Electronic Colloquium on Computational Complexity, Report No. 46 (2016)

Short Interactive Oracle Proofs with Constant Query Complexity, via Composition and Sumcheck

Eli Ben-Sasson
eli@cs.technion.ac.il
Technion

Alessandro Chiesa
alexch@berkeley.edu
UC Berkeley

Ariel Gabizon
arielga@cs.technion.ac.il
Technion

Michael Riabzev
michaelr@cs.technion.ac.il
Technion

Nicholas Spooner
spooner@cs.toronto.edu
University of Toronto

March 22, 2016

Abstract

We study *interactive oracle proofs* (IOPs) (Ben-Sasson, Chiesa, Spooner '16), which combine aspects of probabilistically checkable proofs (PCPs) and interactive proofs (IPs). We present IOP constructions and general techniques that enable us to obtain tradeoffs in proof length versus query complexity that are not known to be achievable via PCPs or IPs alone. Our main results are:

1. *Circuit satisfiability has* 3*-round IOPs with linear proof length (counted in bits) and constant query complexity.*
2. *Reed–Solomon codes have 2-round IOPs of proximity with linear proof length and constant query complexity.*
3. *Tensor product codes have 1-round IOPs of proximity with sublinear proof length and constant query complexity.*

For all of the above, known PCP constructions give *quasilinear* proof length and constant query complexity [BS08, Din07]. In addition, for circuit satisfiability, [BKK$^+$13] obtain PCPs with linear proof length but *sublinear* query complexity. As in [BKK$^+$13], we rely on algebraic-geometry codes to obtain our first result; but, unlike [BKK$^+$13], our use of such codes is much "lighter" because we do not rely on any automorphisms of the code.

We obtain our results in a modular way by proving and combining "IOP-analogues" of two fundamental tools:

- **Interactive proof composition.** Proof composition is used to reduce the query complexity of PCP verifiers, but the proof length of the composed proof system depends exponentially on the randomness complexity of the outer proof system. We prove a composition theorem for IOPs that does not suffer from this inefficiency.
- **Sublinear sumcheck.** The sumcheck protocol is an IP that enables an IP verifier to check the sum of values of a low-degree polynomial on an exponentially-large hypercube, but the verifier running time depends linearly on the individual degree. We prove a sumcheck protocol for IOPs that does not suffer from this inefficiency.

Overall, our work demonstrates that even constant-round IOPs are more efficient than known PCPs and IPs.

Keywords: probabilistically checkable proofs, interactive proofs, proof composition, sumcheck

ISSN 1433-8092

Contents

1 Introduction

We study *interactive oracle proofs* (IOPs) [BCS16], which combine aspects of probabilistically checkable proofs (PCPs) and interactive proofs (IPs). We present IOP constructions and general techniques that enable us to obtain tradeoffs in proof length versus query complexity that are not known to be achievable by either PCPs or IPs alone.

1.1 Motivation

Probabilistically checkable proofs (PCPs) were introduced by [FRS88, BFLS91, AS98, ALM$^+$98]: in a PCP, a probabilistic polynomial-time verifier has oracle access to the proof string. The complexity class $\mathbf{PCP}[\mathsf{r},\mathsf{q}]$ denotes those languages for which the verifier uses at most r random bits and queries at most q proof locations; the proof length is then at most 2^{r}. The PCP Theorem [AS98, ALM$^+$98] states that $\mathbf{NP} = \mathbf{PCP}[O(\log n), O(1)]$: every NP statement has a proof of polynomial length that can be verified via a constant number of queries (say, with soundness error $1/2$).

A fundamental question is how long a PCP needs to be, compared to the corresponding "standard" NP proof. Given $T\colon \mathbb{N} \to \mathbb{N}$, the PCP Theorem states that every language $\mathscr{L}$ in $\mathbf{NTIME}(T)$ has a proof of length $\mathrm{poly}(T(n))$ that can be verified with $O(1)$ queries. A sequence of works [PS94, HS00, GS06, BSVW03, BGH$^+$06, BS08, Din07] gradually reduced the proof length to quasilinear, i.e., $T(n) \cdot \mathrm{polylog}(T(n))$; much of this progress was accompanied by progress on efficient reductions from $\mathbf{NTIME}$ to "PCP-friendly" problems, as well as efficient constructions of PCPs of proximity (PCPPs) for key classes of linear codes. Despite much progress, the following question remains open: *are there PCPs with linear proof length and constant query complexity?*

Ben-Sasson et al. [BKK$^+$13] make progress in this direction by proving that there is $a > 0$ such that for every $\epsilon > 0$ there is a PCP for circuit satisfiability with proof length $2^{a/\epsilon}n$ and query complexity n^{ϵ}. Beyond the sublinear query complexity, [BKK$^+$13]'s result comes with other caveats not affecting most prior constructions: the verifier is *non-uniform*, namely it requires a polynomial-size advice string for every circuit size; and the verifier is not succinct, namely it cannot run in time that is sublinear in the circuit size even if the circuit comes from a uniform circuit family.

In this paper, we continue the study of the tradeoff between proof length and query complexity, but we do so for a more general probabilistic-checking model, which can be thought of as a "multi-round PCP", described next.

Proximity and robustness. We briefly introduce *PCPs of proximity* [DR04, BGH$^+$06] and *robust PCPs* [BGH$^+$06], because we need these notions shortly. To support these discussions, from this point onwards, we switch to using relations instead of languages. We denote by $\mathscr{R}$ a relation consisting of pairs $(\mathbb{x}, \mathbb{w})$, where $\mathbb{x}$ is the *instance* and $\mathbb{w}$ is the *witness*; we think of $\mathscr{R}$ as the relation that is naturally induced by a non-deterministic language $\mathscr{L}$. We denote by $\mathscr{R}|_{\mathbb{x}}$ the (possibly empty) set of witnesses for a given instance $\mathbb{x}$.

PCPs of proximity (PCPPs) strengthen PCPs. On the one hand, a PCP verifier has oracle access to a candidate proof π and decides whether $\mathscr{R}|_{\mathbb{x}} \neq \emptyset$ ($\mathbb{x} \in \mathscr{L}$) or $\mathscr{R}|_{\mathbb{x}} = \emptyset$ ($\mathbb{x} \notin \mathscr{L}$). On the other hand, a PCPP verifier has oracle access to a candidate witness $\mathbb{w}$ and proof π and decides whether $\mathbb{w} \in \mathscr{R}|_{\mathbb{x}}$ or $\mathbb{w}$ is far from $\mathscr{R}|_{\mathbb{x}}$ (in particular, if $\mathscr{R}|_{\mathbb{x}} = \emptyset$, then $\mathbb{w}$ is far from $\mathscr{R}|_{\mathbb{x}}$). A quantity δ known as the *proximity parameter* specifies what "far" means: if $\mathbb{w}$ is δ-far from $\mathscr{R}|_{\mathbb{x}}$ then the PCPP verifier accepts with probability at most ε, where ε is the soundness error.

Robust PCPs also strengthen PCPs, when $\mathscr{R}|_{\mathbb{x}} = \emptyset$: in such a case, the answers to the verifier's queries are, with high probability, far from any answers that make the verifier accept. A quantity ρ known as the *robustness parameter* specifies what "far" means: if $\mathscr{R}|_{\mathbb{x}} = \emptyset$ then, with probability at least $1 - \varepsilon$, the answers are ρ-far from accepting ones.

The two above notions can also be combined, yielding the definition of a *robust PCP of proximity*.

1.2 A more general model: interactive oracle proofs

Ben-Sasson, Chiesa, and Spooner introduce INTERACTIVE ORACLE PROOFS (IOPs) [BCS16], which combine aspects of IPs [Bab85, GMR89] and PCPs [BFLS91, AS98, ALM$^+$98]; also, they generalize interactive PCPs [KR08]. Informally, an IOP extends an IP as follows: whenever the prover sends to the verifier a message, the verifier does not have to read the message in full but may probabilistically query it. In more detail, a k-round IOP comprises k rounds of interaction. In the i-th round of interaction: the verifier sends a message m_i to the prover; then the prover replies with a message f_i to the verifier, which the verifier can query in this and all later rounds (by having oracle access to it). After the k rounds of interaction, the verifier either accepts or rejects.

An *IOP system* for a relation $\mathscr{R}$, with round complexity k and soundness ε is a pair (P, V), where P, V are probabilistic algorithms, that satisfies the naturally-defined notions of perfect completeness and soundness.

Like the IP model, one efficiency measure is the round complexity k. Like the PCP model, two additional efficiency measures are the *proof length* l, which is the total number of alphabet symbols in all of the prover's messages, and the *query complexity* q, which is the total number of locations queried by the verifier across all of the prover's messages. Considering all of these parameters, we say that a relation $\mathscr{R}$ belongs to the complexity class $\mathbf{IOP}[\mathsf{k}, \mathsf{a}, \mathsf{l}, \mathsf{r}, \mathsf{q}, \varepsilon]$ if there is an IOP system for $\mathscr{R}$ in which: (1) the number of rounds is $\mathsf{k}(n)$; (2) the prover messages are over the alphabet $\mathsf{a}(n)$; (3) the proof length over this alphabet is $\mathsf{l}(n)$; (4) the verifier uses $\mathsf{r}(n)$ random bits; (5) the verifier queries the prover messages in $\mathsf{q}(n)$ locations; (6) the soundness error is $\varepsilon(n)$.

Many other definitions for IPs and PCPs carry over naturally. An IOP is *public coin* if m_i is a random string and the verifier postpones any oracle queries until after receiving all the oracles from the prover (i.e., after the k-th round of interaction). An IOP is *non-adaptive* if the query locations do not depend on answers to any previous queries. An IOP has *proximity parameter* δ if the analogous property for PCPs of proximity holds; we can then correspondingly define the complexity class $\mathbf{IOPP}[\mathsf{k}, \mathsf{a}, \mathsf{l}, \mathsf{r}, \mathsf{q}, \varepsilon, \delta]$. See Section 2.3 for details.

Prior work on IOPs. After introducing IOPs and proving basic complexity-theoretic properties about them, [BCS16] prove that public-coin IOPs can be compiled into non-interactive proofs in the random oracle model. Their compiler can be viewed as a generalization of the Fiat–Shamir paradigm for public-coin IPs [FS86, PS96], and of the "CS proof" constructions of Micali [Mic00] and Valiant [Val08] for PCPs. Also, [BCGV16] construct 2-round IOPs (called "duplex PCPs" there) with unconditional zero knowledge and quasilinear proof length; in comparison, short PCPs with unconditional zero knowledge are not known. In this paper, we do not study compilers for cryptographic proofs, nor zero knowledge; instead, we focus on tradeoffs of proof length versus soundness.

Prior work on interactive PCPs. An interactive PCP [KR08] is a PCP followed by a standard IP; in particular, it is an IOP where the verifier sends an empty first message and may query only the first prover message (but must read any other prover messages in full). Prior work on interactive PCPs obtains proof length that depends on the witness size rather than computation size [KR08, GKR08], as well as unconditional zero knowledge [GIMS10]. In this paper we also study proof length, but do not know how to achieve similar results in the more restricted setting of interactive PCPs.

1.3 Results

We obtain several IOP constructions with proof length and query complexity that are not known to be achievable either via PCPs or IPs alone (or even via interactive PCPs [KR08]). First, we show that for circuit satisfiability we can obtain IOPs with linear proof length and constant query complexity; constant round complexity and public coins suffice.

Theorem 1.1 (informal). *Let $\mathscr{R}$ be the relation consisting of instance-witness pairs (ϕ, w) where ϕ is a boolean circuit (of two-input NAND gates) and w is a binary input that satisfies ϕ; we use n to denote the number of gates in ϕ. There exists $a > 0$ and a public-coin IOP system that puts $\mathscr{R}$ in the complexity class*

$$\mathbf{IOP}\begin{bmatrix} \text{rounds} & \mathsf{k}(n) & = & 3 \\ \text{answer alphabet} & \mathsf{a}(n) & = & \mathbb{F}_{O(a)} \\ \text{proof length} & \mathsf{l}(n) & = & a \cdot n \\ \text{query complexity} & \mathsf{q}(n) & = & a \\ \text{soundness error} & \varepsilon(n) & = & 1/2 \end{bmatrix} .$$

In particular, via [PF79]'s reduction from Turing machines to circuits, we deduce that

$$\mathbf{NTIME}(T) \subseteq \mathbf{IOP}\begin{bmatrix} \text{rounds} & \mathsf{k}(T) & = & 3 \\ \text{answer alphabet} & \mathsf{a}(T) & = & \mathbb{F}_{O(a)} \\ \text{proof length} & \mathsf{l}(T) & = & a \cdot T \log T \\ \text{query complexity} & \mathsf{q}(T) & = & a \\ \text{soundness error} & \varepsilon(T) & = & 1/2 \end{bmatrix} .$$

The main points of comparison of the above theorem with prior work are the following.

- For PCPs with constant query complexity, prior work achieved only quasilinear proof length [BS08, Din07], with the "quasilinear" hiding several logarithmic factors. In comparison, we achieve linear proof length for circuit satisfiability, and $O(T \log T)$ proof length for nondeterministic T-time relations.
- Ben-Sasson et al. [BKK+13] show that there is $a > 0$ such that for every $\epsilon > 0$ there is a non-uniform PCP for circuit satisfiability with proof length $2^{a/\epsilon}n$ and query complexity n^{ϵ}; the non-uniformity comes from the use of algebraic-geometry (AG) codes with transitive automorphism groups, for which uniform families are not known. In comparison, we simultaneously achieve linear proof length and constant query complexity; moreover, we make a much "lighter" use of AG codes, which also allows us to avoid non-uniformity. Namely, we rely only on the multiplication properties of AG codes, and do not rely on any code automorphisms. Looking ahead, this is because we do not route circuits on Cayley graphs induced by the automorphisms of the underlying code, unlike [BKK+13].

Second, we show that Reed–Solomon codes over binary fields (fields of characteristic 2) have 2-round IOPs of proximity with linear proof length and constant query complexity. Such codes are a key ingredient for constructing PCPs with quasilinear proof length [BS08].

Theorem 1.2 (informal)**.** *Fix a "fractional degree" $\varrho \in (0,1)$. Define the "Reed–Solomon relation" $\mathscr{R}$ to be the relation consisting of instance-witness pairs $((\mathbb{F}_{2^k}, d), w)$ where $d \leq \varrho 2^k$ and $w \colon \mathbb{F}_{2^k} \to \mathbb{F}_{2^k}$ is the evaluation of a polynomial of degree less than d. For every $\delta \in (0, \frac{1}{2}\varrho)$ there exist $a > 0$ and a public-coin IOP of proximity (P, V) that puts $\mathscr{R}$ in the complexity class*

$$\textbf{IOPP}\begin{bmatrix} \text{rounds} & \mathsf{k}(k) & = & 2 \\ \text{answer alphabet} & \mathsf{a}(k) & = & \mathbb{F}_{2^k} \\ \text{proof length} & \mathsf{l}(k) & = & a \cdot 2^k \\ \text{query complexity} & \mathsf{q}(k) & = & a \\ \text{soundness error} & \varepsilon(k) & = & 1/2 \\ \text{proximity parameter} & \delta(k) & = & \delta \end{bmatrix}.$$

More generally, our result concerns *additive Reed–Solomon codes*, where the domain of a codeword is a k-dimensional affine subspace S of a potentially larger binary field $\mathbb{F}$; in such cases the above statement involves more parameters but achieves the same asymptotics. The main point of comparison of the above theorem with prior work is [BS08, Din07], who achieve PCPs of proximity with the same parameters but superlinear proof length: $a \cdot 2^k \cdot \mathrm{poly}(k)$.

Third, we show that tensor product codes have 1-round IOPs of proximity with sublinear proof length and constant query complexity. Given a positive integer m and a linear code C with domain D and alphabet $\mathbb{F}$, the tensor product code $C^{\otimes m}$ is the linear code that comprises all functions $w \colon D^m \to \mathbb{F}$ whose restriction to any axis-parallel line is in C; the message length, block length, and distance of $C^{\otimes m}$ are each the m-th power of the corresponding parameters of C. Tensor product codes are a large family, and they include Reed–Muller codes (at least when considering the definition that bounds the variables' individual degrees, which we do, as opposed to the one that bounds their sum).

Theorem 1.3 (informal)**.** *Let $m > 4$ be a power of 2 and C a linear code with domain D, alphabet $\mathbb{F}$, and distance d. Let $\ell := |D|$ and suppose that $(\frac{d-1}{\ell})^m \geq \frac{7}{8}$. Define $\mathscr{R}$ to be the relation of instance-witness pairs $\big((C, m), w\big)$ such that $w \in C^{\otimes m}$. For every $\delta \in (0, \frac{1}{2}(\frac{d}{\ell})^m)$ there exist $a > 0$ and a public-coin IOPP system (P, V) that puts $\mathscr{R}$ in the complexity class*

$$\textbf{IOPP}\begin{bmatrix} \text{rounds} & \mathsf{k}(\ell^m) & = & 1 \\ \text{answer alphabet} & \mathsf{a}(\ell^m) & = & \mathbb{F} \\ \text{proof length} & \mathsf{l}(\ell^m) & = & o(\ell^m) \\ \text{query complexity} & \mathsf{q}(\ell^m) & = & a \\ \text{soundness error} & \varepsilon(\ell^m) & = & 1/2 \\ \text{proximity parameter} & \delta(\ell^m) & = & \delta \end{bmatrix}.$$

The main points of comparison of the above theorem with prior work are the following.

