathcal{P}^*$, when $\mathcal{V}$ interacts with $\mathcal{P}^*$, it accepts with probability at most ε.*

The key parameters that we will care about are:

- **Round complexity:** the number of rounds ℓ.
- **Query Complexity:** the number of bits q that the verifier reads from the implicit input x_{imp} and $\mathcal{P}$'s messages.
- **Communication complexity:** the total length cc of $\mathcal{P}$'s messages.
- **Randomness complexity:** the total length r of $\mathcal{V}$'s randomness string $R = (R_1, \ldots, R_\ell)$.
- **Verifier running time:** the total time $T_{\mathcal{V}}$ it takes for $\mathcal{V}$ to generate its randomness string $R = (R_1, \ldots, R_\ell)$, compute the query locations, given the explicit input x_{exp} and the randomness string R, and decide whether to accept or reject based on the query values, the explicit input x_{exp}, and the randomness string R.
- **Prover running time:** the total time $T_{\mathcal{P}}$ it takes for $\mathcal{P}$ to compute its messages. In the context of interactive oracle proof for NP languages we will often assume that the prover is also given as an auxiliary input a witness w proving that the input x belongs to the language.

We note that the notion of PCP corresponds to the special case of IOP, where the round complexity is $\ell = 1$.

A particular special case of interest is that of IOP*s of proximity* [BCS16, RRR21], or IOPP for short. For a pair language $\mathcal{L} \subseteq \{(x_{\text{exp}}, x_{\text{imp}}) \in \{0,1\}^* \times \{0,1\}^*\}$ and $x_{\text{exp}} \in \{0,1\}^*$, we use the notation $\mathcal{L}_{x_{\text{exp}}} := \{x_{\text{imp}} \colon (x_{\text{exp}}, x_{\text{imp}}) \in \mathcal{L}\}$.

Definition 2.4 (Interactive oracle proof of proximity (IOPP))**.** *An ℓ-round IOP of α-proximity (α-IOPP) with soundness error ε for a pair language $\mathcal{L} \subseteq \{(x_{\text{exp}}, x_{\text{imp}}) \in \{0,1\}^* \times \{0,1\}^*\}$ is an ℓ-round IOP with soundness error ε for the promise problem $(\mathsf{YES}, \mathsf{NO})$, where $\mathsf{YES} = \mathcal{L}$ and $\mathsf{NO} = \{(x_{\text{exp}}, y) : y$ is α-far from $\mathcal{L}_{x_{\text{exp}}}\}$.*

We refer to α as the proximity parameter of the IOPP. If $\mathcal{L}$ has no explicit input, then an ℓ-round α-IOPP with soundness error ε for$\mathcal{L}$ is an ℓ-round IOP with soundness error ε for the promise problem $(\mathsf{YES}, \mathsf{NO})$, where $\mathsf{YES} = \mathcal{L}$ and $\mathsf{NO} = \{y : y$ is α-far from $\mathcal{L}\}$. Once more, we note that the notion of PCPP corresponds to the special case of IOPP, when the round complexity is $\ell = 1$.

2.1.1 Composition

A key benefit of IOPPs is that they facilitate *composition*. Specifically, one can compose an "outer" *robust* IOP which has small communication complexity but large query complexity, with an "inner" IOPP which has large communication complexity, but small query complexity, to obtain the best of both worlds: an IOP with small communication and query complexities. We begin with the definition of a robust IOP.

Definition 2.5 (Robustness)**.** *We say that an IOP $(\mathcal{P}, \mathcal{V})$ for a promise problem $(\mathsf{YES}, \mathsf{NO})$ is (α, ε)-robust if the soundness requirement is replaced with the following stronger requirement:*

- **Robustness:** *If $x \in \mathsf{NO}$, then for any prover strategy $\mathcal{P}^*$, when $\mathcal{V}$ interacts with $\mathcal{P}^*$, then the answers to the queries that $\mathcal{V}$ makes to the implicit input x_{imp} and $\mathcal{P}$'s messages are α-close to any view that would make $\mathcal{V}$ accept with probability at most ε.*

We refer to α as the Robustness parameter of the IOP. To also reduce the verifier running time in composition, it will also be useful that each query location in the outer IOP can be computed quickly, as per the following definition.

Definition 2.6. *We say that $\mathcal{V}$ is s-*succinct *if the location of each individual query to the implicit input x_{imp} and $\mathcal{P}$'s messages can be computed in time s, given the explicit input x_{exp} and $\mathcal{V}$'s randomness string $R = (R1, \ldots, R_\ell)$.*

Lemma 2.7. *(Composition) Suppose that the following hold:*

- **(Outer IOP:)** *Let $(\mathsf{YES}, \mathsf{NO})$ be a promise problem with implicit input length n. Suppose that there exists an ℓ-round (α, ε)-robust IOP $(\mathcal{P}, \mathcal{V})$ for $(\mathsf{YES}, \mathsf{NO})$ with communication complexity $\text{cc}(n)$, query complexity $q(n)$, randomness complexity $r(n)$, verifier running time $T_{\mathcal{V}}(n)$, and prover running time $T_{\mathcal{P}}(n)$. Suppose furthermore that $\mathcal{V}$ is $s(n)$-succinct.*
- **(Inner IOPP:)** *Let $\mathcal{L}_{(\mathcal{P},\mathcal{V})}$ denote the pair language consisting of all pairs $((x_{\text{exp}}, R), y)$, so that $\mathcal{V}$ accepts on explicit input x_{exp}, randomness string R, and query values y. Suppose that there exists an ℓ'-round α-IOPP $(\mathcal{P}', \mathcal{V}')$ with soundness error ε' for the pair language $\mathcal{L}_{(\mathcal{P},\mathcal{V})}$ with communication complexity $cc'(n')$, query complexity $q'(n')$, verifier running time $T_{\mathcal{V}'}(n')$, and prover running time $T_{\mathcal{P}'}(n')$, where n' denotes the implicit input length of $\mathcal{L}_{(\mathcal{P},\mathcal{V})}$.*

Then there exists an $(\ell + \ell')$-round IOP $(\mathcal{P}'', \mathcal{V}'')$with soundness error $\varepsilon + \varepsilon'$ for $(\mathsf{YES}, \mathsf{NO})$ with communication complexity $\text{cc}(n) + \text{cc}'(q(n))$, query complexity $q'(q(n))$, verifier running time $r(n) + T_{\mathcal{V}'}(q(n)) \cdot s(n)$, and prover running time $T_{\mathcal{P}}(n) + T_{\mathcal{V}}(n) + T_{\mathcal{P}'}(q(n))$. Moreover, the first (deterministic) message of $\mathcal{P}''$ is identical to that of $\mathcal{P}$.

The use of proof composition originates in [AS98], and is articulated as a composition of a robust PCP with a PCPP in [BGH+06, DR06]. The extension to IOPs is from [BCG+17]. As our setting is slightly different than that of [BCG+17] (in particular, we care about very small factors in the communication complexity), we provide a full proof of Lemma 2.7 in Appendix B.

Lastly, we shall make use of the following PCPP due to [Mie09] as the "inner IOPP" in our composition steps.

Theorem 2.8 ([Mie09, Theorem 1])**.** *Let $\mathcal{L}$ be a pair language decidable in time $T = T(m + n)$, where m, n are the explicit and implicit input lengths, respectively. Then for any constant $\alpha > 0$, there exists an α-PCPP with soundness error $\frac{1}{2}$ for $\mathcal{L}$ with length $\tilde{O}(T)$, constant query complexity, verifier running time $\text{poly}(m, \log n, \log(T))$, and prover running time $\text{poly}(m, n, T)$.*

Remark 2.9. *We remark that [Mie09, Theorem 1] does not explicitly state the prover's running time but it can be 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].*

Instantiating the inner IOPP in Lemma 2.7 with the PCPP given in Theorem 2.8, readily implies the following useful corollary.

Corollary 2.10. *The following holds for any constant $\alpha, \varepsilon' > 0$. Let $(\mathsf{YES}, \mathsf{NO})$ be a promise problem, where m, n are the explicit and implicit input lengths, respectively. Suppose that there exists an ℓ-round (α, ε)-robust IOP $(\mathcal{P}, \mathcal{V})$ for $(\mathsf{YES}, \mathsf{NO})$ with communication complexity $\mathrm{cc}(n)$, query complexity $q(n)$, randomness complexity $r(n)$, verifier running time $T_{\mathcal{V}}(n)$, and prover running time $T_{\mathcal{P}}(n)$. Suppose furthermore that $\mathcal{V}$ is $s(n)$-succinct.*

Then there exists an $(\ell+1)$-round IOP $(\mathcal{P}', \mathcal{V}')$ with soundness error $\varepsilon + \varepsilon'$ for $(\mathsf{YES}, \mathsf{NO})$ with communication complexity $\mathrm{cc}(n) + \tilde{O}(T_{\mathcal{V}}(n))$, constant query complexity, verifier running time $\mathrm{poly}\,(m, r(n), s(n), \log(T_{\mathcal{V}}(n)))$, and prover running time $T_{\mathcal{P}}(n) + \mathrm{poly}(m, T_{\mathcal{V}}(n))$. Moreover, the first (deterministic) message of $\mathcal{P}'$ is identical to that of $\mathcal{P}$.

2.2 Error-correcting codes

Let Σ be a finite alphabet, and n, n' be positive integers (the message length and the block length, respectively). An (error-correcting) code is an injective map $C : \Sigma^n \to \Sigma^{n'}$. The elements in the domain of C are called messages, and the elements in the image of C are called codewords. The rate of a code $C : \Sigma^n \to \Sigma^{n'}$ is the ratio $\rho := \frac{n}{n'}$. The relative distance $\mathrm{dist}(C)$ of C is the maximum $\delta > 0$ such that for every pair of distinct messages $x, y \in \Sigma^n$ it holds that $\mathrm{dist}(C(x), C(y)) \geq \delta$. We say that C is systematic if the message is a prefix of the corresponding codeword, i.e., for every $x \in \Sigma^n$ there exists $z \in \Sigma^{n'-n}$ such that $C(x) = (x, z)$. If $\Sigma = \mathbb{F}$ for some finite field $\mathbb{F}$, and C is a linear map between the vector spaces $\mathbb{F}^n$ and $\mathbb{F}^{n'}$ then we say that C is linear.

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 $\mathcal{C} = \{C_n : \Sigma^n \to \Sigma^{n'}\}_{n \in \mathcal{I}}$ is a countable collection of error correcting codes, one for each message length $n \in \mathcal{I}$. We say that $\mathcal{C}$ has rate $\rho \in (0, 1)$ and relative distance $\delta \in (0, 1)$ if for any $n \in \mathcal{I}$, the code C_n has rate at least ρ and relative distance at least δ. We say that $\mathcal{C}$ is T-time encodable if given $x \in \Sigma^n$, the codeword $C_n(x)$ can be generated in time $T(n)$. The code is linear-time encodable (resp., quasi-linear time encodable) if it is encodable in time $O(n)$ (resp., $\tilde{O}(n)$).

2.2.1 Tensor codes

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

Definition 2.11 (Tensor product code)**.** *The tensor (product) code of linear codes $C_1 : \mathbb{F}^{n_1} \to \mathbb{F}^{n_1'}$ and $C_2 : \mathbb{F}^{n_2} \to \mathbb{F}^{n_2'}$ is the code $C_1 \otimes C_2 : \mathbb{F}^{n_1 \times n_2} \to \mathbb{F}^{n_1' \times n_2'}$, where the encoding $(C_1 \otimes C_2)(M)$ of any message $M \in \mathbb{F}^{n_1 \times n_2}$ is obtained by first encoding each column of M with the code C_1, and then encoding each resulting row with the code C_2.*

Note that by linearity, the codewords of $C_1 \otimes C_2$ are $n_1' \times n_2'$ matrices (over the field $\mathbb{F}$) whose columns belong to the code C_1, and whose rows belong to the code C_2. It is also known that the converse is true: any $n_1' \times n_2'$ matrix, whose columns belong to the code C_1, and whose rows belong to the code C_2, is a codeword of $C_1 \otimes C_2$. Furthermore, swapping the order of encodings, encoding first each row of M with the code C_2, and then encoding each resulting column with the code C_1, results in the same codeword.

The following effects of the tensor product operation on the classical parameters of the code are well known.

Fact 2.12. *Suppose that $C_1 : \mathbb{F}^{n_1} \to \mathbb{F}^{n_1'}$, $C_2 : \mathbb{F}^{n_2} \to \mathbb{F}^{n_2'}$ are linear codes of rates ρ_1, ρ_2 and relative distances δ_1, δ_2 that can be encoded in times T_1, T_2, respectively. Then, the tensor code $C_1 \otimes C_2$ is a linear code of rate $\rho_1 \cdot \rho_2$ and relative distance $\delta_1 \cdot \delta_2$ that can be encoded in time $n_2' \cdot T_1 + n_1' \cdot T_2$.*

For a linear code $C : \mathbb{F}^n \to \mathbb{F}^{n'}$, let $C^{\otimes 1} := C$ and $C^{\otimes t} := C \otimes C^{\otimes(t-1)}$, for any $t \geq 2$. As in the 2-dimensional case, the codewords of $C^{\otimes t} : \mathbb{F}^{n^t} \to \mathbb{F}^{(n')^t}$ can be viewed as t-dimensional cubes, satisfying that their projection on any axis-parallel line is a codeword of C. Once more, we have that the converse is also true. Moreover, by Fact 2.12, if C is a linear code of rate ρ and relative distance δ that can be encoded in time T, then $C^{\otimes t}$ is a linear code of rate ρ^t and relative distance δ^t that can be encoded in time $t \cdot (n')^{t-1} \cdot T$.

2.3 Low degree extension

Let $\mathbb{F}$ be a finite field, $\mathbb{H} \subseteq \mathbb{F}$ 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 $|\mathbb{H}| - 1$ polynomial defined as:

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

The following fact can be verified by inspection.

Fact 2.13. *For every $h, h' \in \mathbb{H}^m$ it holds that $\hat{I}(h, h') = 1$ if $h = h'$ and $\hat{I}(h, h') = 0$ otherwise.*

Proposition 2.14. *For every function $\phi : \mathbb{H}^m \to \mathbb{F}$, there exists a unique extension of ϕ into an individual degree $|\mathbb{H}| - 1$ polynomial $\hat{\phi} : \mathbb{F}^m \to \mathbb{F}$, which agrees with ϕ on $\mathbb{H}^m$ (i.e., $\hat{\phi}|_{\mathbb{H}^m} \equiv \phi$).*

Proof. Consider the polynomial $\hat{\phi}(x) = \sum_{h \in \mathbb{H}^m} \phi(h) \cdot \hat{I}(x, h)$. The fact that $\hat{\phi}$ and ϕ agree on $\mathbb{H}^m$ follows from Fact 2.13. Uniqueness follows from the fact that any two individual degree $|\mathbb{H}| - 1$ polynomials that agree on $\mathbb{H}^m$, agree everywhere. □

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

2.4 The sumcheck protocol

We use (a slight variant of) the celebrated sumcheck protocol of [LFKN92] for verifying the sum of a low-degree polynomial Q 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 2.15 (The Sumcheck Protocol, see e.g., [RRR21, Lemma 7.1])**.** *Let $\mathcal{H} = (\mathbb{H}_n)_{n \in \mathbb{N}}$ and $\mathcal{F} = (\mathbb{F}_n)_{n \in \mathbb{N}}$ be constructible finite field ensembles, so that $\mathbb{F}_n$ is an extension field of $\mathbb{H}_n$ for any $n \in \mathbb{N}$. For every $m = m(n) \geq 1$ and round parameter $\ell = \ell(n) \geq 1$, where $\ell(n)$ divides $m(n)$ for any $n \in \mathbb{N}$, there exists an interactive protocol $(\mathcal{P}, \mathcal{V})$ satisfying the following properties:*

- **Prover $\mathcal{P}$'s Input:** *$\mathcal{P}$ receives as input a polynomial $Q \colon \mathbb{F}_n^m \to \mathbb{F}_n$ of individual degree $d = d(n)$, and a scalar $a \in \mathbb{F}_n$.*
- **Verifier $\mathcal{V}$'s Input:** *$\mathcal{V}$ receives as input $n \in \mathbb{N}$.*
- **Communication phase:** *$\mathcal{P}$ and $\mathcal{V}$ interact for ℓ rounds, where $\mathcal{V}$'s messages are uniformly random strings $R_1, \ldots, R_\ell$.*

- **Query phase:** *At the end of the interaction, $\mathcal{V}$ computes a point $z \in \mathbb{F}_n^m$, depending only on $\mathcal{V}$'s randomness string $R = (R_1, \ldots, R_\ell)$. It then reads the entire $\mathcal{P}$'s messages, and based on the content of these messages and the randomness string R, it either rejects, or outputs z and a scalar $b \in \mathbb{F}_n$.*

- **Completeness:** *If $\sum_{h \in \mathbb{H}_n^m} Q(h) = a$, then when $\mathcal{V}$ interacts with $\mathcal{P}$, with probability 1, the output of $\mathcal{V}$ is (z, b) such that $Q(z) = b$.*

- **Soundness:** *If $\sum_{h \in \mathbb{H}_n^m} Q(h) \neq a$, then for any prover strategy $\mathcal{P}^*$, when $\mathcal{V}$ interacts with $\mathcal{P}^*$, the probability that $\mathcal{V}$ outputs (z, b) such that $Q(z) = b$ is at most $\frac{d \cdot m}{|\mathbb{F}_n|}$.*

The communication complexity is $\ell \cdot (d+1)^{m/\ell} \cdot \log |\mathbb{F}_n| + m \cdot \log |\mathbb{F}_n|$. The verifier runs in time $\ell \cdot d^{O(m/\ell)} \cdot \text{polylog}(|\mathbb{F}_n|)$, and the prover runs in time $|\mathbb{H}_n|^m \cdot \ell \cdot d^{O(m/\ell)} \cdot \text{polylog}(|\mathbb{F}_n|)$.