- Ben-Sasson and Sudan [BS06] construct local testers for tensor product codes with non-constant soundness error and query complexity: $\varepsilon(\ell^m) = 1 - \mathrm{poly}(m)\delta$ and $\mathsf{q}(\ell^m) = \ell^2$, respectively. In contrast, we simultaneously achieve constant soundness error and query complexity, with only sublinear proof length. We inherit the condition that $(\frac{d-1}{\ell})^m \geq \frac{7}{8}$, along with others specified later, from [BS06], since we use their local tester as a subroutine.

- The work of [BS08, Din07] implies PCPs of proximity for tensor product codes with superlinear proof length and query complexity. In contrast, we obtain sublinear proof length, with a single round of interaction.

We obtain the above results via techniques that we believe to be of independent interest: we prove that, in the IOP model, there are more efficient analogues of tools that are fundamental to the construction of PCPs and IPs and we discuss them next.

1.4 Techniques

Recall that IOPs generalize both IPs, by treating the prover's messages as oracle strings, and PCPs, by allowing for multiple rounds of interaction; they also generalize interactive PCPs [KR08]. We prove that IOPs can express two fundamental techniques in a more efficient way than in these prior models: (i) in *interactive proof composition*, the prover is more efficient than in PCP proof composition; and (ii) in *sublinear sumchecks*, the verifier is more efficient than in IP sumcheck protocols. We now discuss both of our new tools, and then how we use them.

Interactive proof composition. Proof composition [AS98] is used to reduce PCP query complexity, cf. [ALM$^+$98, HS00, BGH$^+$06]; it involves two PCPs: an outer one and an inner one. One should think of the outer proof system as having short proofs but large query complexity, while the inner proof system has long proofs but small query complexity.

The composed prover uses the outer prover to send a PCP to the composed verifier, who does not run the outer verifier but, instead, uses the inner verifier to check that the outer verifier would have accepted had it made its queries to the PCP. The composed verifier also needs an auxiliary sub-PCP for the claim that the outer verifier would have accepted; in fact, he needs one sub-PCP for each possible random string of the outer verifier. Hence, the composed prover also sends all of these sub-PCPs along with the first PCP. The benefit is that the query complexity of the composed verifier equals that of the inner verifier, which is typically verifying a much smaller statement than the outer verifier.

Beyond query complexity, most other parameters of the composed proof system are simply the sum of corresponding parameters of the outer and inner proof systems. An exception is the proof length l: it does not simply equal the sum $\mathsf{l}_{\mathsf{out}} + \mathsf{l}_{\mathsf{in}}$, but instead equals $\mathsf{l}_{\mathsf{out}} + 2^{\mathsf{r}_{\mathsf{out}}} \cdot \mathsf{l}_{\mathsf{in}}$, because the composed prover uses the inner proof system to generate a proof of proximity *for each choice of randomness of the outer proof system*. (The same is true for prover running time.)

We prove an **Interactive Proof Composition Theorem** that avoids the above limitations. The outer proof system is a robust PCP $(P_{\mathsf{out}}, V_{\mathsf{out}})$ for a relation $\mathscr{R}$, while the inner one is a k-round IOP $(P_{\mathsf{in}}, V_{\mathsf{in}})$ for V_{out}'s relation; the composed proof system is a $(\mathsf{k}+1)$-round IOP (P, V) for $\mathscr{R}$. The parameters of the composed proof system are exactly as before, except that now the new proof length is *much smaller*: $\mathsf{l}_{\mathsf{out}} + \mathsf{l}_{\mathsf{in}}$. (Ditto for the prover running time.) The crucial observation is that, after the prover sends the outer proof to the verifier, *soundness is not harmed if the verifier tells the prover his choice of outer randomness*; hence, the prover does not have to invest work for all randomness choices but can simply invest work only for the outer randomness that was chosen, which he now knows.

Theorem 1.4 (Interactive Proof Composition — informal)**.** *Suppose that the relation $\mathscr{R}$ satisfies the following:*

***(1)** there exists a robust PCPP system $(P_{\mathsf{out}}, V_{\mathsf{out}})$ that puts $\mathscr{R}$ in the complexity class*

$$\textbf{PCPP}\begin{bmatrix} \text{proof length} & \mathsf{l}_{\mathsf{out}} \\ \text{randomness} & \mathsf{r}_{\mathsf{out}} \\ \text{query complexity} & \mathsf{q}_{\mathsf{out}} \\ \text{soundness error} & \varepsilon_{\mathsf{out}} \\ \text{proximity parameter} & \delta_{\mathsf{out}} \\ \text{robustness parameter} & \rho_{\mathsf{out}} \end{bmatrix}$$

and

***(2)** for every $\mathbb{x}$ there exists an IOPP system $(P_{\mathsf{in}}, V_{\mathsf{in}})$ that puts V_{out}'s relation in the complexity class*

$$\textbf{IOPP}\begin{bmatrix} \text{rounds} & \mathsf{k}_{\mathsf{in}} \\ \text{proof length} & \mathsf{l}_{\mathsf{in}} \\ \text{randomness} & \mathsf{r}_{\mathsf{in}} \\ \text{query complexity} & \mathsf{q}_{\mathsf{in}} \\ \text{soundness error} & \varepsilon_{\mathsf{in}} \\ \text{proximity parameter} & \delta_{\mathsf{in}} \end{bmatrix}$$

If $\delta_{\mathsf{in}} \leq \rho_{\mathsf{out}}$ then there exists an IOPP system (P, V) that puts $\mathscr{R}$ in the complexity class

$$\textbf{IOPP}\begin{bmatrix} \text{rounds} & \mathsf{k} & = & 1 + \mathsf{k}_{\mathsf{in}} \\ \text{proof length} & \mathsf{l} & = & \mathsf{l}_{\mathsf{out}} + \mathsf{l}_{\mathsf{in}} \\ \text{randomness} & \mathsf{r} & = & \mathsf{r}_{\mathsf{out}} + \mathsf{r}_{\mathsf{in}} \\ \text{query complexity} & \mathsf{q} & = & \mathsf{q}_{\mathsf{in}} \\ \text{soundness error} & \varepsilon & = & \varepsilon_{\mathsf{out}} + \varepsilon_{\mathsf{in}} \\ \text{proximity parameter} & \delta & = & \delta_{\mathsf{out}} \end{bmatrix} .$$

The above discussion and informal theorem statement omit many technical details that already arise in non-interactive proof composition (e.g., see lengthy discussions in [BGH$^+$06, BGH$^+$05]), and we also need to deal with. For instance, one has to clarify the size of the sub-claim on which the the inner proof system is invoked; also, one has to carefully define the notion of a verifier to allow for the composed verifier's running time to be *smaller* than the outer verifier's query complexity. For more details, see Section 3.

Sublinear sumcheck. The *sumcheck protocol* [LFKN92, Sha92] is an interactive proof for the claim "$\sum_{\vec{\alpha} \in H^m} w(\vec{\alpha}) = 0$", where w is the evaluation on $\mathbb{F}^m$ of an m-variate polynomial of individual degree d and H is a subset of $\mathbb{F}$. More generally, w may be a codeword in the tensor product code $C^{\otimes m}$, for a given linear code C with domain D and alphabet $\mathbb{F}$, and H is a subset of D [Mei13]. The prover receives H and w as input, while the verifier receives H as input and w as an oracle. The protocol has m rounds and, if C has relative distance τ, the protocol has soundness error $1 - \tau^m$; also, the prover runs in time $\mathrm{poly}(\ell^m)$, and the verifier in time $\mathrm{poly}(\ell + m)$.

In each round, the verifier receives a codeword w_i in C and checks that $\sum_{\alpha \in H} w_i(\alpha)$ equals a certain value γ_{i-1} determined in the previous round; in particular, the verifier reads $\Omega(\ell)$ bits. We show that the verifier complexity can be *sublinear* in ℓ, if the prover and verifier engage in an IOP instead of an IP. The intuition to "go sublinear" is simple: instead of doing these checks explicitly, the verifier uses proximity testers for doing so. Thus, in each round, the prover sends to the verifier two oracles: the codeword in w_i, and a proximity proof attesting that $w_i \in C$ and that $\sum_{\alpha \in H} w_i(\alpha) = \gamma_{i-1}$. The use of proximity proofs complicates the soundness analysis because the verifier only sees noisy codewords, but the backbone of the proof follows that of the standard sumcheck protocol. Overall, we obtain a sumcheck IOP protocol that enables a verifier to efficiently check sumchecks for codes of much larger blocklength than what he can afford in the standard sumcheck protocol.

We state our **Sublinear Sumcheck Theorem** below as a reduction: given a PCP of proximity $(P_{\mathrm{SC}}, V_{\mathrm{SC}})$ for subcodes of the form $C_{H,\gamma} := \{w \in C \text{ s.t. } \sum_{\alpha \in H} w(\alpha) = \gamma\}$, we construct an IOP of proximity (P, V) for sumchecks over H^m for $C^{\otimes m}$. The complexity of the PCPP verifier V_{SC} determines the complexity of the resulting IOPP verifier V; e.g., if the former is sublinear in C's block length ℓ, so is the latter.

Theorem 1.5 (Sublinear Sumcheck — informal)**.** *Let m be a positive integer, and C a linear code with relative distance τ and block length ℓ. Suppose that there is a PCP of proximity for subcodes of the form $C_{H,\gamma} := \{w \in C \text{ s.t. } \sum_{\alpha \in H} w(\alpha) = \gamma\}$ with proof length l_{SC}, query complexity q_{SC}, soundness error $\varepsilon_{\mathrm{SC}}$, proximity parameter δ_{SC}, prover running time $\mathsf{tp}_{\mathrm{SC}}$, and verifier running time $\mathsf{tv}_{\mathrm{SC}}$. Then there is a public-coin IOP for sumchecks over H^m for $C^{\otimes m}$ with the following parameters:*

$$\mathbf{IOP}\left[\begin{array}{lccl} \text{rounds} & \mathsf{k}(\ell^m) & = & m \\ \text{proof length} & \mathsf{l}(\ell^m) & = & m \cdot \mathsf{l}_{\mathrm{SC}}(\ell) + m \cdot \ell \\ \text{query complexity} & \mathsf{q}(\ell^m) & = & m \cdot \mathsf{q}_{\mathrm{SC}}(\ell) + m + 1 \\ \text{soundness error} & \varepsilon(\ell^m) & = & 1 - \tau^m + m \cdot (1 - \delta_{\mathrm{SC}}(\ell)) + \varepsilon_{\mathrm{SC}}(\ell) \\ \text{prover time} & \mathsf{tp}(\ell^m) & = & m \cdot \mathsf{tp}_{\mathrm{SC}}(\ell) + m \cdot \ell^m \\ \text{verifier time} & \mathsf{tv}(\ell^m) & = & m \cdot \mathsf{tv}_{\mathrm{SC}}(\ell) + O(m) \end{array}\right].$$

In later sections, it is more natural to state the theorem without assuming that w is a codeword in $C^{\otimes m}$, so the reduction also takes as input a PCP of proximity $(P_{\otimes}, V_{\otimes})$ for $C^{\otimes m}$ that is invoked on w; this introduces additional terms in the parameters. More generally, both of the PCPs of proximity $(P_{\mathrm{SC}}, V_{\mathrm{SC}})$ and $(P_{\otimes}, V_{\otimes})$ can in fact be IOPs of proximity, and we state our theorem for this more general case, which we need. For more details, see Section 4.

Applying the new tools. We now sketch how we use the above new tools to derive the results of Section 1.3. We begin by discussing our results on proximity testing to codes (stated later); we then turn to circuit satisfiability (stated earlier) because its proof requires one of these results on proximity testing.

Intuition behind Theorem 1.2. The construction of linear-size IOPs of proximity for Reed–Solomon codes over binary fields follows from one invocation of our Interactive Proof Composition Theorem with [BS08]'s robust PCPs of proximity for Reed–Solomon codes as the outer proof system, and [Mie09]'s PCPs of proximity for nondeterministic languages as the inner proof system. Informally, in the first round, the prover sends to the verifier a [BS08] PCP of proximity, which reduces proximity testing of codewords over $\mathbb{F}_{2^k}$ to proximity testing of sub-codewords over $\mathbb{F}_{2^{k/2+O(1)}}$ with only constant overheads; in the second round, the verifier sends his choice of outer randomness, and the prover replies with a [Mie09] PCP of proximity for the sub-codeword. The proof length of this latter component is quasilinear, but is applied to a claim of "square-root size" only, so we obtain linear proof length.

Intuition behind Theorem 1.3. The construction of sublinear-size IOPs of proximity for tensor product codes follows from one invocation of our Interactive Proof Composition Theorem with [BS06]'s robust local tester for tensor product codes as the outer proof system, and [Mie09]'s PCPs of proximity for nondeterministic languages as the inner proof system. Unlike before, we now use one round, because the outer proof system only relies on a local tester rather than a PCP of proximity. The verifier thus simply sends his choice of outer randomness, and the prover replies with a [Mie09] PCP of proximity for a suitable sublinear-size sub-codeword. Since the proof length of this latter component is quasilinear but is applied to a sublinear-size claim, we obtain sublinear proof length.

A summary: overall, we can summarize the above sketches via the following diagram of implications.

$$
\begin{array}{ccccccc}
\underset{\substack{\text{linear-size IOPP}\\ \text{for Reed–Solomon codes}}}{\textbf{Theorem 1.2}} & \longleftarrow & \underset{\substack{\text{interactive}\\ \text{proof composition}}}{\text{Theorem 1.4}} & + & \underset{\substack{\text{robust PCPs of proximity}\\ \text{for Reed–Solomon codes}}}{\text{[BS08]}} & + & \underset{\substack{\text{PCPs of proximity}\\ \text{for } \mathbf{NTIME}}}{\text{[Mie09]}} \\
\underset{\substack{\text{sublinear-size IOPP}\\ \text{for tensor product codes}}}{\textbf{Theorem 1.3}} & \longleftarrow & \underset{\substack{\text{interactive}\\ \text{proof composition}}}{\text{Theorem 1.4}} & + & \underset{\substack{\text{robust local testing}\\ \text{for tensor product codes}}}{\text{[BS06]}} & + & \underset{\substack{\text{PCPs of proximity}\\ \text{for } \mathbf{NTIME}}}{\text{[Mie09]}}
\end{array}
$$

Intuition behind Theorem 1.1. We now turn to how to construct 3-round IOPs for circuit satisfiability with linear proof length and constant query complexity.

The first step of many PCP constructions is to arithmetize the NP statement at hand (in our case, the satisfiability of a boolean circuit) by reducing it to a "PCP-friendly" problem that looks like a constraint satisfaction problem over a well-chosen graph and whose assignments involve codewords in a well-chosen linear code C. Meir observes in [Mei12, Mei13] that key features of C are good relative distance and, moreover, a *multiplication property*: coordinate-wise multiplication of codewords yields codewords in a code whose relative distance is still good. Moreover, to obtain short PCPs, the aforementioned graph is typically chosen so to behave like a routing network [PS94]; for example, [BS08] use De Bruijn graphs, while [BKK$^+$13] use hypercubes. To support such graphs, the automorphism group of C has to be rich enough. This typically holds for Reed–Solomon codes [BS08] which have a doubly-transitive automorphism group, but is a significantly harder condition to fulfill for AG codes [BKK$^+$13], for which obtaining a transitive automorphism group is quite involved and, currently, can only be achieved non-uniformly.

The aforementioned first step would be problematic in our setting, because known routing techniques introduce either logarithmic overheads (as in [BS08]) or large query complexity (as in [BKK$^+$13]), so it is not clear how we could use them. Departing from these prior works, we do not rely on any routing, and instead immediately leverage one round of interaction to directly reduce circuit satisfiability to a sumcheck instance over a given linear code C. Also, we only assume that C has good relative distance and a multiplication property, *but we do not rely on any automorphisms*.

In this first round: (i) the prover sends three codewords w_1, w_2, w_3, the first one encoding values for all left wires, the second one encoding values for all right wires, and the third one encoding values for all output wires; (ii) the verifier then replies with a random field element per gate. At this point the verifier has to check two things. First, that the wire values are "locally consistent" with each gate: for every gate $i \in [n]$, $w_3(i)$ is the NAND of $w_1(i)$ and $w_2(i)$; moreover, the output gate wire value is 0. Second, that the three encodings of wire values are consistent with the circuit topology: namely, if $\ell(i)$ represents the left wire that is used to compute i, and $r(i)$ represents the right wire that is used to compute i, circuit topology requires that $w_3(\ell(i)) = w_1(i)$ and $w_2(r(i)) = w_1(i)$ for every i. Both of these checks can be "bundled" together using the verifier randomness to construct a random subset sum. The random subset sum can then be verified via a sumcheck. (Sampling one field element per gate would require a lot of randomness but this is not necessary since in fact we only need an *evading set*, as we explain later on.)

Below we informally state the reduction, using the following notion: given two linear codes C, C', we say that C' is a *degree d-closure* of C if, for every $w_1, \ldots, w_m \in C$ and m-variate polynomial P of total degree at most d, it holds that $w' \in C'$ where the ith entry of w' is computed by applying P to the ith coordinate of $w_1, \ldots, w_m$. See Section 6 for details of the reduction.

Lemma 1.6 (Circuit SAT to Sumcheck — informal)**.** *Let n be a positive integer, $C \subseteq \Sigma^D$ an n-systematic linear code, and ϕ an n-gate boolean circuit (of two-input NAND gates). There is a 1-round IOP that reduces satisfiability of ϕ to proximity testing to C and a sumcheck over any degree-3 closure of C; moreover, the IOP introduces only constant overheads in all relevant parameters (including proof length and query complexity).*

After reducing circuit satisfiability to sumcheck over the given code C, we are left to choose C so to ensure that the sumcheck can be carried out with 2 additional rounds, linear proof length, and constant query complexity.

For this, our starting point is [GS96, SAK$^+$01]'s efficient construction of a code family with constant rate, relative distance, and alphabet size. Note that since these codes are AG codes, they have a naturally-defined degree-3 closure. Also, their construction is uniform, and thus represents a much "lighter" use of AG codes as compared to in [BKK$^+$13].

If we simply choose C to be a code from this AG code family, then it is not clear how to efficiently conduct the sumcheck. However, what does work is to take C to be the tensor product of $O(1)$ copies of this AG code. Informally, in this way, we can invoke our Sublinear Sumcheck Theorem (Theorem 1.5) on the tensor product code C and we can test proximity to it by Theorem 1.3. See Section 7 for details.

Overall, we can summarize the above sketch via the following diagram of implications.

$$\underset{\substack{\text{linear-size IOP}\\ \text{for circuit SAT}}}{\textbf{Theorem 1.1}} \longleftarrow \underset{\substack{\text{from circuit SAT}\\ \text{to sumcheck}}}{\text{Lemma 1.6}} + \underset{\substack{\text{sublinear}\\ \text{sumcheck}}}{\text{Theorem 1.5}} + \underset{\substack{\text{sublinear-size IOP}\\ \text{for tensor product codes}}}{\text{Theorem 1.3}} + \underset{\substack{\text{efficient construction}\\ \text{of AG codes}}}{\text{[GS96, SAK}^+\text{01]}}$$

1.5 Open questions

The question that originally motivated our work remains open: are there PCPs with linear proof length and constant query complexity? Nevertheless, our work suggests additional questions that may be stepping stones in this and related research directions. First, are there 1-round IOPs for circuit satisfiability with linear proof length and query complexity? (Our construction requires 3 rounds.) Second, are there IOPs for $\mathbf{NTIME}(T)$ with linear proof length and query complexity, for *some* number of rounds? (Our construction only implies proof length $O(T \log T)$.)

1.6 Roadmap

The rest of this paper is organized as follows. In Section 2, we provide basic notations and definitions, including for PCPs and IOPs, and state prior results that we rely on. In Section 3, we state and prove the interactive composition theorem. In Section 4, we state and prove the sublinear sumcheck theorem. In Section 5, we state and prove our results for additive Reed–Solomon codes and tensor product codes. In Section 6, we state and prove the reduction from circuit satisfiability to sumcheck. In Section 7, we state and prove our result for circuit satisfiability.

2 Preliminaries

2.1 Basic notations

Functions, distributions, fields. We use $f\colon D \to R$ to denote a function with domain D and range R; given a subset $\tilde{D}$ of D, we use $f|_{\tilde{D}}$ to denote the restriction of f to $\tilde{D}$. Given a distribution $\mathcal{D}$, we write $x \leftarrow \mathcal{D}$ to denote that x is sampled according to $\mathcal{D}$. We denote by $\mathbb{F}$ a finite field and by $\mathbb{F}_q$ the field of size q; we say $\mathbb{F}$ is a *binary field* if its characteristic is 2. We typically use fields of polynomial size, and take field operations to have constant cost each (and inspection shows that accounting for their actual polylogarithmic cost does not change any of the stated results).

Distances. A distance measure is a function $\Delta\colon \Sigma^n \times \Sigma^n \to [0,1]$ such that for all $x, y, z \in \Sigma^n$: (i) $\Delta(x,x) = 0$, (ii) $\Delta(x,y) = \Delta(y,x)$, and (iii) $\Delta(x,y) \leq \Delta(x,z) + \Delta(z,y)$. We extend Δ to distances to sets: given $x \in \Sigma^n$ and $S \subseteq \Sigma^n$, we define $\Delta(x,S) := \min_{y \in S} \Delta(x,y)$ (or 1 if S is empty). We say that a string x is ϵ-close to another string y if $\Delta(x,y) \leq \epsilon$, and ϵ-far from y if $\Delta(x,y) > \epsilon$; similar terminology applies for a string x and a set S. Unless noted otherwise, we use the *relative Hamming distance* over alphabet Σ (typically implicit): $\Delta(x,y) := |\{i \,|\, x_i \neq y_i\}|/n$.

Languages and relations. We denote by $\mathscr{R}$ a (binary ordered) relation consisting of pairs $(\mathbb{x}, \mathbb{w})$, where $\mathbb{x}$ is the *instance* and $\mathbb{w}$ is the *witness*. We denote by $\mathrm{Lan}(\mathscr{R})$ the language corresponding to $\mathscr{R}$, and by $\mathscr{R}|_{\mathbb{x}}$ the set of witnesses in $\mathscr{R}$ for $\mathbb{x}$. As always, we assume that $|\mathbb{w}|$ is bounded by some computable function of $n := |\mathbb{x}|$; in fact, we are mainly interested in relations arising from nondeterministic languages: $\mathscr{R} \in \mathbf{NTIME}(T)$ if there exists a $T(n)$-time machine M such that $M(\mathbb{x}, \mathbb{w})$ outputs 1 if and only if $(\mathbb{x}, \mathbb{w}) \in \mathscr{R}$. Throughout, we assume that $T(n) \geq n$. We say that $\mathscr{R}$ has relative distance $\delta_{\mathscr{R}}$ if for every $\mathbb{x}$ the minimum relative distance between any two witnesses in $\mathscr{R}|_{\mathbb{x}}$ is at least $\delta_{\mathscr{R}}(|\mathbb{x}|)$.

Polynomials. We denote by $\mathbb{F}[X_1, \dots, X_m]$ the ring of polynomials in m variables over $\mathbb{F}$. Given a polynomial P in $\mathbb{F}[X_1, \dots, X_m]$, $\deg_{X_i}(P)$ is the degree of P in the variable X_i. We denote by $\mathbb{F}^{<d}[X_1, \dots, X_m]$ the subring consisting of $P \in \mathbb{F}[X_1, \dots, X_m]$ with $\deg_{X_i}(P) < d$.

2.2 Probabilistically checkable proofs

We define non-adaptive *PCPs* [BFLS91, AS98, ALM$^+$98], *PCPs of proximity* [DR04, BGH$^+$06], and *robust PCPs of proximity* [BGH$^+$06]; each notion strengthens the former. We follow [BGH$^+$06], where more details can be found.

Informally, given a relation $\mathscr{R}$ and an instance $\mathbb{x}$: a PCP verifier is given oracle access to a candidate proof π and decides whether $\mathscr{R}|_{\mathbb{x}} \neq \emptyset$ ($\mathbb{x}$ is in $\mathscr{R}$'s language) or $\mathscr{R}|_{\mathbb{x}} = \emptyset$ ($\mathbb{x}$ is not in $\mathscr{R}$'s language); in contrast, a PCPP verifier is given oracle access to a candidate witness $\mathbb{w}$ and proof π and decides whether $\mathbb{w} \in \mathscr{R}|_{\mathbb{x}}$ or $\mathbb{w}$ is far from $\mathscr{R}|_{\mathbb{x}}$ (in particular, if $\mathscr{R}|_{\mathbb{x}} = \emptyset$, then $\mathbb{w}$ is far from $\mathscr{R}|_{\mathbb{x}}$). A robust PCPP strengthens a (standard) PCPP when $\mathscr{R}|_{\mathbb{x}} = \emptyset$: in such a case, the answers to the verifier's queries are, with high probability, far from any answers that make the verifier accept.

More formally, in each of the above cases, the proof system is specified by a pair (P, V) that works as follows.

- The *prover* P is a probabilistic algorithm that, given as input an instance-witness pair $(\mathbb{x}, \mathbb{w})$ with $n := |\mathbb{x}|$, outputs a proof $\pi\colon D(n) \to \Sigma(n)$, where the domain $D(n)$ and alphabet $\Sigma(n)$ are finite sets.
- The *verifier* V is a pair $(V^{\mathrm{Q}}, V^{\mathrm{D}})$, where V^{Q} is the *query algorithm* and V^{D} is the *decision algorithm*, where:
  - V^{Q} is probabilistic and, given as input an instance $\mathbb{x}$, outputs a state string σ and a query set I; and
  - V^{D} is deterministic and, given as input the state string σ and the $|I|$ query answers, outputs a bit.

  The verifier V induces the relation $\mathrm{Rel}(V)$ comprising pairs (σ, a) such that, for some choice of $\mathbb{x}$ and I, (σ, I) is in the support of $V^{\mathrm{Q}}(\mathbb{x})$ and $V^{\mathrm{D}}(\sigma, a) = 1$.

We sometimes treat I an an algorithm that implicitly defines the query set (e.g., $I(i)$ is the i-th query) to allow for verifier-efficient interactive proof composition (see Section 3); this follows [BGH$^+$05]'s notion of *verifier specification*. Below, we explicitly denote the prover's and verifier's randomness as r_P and r_V.

The proof system (P, V) is for a relation $\mathscr{R}$ if it satisfies certain completeness and soundness properties. The completeness property is essentially the same in all three types of proof system; the only difference is that in a PCP the verifier only queries the proof π while in a (standard or robust) PCPP the verifier also queries the candidate witness.

Completeness

- **for PCPs:** (P, V) has <u>perfect completeness</u> if for every instance-witness pair $(\mathbb{x}, \mathbb{w})$ in the relation $\mathscr{R}$,

$$\Pr_{r_P, r_V}\left[V^{\mathrm{D}}(\sigma, \pi|_I) = 1 \;\middle|\; \begin{array}{c} \pi \leftarrow P(\mathbb{x}, \mathbb{w}; r_P) \\ (\sigma, I) \leftarrow V^{\mathrm{Q}}(\mathbb{x}; r_V) \end{array} \right] = 1 \ .$$

- **for PCPPs:** (P, V) has <u>perfect completeness</u> if for every instance-witness pair $(\mathbb{x}, \mathbb{w})$ in the relation $\mathscr{R}$,

$$\Pr_{r_P, r_V}\left[V^{\mathrm{D}}(\sigma, (\mathbb{w}\|\pi)|_I) = 1 \;\middle|\; \begin{array}{c} \pi \leftarrow P(\mathbb{x}, \mathbb{w}; r_P) \\ (\sigma, I) \leftarrow V^{\mathrm{Q}}(\mathbb{x}; r_V) \end{array} \right] = 1 \ .$$

The soundness property differs across the three types, with each condition strengthening the previous one:

Soundness

- **for PCPs:** (P, V) has <u>soundness error ε</u> if for every instance $\mathbb{x}$ with $\mathscr{R}|_{\mathbb{x}} = \emptyset$ and proof $\pi \colon D(n) \to \Sigma(n)$,

$$\Pr_{r_V}\left[V^{\mathrm{D}}(\sigma, \pi|_I) = 1 \;\middle|\; (\sigma, I) \leftarrow V^{\mathrm{Q}}(\mathbb{x}; r_V) \right] \le \varepsilon(n) \ .$$

- **for (standard) PCPPs:** (P, V) has <u>soundness error ε with proximity parameter δ</u> if for every instance-witness pair $(\mathbb{x}, \mathbb{w})$ with $\Delta(\mathbb{w}, \mathscr{R}|_{\mathbb{x}}) \ge \delta(n)$ and proof $\pi \colon D(n) \to \Sigma(n)$,

$$\Pr_{r_V}\left[V^{\mathrm{D}}(\sigma, (\mathbb{w}\|\pi)|_I) = 1 \;\middle|\; (\sigma, I) \leftarrow V^{\mathrm{Q}}(\mathbb{x}; r_V) \right] \le \varepsilon(n) \ .$$

- **for robust PCPPs:** (P, V) has <u>soundness error ε with proximity parameter δ and robustness parameter ρ</u> if for every instance-witness pair $(\mathbb{x}, \mathbb{w})$ with $\Delta(\mathbb{w}, \mathscr{R}|_{\mathbb{x}}) \ge \delta(n)$ and proof $\pi \colon D(n) \to \Sigma(n)$,

$$\Pr_{r_V}\left[(\mathbb{w}\|\pi)|_I \text{ is } \rho(n)\text{-close to } \mathrm{Rel}(V)|_\sigma \;\middle|\; (\sigma, I) \leftarrow V^{\mathrm{Q}}(\mathbb{x}; r_V) \right] \le \varepsilon(n) \ .$$

  (We use the familiar "Markov-type" definition for robust soundness, but researchers have also used a definition that imposes a lower bound on the expected distance from $(\mathbb{w}\|\pi)|_I$ to $\mathrm{Rel}(V)|_\sigma$. The two notions are related [BGH+06].)

We now introduce complexity classes for each of the three types of proof systems. A relation $\mathscr{R}$ belongs to the complexity class $\mathbf{PCP}[\mathsf{a}, \mathsf{l}, \mathsf{r}, \mathsf{q}, \mathsf{s}, \varepsilon, \mathsf{tp}, \mathsf{tvq}, \mathsf{tvd}]$ if there is a PCP system for $\mathscr{R}$ in which: (1) the proof alphabet is $\mathsf{a}(n)$ (i.e., $\Sigma(n) = \mathsf{a}(n)$); (2) the proof length over that alphabet is at most $\mathsf{l}(n)$ (i.e., $|D(n)| \le \mathsf{l}(n)$); (3) the verifier uses at most $\mathsf{r}(n)$ random bits (i.e., $|r_V| \le \mathsf{r}(n)$); (4) the verifier queries the witness and proof in at most $\mathsf{q}(n)$ locations (i.e., $|I| \le \mathsf{q}(n)$); (5) the verifier state size is at most $\mathsf{s}(n)$ (i.e., $|\sigma| \le \mathsf{s}(n)$); (6) the soundness error is $\varepsilon(n)$; (7) the prover algorithm runs in time $\mathsf{tp}(n)$; (8) the verifier query algorithm runs in time $\mathsf{tvq}(n)$; (9) the verifier decision algorithm runs in time $\mathsf{tvd}(n)$. Similarly, we say that $\mathscr{R}$ belongs to the complexity class $\mathbf{PCPP}[\mathsf{a}, \mathsf{l}, \mathsf{r}, \mathsf{q}, \mathsf{s}, \varepsilon, \delta, \mathsf{tp}, \mathsf{tvq}, \mathsf{tvd}]$ if there is a PCPP system for $\mathscr{R}$ with the above parameters and having proximity parameter δ; also, we say that $\mathscr{R}$ belongs to the complexity class $\mathbf{PCPP}[\mathsf{a}, \mathsf{l}, \mathsf{r}, \mathsf{q}, \mathsf{s}, \varepsilon, \delta, \rho, \mathsf{tp}, \mathsf{tvq}, \mathsf{tvd}]$ if there is a robust PCPP system for $\mathscr{R}$ with the above parameters and having proximity parameter δ and robustness parameter ρ. Sometimes we write tv to denote the sum of tvq and tvd, i.e., the total running time of the verifier.

Throughout, we assume that the proximity parameter is less than the "unique-decoding radius" of the relation $\mathscr{R}$, i.e., that $\delta < \delta_{\mathscr{R}}/2$ where $\delta_{\mathscr{R}}$ is the relative distance of $\mathscr{R}$ (see Section 2.1).

2.3 Interactive oracle proofs

We define *interactive oracle proofs* (IOPs) [BCS16], which combine aspects of interactive proofs [Bab85, GMR89] and probabilistically checkable proofs [BFLS91, AS98, ALM+98]; also, they generalize interactive PCPs [KR08].

Informally, an IOP extends an interactive proof as follows: whenever the prover sends to the verifier a message, the verifier does not have to read the message in full but may probabilistically query it. In more detail, a k-round IOP comprises k rounds of interaction. In the i-th round of interaction: the verifier sends a message m_i to the prover; then the prover replies with a message f_i to the verifier, which the verifier can query in this and all later rounds (by having oracle access to it). After the k rounds of interaction, the verifier either accepts or rejects.

An *IOP system* for a relation $\mathscr{R}$ with round complexity k and soundness ε is a pair (P, V), where P, V are probabilistic algorithms, that satisfies the following properties. (See [BCS16] for more details.)

Completeness: For every instance-witness pair $(\mathbb{x}, \mathbb{w})$ in the relation $\mathscr{R}$, $\langle P(\mathbb{x}, \mathbb{w}), V(\mathbb{x})\rangle$ is a $\mathsf{k}(n)$-round interactive oracle protocol with accepting probability 1.

Soundness: For every instance $\mathbb{x}$ not in $\mathscr{R}$'s language and unbounded malicious prover $\tilde{P}$, $\langle \tilde{P}, V(\mathbb{x})\rangle$ is a $\mathsf{k}(n)$-round interactive oracle protocol with accepting probability at most $\varepsilon(n)$.

Like the IP model, a fundamental measure of efficiency is the round complexity k. Like the PCP model, two additional fundamental measures of efficiency are the *proof length* l, which is the total number of alphabet symbols in all of the prover's messages, and the *query complexity* q, which is the total number of locations queried by the verifier across all of the prover's messages. Considering all of these parameters, we say that a relation $\mathscr{R}$ belongs to the complexity class $\mathbf{IOP}[\mathsf{k}, \mathsf{a}, \mathsf{l}, \mathsf{r}, \mathsf{q}, \varepsilon, \mathsf{tp}, \mathsf{tv}]$ if there is an IOP system for $\mathscr{R}$ in which: (1) the number of rounds is at most $\mathsf{k}(n)$; (2) the prover messages are over the alphabet $\mathsf{a}(n)$; (3) the proof length over this alphabet is at most $\mathsf{l}(n)$; (4) the verifier uses at most $\mathsf{r}(n)$ random bits; (5) the verifier queries the prover messages in at most $\mathsf{q}(n)$ locations; (6) the soundness error is $\varepsilon(n)$; (7) the prover algorithm runs in time $\mathsf{tp}(n)$; (8) the verifier algorithm runs in time $\mathsf{tv}(n)$.