3 Main results and claims

3.1 Formal statement of our results

Our main result, which is proved in Section 8, is an IOP for every NP relation that can be verified in polynomial time with bounded polynomial space, in which the communication complexity approaches the witness length.

Theorem 3.1 (IOP for NP)**.** *There exists a fixed absolute constant $\xi > 0$ such that the following holds. 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 $\text{poly}(n)$ with space n^ξ. Also, we assume that $m \geq n$ (i.e., instances are not shorter than their corresponding witnesses).*[19]

Then for any constant $\gamma, \varepsilon > 0$, there exists a constant round IOP with soundness error ε for $\mathcal{L}$ with communication complexity $(1+\gamma) \cdot n$ and constant query complexity. The verifier runs in time $\tilde{O}(m)$, and the prover runs in time $\text{poly}(m)$ (given a witness).

Our second main result, proved in Section 7, is an IOPP for bounded space computations in which the communication complexity is slightly less than the input length.

Theorem 3.2 (IOPP for bounded-space computations)**.** *For any constant $\beta > 0$, there exists a constant $\xi > 0$ such that the following holds. Let $\mathcal{L}$ be a language with no explicit input that can be decided in time $\text{poly}(n)$ with space n^ξ. Then for any constant $\gamma, \varepsilon, \alpha > 0$, there exists a constant round α-IOPP with soundness error ε for $\mathcal{L}$ with communication complexity $\gamma \cdot n$ and constant query complexity. The verifier runs in time n^β, and the prover runs in time $\text{poly}(n)$.*

3.2 Main claims

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

We say that a pair language $\mathcal{L}$ has implicit length sequence (resp., input length sequence) $\mathcal{I}$, for $\mathcal{I} \subseteq \mathbb{N}$, if for any $x = (x_{\text{exp}}, x_{\text{imp}}) \in \mathcal{L}$ it holds that $|x_{\text{imp}}| \in \mathcal{I}$ (resp., $|x| \in \mathcal{I}$). For an integer $t > 1$,

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

we let $\mathcal{I}_t = \{i^t \mid i \in \mathbb{N}\}$, that is, $\mathcal{I}_t$ contains all integers that are t-th powers. For a finite field $\mathbb{F}$ and for t vectors $\lambda_1, \ldots, \lambda_t \in \mathbb{F}^n$, we denote by $\lambda_1 \otimes \cdots \otimes \lambda_t \in \mathbb{F}^{n^t}$ the t-dimensional tensor satisfying that $(\lambda_1 \otimes \cdots \otimes \lambda_t)(\bar{i}) = \lambda_1(i_1) \cdots \lambda_t(i_t)$ for any $\bar{i} = (i_1, \ldots, i_t) \in [n]^t$.

We start by defining the notion of a *tensor interactive reduction*, which is an interactive protocol for a pair language $\mathcal{L}$, whose output is t vectors $\lambda_1, \ldots, \lambda_t$ and a scalar b, where the condition $\langle \lambda_1 \otimes \cdots \otimes \lambda_t, x_{imp} \rangle = b$ is interpreted as accepting, and $\langle \lambda_1 \otimes \cdots \otimes \lambda_t, , x_{imp} \rangle \neq b$ is interpreted as rejecting.

Definition 3.3 (Tensor interactive reduction)**.** *Let $t > 1$ be an integer, let $\mathcal{L}$ be a pair language with implicit length sequence $\mathcal{I}_t$, and let $\mathcal{F} = \{\mathbb{F}_n\}_{n \in \mathcal{I}_t}$ be a constructible ensemble of finite fields of characteristic 2. An ℓ-*round *t-*dimensional tensor interactive reduction with soundness error *ε for $\mathcal{L}$* over *$\mathcal{F}$ *is an interactive protocol $(\mathcal{P}, \mathcal{V})$ satisfying the following properties:*

- **Prover $\mathcal{P}$'s Input:** *$\mathcal{P}$ receives as input $x = (x_{\text{exp}}, x_{\text{imp}})$.*
- **Verifier $\mathcal{V}$'s Input:** *$\mathcal{V}$ receives as input x_{exp}.*
- **Communication phase:** *$\mathcal{P}$ and $\mathcal{V}$ interact for ℓ rounds, where $\mathcal{V}$'s messages are uniformly random strings $R_1, \ldots, R_\ell$.*
- **Query phase:** *At the end of the interaction, $\mathcal{V}$ computes t vectors $\lambda_1, \ldots, \lambda_t \in (\mathbb{F}_n)^{n^{1/t}}$, depending only on the explicit input x_{exp} and $\mathcal{V}$'s randomness string $R = (R_1, \ldots, R_\ell)$, where n is the implicit input length. It then reads the entire $\mathcal{P}$'s messages, and based on the content of these messages, the explicit input x_{exp}, and the randomness string R, it either rejects or outputs $\lambda_1, \ldots, \lambda_t$, and a scalar $b \in \mathbb{F}_n$.*
- **Completeness:** *If $x \in \mathcal{L}$, then when $\mathcal{V}$ interacts with $\mathcal{P}$, with probability 1, the output of $\mathcal{V}$ is $(\lambda_1, \ldots, \lambda_t, b)$ such that $\langle \lambda_1 \otimes \cdots \otimes \lambda_t, x_{\text{imp}} \rangle = b$.*
- **Soundness:** *If $x \notin \mathcal{L}$, then for any prover strategy $\mathcal{P}^*$, when $\mathcal{V}$ interacts with $\mathcal{P}^*$, the probability that $\mathcal{V}$ outputs $(\lambda_1, \ldots, \lambda_t, b)$ such that $\langle \lambda_1 \otimes \cdots \otimes \lambda_t, x_{\text{imp}} \rangle = b$ is at most ε.*

We note that we allow the verifier in the tensor interactive reduction to read all of the prover's messages. This is simply because the messages will be quite short (and later on we will even further reduce the query complexity via composition).

For an integer $t > 1$, a code ensemble $\mathcal{C} = \{C_n : \{0,1\}^n \to \{0,1\}^{n'}\}_{n \in \mathcal{I}_t}$, and an ensemble $\mathcal{F} = \{\mathbb{F}_n\}_{n \in \mathcal{I}_t}$ of finite fields of characteristic 2, we let $\mathcal{L}(\mathcal{C}, \mathcal{F}, t)$ denote the pair language

$$\mathcal{L}(\mathcal{C}, \mathcal{F}, t) := \left\{ \Big(\big(\lambda_1, \ldots, \lambda_t, b\big), w \Big) : \exists x \in \{0,1\}^n \text{ s.t. } w = C_n(x) \text{ and } \langle \lambda_1 \otimes \cdots \otimes \lambda_t, x \rangle = b \right\}, \quad (3)$$

where $\lambda_1, \ldots, \lambda_t \in (\mathbb{F}_n)^{n^{1/t}}$ and $b \in \mathbb{F}_n$ are the explicit input, and $w \in \{0,1\}^{n'}$ is the implicit input.

The next lemma, proved in Section 4, shows how to combine a t-dimensional tensor interactive reduction for a language $\mathcal{L}$, together with an IOPP for $\mathcal{L}(\mathcal{C}, \mathcal{F}, t)$, to obtain an IOPP for $\mathcal{L}$.

Lemma 3.4 (Code switching)**.** *Let $t > 1$ be an integer, let $\mathcal{L}$ be a pair language with implicit length sequence $\mathcal{I}_t$, and let $\mathcal{F} = \{\mathbb{F}_n\}_{n \in \mathcal{I}_t}$ be a constructible ensemble of finite fields of characteristic 2. Let $\mathcal{C} = \{C_n : \{0,1\}^n \to \{0,1\}^{n'}\}_{n \in \mathcal{I}_t}$ be a systematic linear code ensemble of rate $1-\gamma$ and relative distance δ that can be encoded in time T. Suppose that the following exist:*

- *An ℓ-round t-dimensional tensor interactive reduction $(\mathcal{P}, \mathcal{V})$ with soundness error ε for $\mathcal{L}$ over $\mathcal{F}$ with communication complexity $\mathrm{cc}(n)$, verifier running time $T_{\mathcal{V}}(n)$, and prover running time $T_{\mathcal{P}}(n)$, where n is the implicit input length of $\mathcal{L}$.*
- *An ℓ'-round α-IOPP $(\mathcal{P}', \mathcal{V}')$, for $\alpha < \frac{\delta}{2}$, with soundness error ε' for $\mathcal{L}(\mathcal{C}, \mathcal{F}, t)$ as given in (3) with communication complexity $\mathrm{cc}'(n')$, query complexity $q'(n')$, verifier running time $T_{\mathcal{V}'}(n')$, and prover running time $T_{\mathcal{P}'}(n')$, where n' is the implicit input length of $\mathcal{L}(\mathcal{C}, \mathcal{F}, t)$. Suppose furthermore that the query locations only depend on the verifier's randomness string.*

Then, there exists an $(\ell + \ell')$-round $\frac{\alpha}{1-\gamma}$-IOPP $(\mathcal{P}'', \mathcal{V}'')$ with soundness error $\varepsilon + \varepsilon'$ for $\mathcal{L}$ with communication complexity $\frac{\gamma}{1-\gamma} \cdot n + \mathrm{cc}(n) + \mathrm{cc}'(n')$, query complexity $cc(n) + q'(n')$, verifier running time $T_{\mathcal{V}}(n) + T_{\mathcal{V}'}(n')$, and prover running time $T(n) + T_{\mathcal{P}}(n) + T_{\mathcal{V}}(n) + T_{\mathcal{P}'}(n')$. Moreover, the communication phase consists of a single message of $\mathcal{P}''$ which is the non-systematic part of $C_n(x_{\mathrm{imp}})$ (i.e., of length $\frac{\gamma}{1-\gamma} \cdot n$), followed by $\mathrm{cc}(n) + \mathrm{cc}'(n')$ additional communication, and the total number of queries made by $\mathcal{V}''$ to the implicit input and the first message of $\mathcal{P}''$ is the same as the number of queries made by $\mathcal{V}'$ to the implicit input.

The following lemma, proved in Section 5, establishes the existence of a tensor interactive reduction for a large class of languages.

Lemma 3.5 (Tensor interactive reduction)**.** *The following holds for any constant $t > 1$. Let $\mathcal{L}$ be a language with no explicit input and with input length sequence $\mathcal{I}_t$ that can be decided in time $\mathrm{poly}(n)$ with space $s = s(n) \geq \log n$. Let $\mathcal{F} = \{\mathbb{F}_n\}_{n \in \mathcal{I}_t}$ be a constructible ensemble of finite fields of characteristic 2, where $\mathrm{polylog}(n) \leq |\mathbb{F}_n| \leq \mathrm{poly}(n)$ for any $n \in \mathcal{I}_t$.*

Then for any constant $\beta > 0$, there exists a constant round t-dimensional tensor interactive reduction with soundness error $\frac{1}{2}$ for $\mathcal{L}$ over $\mathcal{F}$ with communication complexity and verifier running time $n^{\beta} \cdot \mathrm{poly}(s)$ and prover running time $\mathrm{poly}(n)$.

Lastly, the following lemma, proved in Section 6, states the existence of a code ensemble $\mathcal{C}$ for which the language $\mathcal{L}(\mathcal{C}, \mathcal{F}, t)$ given in (3) has an efficient IOPP.

Lemma 3.6 (IOPP for $\mathcal{L}(\mathcal{C}, \mathcal{F}, t)$)**.** *The following holds for any constant integer $t > 1$ and $\delta > 0$. Let $\mathcal{B} = \{B_n : \{0,1\}^n \to \{0,1\}^{n'}\}_{n \in \mathbb{N}}$ be a systematic linear code ensemble of relative distance δ that can be encoded in quasi-linear time, and let $\mathcal{C} = \{C_n = (B_{n^{1/t}})^{\otimes t}\}_{n \in \mathcal{I}_t}$. Let $\mathcal{F} = \{\mathbb{F}_n\}_{n \in \mathcal{I}_t}$ be a constructible ensemble of finite fields of characteristic 2, where $|\mathbb{F}_n| \leq \mathrm{poly}(n)$ for any $n \in \mathcal{I}_t$.*

Then for any constant $\alpha > 0$, there exists a constant round α-IOPP with soundness error $\frac{1}{2}$ for $\mathcal{L}(\mathcal{C}, \mathcal{F}, t)$ as given in (3) with communication complexity, query complexity, and verifier running time $\tilde{O}(n^{2/t})$ and prover running time $\mathrm{poly}(n)$. Moreover, the verifier makes a constant number of queries to the implicit input, where the location of each of these queries only depends on its randomness string.

4 Code switching

In this section we prove the Code Switching Lemma (Lemma 3.4). The IOPP $(\mathcal{P}'', \mathcal{V}'')$ for $\mathcal{L}$ is given in Fig. 1 below.

IOPP $(\mathcal{P}'', \mathcal{V}'')$ **for** $\mathcal{L}$ **on input** $x = (x_{\text{exp}}, x_{\text{imp}})$**:**

Communication phase:

1. $\mathcal{P}''$ sends the non-systematic part z of $C_n(x_{\text{imp}})$ of length $\frac{\gamma}{1-\gamma} \cdot n$.
2. $\mathcal{P}''$ and $\mathcal{V}''$ interact for ℓ rounds according to the communication phase of the tensor interactive reduction $(\mathcal{P}, \mathcal{V})$ for $\mathcal{L}$ over $\mathcal{F}$ on explicit input x_{exp} and implicit input x_{imp}; Let $R = (R_1, \ldots, R_\ell)$ denote $\mathcal{V}$'s randomness string.
3. $\mathcal{P}''$ computes the t vectors $\lambda_1, \ldots, \lambda_t \in (\mathbb{F}_n)^{n^{1/t}}$ and the scalar $b \in \mathbb{F}_n$, based on the explicit input x_{exp}, the randomness string R, and the messages of $\mathcal{P}$.
4. $\mathcal{P}''$ and $\mathcal{V}''$ interact for ℓ' rounds according to the communication phase of the IOPP $(\mathcal{P}', \mathcal{V}')$ for $\mathcal{L}(\mathcal{C}, \mathcal{F}, t)$ on explicit input $(\lambda_1, \ldots, \lambda_t, b)$ and implicit input (x_{imp}, z).

Query phase:

1. $\mathcal{V}''$ reads all of the messages of $\mathcal{P}$; If $\mathcal{V}$ rejects, then $\mathcal{V}''$ rejects and aborts. Otherwise, $\mathcal{V}''$ computes $(\lambda_1, \ldots, \lambda_t, b)$ based on the explicit input x_{exp}, $\mathcal{V}$'s randomness string R, and the messages of $\mathcal{P}$.
2. $\mathcal{V}''$ makes q' queries to the implicit input x_{imp}, to z, and to the messages of $\mathcal{P}'$ according to the query phase of the IOPP $(\mathcal{P}', \mathcal{V}')$ (noting that the location of these queries only depends on $\mathcal{V}$'s randomness string), and accepts if and only if $\mathcal{V}'$ accepts.

Figure 1: IOPP $(\mathcal{P}'', \mathcal{V}'')$ for $\mathcal{L}$

It can be verified that the round complexity, communication complexity, query complexity, verifier running time, and prover running time are all as claimed, and that the requirements in the "moreover" part are satisfied. Next we show completeness and soundness.

Completeness. Suppose that $x = (x_{\text{exp}}, x_{\text{imp}}) \in \mathcal{L}$. Then, by the completeness property of the protocol $(\mathcal{P}, \mathcal{V})$, with probability 1, $\mathcal{V}$ outputs $(\lambda_1, \ldots, \lambda_t, b)$ satisfying $\langle \lambda_1 \otimes \cdots \otimes \lambda_t, x_{\text{imp}} \rangle = b$. Consequently, $((\lambda_1, \ldots, \lambda_t, b), (x_{\text{imp}}, z)) = ((\lambda_1, \ldots, \lambda_t, b), C_n(x_{\text{imp}})) \in \mathcal{L}(\mathcal{C}, \mathcal{F}, t)$, and $\mathcal{V}'$ will accept with probability 1. We conclude that in this case $\mathcal{V}''$ accepts with probability 1.