Finally, we say that an IOP system is *non-adaptive* if the verifier queries are non-adaptive (i.e., the queried locations depend only on the verifier's inputs); also, we say that an IOP system is *public coin* if m_i is a random string and queries to $f_1, \ldots, f_{i-1}$ before the i-th round depend only on $m_1, \ldots, m_{i-1}$.

Proximity. We also study IOPs of proximity, which extend IOPs the same way that PCPs of proximity extend PCPs. An *IOPP system* for a relation $\mathscr{R}$ with round complexity k, soundness error ε, and proximity parameter δ is a pair (P, V) that satisfies the following properties.

Completeness: For every instance-witness pair $(\mathbb{x}, \mathbb{w})$ in the relation $\mathscr{R}$, $\langle P(\mathbb{x}, \mathbb{w}), V^{\mathbb{w}}(\mathbb{x})\rangle$ is a $\mathsf{k}(n)$-round interactive oracle protocol with accepting probability 1.

Soundness: For every instance-witness pair $(\mathbb{x}, \mathbb{w})$ and with $\Delta(\mathbb{w}, \mathscr{R}|_{\mathbb{x}}) \geq \delta(n)$ and unbounded malicious prover $\tilde{P}$, $\langle \tilde{P}, V^{\mathbb{w}}(\mathbb{x})\rangle$ is a $\mathsf{k}(n)$-round interactive oracle protocol with accepting probability at most $\varepsilon(n)$.

Similarly to above, a relation $\mathscr{R}$ belongs to the complexity class $\mathbf{IOPP}[\mathsf{k}, \mathsf{a}, \mathsf{l}, \mathsf{r}, \mathsf{q}, \varepsilon, \delta, \mathsf{tp}, \mathsf{tv}]$ if there is an IOPP system for $\mathscr{R}$ with the corresponding parameters. As in Section 2.2, we always assume that δ is less than $\delta_{\mathscr{R}}/2$.

Robustness. Analogously to non-adaptive PCPs, a robustness parameter ρ can be defined for non-adaptive IOPs. For such IOPs, we can think of the verifier algorithm as two algorithms: a query algorithm that, given as input an instance $\mathbb{x}$, interacts with the prover and then outputs a state string σ and a query set I; then, a decision algorithm takes as input the state string σ and the $|I|$ query answers (across all prover messages), and outputs a bit. Then one requires that, with probability at most $\varepsilon(n)$, the answers to the verifier's queries are $\rho(n)$-close to any answers that make the verifier accept. We do not spell out all the details of this definition because robustness of IOPs will arise in informal discussions only.

Remark 2.1 (comparison with PCPs and IPCPs)**.** An IOP can be viewed as a "multi-round PCP": a PCP is the special case of an IOP where $\mathsf{k} = 1$ (and the first verifier message is empty). Similarly, a PCP of proximity is the special case of an IOP of proximity where $\mathsf{k} = 1$ (and the first verifier message is empty). Also, IOPs generalize interactive PCPs [KR08], which are IOPs where the verifier sends an empty first message and may query only the first prover message (but must read any other prover messages in full).

2.4 Codes

An error correcting code C is a set of functions $w \colon D \to \Sigma$, where D, Σ are finite sets known as the domain and alphabet; in such a case we write $C \subseteq \Sigma^D$ The message length of C is $n := \log_{|\Sigma|} |C|$, its block length is $\ell := |D|$, its rate is $\rho := n/\ell$, its (minimum) distance is $d := \min\{\Delta(w, z) \mid w, z \in C \,,\, w \neq z\}$ when Δ is the (absolute) Hamming distance, and its (minimum) relative distance is $\tau := d/\ell$. At times we write $n(C), \ell(C), \rho(C), d(C), \tau(C)$ to make the code under consideration explicit.

Linearity. A code C is linear if Σ is a finite field and C is a Σ-linear space in Σ^D. The dual code of C is the set $C^\perp$ of functions $z\colon D \to \Sigma$ such that, for all $w\colon D \to \Sigma$, it holds that $\langle z, w\rangle := \sum_{i \in D} z(i)w(i) = 0$. We denote by $\dim(C)$ the dimension of C when viewed as a linear space; it holds that $\dim(C) + \dim(C^\perp) = \ell$. The support of C, denoted $\mathrm{supp}(C)$, is the union of the support (non-zero entries) of functions in C.

Systematicity. Given $s \in \mathbb{N}$, a code $C \subseteq \Sigma^D$ is *s-systematic* if there exists a subset of D identified with $[s]$ such that for every $x \in \Sigma^{[s]}$ there exists $w \in C$ such that $x = w|_{[s]}$.

Tensor products. Given a positive integer m, the *tensor product code* $C^{\otimes m}$ is the linear code with domain D^m, alphabet Σ, message length n^m, block length ℓ^m, and distance d^m that comprises all functions $w\colon D^m \to \Sigma$ whose restriction to any axis-parallel line is in C. Namely, for every $j \in \{1, \dots, m\}$ and $a_1, \dots, a_{j-1}, a_{j+1}, \dots, a_\ell \in D$, the function $w'\colon D \to \Sigma$ defined by $w'(i) := w(a_1, \dots, a_{j-1}, i, a_{j+1}, \dots, a_\ell)$ is in C.

Polynomial closures. Given $C, C' \subseteq \mathbb{F}^D$ and positive integer t, we say that C' is a *degree-t closure* of C if, for every $w_1, \dots, w_m \in C$ and $P \in \mathbb{F}[X_1, \dots, X_m]$ of total degree at most t, it holds that $w' := P(w_1, \dots, w_m)$ is in C', where $w'\colon D \to \Sigma$ is defined coordinate-wise by the equation $w'(i) := P(w_1(i), \dots, w_m(i))$ (i.e., P is applied coordinate-wise to $(w_1, \dots, w_m)$). We now state a simple lemma about polynomial closures.

Claim 2.2. *Let $C, C' \subseteq \mathbb{F}^D$, $t, m \in \mathbb{N}$, and $P \in \mathbb{F}[X_1, \dots, X_m]$. Suppose that C' is a degree-t closure of C, P has total degree at most t, and $f_1, \dots, f_m \in \mathbb{F}^D$ have relative distance less than $\frac{\tau(C')}{2m}$ to C'. Let w' be the codeword in C' closest to $P(f_1, \dots, f_m)$, and, for $i = 1, \dots, m$, let w_j be the codeword in C closest to f_j. Then $w' = P(w_1, \dots, w_m)$.*

Proof. Let T be the set of $i \in D$ for which there exist $j \in [m]$ for which $f_j(i) \neq w_j(i)$. By assumption, $|T|$ is less than $\frac{1}{2}\tau(C')|D|$. The codeword $P(w_1, \dots, w_m)$ is in C' and disagrees with $P(f_1, \dots, f_m)$ in less than $\frac{1}{2}\tau(C')|D|$ places, and so must be the unique codeword in C' closest to $P(f_1, \dots, f_m)$. Thus, $w' = P(w_1, \dots, w_m)$, as desired. $\square$

Code families. We also consider families of codes, and the notation above typically extends in the natural way.

- *Parameters:* A code family $\mathscr{C} = \{C_n\}_{n \in \mathbb{N}}$ has domain $D(\cdot)$ and alphabet $\mathbb{F}(\cdot)$ if each code C_n has domain $D(n)$ and alphabet $\mathbb{F}(n)$. Similarly, $\mathscr{C}$ has block length $\ell(\cdot)$, rate $\rho(\cdot)$, distance $d(\cdot)$, and relative distance $\tau(\cdot)$ if each code C_n has block length $\ell(n)$, rate $\rho(n)$, distance $d(n)$, and relative distance $\tau(n)$.

- *Systematicity:* A code family $\mathscr{C} = \{C_n\}_{n \in \mathbb{N}}$ is systematic if each code C_n is n-systematic.

- *Tensor products:* Given a code family $\mathscr{C} = \{C_n\}_{n \in \mathbb{N}}$ and a positive integer m, $\mathscr{C}^{\otimes m}$ is the family $\{C_n^{\otimes m}\}n \in \mathbb{N}$. Note that if $\mathscr{C}$ is systematic then each $C_n^{\otimes m}$ is n^m-systematic.

- *Polynomial closures:* Given two code families $\mathscr{C} = \{C_n\}_{n \in \mathbb{N}}$ and $\mathscr{D} = \{D_n\}_{n \in \mathbb{N}}$ over the same domain $D(n)$ and alphabet $\mathbb{F}(n)$ and a positive integer t, $\mathscr{D}$ is a degree-t closure of $\mathscr{C}$ if each D_n is a degree-t closure of C_n.

For systematic code families, we also consider the following notion of efficiency. We say that a code family $\mathscr{C} = \{C_n\}_{n \in \mathbb{N}}$ is $T(\cdot)$-efficient if computing a codeword as well as checking that a codeword lies in the code can be done in time $T(\cdot)$. More precisely, there exist two deterministic algorithms Enc and Chk such that, for every $n \in \mathbb{N}$:
– on input $x \in \mathbb{F}(n)^{[n]}$, Enc outputs $w \in C_n$ such that $w|_{[n]} = x$;
– on input $w \in \mathbb{F}(n)^{\ell(n)}$, Chk outputs 1 if and only if $w \in C_n$.
If a code family is not systematic, we talk about its efficiency, but in such a case we only require the second condition (about checking if a codeword is in the code) and ignore the first one (about computing codewords).

We also consider code families $\mathscr{C}$ that are indexed not by $\mathbb{N}$ but by an infinite subset $\mathcal{I}$ thereof that may contain "holes". Most of the above definitions carry over in the natural way, but special attention is warranted for the case of systematicity. Namely, if $\mathscr{C} = \{C_n\}_{n \in \mathcal{I}}$ is such that each C_n is n-systematic, then we view $\mathscr{C}$ as a systematic code family as follows: for each $n \notin \mathcal{I}$, we implicitly define $C_n := C_{n'}$ for the least $n' \in \mathcal{I}$ that is larger than n.

The following additional notation will be convenient for systematic code families. We denote by $\tau(\mathscr{C})$ the infimum of $\tau(C_n)$ over all $n \in \mathbb{N}$. In particular, $\tau(\mathscr{C}) > 0$ if and only if $\mathscr{C}$ has constant relative distance. Similarly, we denote by $\rho(\mathscr{C})$ the infimum of the quantity $n/\ell(C_n)$ over all $n \in \mathbb{N}$. A technicality is that the later quantity might not precisely correspond to the rate, as the dimension of C_n is only *lower bounded* by n.

2.4.1 Reed–Solomon codes

The Reed–Solomon code is the code consisting of evaluations of *univariate* low-degree polynomials: given a field $\mathbb{F}$, subset S of $\mathbb{F}$, and positive integer d with $d \leq |S|$, we denote by $\mathrm{RS}[\mathbb{F}, S, d]$ the linear code consisting of evaluations $w\colon S \to \mathbb{F}$ over S of polynomials in $\mathbb{F}^{<d}[X]$. The code's message length is $n = d$, block length is $\ell = |S|$, rate is $\rho = \frac{d}{|S|}$, and relative distance is $\tau = 1 - \frac{d-1}{|S|}$.

2.4.2 Reed–Muller codes

The Reed–Muller code is the code consisting of evaluations of *multivariate* low-degree polynomials: given a field $\mathbb{F}$, subset S of $\mathbb{F}$, and positive integers m, d with $d \leq |S|$, we denote by $\mathrm{RM}[\mathbb{F}, S, m, d]$ the linear code consisting of evaluations $w\colon S^m \to \mathbb{F}$ over S^m of polynomials in $\mathbb{F}^{<d}[X_1, \ldots, X_m]$ (i.e., we bound individual degrees rather than their sum). The code's message length is $n = d^m$, block length is $\ell = |S|^m$, rate is $\rho = (\frac{d}{|S|})^m$, and relative distance is $\tau = (1 - \frac{d-1}{|S|})^m$. Also, one can verify that (i) $\mathrm{RS}[\mathbb{F}, S, d] = \mathrm{RM}[\mathbb{F}, S, 1, d]$ and (ii) $\mathrm{RM}[\mathbb{F}, S, m, d]$ is the m-wise tensor product of $\mathrm{RS}[\mathbb{F}, S, d]$.

2.4.3 Algebraic-geometry codes

Codewords in an algebraic-geometry (AG) code [Gop81] consist of functions evaluated over domains that arise as the solutions of certain systems of rational equations. AG codes generalize Reed–Solomon codes and "behave" similarly but in contrast to RS codes, they can have arbitrarily large blocklength over *constant-size* fields. More impressively, AG code-families can achieve constant rate and relative distance over fixed alphabet size; and like RS-codes, they can be constructed to be closed under coordinate-wise multiplication. A formal definition of AG codes is not necessary to understand the results in this paper, so we refer the reader to the excellent reference of Stichtenoth [Sti08]. Instead, below we provide a self-contained statement of the result we rely on; it follows from [SAK$^+$01, Theorem 7], which gives an efficient construction of AG codes based on [GS96]'s explicit towers of function fields.

Theorem 2.3. *For every $\tau \in (0,1)$ and $t \in \mathbb{N}$, there exist a prime power q and two code families $\mathscr{A} = \{A_n\}_{n\in\mathbb{N}}$ and $\mathscr{B} = \{B_n\}_{n\in\mathbb{N}}$ that satisfy the following properties for every $n \in \mathbb{N}$:*
1. A_n and B_n are n-systematic codes with alphabet $\mathbb{F}_q$;
2. $\rho(\mathscr{A}), \rho(\mathscr{B}) > 0$. In particular, A_n and B_n all have rate at least ρ for a fixed $\rho > 0$;
3. A_n and B_n have relative distance at least τ;
4. B_n is a degree-t closure of A_n.
Moreover, both $\mathscr{A}$ and $\mathscr{B}$ are $T(n)$-efficient with $T(n) = O(n^3 \cdot \log n)$ (see Section 2.4).

2.5 Evading sets

Given a field $\mathbb{F}$ and positive integer n, an evading set for $\mathbb{F}^n$ is a small set S such that for any non-zero $v \in \mathbb{F}^n$ the inner product $\langle r, v\rangle$, for a uniformly random $r \in S$, does not attain any particular value $a \in \mathbb{F}$ with too high probability. The notion is captured by the following definition.

Definition 2.4. *Let $n \in \mathbb{N}$, $\gamma \in (0,1)$, and $\mathbb{F}$ be a field. A subset S of $\mathbb{F}^n$ is* γ-evading *for $\mathbb{F}^n$ if $\Pr_{r\leftarrow S}[\langle r, v\rangle = a] \leq \gamma$ for every $v \in \mathbb{F}^n$ and $a \in \mathbb{F}$ with $v \neq 0^n$. We say that a family $\mathscr{S} = \{S_n\}_{n\in\mathbb{N}}$ is $T(\cdot)$-efficient $\gamma(\cdot)$-evading for $\mathbb{F}(\cdot)$ if, for every $n \in \mathbb{N}$: (i) S_n is $\gamma(n)$-evading for $\mathbb{F}(n)^n$, and (ii) one can sample a random vector in S_n in time $T(n)$.*

We state a construction of a small evading set, for any finite field. The construction is a straightforward generalization of the one for $\mathbb{F}_2$ in [AGHP92], where it is shown to have the stronger property of being an ϵ-biased set [NN90]; for completeness we also provide a proof sketch.

Lemma 2.5 (based on [AGHP92])**.** *Let $\epsilon \in (0,1)$, $n \in \mathbb{N}$, and $m := \lceil \log_q(n/\epsilon)\rceil$. Let q be a prime power and $\phi\colon \mathbb{F}_{q^m} \to \mathbb{F}_q^m$ an isomorphism of $\mathbb{F}_q$-vector spaces. Define $S := \{s_{x,y}\}_{x\in\mathbb{F}_{q^m}, y\in\mathbb{F}_q^m}$ where $s_{x,y} \in \mathbb{F}_q^n$ is the vector $\big(\langle\phi(1), y\rangle, \langle\phi(x), y\rangle, \ldots, \langle\phi(x^{n-1}), y\rangle\big)$. Then S is $(\epsilon + 1/q)$-evading for $\mathbb{F}_q^n$.*

In particular, there is algorithm that, on input $n \in \mathbb{N}$ and $i \in [q^{2m}]$, runs in time $n \cdot \mathrm{polylog}(nq)$ and outputs the i-th element of a set S of size q^{2m} that is $(\epsilon + 1/q)$-evading for $\mathbb{F}_q^n$.

Proof. We only sketch the proof, because it is very similar to [AGHP92, Proposition 3]. Fix $v \in \mathbb{F}^n$ and $a \in \mathbb{F}$ with $v \neq 0^n$. The inner product $\langle s_{x,y}, v \rangle$ equals the inner product of y with $\phi(P_v(x))$, where P_v is the polynomial $P_v(X) \triangleq \sum_{i=0}^{n-1} v_i \cdot X^i$. Hence, $P_v(x) = 0$ with probability at most ϵ. Given that $\phi(P_v(x)) \neq 0$, the probability its inner product with a uniform y equals a is at most $1/q$. $\square$

2.6 Constructions of proximity testers

We state results about constructions of proximity testers that we use in this paper.

2.6.1 Robust PCPs of proximity for additive Reed–Solomon codes

Ben-Sasson and Sudan [BS08] show that Reed–Solomon codes over binary fields have quasilinear-size PCPs of proximity with constant query complexity. A central ingredient of their construction is a robust PCP of proximity for these codes that reduces proximity testing for dimension k to proximity testing for dimension $k/2 + O(1)$; this also reduces the query complexity from 2^k to $O(2^{k/2})$.