Soundness. Suppose that x_{imp} is $\frac{\alpha}{1-\gamma}$-far from $\mathcal{L}_{x_{\text{exp}}}$. 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 $x = (x_{\text{exp}}, x_{\text{imp}})$) by $\tilde{z}$.

Suppose first that $(x_{\text{imp}}, \tilde{z})$ is α-far from the code C_n. In this case, we also have that $(x_{\text{imp}}, \tilde{z})$ is α-far from $(\mathcal{L}(\mathcal{C}, \mathcal{F}, t))_{(\lambda_1, \ldots, \lambda_t, b)}$, and so $\mathcal{V}'$ will reject with probability at least $1 - \varepsilon'$, in which case $\mathcal{V}''$ will also reject. We conclude that in the case that $(x_{\text{imp}}, \tilde{z})$ is α-far from the code C_n, $\mathcal{V}''$ will reject with probability at least $1 - \varepsilon'$.

Next assume that $(x_{\text{imp}}, \tilde{z})$ is α-close to some codeword $C_n(y)$. First observe that our assumption that $\text{dist}((x_{\text{imp}}, \tilde{z}), C_n(y)) \leq \alpha$ implies that $\text{dist}(x_{\text{imp}}, y) \leq \frac{\alpha \cdot n'}{n} = \frac{\alpha}{1-\gamma}$. By assumption that x_{imp} is $\frac{\alpha}{1-\gamma}$-far from $\mathcal{L}_{x_{\text{exp}}}$, this implies in turn that $y \notin \mathcal{L}_{x_{\text{exp}}}$, so $(x_{\text{exp}}, y) \notin \mathcal{L}$. Consequently, with probability at least $1 - \varepsilon$, $\mathcal{V}$ will either reject (in which case $\mathcal{V}''$ will also reject), or output

$(\lambda_1, \ldots, \lambda_t, b)$ satisfying $\langle \lambda_1 \otimes \cdots \otimes \lambda_t, y \rangle \neq b$.

We claim that in the latter case $(x_{\text{imp}}, \tilde{z})$ is α-far from $(\mathcal{L}(\mathcal{C}, \mathcal{F}, t))_{(\lambda_1,\ldots,\lambda_t,b)}$, and so $\mathcal{V}'$ will reject with probability at least $1-\varepsilon'$, in which case $\mathcal{V}''$ will also reject. To see this, suppose in contradiction that $(x_{\text{imp}}, \tilde{z})$ is α-close to some codeword $C_n(y')$ with $\langle \lambda_1 \otimes \cdots \otimes \lambda_t, y' \rangle = b$. Then we have that $C_n(y)$ and $C_n(y')$ are distinct codewords (since $\langle \lambda_1 \otimes \cdots \otimes \lambda_t, y \rangle \neq b$ but $\langle \lambda_1 \otimes \cdots \otimes \lambda_t, y' \rangle = b$) that are both of distance at most α from $(x_{\text{imp}}, \tilde{z})$. By the triangle inequality and our assumption that $\alpha < \frac{\delta}{2}$ this implies in turn that $\text{dist}(C_n(y), C_n(y')) \leq 2\alpha < \delta$, a contradiction.

We conclude that in the case that $(x_{\text{imp}}, \tilde{z})$ is α-close to the code C_n, $\mathcal{V}''$ will reject with probability at least $(1-\varepsilon) \cdot (1-\varepsilon') \geq 1 - \varepsilon - \varepsilon'$.

5 Tensor interactive reduction

In this section we prove Lemma 3.5. The proof is largely based on a result of Reingold *et al.* [RRR21].

Theorem 5.1 (Doubly efficient interactive proofs for bounded space computations [RRR21, see Corollary 9 and Theorem 10])**.** *Let $\mathcal{L}$ be a language that can be decided in time $\text{poly}(n)$ with space $s = s(n) \geq \log n$. Let $\mathcal{F} = (\mathbb{F}_n)_{n\in\mathbb{N}}$ and $\mathcal{H} = (\mathbb{H}_n)_{n\in\mathbb{N}}$ be constructible finite field ensembles such that $\mathbb{H}_n \subseteq \mathbb{F}_n$, $|\mathbb{H}_n| = \Omega(\log n)$, and $\text{poly}(|\mathbb{H}_n|) \leq |\mathbb{F}_n| \leq \text{poly}(n)$ for any $n \in \mathbb{N}$.*

Then for any constant $\beta > 0$, there exists a constant round public-coin interactive proof with soundness error $\frac{1}{2}$ for $\mathcal{L}$ with communication complexity $n^\beta \cdot \text{poly}(s)$, verifier running time $\tilde{O}(n) + n^\beta \cdot \text{poly}(s)$, and prover running time $\text{poly}(n)$. Moreover, the verifier runs in time $n^\beta \cdot \text{poly}(s)$ given only oracle access to the low degree extension $\hat{x}$ of the input $x \in \{0,1\}^n$ relative to $\mathbb{F}_n$, $\mathbb{H}_n$ and dimension $m = \log_{|\mathbb{H}_n|}(n)$, where the location of the verifier queries to $\hat{x}$ only depends on its randomness string.

Throughout this section we refer to the protocol of Theorem 5.1 as the RRR protocol. At a high level, we would like to prove Lemma 3.5 by simply running the RRR protocol wrt to $\mathcal{L}$. At first glance this seems problematic since the RRR verifier needs to get x as its input, whereas our reduction does not have any access whatsoever to x.

To overcome this difficulty we rely on the furthermore clause of Theorem 5.1, which states that all that the RRR verifier needs is oracle access to the low degree extension of x. 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 x 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 3.5.

The main issue that we encounter when trying to implement this high-level approach is that the verifer in Lemma 3.5 should output a *single* claim about the low degree extension of x, whereas the RRR verifier may make many queries to $\hat{x}$ (which corresponds to generating *many* linear claims). We show how to handle this issue, and generically reduce the number of queries to the low degree extension (via interaction), in Section 5.1 below. Then, in Section 5.2 we use this to derive Lemma 3.5.

5.1 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 q to the low degree extension code (originating in [KR08, GKR15]) is to take a low degree curve passing through all the q points (as well as some additional points), ask the prover for the evaluation on *all* points on the curve and (1) check that the provided values are consistent with a low degree polynomial and (2) check the correctness on a single *random* point. To make this approach work, one needs to use a large field $\mathbb{F}$ of cardinality $|\mathbb{F}| \geq q$. 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., [RRR21]) is to use a high dimensional (low degree) manifold rather than a (univariate) curve. Unfortunately, this idea seems to introduce a quadratic dependence on q (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 5.2. *Let $\mathcal{F} = (\mathbb{F}_n)_{n\in\mathbb{N}}$ and $\mathcal{H} = (\mathbb{H}_n)_{n\in\mathbb{N}}$ be constructible finite field ensembles so that $\mathbb{H}_n \subseteq \mathbb{F}_n$, $|\mathbb{H}_n| \leq \mathrm{polylog}(n)$, and $|\mathbb{F}_n| \leq \mathrm{poly}(n)$ for any $n \in \mathbb{N}$. Let $(\mathcal{P}, \mathcal{V})$ be a public-coin interactive proof with soundness error ε for a language $\mathcal{L}$, in which $\mathcal{V}$ only makes $q(n)$ queries to the low degree extension $\hat{x}$ of the input $x \in \{0,1\}^n$ relative to $\mathbb{F}_n$, $\mathbb{H}_n$, and dimension $m = \log_{|\mathbb{H}_n|}(n)$, where the location of the verifier queries to $\hat{x}$ only depends on its randomness string.*

Then there exists a public-coin interactive proof $(\mathcal{P}', \mathcal{V}')$ with soundness error $\varepsilon + O(m \cdot |\mathbb{H}_n|/|\mathbb{F}_n|)$ for $\mathcal{L}$, in which $\mathcal{V}'$ similarly receives oracle access to $\hat{x}$ but only makes a single query to $\hat{x}$, where the location of this query only depends on its randomness string. For every parameter $\ell \in [m]$, we can implement the foregoing interactive proof with an additional ℓ rounds of communication, additive overhead of $q(n) \cdot \log n + \ell \cdot n^{O(1/\ell)}$ to the communication complexity, $q(n) \cdot \mathrm{polylog}(n) + \ell \cdot n^{O(1/\ell)}$ verification time overhead, and $\mathrm{poly}(n)$ prover time overhead.

Proof. Let $x \in \{0,1\}^n$ be an input to the protocol, and let $\mathbb{F} := \mathbb{F}_n$, $\mathbb{H} := \mathbb{H}_n$, and $q := q(n)$. 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 relative to $\mathbb{F}$, $\mathbb{H}$, 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 2.3).

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 $\leq q$ queries to $\hat{x}$. Let $\{i_1, \ldots, i_q\} \subseteq \mathbb{F}^m$ be the locations that $\mathcal{V}$ wants to query (i.e., $\mathcal{V}$ wants to obtain the values $\hat{x}(i_j)$ for all $j \in [q]$). The prover $\mathcal{P}'$ sends to the verifier $\mathcal{V}'$ a list of alleged values for all these points. That is, values $(y_1, \ldots, y_q)$ which the prover claims are equal to $(\hat{x}(i_1), \ldots, \hat{x}(i_q))$.

Given $(y_1, \ldots, y_q)$, 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

$$\forall j \in [q],\ y_j = \hat{x}(i_j). \tag{4}$$

To do so, $\mathcal{V}'$ selects at random $r_1, \ldots, r_q \in \mathbb{F}$. Observe that if Eq. (4) holds, then $\sum_j r_j y_j = \sum_j r_j \hat{x}(i_j)$. Otherwise however (i.e., if Eq. (4) does not hold) then, with probability $1 - 1/|\mathbb{F}|$, it holds that $\sum_j r_j y_j \neq \sum_j r_j \hat{x}(i_j)$. In other words, after choosing $r_1, \ldots, r_q$, it suffices for $\mathcal{V}'$ to check

the identity

$$\sum_j r_j y_j = \sum_j r_j \hat{x}(i_j). \tag{5}$$

By definition of the low degree extension (see Proposition 2.14), the RHS of Eq. (5) can be expressed as $\sum_j r_j \sum_{h \in \mathbb{H}^m} \hat{x}(h) \cdot \hat{I}(i_j, h)$ (see Section 2.3 for the definition of $\hat{I}$ and more background on the low degree extension). Define $Q(\lambda) = \hat{x}(\lambda) \cdot \left(\sum_j r_j \cdot \hat{I}(i_j, \lambda)\right)$. Observe that Q is an individual degree $2(|\mathbb{H}| - 1)$ polynomial in λ and that Eq. (5) 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 2.15) 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, given in Fig. 2 below.

Protocol $(\mathcal{P}', \mathcal{V}')$ for $\mathcal{L}$ on input x:

Communication phase:

1. $\mathcal{P}'$ and $\mathcal{V}'$ emulate the communication phase of the protocol $(\mathcal{P}, \mathcal{V})$ for $\mathcal{L}$ on input x; Let $R = (R_1, \ldots, R_{\ell'})$ denote $\mathcal{V}$'s randomness string.
2. $\mathcal{P}'$ computes the query set $I = \{i_1, \ldots, i_q\}$, based on the randomness string R, and sends values $(y_1, \ldots, y_q)$ such that $y_j = \hat{x}(i_j)$ for any $j \in [q]$.
3. $\mathcal{V}'$ chooses at random $r_1, \ldots, r_q \in \mathbb{F}$ and sends them to $\mathcal{P}'$.
4. $\mathcal{P}'$ and $\mathcal{V}'$ run the sumcheck protocol (see Lemma 2.15) wrt the claim $\sum_{h \in \mathbb{H}^m} Q(h) = \sum_j r_j y_j$, where $Q(\lambda) = \hat{x}(\lambda) \cdot \left(\sum_j r_j \cdot \hat{I}(i_j, \lambda)\right)$.

Query phase:

1. Let $\tilde{y}_1, \ldots, \tilde{y}_q$ be the values that are sent by the prover on Step 2. $\mathcal{V}'$ checks that $\mathcal{V}$ accepts given the answer sequence $\tilde{y}_1, \ldots, \tilde{y}_q$ and $\mathcal{P}$'s messages; If not, it rejects and aborts.
2. If the sumcheck verifier rejects on Step 4, then $\mathcal{V}'$ rejects and aborts; Otherwise, let $z \in \mathbb{F}^m$ and $b \in \mathbb{F}$ be the output of the sumcheck verifier on Step 4. $\mathcal{V}'$ computes $\hat{x}(z) \cdot \sum_j r_j \cdot \hat{I}(i_j, z)$ using a single query to $\hat{x}$ (noting that z only depends on $\mathcal{V}'$'s randomness string), and accepts if and only if this value equals to b.

Figure 2: Protocol $(\mathcal{P}', \mathcal{V}')$ for $\mathcal{L}$

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

Completeness. Suppose that $x \in \mathcal{L}$. By completeness of $(\mathcal{P}, \mathcal{V})$, on Step 1 $\mathcal{P}'$ generates queries $i_1, \ldots, i_q \in \mathbb{F}^m$ such that $\mathcal{V}$ accepts given the answer sequence $\{y_j = \hat{x}(i_j)\}_j$. Observe that

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

and so by the completeness of Lemma 2.15, the sumcheck verifier outputs a claim (z, b) such that indeed $Q(z) = b$. Thus, the verifier accepts when checking that $b = \hat{x}(z) \cdot \sum_j r_j \cdot \hat{I}(i_j, z)$.

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

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

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 $b \in \mathbb{F}$ such that $Q(z) \neq b$. Assuming the latter, $\mathcal{V}'$ rejects when checking that $b = \hat{x}(z) \cdot \sum_j r_j \cdot \hat{I}(i_j, z) = Q(z)$. Thus, overall, the verifier accepts with probability at most $\varepsilon + O(m \cdot |\mathbb{H}|/|\mathbb{F}|)$. □

5.2 Proof of Lemma 3.5

Next we prove Lemma 3.5, based on the above Theorem 5.1 and Lemma 5.2.

Let $\mathcal{F} = (\mathbb{F}_n)_{n \in \mathbb{N}}$ and $\mathcal{H} = (\mathbb{H}_n)_{n \in \mathbb{N}}$ be constructible finite field ensembles of characteristic 2 such that $\mathbb{H}_n \subseteq \mathbb{F}_n$, $|\mathbb{H}_n| = \Theta(\log n)$, and $\mathrm{poly}(|\mathbb{H}_n|) \leq |\mathbb{F}_n| \leq \mathrm{poly}(n)$ for any $n \in \mathbb{N}$. Let $m = \log_{|\mathbb{H}_n|}(n)$. We choose $\mathbb{H}_n$ so that m is an integer divisible by t.

By Theorem 5.1, there exists a constant-round public-coin interactive proof $(\mathcal{P}, \mathcal{V})$ with soundness error $1/2$ for $\mathcal{L}$ with communication complexity $n^\beta \cdot \mathrm{poly}(s)$ and prover running time $\mathrm{poly}(n)$. By the moreover part of Theorem 5.1, the verifier $\mathcal{V}$ runs in time $n^\beta \cdot \mathrm{poly}(s)$ given oracle access to the low degree extension of the input x (relative to $\mathbb{F}_n$, $\mathbb{H}_n$ and m). Moreover, the location of the verifier queries to $\hat{x}$ only depends on its randomness string. We repeat this protocol a constant number of times (in parallel) to reduce the soundness error to $\frac{1}{4}$.

At this point we apply the transformation of Lemma 5.2 to $(\mathcal{P}, \mathcal{V})$ to obtain an interactive proof with soundness error $\frac{1}{4} + O(m \cdot |\mathbb{H}_n|/|\mathbb{F}_n|) \leq \frac{1}{2}$ in which the verifier only needs to make a single query to the low degree extension of x, where the location of this query only depends on its randomness string. 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 x. Moreover, for a sufficiently

large constant ℓ, the resulting protocol has asymptotically the same communication complexity and verifier and prover running times.