Below, we state [BS08]'s robust PCP of proximity for these codes. The general statement involves the notion of an *additive* Reed–Solomon code $\mathrm{RS}^+[\mathbb{F}, S, d]$, which is a Reed–Solomon code $\mathrm{RS}[\mathbb{F}, S, d]$ where $\mathrm{char}(\mathbb{F}) = 2$ and S is an $\mathbb{F}_2$-linear affine subspace. In this case there exist $\alpha_0 \in \mathbb{F}$ and $\mathbb{F}_2$-linearly-independent $\alpha_1, \ldots, \alpha_k \in \mathbb{F}$ such that S equals the $\mathbb{F}_2$-linear span of $(\alpha_1, \ldots, \alpha_k)$ shifted by α_0; S can then be succinctly represented via the list $(\alpha_0, \alpha_1, \ldots, \alpha_k)$. If $\mathrm{char}(\mathbb{F}) = 2$, we then denote by $\mathrm{Rel}(\mathbb{F}, \varrho)$ the relation of instance-witness pairs $(\mathbb{x}, \mathbb{w}) = \big((S, d), w\big)$ such that $w \in \mathrm{RS}^+[\mathbb{F}, S, d]$, S is k-dimensional, and $d \leq \varrho|S|$; we define the size of $\mathbb{x}$ to be k.

Theorem 2.6. *For every $\varrho > 0$ there exist $\alpha \in (0, 1)$ and $a > 0$ such that for every binary field $\mathbb{F}$ and $\delta \in (0, \frac{1}{2}\varrho)$ there exists a robust PCPP system (P, V) that puts $\mathrm{Rel}(\mathbb{F}, \varrho)$ in the complexity class*

$$\mathbf{PCPP}\begin{bmatrix} \text{answer alphabet} & \mathsf{a}(k) & = & \mathbb{F} \\ \text{proof length} & \mathsf{l}(k) & = & a \cdot 2^k \\ \text{randomness} & \mathsf{r}(k) & = & k + a \\ \text{query complexity} & \mathsf{q}(k) & = & a \cdot 2^{k/2} \\ \text{state size} & \mathsf{s}(k) & = & k/2 + a \\ \text{soundness error} & \varepsilon(k) & = & 1 - \alpha \cdot \delta \\ \text{proximity parameter} & \delta(k) & = & \delta \\ \text{robustness parameter} & \rho(k) & = & \alpha \cdot \delta \\ \text{prover time} & \mathsf{tp}(k) & = & k^a 2^k \\ \text{verifier query time} & \mathsf{tvq}(k) & = & k/2 + a \\ \text{verifier decision time} & \mathsf{tvd}(k) & = & k^a \cdot 2^{k/2} \end{bmatrix} .$$

Moreover, $\mathrm{Rel}(V)$ *is a subset of* $\mathrm{Rel}(\mathbb{F}, \varrho)$.

Remark 2.7. All of the existential quantifiers in the theorem statement are "efficient" in the sense that there is a (uniform) polynomial-time procedure to construct the relevant objects. For example, (P, V) efficiently depends on the field $\mathbb{F}$ and proximity parameter δ.

Also, the running time of the verifier query algorithm in the above statement is $k/2 + a$ while the number of queries is $a \cdot 2^{k/2}$. This is possible because the verifier query algorithm outputs an algorithm that implicitly represents the query set rather than the query set itself; see Section 2.2.

2.6.2 Robust local testers for tensor product codes

Ben-Sasson and Sudan [BS06] show that tensor product codes have robust local testers; their work builds on the tester of Raz and Safra [RS97] for low-degree polynomials. Suitable settings of parameters yield a generic construction of codes with $1/\mathrm{poly}(n)$ rate that are testable with $\mathrm{polylog}(n)$ queries.

Given an integer $m > 1$ and a linear code $C \subseteq \mathbb{F}^D$ with distance d, there is a natural $|D|^{m-1}$-query test for checking that a function $w \colon D^m \to \mathbb{F}$ is close to $C^{\otimes m}$: sample random $i \in D$ and $j \in \{1, \ldots, m\}$ and check that w with the

j-th coordinate restricted to i is in $C^{\otimes m-1}$. Ben-Sasson and Sudan [BS06] show that this test is robust: if $m \geq 3$ and $(\frac{d-1}{|D|})^m \geq \frac{7}{8}$ then the tester is α-robust for a universal constant $\alpha \in (0,1)$. Then, [BS06] prove a composition theorem for Tanner product codes [Tan81] (of which tensor product codes are a special case) and use it, together with the aforementioned tester, to show that the $|D|^2$-query test "*pick a random axis-parallel 2-dimensional plane H in D^m and check that $w|_H \in C^{\otimes 2}$*" is $\alpha^{\log m}$-robust; see [BS06, Lemma 4.3]. (Recall that an n-dimensional axis-parallel plane H is a set of points in D^m obtained by restricting all but n coordinates to constants.)

Below, we state the aforementioned result in a more general form (which we need): for a given $i \in \{1, \ldots, \log m\}$, the test picks a random axis-parallel 2^i-dimensional plane H and checks that $w|_H \in C^{\otimes 2^i}$. This more general form follows by invoking the original result on the tensor code $C_0^{\otimes m/2^{i-1}}$ with $C_0 := C^{\otimes 2^{i-1}}$

Lemma 2.8. *There exists $\alpha \in (0,1)$ such that for every linear code C with domain D, alphabet $\mathbb{F}$, and relative distance τ, and every integer m that is a power of 2, if $m \geq 3$ and $(\frac{d-1}{|D|})^m \geq \frac{7}{8}$ then for every $i \in \{1, \ldots, \log m\}$:*

1. (completeness) *For every $w \colon D^m \to \mathbb{F}$ in the tensor product code $C^{\otimes m}$, it holds that* $\Pr_H[w|_H \in C^{\otimes 2^i}] = 1$.
2. (robust soundness) *For every $w \colon D^m \to \mathbb{F}$, it holds that* $\mathbb{E}_H[\Delta(w|_H, C^{\otimes 2^i})] \geq \alpha^{\frac{\log m}{i}} \cdot \Delta(w, C^{\otimes m})$.

2.6.3 PCPs of proximity for nondeterministic languages

Mie [Mie09] constructs PCPs of proximity for nondeterministic languages, where the prover runs in quasilinear time and the verifier in polylogarithmic time, with constant proximity parameter, soundness error, and query complexity.

Theorem 2.9. *For every relation $\mathscr{R} \in \mathbf{NTIME}(T(n))$ with distance $\delta_{\mathscr{R}}$, every $\delta_0 \in (0, \delta_{\mathscr{R}})$, and every $\varepsilon_0 > 0$, there exist $q_0 > 0$ and a PCPP system (P, V) that puts $\mathscr{R}$ in the complexity class*

$$\mathbf{PCPP}\begin{bmatrix} \text{answer alphabet} & \mathsf{a}(n) & = & \mathbb{F}_2 \\ \text{proof length} & \mathsf{l}(n) & = & \tilde{O}(T(n)) \\ \text{randomness} & \mathsf{r}(n) & = & \log T(n) + O(\log\log T(n)) \\ \text{query complexity} & \mathsf{q}(n) & = & q_0 \\ \text{state size} & \mathsf{s}(n) & = & \operatorname{polylog} T(n) \\ \text{soundness error} & \varepsilon(n) & = & \varepsilon_0 \\ \text{proximity parameter} & \delta(n) & = & \delta_0 \\ \text{prover time} & \mathsf{tp}(n) & = & \operatorname{poly}(n) + \tilde{O}(T(n)) \\ \text{verifier query time} & \mathsf{tvq}(n) & = & \operatorname{poly}(n + \log T(n)) \\ \text{verifier decision time} & \mathsf{tvd}(n) & = & \operatorname{poly}(n + \log T(n)) \end{bmatrix}.$$

Remark 2.10. The statement in [Mie09] only shows that the prover runs in $\operatorname{poly}(T(n))$ time. However, that prover is composed of first running the "outer prover" of [BS08], which was shown to run in $\operatorname{poly}(n) + \tilde{O}(T(n))$ time [BCGT13], and then applying the "inner prover" of [Din07] (which runs in polynomial time) on instances of size $\operatorname{polylog} T(n)$. Combined, the resulting prover runs in $\operatorname{poly}(n) + \tilde{O}(T(n))$ time, as stated above.

3 Interactive proof composition

We prove an interactive proof composition theorem that, when compared to its non-interactive counterpart [AS98], provides significant savings in proof length, as well as prover running time, for the composed proof system. Later on, we leverage this result to obtain short interactive oracle proofs (see Section 5).

Proof composition. Proof composition is a fundamental technique for reducing the query complexity of PCP verifiers; it was introduced in [AS98] and later used and refined in numerous PCP constructions [ALM$^+$98, HS00, BGH$^+$06]. Proof composition involves two probabilistically-checkable proof systems: an outer one and an inner one. One should think of the outer proof system as having short proofs but large query complexity, while the inner proof system has long proofs but small query complexity. Proof composition combines these two so as to obtain a new proof system that inherits the good properties of each; one can think of this as a "proof analogue" of code concatenation [For65].

Informally, the composed prover uses the outer prover to send a PCP to the composed verifier; the composed verifier does not run the outer verifier but, instead, uses the inner verifier to check that the outer verifier would have accepted had it made its queries to the PCP. In order to do so, the composed verifier also needs an auxiliary sub-PCP for the claim that the outer verifier would have accepted; in fact, he needs one such sub-PCP for each possible random string of the outer verifier because each randomness choice induces a corresponding claim. Hence, the composed prover also sends all of these sub-PCPs along with the first PCP. The benefit is that the query complexity of the composed verifier equals that of the inner verifier, which is typically verifying a much smaller statement than the outer verifier (this statement's size is roughly the outer query complexity).

Turning the above sketch into a proof requires distinct properties from the outer PCP and inner PCP. A useful choice is from [BGH$^+$06]: the outer PCP should be a robust PCP while the inner PCP should be a PCP of proximity.

Limitations of proof composition. Beyond query complexity, most other parameters of the composed proof system are simply the sum of the corresponding parameters of the outer and inner proof systems: roughly, the randomness complexity is $\mathsf{r} = \mathsf{r}_{\mathsf{out}} + \mathsf{r}_{\mathsf{in}}$, the soundness error is $\varepsilon = \varepsilon_{\mathsf{out}} + \varepsilon_{\mathsf{in}}$, and the verifier running time is $\mathsf{tv} = \mathsf{tv}_{\mathsf{out}} + \mathsf{tv}_{\mathsf{in}}$. There are two exceptions: the proof length l and prover running time tp are at least $\mathsf{l}_{\mathsf{out}} + 2^{\mathsf{r}_{\mathsf{out}}} \cdot \mathsf{l}_{\mathsf{in}}$ and $\mathsf{tp}_{\mathsf{out}} + 2^{\mathsf{r}_{\mathsf{out}}} \cdot \mathsf{tp}_{\mathsf{in}}$, because the composed prover uses the inner proof system to generate a proof of proximity *for each choice of randomness of the outer proof system*. Thus, the outer randomness complexity puts a significant efficiency limitation on proof composition.

Avoiding the limitations by interacting. We prove an interactive analogue of proof composition that avoids the above limitations. Our theorem involves two proof systems: the outer proof system is a robust PCP $(P_{\mathsf{out}}, V_{\mathsf{out}})$ for a relation $\mathscr{R}$ (as before), while the inner proof system is a k-round IOP $(P_{\mathsf{in}}, V_{\mathsf{in}})$ for V_{out}'s relation;[1] the composed proof system is a $(\mathsf{k}+1)$-round IOP (P, V) for $\mathscr{R}$. The parameters of the composed proof system are exactly as in the non-interactive case, except that now the new proof length and prover running time are *much smaller*: $\mathsf{l}_{\mathsf{out}} + \mathsf{l}_{\mathsf{in}}$ and $\mathsf{tp}_{\mathsf{out}} + \mathsf{tp}_{\mathsf{in}}$, i.e., there is no multiplicative factor of $2^{\mathsf{r}_{\mathsf{out}}}$ in front of l_{in} and $\mathsf{tp}_{\mathsf{in}}$.

The crucial observation is that, after the prover sends the outer proof to the verifier, *soundness is not harmed if the verifier tells the prover his choice of outer randomness*; in this way, the prover does not have to invest work for all randomness choices but can simply invest work for the outer randomness that was actually chosen because he now knows this choice. Thus, after receiving the outer randomness, the prover and verifier use the inner proof system for proving proximity, to a satisfying assignment, of the oracle locations chosen by this outer randomness.

From sketch to proof. The above sketch glosses over many technical details that, in a proof, need to be addressed.

First, we treated the outer verifier as a unit, while instead it consists of two algorithms: a query algorithm that, given the instance $\mathbb{x}$, outputs a state σ and the query set I; and a decision algorithm that, given the state σ and answers to these queries, outputs a bit. The composed prover and verifier explicitly invoke only the query algorithm; the dependence on the decision algorithm is only implicit, via the invocation of the inner proof system on its induced relation. This requires additional bookkeeping in parameters, e.g., the instance size for the inner proof system is not $|\mathbb{x}|$ but $|\sigma|$, which is denoted $\mathsf{s}_{\mathsf{out}}(|\mathbb{x}|)$. These details also arise in the non-interactive case [BGH$^+$06].

Second, most treatments of proof composition obtain a composed verifier that, despite obtaining savings in query complexity, still runs in time that is at least the outer query complexity, because the outer query algorithm outputs the query set I; such uses of proof composition offer no savings in verifier running time [PS94, HS00, GS06, BSVW03,

[1] We could also make the outer proof system interactive by consider a robust IOP but we do not make use of this additional degree of freedom.

BGH$^+$06, BS08]. In contrast, [BGH$^+$05] introduce the notion of *verifier specification*, which allows the query algorithm to implicitly specify I (e.g., via an algorithm), potentially running in time that is polylogarithmic in the query complexity; then, they prove a (non-interactive) composition theorem that leverages this notion to obtain a composed verifier whose running time may be much smaller than the outer verifier's. We have adopted [BGH$^+$05]'s definitions (see Section 2.2), and our interactive proof composition theorem takes [BGH$^+$05] as a starting point so that we too may benefit from savings in verifier running time. (The main difference of our theorem from [BGH$^+$05]'s is that we do not consider verifier specifications for the composed proof system.)

Finally, analogously to the non-interactive case [BGH$^+$06], if the outer PCP has proximity parameter δ_{out}, then the composed proof system is an IOP with proximity parameter $\delta(n) = \delta_{\mathsf{out}}(n)$; moreover, if the inner IOP has robustness parameter ρ_{in} then the composed proof system is an IOP with robustness parameter $\rho(n) = \rho_{\mathsf{in}}(\mathsf{s}_{\mathsf{out}}(n))$. Our theorem below considers only the first case, ignoring any robustness of the inner proof system since we do not make use of it.

We are now ready to state and prove our interactive composition theorem; we have highlighted in blue the parameters for proof length and prover running time, since these are the main differences from the parameters of a non-interactive composition theorem (such as that in [BGH$^+$05]).

Theorem 3.1 (Interactive Proof Composition — formal statement of Theorem 1.4)**.** *Suppose that the relation $\mathscr{R}$ satisfies the following two conditions with $\delta_{\mathsf{in}}(\mathsf{s}_{\mathsf{out}}(n)) \leq \rho_{\mathsf{out}}(n)$:*

***(1)** there exists a robust PCPP system $(P_{\mathsf{out}}, V_{\mathsf{out}})$ that puts $\mathscr{R}$ in the complexity class*

$$\textbf{PCPP}\left[\begin{array}{ll} \text{answer alphabet} & \mathsf{a} \\ \text{proof length} & \mathsf{l}_{\mathsf{out}} \\ \text{randomness} & \mathsf{r}_{\mathsf{out}} \\ \text{query complexity} & \mathsf{q}_{\mathsf{out}} \\ \text{state size} & \mathsf{s}_{\mathsf{out}} \\ \text{soundness error} & \varepsilon_{\mathsf{out}} \\ \text{proximity parameter} & \delta_{\mathsf{out}} \\ \text{robustness parameter} & \rho_{\mathsf{out}} \\ \text{prover time} & \mathsf{tp}_{\mathsf{out}} \\ \text{verifier query time} & \mathsf{tvq}_{\mathsf{out}} \\ \text{verifier decision time} & \mathsf{tvd}_{\mathsf{out}} \end{array}\right]$$

and

***(2)** there exists an IOPP system $(P_{\mathsf{in}}, V_{\mathsf{in}})$ that puts $\mathrm{Rel}(V_{\mathsf{out}})$ in the complexity class*

$$\textbf{IOPP}\left[\begin{array}{ll} \text{rounds} & \mathsf{k}_{\mathsf{in}} \\ \text{answer alphabet} & \mathsf{a} \\ \text{proof length} & \mathsf{l}_{\mathsf{in}} \\ \text{randomness} & \mathsf{r}_{\mathsf{in}} \\ \text{query complexity} & \mathsf{q}_{\mathsf{in}} \\ \text{soundness error} & \varepsilon_{\mathsf{in}} \\ \text{proximity parameter} & \delta_{\mathsf{in}} \\ \text{prover time} & \mathsf{tp}_{\mathsf{in}} \\ \text{verifier time} & \mathsf{tv}_{\mathsf{in}} \end{array}\right]$$

Then there exists an IOPP system (P, V) that puts $\mathscr{R}$ in the complexity class

$$\textbf{IOPP}\left[\begin{array}{llcl} \text{rounds} & \mathsf{k}(n) & = & 1 + \mathsf{k}_{\mathsf{in}}(\mathsf{s}_{\mathsf{out}}(n)) \\ \text{answer alphabet} & \mathsf{a}(n) & & \\ \text{proof length} & \mathsf{l}(n) & = & \mathsf{l}_{\mathsf{out}}(n) + \mathsf{l}_{\mathsf{in}}(\mathsf{s}_{\mathsf{out}}(n)) \\ \text{randomness} & \mathsf{r}(n) & = & \mathsf{r}_{\mathsf{out}}(n) + \mathsf{r}_{\mathsf{in}}(\mathsf{s}_{\mathsf{out}}(n)) \\ \text{query complexity} & \mathsf{q}(n) & = & \mathsf{q}_{\mathsf{in}}(\mathsf{s}_{\mathsf{out}}(n)) \\ \text{soundness error} & \varepsilon(n) & = & \varepsilon_{\mathsf{out}}(n) + \varepsilon_{\mathsf{in}}(\mathsf{s}_{\mathsf{out}}(n)) - \varepsilon_{\mathsf{out}}(n)\varepsilon_{\mathsf{in}}(\mathsf{s}_{\mathsf{out}}(n)) \\ \text{proximity parameter} & \delta(n) & = & \delta_{\mathsf{out}}(n) \\ \text{prover time} & \mathsf{tp}(n) & = & \mathsf{tp}_{\mathsf{out}}(n) + \mathsf{tvq}_{\mathsf{out}}(n) + \mathsf{tp}_{\mathsf{in}}(\mathsf{s}_{\mathsf{out}}(n)) \\ \text{verifier time} & \mathsf{tv}(n) & = & \mathsf{tvq}_{\mathsf{out}}(n) + \mathsf{tv}_{\mathsf{in}}(\mathsf{s}_{\mathsf{out}}(n)) \end{array}\right].$$

Moreover, if V_{in}'s queries are non-adaptive so are V's queries; also, if V_{in} is public coin so is V.