Let $(z, b) \in \mathbb{F}_n^m \times \mathbb{F}_n$ be the claim generated by the verifier. That is, the verifier would like to check that $\hat{x}(z) = b$ (where $\hat{x} : \mathbb{F}_n^m \to \mathbb{F}_n$ is the low degree extension of x). Using Proposition 2.14, this precisely corresponds to $\sum_{h \in \mathbb{H}_n^m} x(h) \cdot \hat{I}(z, h) = b$, a linear claim about x. In more detail, consider a vector $\lambda \in \mathbb{F}_n^n$ defined as $\lambda(h) = \hat{I}(z, h)$ for every $h \in \mathbb{H}_n^m$ (recall that we associate the set $[n]$ with $\mathbb{H}_n^m$). Then the claim that the verifier generates is that $\langle \lambda, x \rangle = b$. Observe further that since $\hat{I}(z, h) = \prod_{i=1}^m \hat{I}(z_i, h_i)$, and m is divisible by t, it holds that

$$\lambda = \lambda_1 \otimes \cdots \otimes \lambda_m = \lambda'_1 \otimes \cdots \otimes \lambda'_t,$$

where $\lambda_i(g) = \hat{I}(z_i, g)$ for $g \in \mathbb{H}_n$, and $\lambda'_i = \lambda_{(i-1)\cdot\frac{m}{t}+1} \otimes \cdots \otimes \lambda_{i\cdot\frac{m}{t}}$ for any $i = 1, \ldots, t$. Finally, note that $\lambda'_1, \ldots, \lambda'_t$ only depend on the verifier's randomness string.

6 IOPP for $\mathcal{L}(\mathcal{C}, \mathcal{F}, t)$

In this section we prove Lemma 3.6, showing the existence of an efficient IOPP for the language $\mathcal{L}(\mathcal{C}, \mathcal{F}, t)$, as given in Eq. (3) above. The IOPP relies on known properties of tensor codes such as the sumcheck protocol [LFKN92, Mei13], local testing [BV15, Vid15], and a relaxed local correcting procedure [GRR18].

We start with the following lemma, which provides a sumcheck protocol for rank 1 tensor coefficients 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 to the nearest codeword) procedures for tensor codes.

Lemma 6.1 (Sumcheck protocol for rank 1 tensor coefficients)**.** *The following holds for any integer $t > 1$ and $\delta > 0$. Let $\mathcal{C} = \{C_n : \{0,1\}^n \to \{0,1\}^{n'}\}_{n \in \mathbb{N}}$ be a systematic linear code ensemble of relative distance δ that can be encoded in time $T = T(n)$, and let $\mathcal{F} = \{\mathbb{F}_n\}_{n \in \mathbb{N}}$ be a constructible finite field ensemble of characteristic 2. Let $(\mathsf{YES}, \mathsf{NO})$ be the following promise problem:*

$$\mathsf{YES} = \left\{((\lambda_1, \ldots, \lambda_t, b), C_n^{\otimes t}(x)) \ : \ \langle \lambda_1 \otimes \cdots \otimes \lambda_t, x \rangle = b \right\},$$
$$\mathsf{NO} = \left\{((\lambda_1, \ldots, \lambda_t, b), C_n^{\otimes t}(x)) \ : \ \langle \lambda_1 \otimes \cdots \otimes \lambda_t, x \rangle \neq b \right\},$$

where $\lambda_1, \ldots, \lambda_t \in \mathbb{F}_n^n$ and $b \in \mathbb{F}_n$ are the explicit input, and $C_n^{\otimes t}(x) \in \{0,1\}^{[n']^t}$ is the implicit input, where $x \in \{0,1\}^{[n]^t}$.

Then there exists a t-round IOP with soundness error $1 - \delta^t$ for the promise problem $(\mathsf{YES}, \mathsf{NO})$ with communication complexity $t \cdot n' \cdot \log(|\mathbb{F}_n|)$, query complexity $t \cdot n' \cdot \log(|\mathbb{F}_n|) + 1$, verifier running time $t \cdot T(n) \cdot \mathrm{polylog}(|\mathbb{F}_n|)$, and prover running time $t^2 \cdot n' \cdot n^{t-1} \cdot \mathrm{polylog}(|\mathbb{F}_n|)$. Moreover, the verifier makes a single query to the implicit input, where the location of this query only depends on its randomness string.

We prove the above Lemma 6.1 in Section 6.1. The next lemma gives an IOPP for verifying membership in a tensor code.

Lemma 6.2. *(Local testing protocol for tensor codes). The following holds for any constant integer $t > 1$ and $\delta > 0$. Let $\mathcal{C} = \{C_n : \{0,1\}^n \to \{0,1\}^{n'}\}_n$ be a systematic linear code ensemble of relative distance δ that can be encoded in time $T = T(n)$. Then for any constant $\alpha > 0$, there exists a 2-round α-IOPP with soundness error $\frac{1}{2}$ for the language $\mathcal{L} = \bigcup_n C_n^{\otimes t}$ (where we view the input to $\mathcal{L}$ as the implicit input and there is no explicit input) with communication complexity $n' \cdot \tilde{O}(T(n))$, constant query complexity, verifier running time $\mathrm{polylog}(T(n))$, and prover running time $\mathrm{poly}(T(n))$.*

Lemma 6.2 follows by composing the local tester for tensor codes of [Vid15] with an inner PCPP to reduce the query complexity. For completeness, we provide a full proof of this lemma in Appendix C.1. Finally, the following lemma gives a relaxed local correction protocol for tensor codes.

Lemma 6.3. *(Relaxed local correction protocol for tensor codes). The following holds for any constant integer $t > 1$ and $\delta > 0$. Let $\mathcal{C} = \{C_n : \{0,1\}^n \to \{0,1\}^{n'}\}_n$ be a systematic linear code ensemble of relative distance δ that can be encoded in time $T = T(n)$. Let $(\mathsf{YES}, \mathsf{NO})$ be the following promise problem:*

$$\mathsf{YES} = \left\{(\bar{i}, w) \ : \ w \in C_n^{\otimes t}\right\},$$
$$\mathsf{NO} = \left\{(\bar{i}, w) \ : \ \mathrm{dist}(w, c) < \left(\frac{\delta}{4}\right)^t \text{ for } c \in C_n^{\otimes t} \text{ with } w(\bar{i}) \neq c(\bar{i})\right\},$$

where $\bar{i} \in [n']^t$ is the explicit input, and $w \in \{0,1\}^{[n']^t}$ is the implicit input.

Then there exists a 2-round IOP with soundness error $\frac{1}{2}$ for the promise problem $(\mathsf{YES}, \mathsf{NO})$ with communication complexity $\tilde{O}(T(n))$, constant query complexity, verifier running time $\mathrm{polylog}(T(n))$, and prover running time $\mathrm{poly}(T(n))$.

Lemma 6.3 follows by composing a robust version of the relaxed local corrector for tensor codes of [GRR18] with an inner PCPP to reduce the query complexity. For completeness, we provide a full proof of this lemma in Appendix C.2.

Next we prove Lemma 3.6, based on the above Lemmas 6.1 to 6.3.

Proof of Lemma 3.6. On explicit input $\lambda_1, \ldots, \lambda_t \in (\mathbb{F}_n)^{n^{1/t}}$ and $b \in \mathbb{F}_n$ and implicit input $w \in \{0,1\}^{n'}$, the prover and the verifier run the sumcheck protocol for rank 1 tensor coefficients given by Lemma 6.1 for the base code $B_{n^{1/t}}$ (recalling that $C_n = (B_{n^{1/t}})^{\otimes t}$), suppose that this protocol queries the $\bar{i}$'th entry of w. The prover and the verifier then run the local testing protocol given by Lemma 6.2 for the base code $B_{n^{1/t}}$ and proximity parameter $\alpha' := \min\{\alpha, \left(\frac{\delta}{4}\right)^t\}$ on input w, and the relaxed local correction protocol given by Lemma 6.3 for the base code $B_{n^{1/t}}$ on explicit input $\bar{i}$ and implicit input w. If the verifier in any of the executions rejects, the verifier rejects, otherwise it accepts.

It can be readily verified that the round complexity, communication complexity, query complexity, verifier running time, and prover running time are all as claimed, and that the requirements in the "moreover" part are satisfied. Completeness is also straightforward. Next we show soundness.

To this end, suppose that $w \in \{0,1\}^{n'}$ is α-far from $\mathcal{L}(\mathcal{C}, \mathcal{F}, t)_{(\lambda_1,\ldots,\lambda_t,b)}$. If w is α'-far from the code $C_n = (B_{n^{1/t}})^{\otimes t}$, then the verifier in the local testing protocol will reject with probability at least $\frac{1}{2}$. Hence, we may assume that w is α'-close to a codeword $c = C_n(x)$, where $\alpha' = \min\{\alpha, \left(\frac{\delta}{4}\right)^t\}$. By assumption that w is α-far from $\mathcal{L}(\mathcal{C}, \mathcal{F}, t)_{(\lambda_1,\ldots,\lambda_t,b)}$, this implies in turn that $\langle \lambda_1 \otimes \cdots \lambda_t, x \rangle \neq b$.

Moreover, if $w(\bar{i}) \neq c(\bar{i})$, then the verifier in the relaxed local correction protocol will reject with probability at least $\frac{1}{2}$, and so we may assume that $w(\bar{i}) = c(\bar{i})$.

But this implies in turn that the sumcheck protocol is exectued on the codeword $c = C_n(x)$ which satisfies that $\langle \lambda_1 \otimes \cdots \otimes \lambda_t, x \rangle \neq b$, and consequently the verifier will reject with probability at least δ^t. Finally, the rejection probability can be increased to $\frac{1}{2}$ by executing the protocol independently $O(1/\delta^t)$ times, and accepting if and only if all invocations accepts. As t and δ are constant, this will only incur a constant multiplicative overhead in the communication complexity, query complexity, verifier running time, and prover running time. □

6.1 Sumcheck for rank 1 tensor coefficients

In this section we prove Lemma 6.1, giving a sumcheck protocol for rank 1 tensor coefficients. The proof 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 coefficients to lie in an extension field of the field over which our code is defined.

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

The sumcheck protocol for rank 1 tensor coefficients is given in Fig. 3 below.

It can be verified that the round complexity, communication complexity, query complexity, verifier running time, and prover running time are all as claimed (the verifier running time follows since checking whether a given string $y \in \{0,1\}^{n'}$ is a codeword of C_n can be done by encoding the systematic part of y via the code C_n, and checking whether the resulting codeword equals y), and that the requirements in the 'moreover' part are satisfied. Next we show completeness and soundness.

We first note the following.

Claim 6.4. *For any $\ell = 1, \ldots, t-1$, we have that*

$$w_\ell(r_\ell) = \sum_{j \in [n]} \lambda_{\ell+1}(j) \cdot w_{\ell+1}(j).$$

Proof. By Eq. (6) we have that,

$$\begin{aligned}
\sum_{j \in [n]} \lambda_{\ell+1}(j) \cdot w_{\ell+1}(j) &= \sum_{j \in [n]} \lambda_{\ell+1}(j) \sum_{i_{\ell+2}, \ldots, i_t \in [n]} \lambda_{\ell+2}(i_{\ell+2}) \cdots \lambda_t(i_t) \cdot c(r_1, \ldots, r_\ell, j, i_{\ell+2}, \ldots, i_t) \\
&= \sum_{i_{\ell+1}, \ldots, i_t \in [n]} \lambda_{\ell+1}(i_{\ell+1}) \cdots \lambda_t(i_t) \cdot c(r_1, \ldots, r_\ell, i_{\ell+1}, \ldots, i_t) \\
&= w_\ell(r_\ell).
\end{aligned}$$

□

Sumcheck

- ***Explicit input:*** $\lambda_1, \ldots, \lambda_t \in \mathbb{F}_n^n$ and $b \in \mathbb{F}_n$.
- ***Implicit input:*** A codeword $c = C_n^{\otimes t}(x) \in \{0,1\}^{[n']^t}$, where $x \in \{0,1\}^{[n]^t}$.

Communication phase:

For $\ell = 1, ..., t$:

1. Consider the string $w_\ell \in \mathbb{F}_n^{n'}$ defined as:

$$w_\ell(j) = \sum_{i_{\ell+1},\ldots,i_t \in [n]} \lambda_{\ell+1}(i_{\ell+1}) \cdots \lambda_t(i_t) \cdot c(r_1, \ldots, r_{\ell-1}, j, i_{\ell+1}, \ldots, i_t), \tag{6}$$

   for every $j \in [n']$. The prover computes w_ℓ and sends it to the verifier.

2. The verifier sends a uniform random $r_\ell \in [n']$.

Query phase:

1. The verifier sets $b_0 \leftarrow b$.
2. For $\ell = 1, ..., t$:
   (a) Let $\tilde{w}_\ell$ be the string sent by the prover, which is allegedly equal to w_ℓ. The verifier checks that $\tilde{w}_\ell|_a$ is a codeword of C_n for any $a \in A_n$ and that $\sum_{j \in [n]} \lambda_\ell(j) \cdot \tilde{w}_\ell(j) = b_{\ell-1}$; If not, then it rejects and aborts.
   (b) The verifier sets $b_\ell \leftarrow \tilde{w}_\ell(r_\ell)$.
3. The verifier explicitly queries $c(r_1, \ldots, r_t)$, and accepts if and only if $c(r_1, \ldots, r_t) = b_t$.

Figure 3: Sumcheck protocol for rank 1 tensor coefficients

Completeness. Completeness relies on the following claim.

Claim 6.5. *Suppose that $\langle \lambda_1 \otimes \cdots \otimes \lambda_t, x \rangle = b$. Then when $\mathcal{V}$ interacts with $\mathcal{P}$, for any $\ell = 1, \ldots, t$ it holds that*

$$\sum_{j \in [n]} \lambda_\ell(j) \cdot w_\ell(j) = b_{\ell-1}.$$

Proof. For $\ell = 1$, we have that

$$\begin{aligned}
\sum_{j \in [n]} \lambda_1(j) \cdot w_1(j) &= \sum_{j \in [n]} \lambda_1(j) \sum_{i_2,\ldots,i_t \in [n]} \lambda_2(i_2) \cdots \lambda_t(i_t) \cdot c(j, i_2, \ldots, i_t) \\
&= \sum_{i_1,\ldots,i_t \in [n]} \lambda_1(i_1) \cdots \lambda_t(i_t) \cdot c(i_1, \ldots, i_t) \\
&= \langle \lambda_1 \otimes \cdots \otimes \lambda_t, x \rangle \\
&= b_0.
\end{aligned}$$

For $\ell = 2, \ldots, t$, by Claim 6.4, we have that

$$\sum_{j \in [n]} \lambda_\ell(j) \cdot w_\ell(j) = w_{\ell-1}(r_{\ell-1}) = b_{\ell-1}.$$

□

Next assume that $\langle \lambda_1 \otimes \cdots \otimes \lambda_t, x \rangle = b$. We first claim that for any $\ell = 1, \ldots, t$, $w_\ell|_a$ is a codeword of C_n for any $a \in A_n$. To see this, note that by the properties of tensor codes, for any fixed $i_{\ell+1}, \ldots, i_t \in [n]$ we have that $c(r_1, \ldots, r_{\ell-1}, \cdot, i_{\ell+1}, \ldots, i_t)$ is a codeword of C. Consequently,

$$w_\ell = \sum_{i_{\ell+1}, \ldots, i_t \in [n]} \lambda_{\ell+1}(i_{\ell+1}) \cdots \lambda_t(i_t) \cdot c(r_1, \ldots, r_{\ell-1}, \cdot, i_{\ell+1}, \ldots, i_t)$$

is a linear combination of codewords of C_n with coefficients in $\mathbb{F}_n$. By linearity, this implies in turn that $w_\ell|_a$ is a codeword of C_n for any $a \in A_n$. Moreover, by Claim 6.5, we have that $\sum_{j \in [n]} \lambda_\ell(j) \cdot w_\ell(j) = b_{\ell-1}$ for any $\ell = 1, \ldots, t$. We conclude that all verifier's checks on Step 2a will pass with probability 1.

Moreover, we clearly have that $c(r_1, \ldots, r_t) = w_t(r_t) = b_t$, and consequently the verifier's check on Step 3 will pass with probability 1 as well. We conclude that all tests will pass with probability 1, and so the verifier accepts with probability 1.

Soundness. Fix a deterministic prover strategy $\mathcal{P}^*$. Soundness relies on the following claims.

Claim 6.6. *Suppose that in some round $\ell \in \{1, \ldots, t-1\}$, $\sum_{j \in [n]} \lambda_\ell(j) \cdot w_\ell(j) \neq b_{\ell-1}$, and the verifier's checks on Step 2a pass. Then with probability at least δ over the choice of r_ℓ it holds that $\sum_{j \in [n]} \lambda_{\ell+1}(j) \cdot w_{\ell+1}(j) \neq b_\ell$.*

Proof. By our assumption that the verifier's checks on Step 2a pass, we have that $\sum_{j \in [n]} \lambda_\ell(j) \cdot \tilde{w}_\ell(j) = b_{\ell-1}$. By our assumption that $\sum_{j \in [n]} \lambda_\ell(j) \cdot w_\ell(j) \neq b_\ell$, this implies in turn that $w_\ell \neq \tilde{w}_\ell$, and so $w_\ell|_a \neq \tilde{w}_\ell|_a$ for some $a \in A_n$. Moreover, we have that both $w_\ell|_a$ and $\tilde{w}_\ell|_a$ are codewords of C_n, and since C_n has relative distance δ, these two codewords differ on at least a δ-fraction of the coordinates. But this means that w_ℓ and $\tilde{w}_\ell$ differ on at least a δ-fraction of the coordinates, and so with probability at least δ over the choice of r_ℓ it holds that $w_\ell(r_\ell) \neq \tilde{w}_\ell(r_\ell)$. Recalling that on the one hand $w_\ell(r_\ell) = \sum_{j \in [n]} \lambda_{\ell+1}(j) \cdot w_{\ell+1}(j)$ by Claim 6.4, and on the other hand $\tilde{w}_\ell(r_\ell) = b_\ell$ by Step 2b, we conclude that in this case $\sum_{j \in [n]} \lambda_{\ell+1}(j) \cdot w_{\ell+1}(j) \neq b_\ell$. So it holds that $\sum_{j \in [n]} \lambda_{\ell+1}(j) \cdot w_{\ell+1}(j) \neq b_\ell$ with probability at least δ over the choice of r_ℓ, as stated. □

Next assume that $\langle \lambda_1 \otimes \cdots \otimes \lambda_t, x \rangle \neq b$, we shall show that the verifier rejects with probability at least δ^t. First, we may assume that all verifier's check on Step 2a pass, otherwise the verifier clearly rejects. First note that by our assumption that $\langle \lambda_1 \otimes \cdots \otimes \lambda_t, x \rangle \neq b$, we have that

$$\begin{aligned}
\sum_{j \in [n]} \lambda_1(j) \cdot w_1(j) &= \sum_{j \in [n]} \lambda_1(j) \sum_{i_2, \ldots, i_t \in [n]} \lambda_2(i_2) \cdots \lambda_t(i_t) \cdot c(j, i_2, \ldots, i_t) \\
&= \sum_{i_1, \ldots, i_t \in [n]} \lambda_1(i_1) \cdots \lambda_t(i_t) \cdot c(i_1, \ldots, i_t) \\
&= \langle \lambda_1 \otimes \cdots \otimes \lambda_t, x \rangle \\
&\neq b_0.
\end{aligned}$$

Consequently, by Claim 6.6, assuming that all verifier's checks on Step 2a pass, with probability at least δ over the choice of r_1 we have that $\sum_{j\in[n]} \lambda_2(j)\cdot w_2(j) \neq b_1$. If this is the case, then with probability at least δ over the choice of r_2, we have that $\sum_{j\in[n]} \lambda_3(j)\cdot w_3(j) \neq b_2$. Continuing this way, we have that $\sum_{j\in[n]} \lambda_t(j)\cdot w_t(j) \neq b_{t-1}$ with probability at least δ^{t-1}.

Assuming this, and by assumption that $\sum_{j\in[n]} \lambda_t(j)\cdot \tilde{w}_t(j) = b_{t-1}$, the above gives that $w_t \neq \tilde{w}_t$. As before, this implies in turn that w_t and $\tilde{w}_t$ differ on at least a δ-fraction of the coordinates, and so with probability at least δ over the choice of r_t we have that $w_t(r_t) \neq \tilde{w}_t(r_t)$. Recalling that $w_t(r_t) = c(r_1,\ldots,r_t)$ and $b_t = \tilde{w}_t(r_t)$, we conclude that in this case the verifier will reject on Step 3 with probability at least δ over the choice of r_t.

We conclude that in the case when $\langle \lambda_1 \otimes \cdots \otimes \lambda_t, x\rangle \neq b$, the verifier rejects with probability at least δ^t.

7 IOPP for bounded space computation

In this section, building on the results established in Sections 4 to 6, we construct a short IOPP for bounded space languages, thereby proving Theorem 3.2. Actually, we will prove the following more general lemma, from which Theorem 3.2 immediately follows. The additional technical details in this lemma will be useful for us in Section 8 when we construct a (short) IOP for NP.

Lemma 7.1. *For any constant $\beta > 0$, there exists an integer $t > 1$ so that the following holds for any constant $\gamma, \delta > 0$. Let $\mathcal{L}$ be a language with no explicit input and with input length sequence $\mathcal{I}_t$ that can be decided in time $\mathrm{poly}(n)$ with space $s = s(n) \geq \log n$. Let $\mathcal{B} = \{B_n : \{0,1\}^n \to \{0,1\}^{n'}\}_{n\in\mathbb{N}}$ be a systematic linear code ensemble of rate $1-\gamma$ and relative distance δ that can be encoded in quasi-linear time, and let $\mathcal{C} = \{C_n = (B_{n^{1/t}})^{\otimes t}\}_{n\in\mathcal{I}_t}$.*

Then for any constant $\alpha > 0$, there exists a constant round α-IOPP with soundness error $\frac{1}{2}$ for $\mathcal{L}$ with communication complexity $(\frac{1}{(1-\gamma)^t} - 1)\cdot n + n^\beta \cdot \mathrm{poly}(s)$, constant query complexity, verifier running time $n^\beta \cdot \mathrm{poly}(s)$, and prover running time $\mathrm{poly}(n)$. Moreover, the communication phase consists of a single prover message which is the non-systematic part of $C_n(x)$ of length $(\frac{1}{(1-\gamma)^t} - 1)\cdot n$, followed by $n^\beta \cdot \mathrm{poly}(s)$ additional communication.

Proof of Theorem 3.2. Let $\xi = \xi(\beta) > 0$ be a sufficiently small constant to be determined later on, and let $t := \lfloor 2/\xi \rfloor + 1$. Let $\bar{\mathcal{L}}$ be the language obtained by padding each input belonging to $\mathcal{L}$ of length n with $\bar{n}-n$ zeros, where $\bar{n} := \left(\lceil n^{1/t} \rceil\right)^t \leq (1+o(1))\cdot n$. Let $\mathcal{B} = \{B_n : \{0,1\}^n \to \{0,1\}^{n'}\}_{n\in\mathbb{N}}$ be any systematic linear code ensemble of rate $\left(\frac{1}{1+\gamma/2}\right)^{1/t}$ and constant relative distance that can be encoded in quasi-linear time.

Applying Lemma 7.1 on the language $\bar{\mathcal{L}}$ and the code ensemble $\mathcal{B}$ gives a constant round α-IOPP with soundness error $\frac{1}{2}$ for $\mathcal{L}$ with communication complexity $\frac{\gamma}{2}\cdot\bar{n} + n^{O(\xi)}$, constant query complexity, verifier running time $n^{O(\xi)}$, and prover running time $\mathrm{poly}(n)$. Moreover, the communication phase consists of a single prover message which is the non-systematic part of $C_{\bar{n}}(x \circ 0^{\bar{n}-n})$ of length $\frac{\gamma}{2}\cdot\bar{n}$, followed by $n^{O(\xi)}$ additional communication.

Theorem 3.2 then follows by letting $\xi > 0$ be a sufficiently small constant depending on β, and noting that the soundness error can be reduced to any constant $\varepsilon > 0$ by repeating the protocol independently a constant number of times, and accepting if and only if all invocations accept. Since the first prover message is deterministic, one does not need to repeat this message, and therefore the protocol has the same asymptotic guarantees. □

To prove the above Lemma 7.1, we first use Lemmas 3.4 to 3.6, established in Sections 4 to 6, to construct a constant-round short *robust* IOPP for bounded space languages with *sublinear* query complexity. Later we shall use Corollary 2.10 to reduce the query complexity to a constant by composition.

Lemma 7.2. *The following holds for any constant $\beta, \gamma, \delta > 0$ and integer $t > \lfloor 2/\beta \rfloor$. Let $\mathcal{L}$ be a language with no explicit input and with input length sequence $\mathcal{I}_t$ that can be decided in time $\mathrm{poly}(n)$ with space $s = s(n) \geq \log n$. Let $\mathcal{B} = \{B_n : \{0,1\}^n \to \{0,1\}^{n'}\}_{n \in \mathbb{N}}$ be a systematic linear code ensemble of rate $1-\gamma$ and relative distance δ that can be encoded in quasi-linear time, and let $\mathcal{C} = \{C_n = (B_{n^{1/t}})^{\otimes t}\}_{n \in \mathcal{I}_t}$.*

Then for any constant $\alpha > 0$, there exists a constant round $(\Omega(1), \frac{1}{2})$-robust α-IOPP for $\mathcal{L}$ with communication complexity $(\frac{1}{(1-\gamma)^t} - 1) \cdot n + n^\beta \cdot \mathrm{poly}(s)$, query complexity and verifier running time $n^\beta \cdot \mathrm{poly}(s)$, and prover running time $\mathrm{poly}(n)$. Moreover, the communication phase consists of a single prover message which is the non-systematic part of $C_n(x)$ of length $(\frac{1}{(1-\gamma)^t} - 1) \cdot n$, followed by $n^\beta \cdot \mathrm{poly}(s)$ additional communication.

To prove the above lemma, we shall also need the following lemma which shows how to turn a constant round IOP with a certain structure into a robust IOP.

Lemma 7.3. *Let $(\mathcal{P}, \mathcal{V})$ be a constant round IOP with soundness error ε for a promise problem $(\mathsf{YES}, \mathsf{NO})$ with no explicit input, satisfying the following properties:*

- *The communication phase consists of a single prover message of length cc, followed by cc' additional communication.*
- *The verifier makes a constant number of queries to the input and the first prover message, and reads all the rest of the prover's messages of total length cc'.*

Suppose furthermore that $(\mathcal{P}, \mathcal{V})$ has verifier running time $T_{\mathcal{V}}$ and prover running time $T_{\mathcal{P}}$.

Then there exists a constant round $(\Omega(1), \varepsilon)$-robust IOP $(\mathcal{P}', \mathcal{V}')$ for $(\mathsf{YES}, \mathsf{NO})$ with communication complexity $\mathrm{cc} + O(\mathrm{cc}')$, query complexity $O(\mathrm{cc}')$, verifier running time $T_{\mathcal{V}} + O(\mathrm{cc}')$, and prover running time $T_{\mathcal{P}} + O(\mathrm{cc}')$. Moreover, the first message of $\mathcal{P}'$ is identical to that of $\mathcal{P}$, and the rest of the communication has length $O(\mathrm{cc}')$.

We prove the above lemma in Section 7.1. We now turn to the proof of Lemma 7.2.

Proof of Lemma 7.2. Let $\mathcal{F} = \{\mathbb{F}_n\}_{n \in \mathcal{I}_t}$ be a constructible ensemble of finite fields of characteristic 2, where each $\mathbb{F}_n$ has size $\mathrm{polylog}(n)$. By Lemma 3.5, there exists a constant round t-dimensional tensor interactive reduction with soundness error $\frac{1}{2}$ for $\mathcal{L}$ over $\mathcal{F}$ with communication complexity and verifier running time $n^\beta \cdot \mathrm{poly}(s)$ and prover running time $\mathrm{poly}(n)$.

Let $\alpha' := \min\{\alpha \cdot (1-\gamma)^t, \frac{\delta^t}{2}\}$. By Lemma 3.6, there exists a constant round α'-IOPP with soundness error $\frac{1}{2}$ for $\mathcal{L}(\mathcal{C}, \mathcal{F}, t)$ with communication, query complexity, and verifier running time $\tilde{O}(n^{2/t})$ and prover running time $\mathrm{poly}(n)$. Moreover, the verifier makes a constant number of queries to the implicit input, where the location of each of these queries only depends on its randomness string.

We further note that the soundness error in both protocols can be reduced to $\frac{1}{4}$ by repeating the protocols independently a constant number of times, while incurring only a constant multiplicative overhead to all parameters.

Lemma 3.4 implies in turn the existence of a constant round α-IOPP with soundness error $\frac{1}{2}$ for $\mathcal{L}$ with communication complexity $(\frac{1}{(1-\gamma)^t}-1)\cdot n + n^\beta \cdot \mathrm{poly}(s) + \tilde{O}(n^{2/t})$, query complexity and verifier running time $n^\beta \cdot \mathrm{poly}(s) + \tilde{O}(n^{2/t})$, and prover running time $\mathrm{poly}(n)$. Moreover, the communication phase consists of a single prover message which is the non-systematic part of $C_n(x)$ of length $(\frac{1}{(1-\gamma)^t}-1)\cdot n$, followed by $n^\beta \cdot \mathrm{poly}(s) + \tilde{O}(n^{2/t})$ additional communication, and the verifier makes a constant number of queries to both the input and the first prover message. We may further assume that the verifier reads all prover messages, except for the first message, as this does not change the asymptotic query complexity of the verifier.

Finally, Lemma 7.3 implies the existence of a constant round $(\Omega(1), \frac{1}{2})$-robust α-IOPP for $\mathcal{L}$ with communication complexity $(\frac{1}{(1-\gamma)^t}-1)\cdot n + n^\beta \cdot \mathrm{poly}(s) + \tilde{O}(n^{2/t})$, query complexity and verifier running time $n^\beta \cdot \mathrm{poly}(s) + \tilde{O}(n^{2/t})$, and prover running time $\mathrm{poly}(n)$. Moreover, the communication phase consists of a single prover message which is the non-systematic part of $C_n(x)$ of length $(\frac{1}{(1-\gamma)^t}-1)\cdot n$, followed by $n^\beta \cdot \mathrm{poly}(s) + \tilde{O}(n^{2/t})$ additional communication. Lemma 7.2 then follows by our assumption that $t > \lfloor 2/\beta \rfloor$. □

Next we prove Lemma 7.1, based on Lemma 7.2 and Corollary 2.10.

Proof of Lemma 7.1. We apply Corollary 2.10 on the robust IOPP given by Lemma 7.2 to reduce the query complexity.

In more detail, let $\beta' = \beta'(\beta) \in (0, \beta)$ be a sufficiently small constant to be determined later on, and let $t := \lfloor 2/\beta' \rfloor + 1$. By Lemma 7.2, there exists a constant round $(\Omega(1), \frac{1}{2})$-robust α-IOPP for $\mathcal{L}$ with communication complexity $(\frac{1}{(1-\gamma)^t}-1)\cdot n + n^{\beta'} \cdot \mathrm{poly}(s)$, query complexity and verifier running time $n^{\beta'} \cdot \mathrm{poly}(s)$, and prover running time $\mathrm{poly}(n)$. Moreover, the communication phase consists of a single prover message which is the non-systematic part of $C_n(x)$ of length $(\frac{1}{(1-\gamma)^t}-1)\cdot n$, followed by $n^{\beta'} \cdot \mathrm{poly}(s)$ additional communication.

Applying Corollary 2.10 on the resulting IOPP gives a constant round α-IOPP with soundness error $\frac{3}{4}$ for $\mathcal{L}$ with communication complexity $(\frac{1}{(1-\gamma)^t}-1)\cdot n + n^{\beta'} \cdot \mathrm{poly}(s)$, constant query complexity, verifier running time $n^{O(\beta')} \cdot \mathrm{poly}(s)$, and prover running time $\mathrm{poly}(n)$. Moreover, the communication phase consists of a single prover message which is the non-systematic part of $C_n(x)$ of length $(\frac{1}{(1-\gamma)^t}-1)\cdot n$, followed by $n^{\beta'} \cdot \mathrm{poly}(s)$ additional communication. Lemma 7.1 then follows by letting β' be a sufficiently small constant, depending on β, and noting that the soundness error can be reduced to $\frac{1}{2}$ by repeating the protocol independently a constant number of times. Since the first prover message is deterministic, one does not need to repeat this message, and so the protocol has the same asymptotic guarantees. □

7.1 Robustness

In this section we prove Lemma 7.3, which shows how to turn a constant round IOP with a certain structure into a robust IOPP.

Let $\ell = O(1)$ denote the number of rounds in the protocol $(\mathcal{P}, \mathcal{V})$, and let $m_1, \ldots, m_\ell$ denote the prover messages. Let $q = O(1)$ denote the total number of queries made by the verifier to the input x and the message m_1. Let E be the systematic linear-time encodable code ensemble of [Spi96] of constant rate $\rho > 0$ and constant relative distance $\delta > 0$.

The protocol $(\mathcal{P}', \mathcal{V}')$ is obtained from $(\mathcal{P}, \mathcal{V})$ via the following modifications. First, for simplicity, assume that m_i has length exactly cc' for any $i > 1$, which can be achieved by padding zeros to

shorter messages. In the communication phase, in the first round $\mathcal{P}'$ sends $w_1 = m_1$, and in each round $i = 2, \ldots, \ell$, $\mathcal{P}'$ first encodes $m_i \in \{0,1\}^{cc'}$ via the code E, and then sends the resulting codeword $w_i = E(m_i) \in \{0,1\}^{\mathrm{cc}'/\rho}$ to $\mathcal{V}'$. Let $\tilde{w}_1, \ldots, \tilde{w}_\ell$ denote the messages of $\mathcal{P}'$ in the modified protocol, which are allegedly equal to $w_1, \ldots, w_\ell$.