Proof. We construct the IOPP system (P, V), then analyze its completeness, its soundness, and efficiency parameters.

Construction. Construct the IOPP system (P, V) as follows. Let $(\mathbb{x}, \mathbb{w})$ be an instance-witness pair in the relation $\mathscr{R}$ with $n := |\mathbb{x}|$; the prover P receives $(\mathbb{x}, \mathbb{w})$ as input, while the verifier V receives $\mathbb{x}$ as input and $\mathbb{w}$ as oracle. In the first round: V sends an empty message to P; then P computes the proximity proof $\pi_{\mathsf{out}} \leftarrow P_{\mathsf{out}}(\mathbb{x}, \mathbb{w})$ and sends π_{out} to V. Next, V samples randomness r_{out} for $V_{\mathsf{out}}^{\mathrm{Q}}(\mathbb{x})$ and sends r_{out} to P; both P and V compute $(\sigma, I) \leftarrow V_{\mathsf{out}}^{\mathrm{Q}}(\mathbb{x}; r_{\mathsf{out}})$; in parallel, P and V engage in an interactive oracle protocol attesting to the proximity of $(\mathbb{w}\|\pi_{\mathsf{out}})|_I$ to $\mathrm{Rel}(V)|_\sigma$, by invoking $P_{\mathsf{in}}(\sigma, (\mathbb{w}\|\pi_{\mathsf{out}})|_I)$ and $V_{\mathsf{in}}^{(\mathbb{w}\|\pi_{\mathsf{out}})|_I}(\sigma)$, respectively. The verifier V accepts if and only if V_{in} does.

Completeness. Completeness of (P, V) follows from that of $(P_{\mathsf{out}}, V_{\mathsf{out}})$ and $(P_{\mathsf{in}}, V_{\mathsf{in}})$. Namely, for every instance-witness pair $(\mathbb{x}, \mathbb{w})$ in the relation $\mathscr{R}$, $(\sigma, (\mathbb{w} \| \pi_{\mathsf{out}})|_I) \in \mathrm{Rel}(V_{\mathsf{out}})$ with probability 1, where $\pi_{\mathsf{out}} \leftarrow P_{\mathsf{out}}(\mathbb{x}, \mathbb{w})$ and $(\sigma, I) \leftarrow V_{\mathsf{out}}^{\mathrm{Q}}(\mathbb{x})$. Hence, when interacting with $P_{\mathsf{in}}(\sigma, (\mathbb{w} \| \pi_{\mathsf{out}})|_I)$, $V_{\mathsf{in}}^{(\mathbb{w} \| \pi_{\mathsf{out}})|_I}(\sigma)$ accepts with probability 1.

Soundness. Soundness of (P, V) follows from that of $(P_{\mathsf{out}}, V_{\mathsf{out}})$ and $(P_{\mathsf{in}}, V_{\mathsf{in}})$, as we now explain. Consider any instance-witness pair $(\mathbb{x}, \mathbb{w})$ and with $\Delta(\mathbb{w}, \mathscr{R}|_{\mathbb{x}}) \geq \delta_{\mathsf{out}}(n)$, and an unbounded malicious prover $\tilde{P}$. Letting $\tilde{\pi}_{\mathsf{out}}$ be $\tilde{P}$'s first message, we know that $\Delta((\mathbb{w} \| \tilde{\pi}_{\mathsf{out}})|_I, \mathrm{Rel}(V_{\mathsf{out}})|_\sigma) \leq \rho_{\mathsf{out}}(n)$ with probability at most $\varepsilon_{\mathsf{out}}(n)$ over r_{out}, where $(\sigma, I) \leftarrow V^{\mathrm{Q}}(\mathbb{x}; r_{\mathsf{out}})$. Call r_{out} bad if the distance in the previous sentence is at most $\rho_{\mathsf{out}}(n)$; else call r_{out} good. For any bad r_{out}, we know only that $V_{\mathsf{in}}^{(\mathbb{w} \| \tilde{\pi}_{\mathsf{out}})|_I}(\sigma)$ accepts with probability at most 1; for any good r_{out}, we know that $V_{\mathsf{in}}^{(\mathbb{w} \| \tilde{\pi}_{\mathsf{out}})|_I}(\sigma)$ accepts with probability at most $\varepsilon_{\mathsf{in}}(\mathsf{s}_{\mathsf{out}}(n))$, because of the hypothesis that $\delta_{\mathsf{in}}(\mathsf{s}_{\mathsf{out}}(n)) \leq \rho_{\mathsf{out}}(n)$. Overall, we deduce that V accepts with probability at most $\varepsilon_{\mathsf{out}}(n) \cdot 1 + (1 - \varepsilon_{\mathsf{out}}(n)) \cdot \varepsilon_{\mathsf{in}}(\mathsf{s}_{\mathsf{out}}(n))$.

Efficiency parameters. The constructed IOPP system has $\mathsf{k}(n) := 1 + \mathsf{k}_{\mathsf{in}}(\mathsf{s}_{\mathsf{out}}(n))$ rounds, because in the first round the verifier sends an empty message and the prover replies with a proximity proof; in the remaining $\mathsf{k}_{\mathsf{in}}(\mathsf{s}_{\mathsf{out}}(n))$ rounds, the prover and verifier run $(P_{\mathsf{in}}, V_{\mathsf{in}})$ on σ (and note that the random string r_{out} can be sent in parallel to the first verifier message of $(P_{\mathsf{in}}, V_{\mathsf{in}})$). The proof length is $\mathsf{l}(n) := \mathsf{l}_{\mathsf{out}}(n) + \mathsf{l}_{\mathsf{in}}(\mathsf{s}_{\mathsf{out}}(n))$ because the first term accounts for the proximity proof and the second for the prover messages in the subsequent interactive oracle protocol. The randomness complexity is $\mathsf{r}(n) := \mathsf{r}_{\mathsf{out}}(n) + \mathsf{r}_{\mathsf{in}}(\mathsf{s}_{\mathsf{out}}(n))$ because the first term accounts for running $V_{\mathsf{out}}^{\mathrm{Q}}$ and the second term for running V_{in}. The query complexity is $\mathsf{q}(n) := \mathsf{q}_{\mathsf{in}}(\mathsf{s}_{\mathsf{out}}(n))$ because verifier queries are due only to V_{in}. The prover running time is $\mathsf{tp}(n) := \mathsf{tp}_{\mathsf{out}}(n) + \mathsf{tvq}_{\mathsf{out}}(n) + \mathsf{tp}_{\mathsf{in}}(\mathsf{s}_{\mathsf{out}}(n))$ because the prover runs P_{out} and $V_{\mathsf{out}}^{\mathrm{Q}}$ on $\mathbb{x}$ and P_{in} on σ. The verifier running time is $\mathsf{tv}(n) := \mathsf{tvq}_{\mathsf{out}}(n) + \mathsf{tv}_{\mathsf{in}}(\mathsf{s}_{\mathsf{out}}(n))$. Finally, the construction clearly preserves non-adaptivity of queries and public coins, if present. $\square$

4 Sublinear sumcheck

We show how to use IOPs to obtain a sumcheck protocol in which the verifier complexity is *sublinear* in the individual degree of the polynomial being verified. More generally, we phrase our result for tensor product codes [Wol65, WE63, Tan81], and the verifier complexity is then sublinear in the block length of one copy of the code. Later on, we leverage this result to obtain interactive oracle proofs for circuit satisfiability (see Section 7). We now review the sumcheck protocol, discuss intuition behind our result, and then formally state and prove it.

The sumcheck protocol. The sumcheck protocol [LFKN92, Sha92] is a fundamental building block of numerous results in complexity theory and cryptography. The protocol consists of an interactive proof for the claim "$\sum_{\vec{\alpha} \in H^m} w(\vec{\alpha}) = 0$", where w is the evaluation on $\mathbb{F}^m$ of an m-variate polynomial of individual degree d and H is a subset of $\mathbb{F}$. The prover receives H and w as input, while the verifier receives H as input and w as an oracle. The sumcheck protocol has soundness error $1 - (1 - \frac{d}{|\mathbb{F}|})^m$, the prover runs in time $\mathrm{poly}(|\mathbb{F}|^m)$, the verifier runs in time $\mathrm{poly}(|\mathbb{F}| + m)$, the communication complexity is $\mathrm{poly}(|\mathbb{F}| + m)$, and the number of rounds is m; moreover, the protocol is public coin and the verifier queries w only at one random index.[2]

Limitations, and how to avoid them. In each of the m rounds, the prover sends to the verifier the evaluation on $\mathbb{F}$ of a univariate polynomial of degree d or, alternatively, the prover sends the coefficients of this polynomial; then the verifier checks that the sum of this polynomial over H equals a certain value determined in the previous round. In particular, the verifier reads $\Omega(md)$ bits and its running time is also $\Omega(md)$. We show that the verifier complexity can be *sublinear* in d, if the prover and verifier engage in an interactive oracle proof (rather than in an interactive proof).

Recall that, in an IOP, the verifier has oracle access to the prover's messages, so the verifier may read as many locations of these as are sufficient to perform the necessary checks. The intuition to "go sublinear" is simple: instead of performing these checks explicitly, the verifier relies on proximity testers for doing them. Thus, in each of the m rounds, the prover sends to the verifier two oracles: the evaluation on $\mathbb{F}$ of a univariate polynomial of degree d, and a proximity proof attesting that this evaluation has degree d and has the appropriate sum over H. The use of proximity proofs somewhat complicates the soundness analysis (e.g., the verifier only sees noisy codewords) but the backbone of the proof follows that of the standard sumcheck protocol; overall, this high level intuition can be turned into a proof.

More generally, instead of sending (non-interactive) proximity proofs, the prover may interact with the verifier in an interactive oracle sub-proof of proximity for the appropriate codewords.

Beyond Reed–Muller. The sumcheck protocol can be phrased in a more general setting [Mei13]: an interactive proof for the claim "$\sum_{\vec{\alpha} \in H^m} w(\vec{\alpha}) = 0$" where w is a codeword in the tensor product code $C^{\otimes m}$, for a given linear code C with domain D and alphabet $\mathbb{F}$, and H is a subset of D. Low-degree polynomials are a special case: the Reed–Muller code is a tensor of the Reed–Solomon code. Conveniently, the parameters in the more general case are analogous to the one of low-degree polynomials: the soundness error becomes $1 - \tau^m$ where τ is C's relative distance ($\tau = 1 - \frac{d}{|\mathbb{F}|}$ for the Reed–Solomon code), and C's block length ℓ replaces the field size $|\mathbb{F}|$ in the running times and communication complexity. In particular, the verifier reads $\Omega(m\ell)$ bits and its running time is also $\Omega(m\ell)$. Below, we phrase our sublinear sumcheck result in the language of tensor product codes not only because of the greater generality but also because we invoke this result on algebraic-geometry codes that are not the usual Reed–Muller codes (see Section 7).

We state our theorem as a reduction: given a PCP of proximity $(P_{\mathrm{SC}}, V_{\mathrm{SC}})$ for subcodes of the form $C_{H,\gamma} := \{w \in C \text{ s.t. } \sum_{\alpha \in H} w(\alpha) = \gamma\}$, we construct an IOP of proximity (P, V) for sumchecks over H^m for $C^{\otimes m}$. The complexity of the PCPP verifier V_{SC} determines the complexity of the resulting IOPP verifier V; e.g., if the former is sublinear in C's block length ℓ, so is the latter. In fact, we find it more natural to state the theorem without assuming that w is promised to be a codeword in $C^{\otimes m}$, so the reduction also takes as input a PCP of proximity $(P_{\otimes}, V_{\otimes})$ for $C^{\otimes m}$ that is invoked on w.[3] More generally, both PCPPs can in fact be IOPPs, and we state our theorem for this more general case.

As an example of instantiation, in the case of low-degree polynomials, one can invoke the theorem with a low-degree test [BFL90, BFLS91, ALM$^+$98, AS03] for the tensor product, and proximity proofs of [BS08] for the subcodes of the Reed–Solomon code; this yields sumcheck for low-degree polynomials where the verifier complexity is polylogarithmic in the individual degree. At the other extreme, one can invoke the theorem with the "trivial" proximity testers that read a codeword in full; this collapses our construction to the standard non-sublinear sumcheck protocol.

[2]The sumcheck protocol's standard analysis yields a soundness error of $\frac{md}{|\mathbb{F}|}$, but a more careful analysis yields the smaller error of $1 - (1 - \frac{d}{|\mathbb{F}|})^m$.

[3]Indeed, since we think of ℓ as large, the setting in which the verifier knows all of w (as in [LFKN92, Sha92]) does not apply in general; that said, one can easily specialize the theorem's statement and proof to the cases where the promise "$w \in C^{\otimes m}$" holds.

Below, we use the following notation. Given a code family $\mathscr{C}$, we denote by $\mathrm{Rel}(\mathscr{C}, m)$ the relation of instance-witness pairs $(\mathbb{x}, \mathbb{w}) = \big((C, m), w\big)$ such that $C \in \mathscr{C}$ and $w \in C^{\otimes m}$. In addition, given a function $\mathscr{H}$ that maps every $C \in \mathscr{C}$ to a subset H of C's domain, we denote by $\mathrm{Rel}(\mathscr{C}, m, \mathscr{H})$ the relation of instance-witness pairs $(\mathbb{x}, \mathbb{w}) = \big((C, m, H, \gamma), w\big)$ such that $C \in \mathscr{C}$, $w \in C^{\otimes m}$, $H = \mathscr{H}(C)$, and $\sum_{\vec{\alpha} \in H^m} w(\vec{\alpha}) = \gamma$. With this notation, our theorem can be viewed as a reduction from proximity testing to $\mathrm{Rel}(\mathscr{C}, m, \mathscr{H})$ to proximity testing to $\mathrm{Rel}(\mathscr{C}, m)$ and $\mathrm{Rel}(\mathscr{C}, 1, \mathscr{H})$. (Note that both a particular code C and subset H need not be represented explicitly, via a generator matrix and a set of indices; they may be represented in a more succinct way, when possible.) When reading the theorem statement, it is helpful to keep in mind that the constructed IOP relies on a single invocation of proximity testing to $\mathrm{Rel}(\mathscr{C}, m)$, and m invocations of proximity testing to $\mathrm{Rel}(\mathscr{C}, 1, \mathscr{H})$; it is also helpful to keep in mind that the term $1 - \tau^m$ in the soundness error (highlighted in blue) is inherited from the standard sumcheck protocol while the other terms are due to proximity testing.