In the query phase, $\mathcal{V}'$ first makes q queries to the input x and to $\tilde{w}_1$ according to the query phase of the protocol $(\mathcal{P}, \mathcal{V})$, where each of the q queries is repeated for cc'/ρ times; If for any of these queries, the answers to the repeated queries are not identical, then $\mathcal{V}'$ rejects and aborts (these seemingly redundant queries are needed to guarantee robustness). Then $\mathcal{V}'$ reads $\tilde{w}_i$ for each $i = 2, \ldots, \ell$, and checks that it is a codeword of E (which can be done by encoding the systematic part of $\tilde{w}_i$ via the code E, and checking that the resulting codeword is identical to $\tilde{w}_i$); If any of these messages is not a codeword of E, then $\mathcal{V}'$ rejects and aborts. Finally, $\mathcal{V}'$ accepts if and only if $\mathcal{V}$ accepts when given the values of the q queries made to the input x and to $\tilde{w}_1$, and the systematic part of $\tilde{w}_i$ for $i = 2, \ldots, \ell$.

It can be verified that the round complexity, communication complexity, query complexity, verifier running time, and prover running time of the protocol $(\mathcal{P}', \mathcal{V}')$ are all as claimed, and that the requirements in the 'moreover' part are satisfied. Completeness is also straightforward. Next we show that the protocol $(\mathcal{P}', \mathcal{V}')$ is (α, ε)-robust for $\alpha := \frac{\delta}{2(\ell+q)} = \Omega(1)$. To this end, assume that $x \in \mathsf{NO}$, and recall that the verifier's view consists of the values $a_1, \ldots, a_q$ of the q queries made to the input x and to $\tilde{w}_1$, where each such query is repeated cc'/ρ times, followed by all messages $\tilde{w}_i \in \{0,1\}^{\mathrm{cc}'/\rho}$ for $i = 2, \ldots, \ell$. For $i = 2, \ldots, \ell$, let $c_i = E(\tilde{m}_i)$ be the codeword that is closest to $\tilde{w}_i$.

Next assume that $v = ((u_i)_{i\in[q]}, (v_i)_{i=2,\ldots,\ell})$ is a view that is α-close to the verifier's view, where $u_i, v_i \in \{0,1\}^{\mathrm{cc}'/\rho}$ for all i. If there exists $i \in [q]$ so that the entries in u_i are non-identical then the verifier clearly rejects. Hence we may assume that for any $i \in [q]$ all values in u_i are identical. Moreover, by assumption that v is α-close to the verifier's view, we must have that for any $i \in [q]$ all entries in u_i are equal to a_i. Similarly, if there exists $i \in \{2, \ldots, \ell\}$ so that v_i is not a codeword of E, then the verifier clearly rejects. Hence we may assume that all v_i are codewords of E. Moreover, by assumption that v is α-close to the verifier's view, we must have that $\mathrm{dist}(v_i, \tilde{w}_i) < \frac{\delta}{2}$ for any $i \in \{2, \ldots, \ell\}$. But since E has relative distance at least δ, this implies in turn that $v_i = c_i$ for any $i = 2, \ldots, \ell$. But by assumption that $x \in \mathsf{NO}$, and by the soundness property of the protocol $(\mathcal{P}, \mathcal{V})$, we have that $\mathcal{V}$ rejects with probability at least $1 - \varepsilon$ on the view $((a_i)_{i\in[q]}, (\tilde{m}_i)_{i=2,\ldots,\ell})$. Consequently, $\mathcal{V}'$ also rejects v with probability at least $1 - \varepsilon$.

8 IOP for NP

We now prove Theorem 3.1, which gives a short IOP for NP languages. The proof of this theorem 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 8.1. *The following holds for any constant $\gamma, \delta > 0$. 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 $m \geq n$. Let*

$\mathcal{E} = \{E_m : \{0,1\}^m \to \{0,1\}^{\tilde{m}}\}_{m\in\mathbb{N}}$ *be a code ensemble of relative distance* δ *that can be encoded in quasi-linear time, and let* $\mathcal{C} = \{C_n : \{0,1\}^n \to \{0,1\}^{n'}\}_{n\in\mathbb{N}}$ *be a systematic linear code ensemble of rate* $1-\gamma$ *that can be encoded in quasi-linear time. Further assume that* n *divides* $\tilde{m}$*, and let* $C_{m,n} := C_{\frac{\tilde{m}}{n}+1} \otimes C_n$*.*

Let $\mathcal{L}' = \big\{(E_m(x), w) : (x, w) \in \mathcal{R}_{\mathcal{L}}\big\}$*, where we view the tuple* $(E_m(x), w)$ *as the implicit input to* $\mathcal{L}'$ *and there is no explicit input. Suppose that there exists an* ℓ*-round* $\frac{\delta}{2}$-IOPP $(\mathcal{P}, \mathcal{V})$ *with soundness error* ε *for* $\mathcal{L}'$ *with query complexity* $q(m,n)$*, verifier running time* $T_{\mathcal{V}}(m,n)$*, and prover running time* $T_{\mathcal{P}}(m,n)$*. Suppose furthermore that the communication phase consists of a single prover message which is the non-systematic part of* $C_{m,n}(E_m(x), w)$*, followed by* $\mathrm{cc}(m,n)$ *additional communication.*

Then there exists an ℓ*-round* IOP $(\mathcal{P}', \mathcal{V}')$ *with soundness error* ε *for* $\mathcal{L}$ *with communication complexity* $\frac{1}{1-\gamma}\cdot n + \mathrm{cc}(m,n)$*, query complexity* $q(m,n)$*, verifier running time* $T_{\mathcal{V}}(m,n) + \tilde{O}(m)$*, and prover running time* $T_{\mathcal{P}}(m,n) + \tilde{O}(m)$ *(given a witness). Moreover, the communication phase consists of a single prover message* $C_n(w)$ *of length* $\frac{1}{1-\gamma}\cdot n$*, followed by* $\mathrm{cc}(m,n)$ *additional communication.*

The proof of the above lemma is deferred to Section 8.1. Next we prove our Main Theorem 3.1 based on this lemma.

Proof of Theorem 3.1. Let $\mathcal{L} \in$ NP with corresponding relation $\mathcal{R}_{\mathcal{L}}$ in which the instance x has length m and the witness w has length n, where $n \le m \le n^a$ for a constant a. Let $\beta = 1/(2a)$ (so that $m^\beta \le \sqrt{n}$), and let t be the integer guaranteed by Lemma 7.1 for the constant β. Let $\mathcal{E} = \{E_m : \{0,1\}^m \to \{0,1\}^{\tilde{m}}\}_{m\in\mathbb{N}}$ be the systematic linear code ensemble of [Jus72] of constant rate and constant relative distance δ that can be encoded in time $\mathrm{poly}(m)$ with space $\mathrm{polylog}(m)$, or alternatively, in time $\tilde{O}(m)$ with space $\tilde{O}(m)$. Let $\mathcal{B} = \{B_n : \{0,1\}^n \to \{0,1\}^{n'}\}_{n\in\mathbb{N}}$ be any systematic linear code ensemble of rate $\left(\frac{1}{1+\gamma/2}\right)^{1/t}$ and constant relative distance that can be encoded in time $\tilde{O}(n)$.

For an integer n, let $\bar{n} := \left(\lceil n^{1/t}\rceil\right)^t \le (1+o(1))n$. We first claim that without loss of generality we may assume that n divides $\tilde{m}$, and that both $\frac{\tilde{m}}{n}+1$ and n belong to $\mathcal{I}_t$. To achieve this, one can pad w with zeros to obtain a string of length $\bar{n}$, noting that this does not change the asymptotic guarantees of the theorem. Moreover, one can pad $E_m(x)$ with zeros to obtain a string of length $\hat{m} := 2^t \cdot \bar{b}\cdot\bar{n} - \bar{n} = O(m)$, where $b := \lceil \frac{\tilde{m}}{n}\rceil$, noting that this only decreases the rate and relative distance of E_m by a constant factor. Note that indeed $\bar{n}$ divides $\hat{m}$, and that $\frac{\hat{m}}{\bar{n}}+1, \bar{n} \in \mathcal{I}_t$. Thus we may assume that n divides $\tilde{m}$, and that $\frac{\tilde{m}}{n}+1, n \in \mathcal{I}_t$, and we let

$$C_{m,n} := (B_{(\frac{\tilde{m}}{n}+1)^{1/t}})^{\otimes t} \otimes (B_{n^{1/t}})^{\otimes t} = (B_{(\frac{\tilde{m}}{n}+1)^{1/t}} \otimes B_{n^{1/t}})^{\otimes t}.$$

Let $\mathcal{L}' = \big\{(E_m(x), w) : (x, w) \in \mathcal{R}_{\mathcal{L}}\big\}$. Next observe that membership in $\mathcal{L}'$ can be decided in time $\mathrm{poly}(m)$ with space $\mathrm{polylog}(m) + n^\xi \le O(n^\xi)$ (since membership in E_m can be verified by encoding the systematic part of a given string y in time $\mathrm{poly}(m)$ with space $\mathrm{polylog}(m)$, and checking whether the resulting codeword equals y). Thus, applying Lemma 7.1 on $\mathcal{L}'$ gives a constant round $\frac{\delta}{2}$-IOPP with soundness error $\frac{1}{2}$ for $\mathcal{L}'$ with constant query complexity, verifier running time $m^\beta \cdot n^{O(\xi)} \le n^{1/2+O(\xi)}$, and prover running time $\mathrm{poly}(m)$. Moreover, the communication phase consists of a single prover message which is the non-systematic part of $C_{m,n}(E_m(x), w)$, followed by $n^{1/2+O(\xi)}$ additional communication.

Lemma 8.1 implies in turn the existence of a constant round IOP with soundness error $\frac{1}{2}$ for $\mathcal{L}$ with communication complexity $(1+\frac{\gamma}{2})\cdot n + n^{1/2+O(\xi)}$, constant query complexity, verifier running time $\tilde{O}(m)+n^{1/2+O(\xi)}$, and prover running time $\mathrm{poly}(m)$ (given a witness). Moreover, the communication phase consists of a single prover message $((B_{n^{1/t}})^{\otimes t})(w)$ of length $(1+\frac{\gamma}{2})\cdot n$, followed by $n^{1/2+O(\xi)}$ additional communication. Theorem 3.1 then follows for a sufficiently small constant $\xi > 0$, and noting that the soundness error can be reduced to any constant $\varepsilon > 0$ by repeating the protocol independently a constant number of times. Since the first prover message is deterministic, one does not need to repeat this message. □

8.1 From IOPP for P to IOP for NP

In this section we prove Lemma 8.1, which gives a transformation from IOPPs for deterministic languages to IOPs for non-deterministic languages.

The IOP $(\mathcal{P}', \mathcal{V}')$ for $\mathcal{L}$ proceeds as follows. Recall that $\mathcal{V}'$ gets as input $x \in \{0,1\}^m$, and $\mathcal{P}'$ gets $x \in \{0,1\}^m$ and a witness $w \in \{0,1\}^n$, where n divides $\tilde{m}$. In the communication phase, $\mathcal{P}'$ first sends w to $\mathcal{V}'$. Then $\mathcal{P}'$ and $\mathcal{V}'$ run the IOPP $(\mathcal{P}, \mathcal{V})$ for $\mathcal{L}'$ with respect to the implicit input $(E_m(x), w)$ (we once again emphasize that $(E_m(x), w)$ is an implicit input to the IOPP and there is no explicit input),[20] except that the first prover message is the non-systematic part of $C_n(w)$ instead of the non-systematic part of $C_{m,n}(E_m(x), w)$.

In the query phase, $\mathcal{V}'$ emulates the queries of $\mathcal{V}$ to the codeword $c := C_{m,n}(E_m(x), w)$, as follows. We view the message $(E_m(x), w) \in \{0,1\}^{\tilde{m}+n}$ as an $(\frac{\tilde{m}}{n}+1) \times n$ matrix M, whose first $\frac{\tilde{m}}{n}$ rows are $(E_m(x))_1, \ldots, (E_m(x))_{\frac{\tilde{m}}{n}}$, where $(E_m(x))_i$ denotes the i-th block of $E_m(x) \in \{0,1\}^{\tilde{m}}$ of length n, and whose $\frac{\tilde{m}}{n}+1$ row is $w \in \{0,1\}^n$. The codeword c can then be obtained by encoding each row of M via the code C_n, and encoding each resulting column via the code $C_{\frac{\tilde{m}}{n}+1}$.

The verifier $\mathcal{V}'$ first computes the first $\frac{\tilde{m}}{n}$ rows of c by encoding each block $(E_m(x))_i$ with the code C_n, which takes time $\frac{\tilde{m}}{n}\cdot\tilde{O}(n) = \tilde{O}(m)$ (recall that $\mathcal{V}'$ has explicit access to x, and so can compute these encodings by itself). Then to compute an entry $c(i,j)$, $\mathcal{V}'$ first retrieves the systematic part of the j'th column by making an additional query to the j'th entry of $C_n(w)$, and then computes $c(i,j)$ by encoding the systematic part of the j'th column via the code $C_{\frac{\tilde{m}}{n}+1}$, which takes time at most $\tilde{O}(\frac{m}{n})$.

It can be verified that the round complexity, communication complexity, query complexity, verifier running time, and prover running time are all as claimed, and that the requirements in the 'moreover' part are satisfied. Completeness is also straightforward. Next we show soundness.

To this end, 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 $\frac{\delta}{2}$-far from $\mathcal{L}'$. Thus, by soundness of the IOPP $(\mathcal{P}, \mathcal{V})$ for $\mathcal{L}'$, the verifier accepts with probability at most ε.

[20] 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.

Acknowledgements

We thank Yuval Ishai, Swastik Kopparty and Or Meir for useful discussions.

References

[ACY22a] Gal Arnon, Alessandro Chiesa, and Eylon Yogev. Hardness of approximation for stochastic problems via interactive oracle proofs. In *proceedings of the 37th Computational Complexity Conference (CCC)*, volume 234 of *LIPIcs*, pages 24:1–24:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[ACY22b] Gal Arnon, Alessandro Chiesa, and Eylon Yogev. A PCP theorem for interactive proofs and applications. In *Proceedings of the 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)*, volume 13276, pages 64–94. Springer, 2022.

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

[App17] Benny Applebaum. Exponentially-hard gap-csp and local PRG via local hardcore functions. In *proceedings of the 58th IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, pages 836–847. IEEE Computer Society, 2017.

[ARW17] Amir Abboud, Aviad Rubinstein, and Ryan Williams. Distributed PCP theorems for hardness of approximation in P. In *proceedings of the 58th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 25–36. IEEE Computer Society, 2017.

[AS98] Sanjeev Arora and Shmuel Safra. Probabilistic checkable proofs: A new characterization of NP. *Journal of the 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 *Proceedings of the 45th International Colloquium on Automata, Languages and Programming (ICALP)*, pages 14:1–14:17. Springer, 2018.

[BBHR19] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. Scalable zero knowledge with no trusted setup. In *proceedings of the 39th Annual International Cryptology Conference (Crypto)*, Lecture Notes in Computer Science, pages 701–732. Springer, 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 *Proceedings of the 44th International Colloquium on Automata, Languages and Programming (ICALP)*, pages 40:1–40:15. Springer, 2017.

[BCG20] Jonathan Bootle, Alessandro Chiesa, and Jens Groth. Linear-time arguments with sublinear verification from tensor codes. In *proceedings of the 18th International Theory of Cryptography Conference (TCC)*, volume 12551 of *Lecture Notes in Computer Science*, pages 19–46. Springer, 2020.

[BCGT13] Eli Ben-Sasson, Alessandro Chiesa, Daniel Genkin, and Eran Tromer. On the concrete efficiency of probabilistically-checkable proofs. In *Proceedings of the 45th Annual ACM Symposium on Theory of Computing (STOC)*, pages 585–594. ACM Press, 2013.

[BCL22] Jonathan Bootle, Alessandro Chiesa, and Siqi Liu. Zero-knowledge iops with linear-time prover and polylogarithmic-time verifier. In *Proceedings of the 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)*, volume 13276 of *Lecture Notes in Computer Science*, pages 275–304. Springer, 2022.

[BCR+19] Eli Ben-Sasson, Alessandro Chiesa, Michael Riabzev, Nicholas Spooner, Madars Virza, and Nicholas P. Ward. Aurora: Transparent succinct arguments for R1CS. In *Proceedings of the 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)*, Lecture Notes in Computer Science, pages 103–128. Springer, 2019.

[BCS16] Eli Ben-Sasson, Alessandro Chiesa, and Nicholas Spooner. Interactive oracle proofs. In *Proceedings of the 14th IACR Theory of Cryptography Conference (TCC)*, pages 31–60. Springer, 2016.