Theorem 4.1 (Sublinear Sumcheck — formal statement of Theorem 1.5)**.** *Let m be a positive integer, $\mathscr{C}$ a family of linear codes with relative distance τ and block length ℓ, and $\mathscr{H}$ a function that maps every $C \in \mathscr{C}$ to a subset H of C's domain. Suppose that the following two conditions hold:*

***(1)** there exists an IOPP system $(P_\otimes, V_\otimes)$ that puts $\mathrm{Rel}(\mathscr{C}, m)$ in the complexity class*

$$\textbf{IOPP}\left[\begin{array}{ll} \text{rounds} & \mathsf{k}_\otimes \\ \text{answer alphabet} & \mathsf{a} \\ \text{proof length} & \mathsf{l}_\otimes \\ \text{randomness} & \mathsf{r}_\otimes \\ \text{query complexity} & \mathsf{q}_\otimes \\ \text{soundness error} & \varepsilon_\otimes \\ \text{proximity parameter} & \delta_\otimes \\ \text{prover time} & \mathsf{tp}_\otimes \\ \text{verifier time} & \mathsf{tv}_\otimes \end{array}\right]$$

and

***(2)** there exists an IOPP system $(P_{\mathrm{SC}}, V_{\mathrm{SC}})$ that puts $\mathrm{Rel}(\mathscr{C}, 1, \mathscr{H})$ in the complexity class*

$$\textbf{IOPP}\left[\begin{array}{ll} \text{rounds} & \mathsf{k}_{\mathrm{SC}} \\ \text{answer alphabet} & \mathsf{a} \\ \text{proof length} & \mathsf{l}_{\mathrm{SC}} \\ \text{randomness} & \mathsf{r}_{\mathrm{SC}} \\ \text{query complexity} & \mathsf{q}_{\mathrm{SC}} \\ \text{soundness error} & \varepsilon_{\mathrm{SC}} \\ \text{proximity parameter} & \delta_{\mathrm{SC}} \\ \text{prover time} & \mathsf{tp}_{\mathrm{SC}} \\ \text{verifier time} & \mathsf{tv}_{\mathrm{SC}} \end{array}\right]$$

Then there exists a public-coin IOPP system (P, V) that puts $\mathrm{Rel}(\mathscr{C}, m, \mathscr{H})$ in the complexity class

$$\textbf{IOPP}\left[\begin{array}{llcl} \text{rounds} & \mathsf{k}(\ell^m) & = & \max\{\mathsf{k}_\otimes(\ell^m), m \cdot \mathsf{k}_{\mathrm{SC}}(\ell)\} \\ \text{answer alphabet} & \mathsf{a} & & \\ \text{proof length} & \mathsf{l}(\ell^m) & = & \mathsf{l}_\otimes(\ell^m) + m \cdot \mathsf{l}_{\mathrm{SC}}(\ell) + m \cdot \ell \\ \text{randomness} & \mathsf{r}(\ell^m) & = & \mathsf{r}_\otimes(\ell^m) + m \cdot \mathsf{r}_{\mathrm{SC}}(\ell) + m \cdot \log(\ell) \\ \text{query complexity} & \mathsf{q}(\ell^m) & = & \mathsf{q}_\otimes(\ell^m) + m \cdot \mathsf{q}_{\mathrm{SC}}(\ell) + m + 1 \\ \text{soundness error} & \varepsilon(\ell^m) & & \\ \text{proximity parameter} & \delta(\ell^m) & = & \delta_\otimes(\ell^m) \\ \text{prover time} & \mathsf{tp}(\ell^m) & = & \mathsf{tp}_\otimes(\ell^m) + m \cdot \mathsf{tp}_{\mathrm{SC}}(\ell) + m \cdot \ell^m \\ \text{verifier time} & \mathsf{tv}(\ell^m) & = & \mathsf{tv}_\otimes(\ell^m) + m \cdot \mathsf{tv}_{\mathrm{SC}}(\ell) + O(m) \end{array}\right]$$

where the soundness error $\varepsilon(\ell^m)$ is

$$\varepsilon(\ell^m) = \max\Big\{\varepsilon_\otimes(\ell^m)\,,\ \varepsilon_{\mathrm{SC}}(\ell)\,,\ 1 - \tau^m + (1 - \delta_\otimes(\ell^m)) + m \cdot (1 - \delta_{\mathrm{SC}}(\ell))\Big\}$$

Moreover, if $V_\otimes$'s and V_{SC}'s queries are non-adaptive, then so are V's.

Proof. We first prove the theorem in the case where both of the given IOPPs are in fact PCPPs (in particular, $\mathsf{k}_\otimes = \mathsf{k}_{\mathrm{SC}} = 1$); at the end of the proof we explain the straightforward extension to the general case. So now, for the case of PCPPs, we construct the IOPP system (P, V), then analyze its completeness, its soundness, and efficiency parameters.

Construction. Construct the IOPP system (P, V) as follows. Let $(\mathbb{x}, \mathbb{w}) = \big((C, m, H, \gamma_0), w\big)$ be an instance-witness pair in the relation $\mathrm{Rel}(\mathscr{C}, m, \mathscr{H})$; the prover P receives the instance and witness as input, while the verifier V receives the instance as input and the witness as an oracle.

- In the first round, the verifier V sends an empty message to P; next, the prover P proceeds as follows:

- compute the proximity proof $\pi_0 \leftarrow P_\otimes\big((C, m), w\big)$, which attests that w is in the tensor product code $C^{\otimes m}$;
- compute the codeword $w_1 \colon D \to \mathbb{F}$ defined by $w_1(x) := \sum_{a_2,\dots,a_m \in H} w(x, a_2, \dots, a_m)$;
- compute the proximity proof $\pi_1 \leftarrow P_{\mathrm{SC}}\big((C, H, \gamma_0), w_1\big)$, which attests that w_1 is in the subcode C_{H,γ_0};
- send the proof string (π_0, w_1, π_1) to the verifier V.

- For $i = 2, \dots, m$, in the i-th round, the verifier V draws $r_{i-1} \in D$ uniformly and independently at random, and sends r_{i-1} to P; next, the prover P proceeds as follows:
  - compute the codeword $w_i \colon D \to \mathbb{F}$ defined by $w_i(x) := \sum_{a_{i+1},\dots,a_m \in H} w(r_1, \dots, r_{i-1}, x, a_{i+1}, \dots, a_m)$;
  - set $\gamma_{i-1} := w_{i-1}(r_{i-1})$;
  - compute the proximity proof $\pi_i \leftarrow P_{\mathrm{SC}}\big((C, H, \gamma_{i-1}), w_i\big)$, which attests that w_i is in the subcode $C_{H,\gamma_{i-1}}$;
  - send (w_i, π_i) to V.

- After the m-th round, the verifier V proceeds as follows:
  - set $\gamma_i := w_i(r_i)$ for every $i \in \{1, \dots, m\}$;
  - check that $V_\otimes^{w,\pi_0}\big((C, m)\big)$ accepts;
  - check that $V_{\mathrm{SC}}^{w_i,\pi_i}\big((C, H, \gamma_{i-1})\big)$ accepts for every $i \in \{1, \dots, m\}$;
  - check that $w(r_1, \dots, r_m) = \gamma_m$.

Completeness. Completeness of (P, V) follows from that of $(P_\otimes, V_\otimes)$ and $(P_{\mathrm{SC}}, V_{\mathrm{SC}})$, and the fact that the partial sums belong to the appropriate subcodes. Namely, for every instance-witness pair $(\mathbb{x}, \mathbb{w}) = \big((C, m, H, \gamma_0), w\big)$ in the relation $\mathrm{Rel}(\mathscr{C}, m, \mathscr{H})$ it holds that:
- w is in the tensor product code $C^{\otimes m}$, so that $V_\otimes^{w,\pi_0}(C, m)$ accepts with probability 1, and its sum over H^m is γ_0;
- for every $i \in \{1, \dots, m\}$ and $r_1, \dots, r_{i-1} \in D$, the codeword w_i, which depends on $r_1, \dots, r_{i-1}$, is in the code C and its sum over H is $\gamma_{i-1} = w_{i-1}(r_{i-1})$ so that $V_{\mathrm{SC}}^{w_i,\pi_i}\big((C, H, \gamma_{i-1})\big)$ accepts with probability 1;
- $w(r_1, \dots, r_m) = \gamma_m = w_m(r_m)$.

We conclude that P makes V accept with probability 1.

Soundness. Consider any instance-witness pair $(\mathbb{x}, \mathbb{w}) = \big((C, m, H, \gamma_0), w\big)$ and unbounded malicious prover $\tilde{P}$. Suppose for now that w does not sum to γ_0 on H^m but is in the tensor product code $C^{\otimes m}$ and, moreover, each w_i sent by $\tilde{P}$ is in the subcode $C_{H,\gamma_{i-1}}$. In this case, the standard soundness analysis of the sumcheck protocol (when extended to tensor product codes) shows that the probability that the verifier accepts is at most $1 - \tau^m$, where τ is the relative distance of C. However, the verifier does not explicitly check if each w_i is in $C_{H,\gamma_{i-1}}$ but, instead, relies on the PCPP verifier V_{SC} to test proximity to this code; also, the verifier is not guaranteed that w is in $C^{\otimes m}$ but, instead, relies on the PCPP verifier $V_\otimes$ to test proximity to this code. Overall, this means that the verifier incurs additional soundness errors due to the proximity testing and, hence, accessing noisy codewords. We now describe how to account for these.

Suppose that w is $\delta_\otimes(\ell^m)$-far from any codeword in $C^{\otimes m}$ that sums to γ_0 on H^m. We distinguish among the following cases:

- *Case 1: w is $\delta_\otimes(\ell^m)$-far from the tensor product code $C^{\otimes m}$.* In this case, the PCPP verifier $V_\otimes^{w,\pi_0}\big((C, m)\big)$ accepts with probability at most $\varepsilon_\otimes(\ell^m)$.

- *Case 2: there exists i such that w_i is $\delta_{\mathrm{SC}}(\ell)$-far from the subcode $C_{H,\gamma_{i-1}}$.* In this case, the PCPP verifier $V_{\mathrm{SC}}^{w_i,\pi_i}\big((C, H, \gamma_{i-1})\big)$ accepts with probability at most $\varepsilon_{\mathrm{SC}}(\ell)$.

- *Case 3: the above two cases do not happen.* In this case, let $\hat{w}$ be the unique codeword in $C^{\otimes m}$ closest to w and, for each i, let $\hat{w}_i$ be the unique codeword in $C_{H,\gamma_{i-1}}$ that is closest to w_i; recall that proximity parameters are less than the unique-decoding radius (see Section 2.2) so these unique codewords exist. Note that $\hat{w}$ cannot sum to γ_0 on H^m (because we have assumed that w is $\delta_\otimes(\ell^m)$-far from any codeword in $C^{\otimes m}$ that sums to γ_0 on H^m). At this point we apply the standard analysis of the sumcheck, but relative to the codewords $\hat{w}$ and $\hat{w}_1, \dots, \hat{w}_m$: if the verifier has access to these codewords, then the verifier accepts with probability at most $1 - \tau^m$. However the verifier only has access to functions that are close to these, which incurs an additional soundness error of $(1 - \delta_\otimes(\ell^m)) + m \cdot (1 - \delta_{\mathrm{SC}}(\ell))$ because the verifier queries w and $w_1, \dots, w_m$ each at a random location.

We deduce that the verifier accepts with probability that is at most the maximum of the acceptance probability across the three cases, namely,

$$\max\left\{\varepsilon_{\otimes}(\ell^m)\,,\ \varepsilon_{\mathrm{SC}}(\ell)\,,\ 1-\tau^m+(1-\delta_{\otimes}(\ell^m))+m\cdot(1-\delta_{\mathrm{SC}}(\ell))\right\}\ .$$

For any particular choice of code C and explicit bounds on the soundness errors and proximity parameters, the final soundness error can be further improved by "balancing" the above three cases; we do not do so for the general case.

Efficiency parameters. The constructed IOPP system has $\mathsf{k}(\ell^m) := m$ rounds. The proof length is $\mathsf{l}(\ell^m) := \mathsf{l}_{\otimes}(\ell^m) + m\cdot \mathsf{l}_{\mathrm{SC}}(\ell) + m\cdot\ell$ because the prover sends the proximity proof π_0 output by $P_{\otimes}$, m proximity proofs output by P_{SC}, and m codewords with block length ℓ. The randomness complexity is $\mathsf{r}(\ell^m) := \mathsf{r}_{\otimes}(\ell^m)+m\cdot\mathsf{r}_{\mathrm{SC}}(\ell)+m\cdot\log(\ell)$ because the verifier runs $V_{\otimes}$, runs m times V_{SC}, and samples m elements in D. The query complexity is $\mathsf{q}(\ell^m) := \mathsf{q}_{\otimes}(\ell^m) + m\cdot\mathsf{q}_{\otimes}(\ell) + m + 1$ because the verifier runs $V_{\otimes}$, runs m times V_{SC}, and makes $m+1$ additional queries (one to each of $w_1,\dots,w_m,w$). The prover running time is $\mathsf{tp}(\ell^m) := \mathsf{tp}_{\otimes}(\ell^m) + m\cdot\mathsf{tp}_{\mathrm{SC}}(\ell) + m\cdot\ell^m$ because the prover runs $P_{\otimes}$, runs m times P_{SC}, and computes m partial sums over domains of size at most ℓ^m. The verifier running time is $\mathsf{tv}(\ell^m) := \mathsf{tv}_{\otimes}(\ell^m)+m\cdot\mathsf{tv}_{\mathrm{SC}}(\ell)+O(m)$ because the verifier runs $V_{\otimes}$, runs m times V_{SC}, and performs $O(m)$ additional work. Finally, the protocol is clearly public coins.

From PCPPs to IOPPs. The extension from PCPPs to IOPPs is straightforward: whenever the prover would have sent to the verifier a (non-interactive) proof of proximity, the prover now interacts with the verifier in an interactive oracle proof of proximity. Thus, testing proximity of w to $C^{\otimes m}$ takes $\mathsf{k}_{\otimes}(\ell^m)$ rounds, while testing proximity of each of w_i to $C_{H,\gamma_{i-1}}$ takes $\mathsf{k}_{\mathrm{SC}}(\ell)$ rounds. The first can be done in parallel to the second, so the overall number of rounds is now $\max\{\mathsf{k}_{\otimes}(\ell^m), m\cdot\mathsf{k}_{\mathrm{SC}}(\ell)\}$. The rest of the proof, mutatis mutandis, is unaffected. □

5 Short IOPs of proximity with constant query complexity

We use interactive proof composition (see Section 3) to obtain results on proximity testing for notable classes of linear codes: we obtain IOPs of proximity with proof length and query complexity that are not known to be achievable by any PCP of proximity. We consider the following two classes of codes.

- **Additive Reed–Solomon codes** (Section 5.1)

  We show that additive Reed–Solomon codes over binary fields have *IOPs of proximity with linear proof length and constant query complexity*; moreover, 2 rounds of interaction and public coins suffice. In contrast, for these codes, we only know how to construct PCPs of proximity with *quasilinear* proof length and constant query complexity [BS08, Din07, Mie09].

- **Tensor product codes** (Section 5.2)

  We show that tensor product codes have *IOPs of proximity with sublinear proof length and constant query complexity*; moreover, 1 round of interaction and public coins suffice. In contrast, for these codes, we only know how to construct local testers with sublinear query complexity [BS06], or PCPs of proximity with superlinear proof length and constant query complexity [Mie09].

  Special cases of tensor product codes include Reed–Muller codes.

Above, all statements are relative to a constant soundness error (with a necessary linear dependence on the proximity parameter), and involve a polylogarithmic-time verifier. We now describe, state, and prove the above results.

5.1 For additive Reed–Solomon codes

We show that additive Reed–Solomon codes over binary fields have *linear*-size IOPs of proximity with constant query complexity; moreover, 2 rounds of interaction and public coins suffice. The construction follows from one invocation of our interactive composition theorem with [BS08]'s robust PCPs of proximity for additive Reed–Solomon codes as the outer proof system, and [Mie09]'s PCPs of proximity for nondeterministic languages as the inner proof system. See Section 2.6.1 and Section 2.6.3 for these two components; also, see Section 2.6.1 for the definition of the relation $\mathrm{Rel}(\mathbb{F}, \varrho)$, corresponding to the class of additive Reed–Solomon codes, over a binary field $\mathbb{F}$, with fractional degree ϱ.

Informally, [BS08]'s robust PCPs of proximity reduce proximity testing for $\mathrm{Rel}(\mathbb{F}, \varrho)$ from dimension k to dimension $k/2 + O(1)$; this also reduces the query complexity from 2^k to $O(2^{k/2})$. Thus, in our 2-round IOP, in the first round the prover sends a [BS08]-type PCP of proximity for the function over a domain of dimension k and, after receiving the randomness from the verifier, in the second round the prover sends a [Mie09]-type PCP of proximity for the statement that [BS08]'s verifier accepts. Since this statement lies in $\mathbf{NTIME}(\tilde{O}(2^{k/2}))$ and this latter PCP of proximity is quasilinear in the decider running time, we obtain the desired result.

Below, we state the theorem. After the theorem, we also give a weaker theorem that forgoes the use of a "heavy" tool such as [Mie09], incurring a larger round complexity and soundness error but with better concrete constants.

Theorem 5.1 (formal statement of Theorem 1.2)**.** *Fix $\varrho > 0$ and a binary field $\mathbb{F}$, and define $\mathscr{R} := \mathrm{Rel}(\mathbb{F}, \varrho)$ and $\delta_{\mathscr{R}} := \frac{1}{2}\varrho$. For every $\delta \in (0, \delta_{\mathscr{R}})$ there exists a public-coin IOPP system (P, V) that puts $\mathscr{R}$ in the complexity class*

$$\mathbf{IOPP}\left[\begin{array}{lccl} \text{rounds} & \mathsf{k}(k) & = & 2 \\ \text{answer alphabet} & \mathsf{a}(k) & = & \mathbb{F} \\ \text{proof length} & \mathsf{l}(k) & = & O(2^k) \\ \text{randomness} & \mathsf{r}(k) & = & O(k) \\ \text{query complexity} & \mathsf{q}(k) & = & O(1) \\ \text{soundness error} & \varepsilon(k) & = & 1/2 \\ \text{proximity parameter} & \delta(k) & = & \delta \\ \text{prover time} & \mathsf{tp}(k) & = & \tilde{O}(2^k) \\ \text{verifier time} & \mathsf{tv}(k) & = & \mathrm{poly}(k) \end{array}\right].$$

Proof. We invoke the interactive proof composition theorem (Theorem 3.1) as follows.

- The PCPP system for $\mathscr{R} = \mathrm{Rel}(\mathbb{F}, \varrho)$ of Theorem 2.6 as the "outer" proof system.