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

[BFLR20] Omri Ben-Eliezer, Eldar Fischer, Amit Levi, and Ron D. Rothblum. Hard properties with (very) short pcpps and their applications. In *proceedings of the 11th Innovations in Theoretical Computer Science Conference (ITCS)*, volume 151 of *LIPIcs*, pages 9:1–9:27. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[BFLS91] László Babai, Lance Fortnow, Leonid Levin, and Mario Szegedy. Checking computations in polylogarithmic time. In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 21–31. ACM Press, 1991.

[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 Journal on Computing*, 36(4):889–974, 2006.

[BGKS20] Eli Ben-Sasson, Lior Goldberg, Swastik Kopparty, and Shubhangi Saraf. DEEP-FRI: sampling outside the box improves soundness. In *proceedings of the 11th Innovations in Theoretical Computer Science Conference (ITCS)*, volume 151 of *LIPIcs*, pages 5:1–5:32. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[BKK+16] Eli Ben-Sasson, Yohay Kaplan, Swastik Kopparty, Or Meir, and Henning Stichtenoth. Constant rate PCPs for circuit-sat with sublinear query complexity. *Journal of the 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 Journal on Computing*, 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.

[CGL+19] Lijie Chen, Shafi Goldwasser, Kaifeng Lyu, Guy Rothblum, and Aviad Rubinstein. Fine-grained complexity meets ip = pspace. In *Proceedings of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1–20. SIAM, 2019.

[DEL+22] Irit Dinur, Shai Evra, Ron Livne, Alexander Lubotzky, and Shahar Mozes. Locally testable codes with constant rate, distance, and locality. In *Proceedings of the 54th Annual ACM Symposium on Theory of Computing (STOC)*, pages 357–374. ACM Press, 2022.

[Din07] Irit Dinur. The PCP theorem by gap amplification. *Journal of the 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 Journal on Computing*, 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. *Information and Computation*, 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. *Journal of the ACM*, 43(2):268–292, 1996.

[FS11] Lance Fortnow and Rahul Santhanam. Infeasibility of instance compression and succinct PCPs for NP. *Journal of Computer and System Sciences*, 77(1):91–106, 2011.

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

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

[GLS+21] Alexander Golovnev, Jonathan Lee, Srinath Setty, Justin Thaler, and Riad Wahby. Brakedown: Linear-time and post-quantum snarks for R1CS. Cryptology ePrint Archive, Report 2021/1043, 2021. https://ia.cr/2021/1043.

[GM12] Oded Goldreich and Or Meir. The tensor product of two good codes is not necessarily locally testable. *Information Processing Letters*, 112(8-9):351–355, 2012.

[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.

[GRR18] Tom Gur, Govind Ramnarayan, and Ron D. Rothblum. Relaxed locally correctable codes. In *proceedings of the 9th Innovations in Theoretical Computer Science Conference (ITCS)*, volume 94 of *LIPIcs*, pages 27:1–27:11. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 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. *Journal of the ACM*, 48(4):798–859, 2001.

[Imm88] Neil Immerman. Nondeterministic space is closed under complementation. *SIAM Journal on Computing*, 17(5):935–938, 1988.

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

[Kar75] R. M. Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1975.

[Kil92] Joe Kilian. A note on efficient zero-knowledge proofs and arguments. In *Proceedings of the 24th Annual ACM Symposium on Theory of Computing (STOC)*, pages 723–732. ACM Press, 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. *Journal of the ACM*, 64(2):11:1–11:42, 2017.

[KR08] Yael Tauman Kalai and Ran Raz. Interactive PCP. In *proceedings of the 35th International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 536–547. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2008.

[KR15] Yael Tauman Kalai and Ron D. Rothblum. Arguments of proximity. In *proceedings of the 35th Annual International Cryptology Conference (Crypto)*, Lecture Notes in Computer Science, pages 422–442. Springer, 2015.

[KSY14] Swastik Kopparty, Shubhangi Saraf, and Sergey Yekhanin. High-rate codes with sublinear-time decoding. *Journal of the ACM*, 61(5):28, 2014.

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

[Mei13] Or Meir. IP = PSPACE using error-correcting codes. *SIAM Journal on Computing*, 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 Journal on Computing*, 30(4):1253–1298, 2000.

[Mie09] Thilo Mie. Short PCPPs verifiable in polylogarithmic time with O (1) queries. *Annals of Mathematics and Artificial Intelligence*, 56(3-4):313–338, 2009.

[MR17] Pasin Manurangsi and Prasad Raghavendra. A birthday repetition theorem and complexity of approximating dense csps. In *proceedings of the 44th International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 80 of *LIPIcs*, pages 78:1–78:15. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017.

[Mul54] David Muller. Application of boolean algebra to switching circuit design and to error detection. *Transactions of the IRE Professional Group on Electronic Computers*, 3(3):6–12, 1954.

[NR22] Shafik Nassar and Ron D. Rothblum. Succinct interactive oracle proofs: Applications and limitations. *IACR Cryptology ePrint Archive*, page 281, 2022.

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

[PK22] Pavel Panteleev and Gleb Kalachev. Asymptotically good quantum and locally testable classical LDPC codes. In *Proceedings of the 54th Annual ACM Symposium on Theory of Computing (STOC)*, pages 375–388. ACM Press, 2022.

[Ran13] Hugues Randriambololona. An upper bound of singleton type for componentwise products of linear codes. *IEEE Transactions on Information Theory*, 59(12):7936–7939, 2013.

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

[RR19] Noga Ron-Zewi and Ron Rothblum. Local proofs approaching the witness length. *Electronic Colloquium on Computational Complexity*, page 127, 2019. https://eccc.weizmann.ac.il/report/2019/127/,.

[RR20] Guy N. Rothblum and Ron D. Rothblum. Batch verification and proofs of proximity with polylog overhead. In *Proceedings of the 18th IACR Theory of Cryptography Conference (TCC)*, volume 12551 of *Lecture Notes in Computer Science*, pages 108–138. Springer, 2020.

[RR22] Noga Ron-Zewi and Ron Rothblum. Proving as fast as computing: Succinct arguments with constant prover overhead. In *Proceedings of the 54th Annual ACM Symposium on Theory of Computing (STOC)*, pages 1353–1363. ACM Press, 2022.

[RRR17] Omer Reingold, Guy N. Rothblum, and Ron D. Rothblum, 2017. Personal Communication.

[RRR21] Omer Reingold, Guy N. Rothblum, and Ron D. Rothblum. Constant-round interactive proofs for delegating computation. *SIAM Journal on Computing*, 50(3), 2021.

[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 Symposium on Theory of Computing (STOC)*, pages 1260–1268. ACM Press, 2018.

[RVW13] Guy N. Rothblum, Salil P. Vadhan, and Avi Wigderson. Interactive proofs of proximity: delegating computation in sublinear time. In *Proceedings of the 45th Annual Symposium on Theory of Computing (STOC)*, pages 793–802. ACM Press, 2013.

[Sho88] Victor Shoup. New algorithms for finding irreducible polynomials over finite fields. In *Proceedings of the 29th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 283–290. IEEE Computer Society, 1988.

[Spi96] Daniel A. Spielman. Linear-time encodable and decodable error-correcting codes. *IEEE Transactions on Information Theory*, 42(6):1723–1731, 1996.

[Sti06] Henning Stichtenoth. Transitive and self-dual codes attaining the tsfasman-vladut-zink bound. *IEEE Transactions on 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.

[Sze87] Róbert Szelepcsényi. The method of forcing for nondeterministic automata. *Bulletin of the EATCS*, 33:96–99, 1987.

[Val05] Paul Valiant. The tensor product of two codes is not necessarily robustly testable. In *proceedings of the 9th International Workshop on Randomization and Computation (RANDOM)*, pages 472–481. Springer, 2005.

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

A Optimal communication complexity for IOPs

In this section we show the communication complexity in our IOP is close to optimal under the randomized strong exponential time hypothesis (RSETH) assumption. We first formally state the RSETH:

Definition A.1 (RSETH)**.** *The* randomized strong exponential time hypothesis (RSETH) *states that for every $\varepsilon > 0$ there exists k such that the language k-SAT (consisting of satisfiable k-CNF formulas on n variables) is not contained in* $\mathsf{BPTIME}\big(2^{(1-\varepsilon)\cdot n}\big)$.

We will also use the following result due to Goldreich and Håstad [GH98].

Theorem A.2 ([GH98, Proposition 6])**.** *If $\mathcal{L}$ has a constant-round public-coin interactive proof in which the prover sends at most b bits to the verifier, then $\mathcal{L} \in \mathsf{BPTIME}(2^b \cdot \mathrm{poly}(n))$ (where n is the input length).*

We remark that the conclusion stated in [GH98, Proposition 6] is that $\mathcal{L} \in \mathsf{BPTIME}(2^{O(b)} \cdot \mathrm{poly}(n))$ but inspection of their proof shows that it actually implies the stronger form as given in Theorem A.2 (i.e., $\mathcal{L} \in \mathsf{BPTIME}\big(2^b \cdot \mathrm{poly}(n)\big)$).

From Theorem A.2 we immediately derive the following conclusion:

Corollary A.3. *If* RSETH *holds then for every $\gamma > 0$ there exists a k such that k-SAT does not have a constant-round public-coin interactive proof in which the prover sends less than $(1-\gamma)\cdot n$ bits.*

B IOP Composition

In this section we prove Lemma 2.7, restated below, which gives a composition method for IOPs.

Lemma 2.7. *(Composition) Suppose that the following hold:*

- **(Outer IOP:)** *Let $(\mathsf{YES}, \mathsf{NO})$ be a promise problem with implicit input length n. Suppose that there exists an ℓ-round (α, ε)-robust IOP $(\mathcal{P}, \mathcal{V})$ for $(\mathsf{YES}, \mathsf{NO})$ with communication complexity $\mathrm{cc}(n)$, query complexity $q(n)$, randomness complexity $r(n)$, verifier running time $T_{\mathcal{V}}(n)$, and prover running time $T_{\mathcal{P}}(n)$. Suppose furthermore that $\mathcal{V}$ is $s(n)$-succinct.*
- **(Inner IOPP:)** *Let $\mathcal{L}_{(\mathcal{P},\mathcal{V})}$ denote the pair language consisting of all pairs $((x_{\mathrm{exp}}, R), y)$, so that $\mathcal{V}$ accepts on explicit input x_{exp}, randomness string R, and query values y. Suppose that there exists an ℓ'-round α-IOPP $(\mathcal{P}', \mathcal{V}')$ with soundness error ε' for the pair language $\mathcal{L}_{(\mathcal{P},\mathcal{V})}$ with communication complexity $\mathrm{cc}'(n')$, query complexity $q'(n')$, verifier running time $T_{\mathcal{V}'}(n')$, and prover running time $T_{\mathcal{P}'}(n')$, where n' denotes the implicit input length of $\mathcal{L}_{(\mathcal{P},\mathcal{V})}$.*

Then there exists an $(\ell + \ell')$-round IOP $(\mathcal{P}'', \mathcal{V}'')$with soundness error $\varepsilon + \varepsilon'$ for $(\mathsf{YES}, \mathsf{NO})$ with communication complexity $\mathrm{cc}(n) + \mathrm{cc}'(q(n))$, query complexity $q'(q(n))$, verifier running time $r(n) + T_{\mathcal{V}'}(q(n)) \cdot s(n)$, and prover running time $T_{\mathcal{P}}(n) + T_{\mathcal{V}}(n) + T_{\mathcal{P}'}(q(n))$. Moreover, the first (deterministic) message of $\mathcal{P}''$ is identical to that of $\mathcal{P}$.

Proof. The composed protocol $(\mathcal{P}'', \mathcal{V}'')$ is given in Fig. 4.

It can be verified that the round complexity, communication complexity, query complexity, verifier running time, and prover running time are all as stated, and that the 'moreover' part is satisfied as well. Next we show completeness and soundness.

The composed protocol $(\mathcal{P}'', \mathcal{V}'')$ for $(\mathsf{YES}, \mathsf{NO})$ on input $x = (x_{\text{imp}}, x_{\text{exp}})$:

Communication phase:

1. $\mathcal{P}''$ and $\mathcal{V}''$ interact for ℓ rounds according to the communication phase of the outer IOP $(\mathcal{P}, \mathcal{V})$ for $(\mathsf{YES}, \mathsf{NO})$ on explicit input x_{exp} and implicit input x_{imp}; Let $R = (R_1, \ldots, R_\ell)$ denote $\mathcal{V}$'s randomness string.
2. $\mathcal{P}''$ computes the query set I, based on the explicit input x_{exp} and the randomness string R; Let y denote the restriction of the implicit input x_{imp} and the messages of $\mathcal{P}$ to the query set I.
3. $\mathcal{P}''$ and $\mathcal{V}''$ interact for ℓ' rounds according to the communication phase of the inner IOPP $(\mathcal{P}', \mathcal{V}')$ for $\mathcal{L}_{(\mathcal{P},\mathcal{V})}$ on explicit input (x_{exp}, R) and implicit input y.

Query phase:

$\mathcal{V}''$ makes q' queries to the implicit input x_{imp} and the messages of $\mathcal{P}$ and $\mathcal{P}'$ according to the query phase of the IOPP $(\mathcal{P}', \mathcal{V}')$, and accepts if and only if $\mathcal{V}'$ accepts.

Figure 4: The Composed Protocol $(\mathcal{P}'', \mathcal{V}'')$

Completeness. Suppose that $x = (x_{\text{exp}}, x_{\text{imp}}) \in \mathsf{YES}$. Then by the completeness property of the outer IOP $(\mathcal{P}, \mathcal{V})$, the verifier $\mathcal{V}$ accepts given x_{exp}, R, and y with probability 1. Consequently, by the completeness proprety of the inner IOPP $(\mathcal{P}', \mathcal{V}')$, the verifier $\mathcal{V}'$ will accept with probability 1. We conclude that in this case $\mathcal{V}''$ accepts with probability 1.

Soundness. Suppose that $x = (x_{\text{exp}}, x_{\text{imp}}) \in \mathsf{NO}$, and let $(\mathcal{P}'')^*$ be a prover strategy. By the robustness property of the outer IOP $(\mathcal{P}, \mathcal{V})$, with probability at least $1 - \varepsilon$, y is α-far from any view that would make $\mathcal{V}$ accepts. Consequently, by the soundness property of the inner IOPP $(\mathcal{P}', \mathcal{V}')$, the verifier $\mathcal{V}'$ will reject with probability at least $1 - \varepsilon'$. We conclude that in this case $\mathcal{V}''$ rejects with probability at least $(1 - \varepsilon) \cdot (1 - \varepsilon')$. Consequently, the soundness error is at most $1 - (1 - \varepsilon) \cdot (1 - \varepsilon') \leq \varepsilon + \varepsilon'$. □

C Local testing and relaxed local correction protocols for tensor codes

C.1 Local testing protocol for tensor codes

In this section we prove Lemma 6.2, restated below, which gives a local testing protocol for tensor codes.

Lemma 6.2. *(Local testing protocol for tensor codes). The following holds for any constant integer $t > 1$ and $\delta > 0$. Let $\mathcal{C} = \{C_n : \{0,1\}^n \to \{0,1\}^{n'}\}_n$ be a systematic linear code ensemble of relative distance δ that can be encoded in time $T = T(n)$. Then for any constant $\alpha > 0$, there exists a 2-round α-IOPP with soundness error $\frac{1}{2}$ for the language $\mathcal{L} = \bigcup_n C_n^{\otimes t}$ (where we view the input to $\mathcal{L}$ as the implicit input and there is no explicit input) with communication complexity $n' \cdot \tilde{O}(T(n))$, constant query complexity, verifier running time* $\text{polylog}(T(n))$*, and prover running time* $\text{poly}(T(n))$*.*

The proof follows by composing the following local testing procedure for tensor codes from [Vid15] with an inner PCPP to reduce the query complexity.

Theorem C.1 (Local testing of tensor codes, [Vid15], Theorem 3.1)**.** *The following holds for any integer $t > 1$ and $\delta > 0$. Let $C : \{0,1\}^n \to \{0,1\}^{n'}$ be a systematic linear code of relative distance δ. Then there exists a randomized oracle algorithm $\mathcal{A}$ satisfying the following properties.*

- **Input:** *$\mathcal{A}$ gets oracle access to a string $w \in \{0,1\}^{[n']^t}$.*
- **Query complexity:** *$\mathcal{A}$ makes $(n')^2$ queries to w.*
- **Completeness:** *If w is a codeword of $C^{\otimes t}$, then $\mathcal{A}$ accepts with probability 1.*
- **Robustness:** *If $\mathrm{dist}(w, C^{\otimes t}) \geq \alpha$, then, in expectation, $\mathcal{A}$'s view is $(\delta^{O(t)} \cdot \alpha)$-far from an accepting view.*

If C can be encoded in time T, then $\mathcal{A}$ has running time $O(n' \cdot T)$. Moreover, the randomness complexity of $\mathcal{A}$ is $O(t \log(n'))$, and the location of each individual query can be computed in time $O(t \log(n'))$.

Remark C.2. *We remark on the following differences from [Vid15, Theorem 3.1]:*

1. *[Vid15, Theorem 3.1] only bounds the rejection probability of the $(n')^2$-query test that chooses a random two-dimensional axis-parallel plane (according to some specified distribution), and checks that the projection of w to the plane is a codeword of $C \otimes C$. However, the proof shows that this test is robust, in the sense that the average view of the tester is $(\delta^{O(t)} \cdot \alpha)$-far from $C \otimes C$.*
2. *The running time is not stated explicitly in [Vid15]. However, checking whether a given string $w' \in \{0,1\}^{n' \times n'}$ is a codeword of $C \otimes C$ can be done by first encoding the systematic part of w' via $C \otimes C$, and then checking that the resulting codeword equals w'. As $C \otimes C$ can be encoded in time $O(n' \cdot T)$ (cf., Fact 2.12), this results in a total running time of $O(n' \cdot T)$.*
3. *Randomness complexity is not stated explicitly in [Vid15]. However, inspection shows that the randomness complexity required for sampling the random two-dimensional plane, as well as the time required for computing the location of each individual point in this plane are $O(t \log(n'))$.*

Next we prove Lemma 6.2, based on the above Theorem C.1.

Proof of Lemma 6.2. We apply Corollary 2.10 on the local tester given by Theorem C.1 (which is in particular also a 1-round robust IOPP for the language $\mathcal{L} = \bigcup_n C_n^{\otimes t}$).

In more detail, by Theorem C.1 and Markov's inequality, there exists a 1-round $(\delta^{O(t)} \cdot \alpha, 1 - \delta^{O(t)} \cdot \alpha)$-robust α-IOPP $(\mathcal{P}, \mathcal{V})$ for the language $\mathcal{L} = \bigcup_n C_n^{\otimes t}$ with no communication complexity, query complexity $(n')^2$, randomness complexity $O(\log(n'))$, verifier running time $O(n' \cdot T(n))$, and no prover running time. Moreover, $\mathcal{V}$ is $O(\log(n'))$-succinct.

Consequently, by Corollary 2.10 there exists a 2-round α-IOPP $(\mathcal{P}', \mathcal{V}')$ with soundness error $1 - \delta^{O(t)} \cdot \alpha$ for $\mathcal{L}$ with communication complexity $n' \cdot \tilde{O}(T(n))$, constant query complexity, verifier running time $\mathrm{polylog}(T(n))$, and prover running time $\mathrm{poly}(T(n))$, where we used that $T(n) \geq n'$. Finally, the soundness error can be reduced to $\frac{1}{2}$ by repeating the protocol independently a constant number of times, and accepting if and only if all invocations accept. □

C.2 Relaxed local correction protocol for tensor codes

In this section we prove Lemma 6.3, restated below, which gives a relaxed local correction protocol for tensor codes.

Lemma 6.3. *(Relaxed local correction protocol for tensor codes). The following holds for any constant integer $t > 1$ and $\delta > 0$. Let $\mathcal{C} = \{C_n : \{0,1\}^n \to \{0,1\}^{n'}\}_n$ be a systematic linear code ensemble of relative distance δ that can be encoded in time $T = T(n)$. Let $(\mathsf{YES}, \mathsf{NO})$ be the following promise problem:*

$$\mathsf{YES} = \left\{(\bar{i}, w) \ : \ w \in C_n^{\otimes t}\right\},$$
$$\mathsf{NO} = \left\{(\bar{i}, w) \ : \ \mathrm{dist}(w, c) < \left(\frac{\delta}{4}\right)^t \text{ for } c \in C_n^{\otimes t} \text{ with } w(\bar{i}) \neq c(\bar{i})\right\},$$

where $\bar{i} \in [n']^t$ is the explicit input, and $w \in \{0,1\}^{[n']^t}$ is the implicit input.

Then there exists a 2-round IOP with soundness error $\frac{1}{2}$ for the promise problem $(\mathsf{YES}, \mathsf{NO})$ with communication complexity $\tilde{O}(T(n))$, constant query complexity, verifier running time $\mathrm{polylog}(T(n))$, and prover running time $\mathrm{poly}(T(n))$.

The proof follows by composing the following *robust* relaxed local correction procedure for tensor codes with an inner PCPP to reduce the verifier's running time and query complexity.

Lemma C.3 (Relaxed local correction of tensor codes)**.** *The following holds for any integer $t > 1$ and $\delta > 0$. Let $C : \{0,1\}^n \to \{0,1\}^{n'}$ be a systematic linear code of relative distance δ. Then there exists a randomized oracle algorithm $\mathcal{A}$ satisfying the following properties.*

- **Input:** *$\mathcal{A}$ takes as input a tuple $\bar{i} \in [n']^t$, and also gets oracle access to a string $w \in \{0,1\}^{[n']^t}$.*
- **Query complexity:** *$\mathcal{A}$ makes $2tn'$ queries to w.*
- **Completeness:** *If w is a codeword of $C^{\otimes t}$, then $\mathcal{A}$ accepts with probability 1.*
- **Robustness:** *If $\mathrm{dist}(w, c) < \left(\frac{\delta}{4}\right)^t$ for some codeword $c \in C^{\otimes t}$ with $c(\bar{i}) \neq w(\bar{i})$, then with probability at least $\left(\frac{\delta}{4}\right)^t$, the view of $\mathcal{A}$ is $\frac{\delta}{8t}$-far from an accepting view.*

If C can be encoded in time T, then $\mathcal{A}$ has running time $O(t \cdot T)$. Moreover, the randomness complexity of $\mathcal{A}$ is $t^2 \log(n')$, and the location of each individual query can be computed in time $O(t \log(n'))$.

Remark C.4. *Usually a relaxed local corrector is defined as a randomized oracle algorithm that is allowed to output either a symbol in $\{0,1\}$ or a special symbol $\perp$. The completeness requirement then is that if w is a codeword of $C^{\otimes t}$, then $\mathcal{A}$ outputs $w(\bar{i})$ with probability 1, while the soundness requirement is that if w is sufficiently close to a codeword c of $C^{\otimes t}$, then $\mathcal{A}$ outputs a symbol in $\{c(\bar{i}), \perp\}$ with sufficiently high probability. Note, however, that our relaxed local corrector can be modified to satisfy these requirements, by outputting $\perp$ if it rejects, and outputting $w(\bar{i})$ otherwise. To facilitate composition, it will be more convenient for us to work with our (stronger) requirements.*

Lemma C.3 is a robust version of [GRR18, Lemma 5.5.]. For completeness, we provide a full proof in Appendix C.2.1 below. Next we prove Lemma 6.3, based on Lemma C.3.

Proof of Lemma 6.3. We apply Corollary 2.10 on the relaxed local corrector given by Lemma C.3 (which is in particular also a 1-round robust IOP).

In more detail, by Lemma C.3, there exists a 1-round $(\frac{\delta}{8t}, 1-\left(\frac{\delta}{4}\right)^t)$-robust IOP $(\mathcal{P},\mathcal{V})$ for the promise problem $(\mathsf{YES},\mathsf{NO})$ with no communication complexity, query complexity $O(n')$, randomness complexity $O(\log(n'))$, verifier running time $O(T(n))$, and no prover running time. Moreover, $\mathcal{V}$ is $O(\log(n'))$-succinct.

Consequently, by Corollary 2.10 there exists a 2-round IOP $(\mathcal{P}',\mathcal{V}')$ with soundness error $1-\delta^{O(t)}$ for $(\mathsf{YES},\mathsf{NO})$ with communication complexity $\tilde{O}(T(n))$, constant query complexity, verifier running time $\mathrm{polylog}(T(n))$, and prover running time $\mathrm{poly}(T(n))$, where we used that $T(n) \geq n'$. Finally, the soundness error can be reduced to $\frac{1}{2}$ by repeating the protocol independently a constant number of times, and accepting if and only if all invocations accept. □

C.2.1 Relaxed local correction for tensor codes

In this section we prove Lemma C.3, which gives a robust relaxed local correction procedure for tensor codes. The relaxed local corrector $\mathcal{A}$ is given in Fig. 5 below.

Relaxed Local Corrector $\mathcal{A}$

Input: Coordinate $\bar{i} = (i_1,\ldots,i_t) \in [n']^t$, oracle access to $w \in \{0,1\}^{[n']^t}$

1. For $\ell = 1,...,t$:
   (a) Pick uniform random indices $r_{\ell+1}^{(\ell)},\ldots,r_t^{(\ell)} \in [n']$.
   (b) Consider the string $w_\ell \in \{0,1\}^{n'}$ defined as:
   $$w_\ell(j) = w(i_1,\ldots,i_{\ell-1},j,r_{\ell+1}^{(\ell)},\ldots,r_t^{(\ell)}),$$
   for every $j \in [n']$. Query all points in w_ℓ, as well as n' additional copies of $w_\ell(i_\ell)$.[a]
   (c) Check that w_ℓ is a codeword of C, and that all $n'+1$ copies of $w_\ell(i_\ell)$ are identical; If not, reject and abort.
2. Accept and abort.

[a] These (seemingly redundant) queries are made in order to get *robustness*.

Figure 5: Relaxed Local Corrector for Tensor Codes

It can be verified that query complexity and the running time are as claimed (the running time follows since checking whether a given string $y \in \{0,1\}^{n'}$ is a codeword of C can be done by encoding the systematic part of y via the code C, and checking whether the resulting codeword equals y), and that the requirements in the moreover part are satisfied. Next we show completeness and robustness.

Completeness. Suppose that w is a codeword of $C^{\otimes t}$. Then by properties of tensor codes, we have that w_ℓ is a codeword of C for any $\ell = 1,\ldots,t$. Consequently, for any $\ell = 1,\ldots,t$, $\mathcal{A}$ will not reject in Step 1c. But in this case it will accept on Step 2, as required.

Robustness. Suppose that $\mathrm{dist}(w, c) < \left(\frac{\delta}{4}\right)^t$ for some codeword $c \in C^{\otimes t}$ with $c(\bar{i}) \neq w(\bar{i})$. For $\ell = 1, \ldots, t+1$, let

$$\hat{c}_\ell := c(i_1, \ldots, i_{\ell-1}, \cdot, \ldots, \cdot) \in C^{\otimes(t-\ell+1)},$$

and

$$\hat{w}_\ell := w(i_1, \ldots, i_{\ell-1}, \cdot, \ldots, \cdot) \in \{0,1\}^{[n']^{t-\ell+1}}.$$

Note that $\hat{w}_1 = w$, $\hat{c}_1 = c$, $\hat{w}_{t+1} = w(\bar{i})$, and $\hat{c}_{t+1} = c(\bar{i})$. Robustness relies on the following claim.

Claim C.5. *Suppose that for some $\ell \in \{1, \ldots, t\}$ it holds that $\mathrm{dist}(\hat{w}_\ell, \hat{c}_\ell) < \left(\frac{\delta}{4}\right)^{t-\ell+1}$, but $\mathrm{dist}(\hat{w}_{\ell+1}, \hat{c}_{\ell+1}) \geq \left(\frac{\delta}{4}\right)^{t-\ell}$. Then on the ℓ-th iteration, with probability at least $\frac{1}{2} \cdot \left(\frac{\delta}{4}\right)^{t-\ell}$ over the choice of $r_{\ell+1}^{(\ell)}, \ldots, r_t^{(\ell)}$ on Step 1a, any $v \in \{0,1\}^{2n'}$ that is $\frac{\delta}{8}$-close to the view of $\mathcal{A}$ on Step 1b causes $\mathcal{A}$ to reject.*

Proof. Fix $\ell \in \{1, \ldots, t\}$, and suppose that $\mathrm{dist}(\hat{w}_\ell, \hat{c}_\ell) < \left(\frac{\delta}{4}\right)^{t-\ell+1}$, but $\mathrm{dist}(\hat{w}_{\ell+1}, \hat{c}_{\ell+1}) \geq \left(\frac{\delta}{4}\right)^{t-\ell}$. Then by assumption that $\mathrm{dist}(\hat{w}_\ell, \hat{c}_\ell) < \left(\frac{\delta}{4}\right)^{t-\ell+1}$, with probability at least $1 - \frac{1}{2} \cdot \left(\frac{\delta}{4}\right)^{t-\ell}$ over the choice of $r_{\ell+1}^{(\ell)}, \ldots, r_t^{(\ell)}$, we have that

$$\mathrm{dist}(w_\ell, c_\ell) \leq \frac{\delta}{2}, \tag{7}$$

where $c_\ell := c(i_1, \ldots, i_{\ell-1}, \cdot, r_{\ell+1}^{(\ell)}, \ldots, r_t^{(\ell)})$. On the other hand, by the assumption that $\mathrm{dist}(\hat{w}_{\ell+1}, \hat{c}_{\ell+1}) \geq \left(\frac{\delta}{4}\right)^{t-\ell}$, with probability at least $\left(\frac{\delta}{4}\right)^{t-\ell}$ over the choice of $r_{\ell+1}^{(\ell)}, \ldots, r_t^{(\ell)}$, we have that

$$w_\ell(i_\ell) = w(i_1, \ldots, i_\ell, r_{\ell+1}^{(\ell)}, \ldots, r_t^{(\ell)}) \neq c(i_1, \ldots, i_\ell, r_{\ell+1}^{(\ell)}, \ldots, r_t^{(\ell)}) = c_\ell(i_\ell). \tag{8}$$

Next assume that both events (7) and (8) above hold, which happens with probability at least $\frac{1}{2} \cdot \left(\frac{\delta}{4}\right)^{t-\ell}$. Suppose that $v \in \{0,1\}^{2n'}$ is $\frac{\delta}{8}$-close to the view of $\mathcal{A}$ on Step 1b on the ℓ-th iteration, we shall show that v causes $\mathcal{A}$ to reject. Let $v = (v^{(1)}, v^{(2)})$ where $v^{(1)}, v^{(2)} \in \{0,1\}^{n'}$.

First observe that by our assumption that v is $\frac{\delta}{8}$-close to $\mathcal{A}$'s view, we have that $\mathrm{dist}(v^{(1)}, w_\ell) \leq \frac{\delta}{4}$. By Eq. (7) and the triangle inequality, this implies in turn that $\mathrm{dist}(v^{(1)}, c_\ell) \leq \frac{3\delta}{4}$. Since $c_\ell \in C$, and C has relative distance δ, we conclude that either $v^{(1)} = c_\ell$, or $v^{(1)} \notin C$. In the latter case $\mathcal{A}$ clearly rejects, and so we may assume that $v^{(1)} = c_\ell$.

Next observe that by our assumption that v is $\frac{\delta}{8}$-close to $\mathcal{A}$'s view, we also have that at least a $(1 - \frac{\delta}{4})$-fraction of the entries in $v^{(2)}$ are equal to $w_\ell(i_\ell)$. On the other hand, by sEq. (8), we have that $c_\ell(i_\ell) \neq w_\ell(i_\ell)$. We conclude that there exists an entry in $v^{(2)}$ that is not identical to $c_\ell(i_\ell) = v^{(1)}(i_\ell)$, and consequently $\mathcal{A}$ rejects on v. □

Next observe that by our assumptions that $\mathrm{dist}(w, c) < \left(\frac{\delta}{4}\right)^t$ and $c(\bar{i}) \neq w(\bar{i})$, we have that $\mathrm{dist}(\hat{w}_1, \hat{c}_1) < \left(\frac{\delta}{4}\right)^t$, but $\mathrm{dist}(\hat{w}_{t+1}, \hat{c}_{t+1}) = 1 = \left(\frac{\delta}{4}\right)^0$. Consequently, there exists $\ell \in \{1, \ldots, t\}$ for which $\mathrm{dist}(\hat{w}_\ell, \hat{c}_\ell) < \left(\frac{\delta}{4}\right)^{t-\ell+1}$, but $\mathrm{dist}(\hat{w}_{\ell+1}, \hat{c}_{\ell+1}) \geq \left(\frac{\delta}{4}\right)^{t-\ell}$. But by the above Claim C.5, this implies in turn that with probability at least $\frac{1}{2} \cdot \left(\frac{\delta}{4}\right)^{t-\ell} \geq \left(\frac{\delta}{4}\right)^t$, any view that is $\frac{\delta}{8}$-close to $\mathcal{A}$'s view on Step 1b on the ℓ-th iteration, would cause it to reject. We conclude that with probability at least $\left(\frac{\delta}{4}\right)^t$, any view that is $\frac{\delta}{8t}$-close to $\mathcal{A}$'s view will cause $\mathcal{A}$ to reject.

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