  We invoke the theorem with the same fractional degree ϱ as in this proof, which gives us $\alpha \in (0,1)$ and $a > 0$ such that for every binary field $\mathbb{F}_{\mathsf{out}}$ and $\delta_{\mathsf{out}} \in (0, \frac{1}{2}\varrho)$ there exists a robust PCPP system $(P_{\mathsf{out}}, V_{\mathsf{out}})$ for $\mathrm{Rel}(\mathbb{F}, \varrho)$ with the parameters described in the theorem statement. In this proof, we choose $\mathbb{F}_{\mathsf{out}} := \mathbb{F}$ and $\delta_{\mathsf{out}} := \delta$.

- The PCPP system for nondeterministic languages of Theorem 2.9 as the "inner" one.

  The relation that we choose is $\mathscr{R}_{\mathsf{in}} := \mathrm{Rel}(V_{\mathsf{out}})$; hence, $\delta_{\mathscr{R}_{\mathsf{in}}} = \delta_{\mathscr{R}} = \frac{1}{2}\varrho$. Because the state size of the outer proof system is $\mathsf{s}_{\mathsf{out}}(k) = k/2 + a$, we deduce that we can decide if $(\sigma, a) \in \mathscr{R}_{\mathsf{in}}$ in $T := \tilde{O}(2^{k/2+a})$ time. We thus get that for every $\delta_{\mathsf{in}} \in (0, \delta_{\mathscr{R}_{\mathsf{in}}})$ and $\varepsilon_{\mathsf{in}} > 0$ there exist $q_{\mathsf{in}} > 0$ and a PCPP system $(P_{\mathsf{in}}, V_{\mathsf{in}})$ for $\mathscr{R}_{\mathsf{in}}$ with the parameters described in the theorem statement. In this proof, we choose $\delta_{\mathsf{in}} := \rho_{\mathsf{out}}(k) = \alpha \cdot \delta_{\mathsf{out}}$ and $\varepsilon_{\mathsf{in}} := \frac{1}{100}$.

This composition gives us an IOPP system (P, V) that puts $\mathscr{R}$ as a subset of the complexity class

$$\mathbf{IOPP}\left[\begin{array}{lll} \text{rounds} & \mathsf{k}(k) & = 1 + \mathsf{k}_{\mathsf{in}}(\mathsf{s}_{\mathsf{out}}(k)) = 1 + \mathsf{k}_{\mathsf{in}}(k/2+a) = 1+1 \\ \text{answer alphabet} & \mathsf{a}(k) & = \mathbb{F} \\ \text{proof length} & \mathsf{l}(k) & = \mathsf{l}_{\mathsf{out}}(k) + \mathsf{l}_{\mathsf{in}}(\mathsf{s}_{\mathsf{out}}(k)) = a2^k + \mathsf{l}_{\mathsf{in}}(k/2+a) = a2^k + \tilde{O}(2^{(k/2+a)}) \\ \text{randomness} & \mathsf{r}(k) & = \mathsf{r}_{\mathsf{out}}(k) + \mathsf{r}_{\mathsf{in}}(\mathsf{s}_{\mathsf{out}}(k)) = (k+a) + \mathsf{r}_{\mathsf{in}}(k/2+a) = (k+a) + (k/2+a) + O(\log(k/2+a)) \\ \text{query complexity} & \mathsf{q}(k) & = \mathsf{q}_{\mathsf{in}}(\mathsf{s}_{\mathsf{out}}(k)) = \mathsf{q}_{\mathsf{in}}(k/2+a) = q_{\mathsf{in}} \\ \text{soundness error} & \varepsilon(k) & = \varepsilon_{\mathsf{out}}(k) + \varepsilon_{\mathsf{in}}(\mathsf{s}_{\mathsf{out}}(k)) - \varepsilon_{\mathsf{out}}(n)\varepsilon_{\mathsf{in}}(\mathsf{s}_{\mathsf{out}}(n)) = 1 - \alpha \cdot \delta_{\mathsf{out}} + \varepsilon_{\mathsf{in}}(k/2+a) = 1 - \alpha \cdot \delta_{\mathsf{out}} \cdot (1 - \varepsilon_{\mathsf{in}}) \\ \text{proximity parameter} & \delta(k) & = \delta_{\mathsf{out}}(k) = \delta_{\mathsf{out}} \\ \text{prover time} & \mathsf{tp}(k) & = \mathsf{tp}_{\mathsf{out}}(k) + \mathsf{tvq}_{\mathsf{out}}(k) + \mathsf{tp}_{\mathsf{in}}(\mathsf{s}_{\mathsf{out}}(k)) = k^a 2^k + (k/2+a) + \mathsf{tp}_{\mathsf{in}}(k/2+a) = k^a 2^{k+a} + (k/2+a) + \tilde{O}(2^{k/2+a}) \\ \text{verifier time} & \mathsf{tv}(k) & = \mathsf{tvq}_{\mathsf{out}}(k) + \mathsf{tv}_{\mathsf{in}}(\mathsf{s}_{\mathsf{out}}(k)) = (k/2+a) + \mathsf{tv}_{\mathsf{in}}(k/2+a) = (k/2+a) + \mathrm{poly}(k/2+a) \end{array}\right],$$

which implies the theorem statement. □

The alternative construction is an IOP of proximity with $O(\log k)$ rounds, and follows from recursively invoking our interactive proof composition theorem $O(\log k)$ times on [BS08]'s PCPs of proximity for additive Reed–Solomon codes. Here we exploit the fact that the relation of [BS08]'s verifier is itself a subrelation of $\mathrm{Rel}(\mathbb{F}, \varrho)$, so that we can again use the same PCP of proximity without going through a generic reduction. While this alternative construction has $O(\log k)$ rounds rather than 2 and a weaker soundness guarantee, the construction is simpler and the underlying constants (other than soundness) are smaller because the "inner" proof system is much less complex.

Theorem 5.2. *Fix $\varrho > 0$ and a binary field $\mathbb{F}$, and define $\mathscr{R} := \mathrm{Rel}(\mathbb{F}, \varrho)$ and $\delta_{\mathscr{R}} := \frac{1}{2}\varrho$. For every $\delta \in (0, \delta_{\mathscr{R}})$ there exists a public-coin IOPP system (P, V) that puts $\mathscr{R}$ in the complexity class*

$$\mathbf{IOPP}\left[\begin{array}{lll} \text{rounds} & \mathsf{k}(k) & = O(\log k) \\ \text{answer alphabet} & \mathsf{a}(k) & = \mathbb{F} \\ \text{proof length} & \mathsf{l}(k) & = O(2^k) \\ \text{randomness} & \mathsf{r}(k) & = O(k) \\ \text{query complexity} & \mathsf{q}(k) & = O(1) \\ \text{soundness error} & \varepsilon(k) & = 1 - \mathrm{poly}(k)\delta \\ \text{proximity parameter} & \delta(k) & = \delta \\ \text{prover time} & \mathsf{tp}(k) & = \tilde{O}(2^k) \\ \text{verifier time} & \mathsf{tv}(k) & = \mathrm{poly}(k) \end{array}\right].$$

5.2 For tensor product codes

We show that tensor product codes have *sublinear*-size IOPs of proximity with constant query complexity; moreover, 1 round of interaction and public coins suffice. The construction follows from one invocation of our interactive composition theorem with [BS06]'s robust local testers as the outer proof system, and [Mie09]'s PCPs of proximity for nondeterministic languages as the inner proof system. (See Section 2.6.2 and Section 2.6.3 for these two components.)

Let $m > 1$ and $i \in \{1, \dots, \log m\}$ be integers, and $C \subseteq \mathbb{F}^D$ a linear code. Also, let $b > 0$ be a constant such that there is an $(\ell^m)^b$-time algorithm for deciding if a function $f \colon D^m \to \mathbb{F}$ belongs to $C^{\otimes m}$; e.g., the algorithm could compute the parity check matrix of C and use it to check every axis-parallel line of f; typically, $b = 1 + o(1)$.

Informally, [BS06]'s robust local tester reduces proximity testing of $C^{\otimes m}$ to $C^{\otimes 2^i}$ by restricting a function $w \colon D^m \to \mathbb{F}$ to a random axis-parallel plane H, provided certain conditions hold (m must be at least 3 and a power of 2, and C's relative distance must be large enough). In our 1-round IOP of proximity, the verifier sends the random plane H to the prover, who replies with a PCP of proximity of [Mie09] for the claim "$w|_H \in C^{\otimes 2^i}$". We then obtain the desired result, provided that [BS06]'s result applies, and the claim "$w|_H \in C^{\otimes 2^i}$" lies in $\mathbf{NTIME}(o(\ell^m))$, where $\ell := |D|$ is C's block length. This latter condition generally holds for $m \geq 3$ and $i \in \{1, \dots, \log m\}$ because it suffices to have $m > b2^i$ (and, as explained above, we typically expect $b = 1 + o(1)$).

We do not prove the theorem via a black box invocation of the interactive composition theorem because we would obtain sub-optimal parameters: the outer proof system is a robust local tester rather than a robust PCPP, so that we would obtain a 2-round IOP of proximity with one empty round. Thus, in the proof below, we perform interactive composition directly for [BS06]'s robust local tester and [Mie09]'s PCPP, obtaining a 1-round IOP of proximity.

Below, we denote by $\mathrm{Rel}(C, m)$ the relation of instance-witness pairs $(\mathbb{x}, \mathbb{w}) = \big((C, m), w\big)$ such that $w \in C^{\otimes m}$. In order to obtain constant soundness, it suffices to set $i = \Omega(\log m)$ while maintaining the condition $m > b2^i$, where b is the constant mentioned above that determines the decider algorithm's runtime.

Theorem 5.3 (formal statement of Theorem 1.3)**.** *Let m be a power of 2, $i \in \{1, \dots, \log m\}$, and C a linear code with domain D, alphabet $\mathbb{F}$, and distance d; let $\ell := |D|$. Suppose that (i) $m \geq 3$, (ii) $(\frac{d-1}{\ell})^m \geq \frac{7}{8}$, and (iii) $m > b2^i$. Define $\mathscr{R} := \mathrm{Rel}(C, m)$ and $\delta_{\mathscr{R}} := \frac{1}{2}(\frac{d}{\ell})^m$. There exists $\alpha_0 \in (0, 1)$ such that for every $\delta \in (0, \delta_{\mathscr{R}})$ there exist $q_0 > 0$ and a public-coin IOPP system (P, V) that puts $\mathscr{R}$ in the complexity class*

$$\mathbf{IOPP}\left[\begin{array}{lccl} \text{rounds} & \mathsf{k}(\ell^m) & = & 1 \\ \text{answer alphabet} & \mathsf{a}(\ell^m) & = & \mathbb{F} \\ \text{proof length} & \mathsf{l}(\ell^m) & = & o(\ell^m) \\ \text{randomness} & \mathsf{r}(\ell^m) & = & O(m \log \ell) \\ \text{query complexity} & \mathsf{q}(\ell^m) & = & q_0 \\ \text{soundness error} & \varepsilon(\ell^m) & = & 1 - \alpha_0^{\frac{\log m}{i}} \delta \\ \text{proximity parameter} & \delta(\ell^m) & = & \delta \\ \text{prover time} & \mathsf{tp}(\ell^m) & = & o(\ell^m) \\ \text{verifier time} & \mathsf{tv}(\ell^m) & = & \mathrm{poly}(m + \log \ell) \end{array}\right].$$

Proof. We construct the IOPP system (P, V), then analyze its completeness, its soundness, and efficiency parameters.

Construction. Construct the IOPP system (P, V) as follows. Let $(\mathbb{x}, \mathbb{w}) = \big((C, m), w\big)$ be an instance-witness pair in the relation $\mathrm{Rel}(C, m)$; the prover P receives the instance and witness as input, while the verifier V receives the instance as input and the witness as oracle. In the first round, the verifier V samples a random 2^i-dimensional plane H in D^m and then sends (the description of) H to P; the prover P then computes a proximity proof π attesting to the statement "$w|_H \in C^{\otimes 2^i}$", and sends the proof string π to V. The verifier V checks π and accepts if the check passes.

We are left to specify which PCPP system to use for this task: we rely on the PCPP system for nondeterministic languages of Theorem 2.9. We apply the theorem to the relation $\mathscr{R}_{\mathsf{in}}$ for which instances are tuples $(C, 2^i)$ and witnesses are functions $f \colon D^{2^i} \to \mathbb{F}$ such that $f \in C^{\otimes 2^i}$; the relation $\mathscr{R}_{\mathsf{in}}$ can be decided in time $T = \ell^{b2^i} = o(\ell^m)$ (since $m > b2^i$). By Theorem 2.9, we obtain that for every $\delta_{\mathsf{in}} \in (0, \delta_{\mathscr{R}_{\mathsf{in}}})$ and $\varepsilon_{\mathsf{in}} > 0$ there exist $q_{\mathsf{in}} > 0$ and a PCPP system $(P_{\mathsf{in}}, V_{\mathsf{in}})$ for $\mathscr{R}_{\mathsf{in}}$ with the parameters described in the theorem statement. In this proof, we choose $\delta_{\mathsf{in}} := \min\{\delta_{\mathscr{R}_{\mathsf{in}}}, \rho - \varepsilon'\}$ and $\varepsilon_{\mathsf{in}} := \frac{1}{100}$, where the constants $\rho, \varepsilon' \in (0, 1)$ are chosen in the soundness analysis below.

Completeness. Consider any instance-witness pair $(\mathbb{x}, \mathbb{w}) = \big((C, m), w\big)$ in the relation $\mathrm{Rel}(C, m)$. By Theorem 2.8, $w|_H \in C^{\otimes 2^i}$ for every 2^i-dimensional plane $H \in D^m$; hence, $V_{\mathsf{in}}^{w|_H, \pi}(C, 2^i)$ always accepts. We conclude that $P(\mathbb{x}, \mathbb{w})$ makes $V^{\mathbb{w}}(\mathbb{x})$ accept with probability 1.

Soundness. Consider any instance-witness pair $(\mathbb{x}, \mathbb{w}) = \big((C, m), w\big)$ and unbounded malicious prover $\tilde{P}$. Suppose that w is $\delta(\ell^m)$-far from any codeword in $C^{\otimes m}$. By Theorem 2.8, the expected distance of $w|_H$ to $C^{\otimes 2^i}$ is at least $\rho := \alpha^{\frac{\log m}{i}} \delta(\ell^m)$ for a universal constant $\alpha \in (0, 1)$. By [BGH⁺06, Proposition 2.10], for any $\varepsilon' \leq \rho$, the distance is at most $\rho - \varepsilon'$ with probability at most $1 - \varepsilon'$. Call H bad if its distance to $C^{\otimes 2^i}$ is at most $\rho - \varepsilon'$; else call H good. For any bad H, we only know that $V_{\mathsf{in}}^{w|_H, \pi}(C, 2^i)$ accepts with probability at most 1; for any good H, we know that $V_{\mathsf{in}}^{w|_H, \pi}(C, 2^i)$ accepts with probability at most $\varepsilon_{\mathsf{in}}$ because $\delta_{\mathsf{in}} \leq \rho - \varepsilon'$. Overall, we deduce that V accepts with probability at most $(1 - \varepsilon') \cdot 1 + \varepsilon' \cdot \varepsilon_{\mathsf{in}} = 1 - \varepsilon' \cdot (1 - \varepsilon_{\mathsf{in}})$. Now we pick $\varepsilon' := \rho/2$, and we conclude that V accepts with probability at most $1 - \rho/2 \cdot (1 - \varepsilon_{\mathsf{in}}) = 1 - \frac{\alpha^{\frac{\log m}{i}} \delta(\ell^m)}{2} \cdot (1 - \frac{1}{100})$, and the claimed soundness follows for $\alpha_0 \in (0,1)$ sufficiently larger than α.

Efficiency parameters. The constructed IOPP system has $\mathsf{k}(\ell^m) := 1$ rounds. The proof length is $\mathsf{l}(\ell^m) := O(\ell^{b2^i}) = o(\ell^m)$ because the prover sends the proximity proof π output by P_{in}. The randomness complexity is $\mathsf{r}(\ell^m) := O(m \log \ell)$ because the verifier samples a random 2^i-dimensional plane in D^m and then runs V_{in}. The query complexity is $\mathsf{q}(\ell^m) := q_{\mathsf{in}}$ because the verifier runs V_{in}. The prover running time is $\mathsf{tp}(\ell^m) := \tilde{O}(\ell^{b2^i}) = o(\ell^m)$ because the prover runs P_{in}. The verifier running time is $\mathsf{tv}(\ell^m) := \mathrm{poly}(m + \log \ell)$ because the verifier samples a random 2^i-dimensional plane in D^m and then runs V_{in}. Finally, the protocol is clearly public coins. □

6 From circuit satisfiability to sumcheck

We prove that, with 1 round of IOP interaction, we can reduce circuit satisfiability to proximity testing to a linear code and a sumcheck over any degree-3 closure of it; moreover, the IOP introduces only constant overheads. We use this reduction in Section 7, along with other ingredients, to construct 3-round IOPs for circuit satisfiability with linear proof length and constant query complexity. We begin by recalling the notion of boolean circuits and their satisfiability.

Definition 6.1. *A* boolean circuit *ϕ with n gates and s inputs is a directed acyclic graph with n vertices of which s are sources and 1 is a sink; vertices represent gates while directed edges represent wires among them. We define ϕ's size to be n, and label the gates as $g_1, \dots, g_n$ so that $g_1, \dots, g_s$ are the input gates and g_n is the output gate. We assume that all gates (except input gates) are two-input NAND gates. Denote by $\ell \colon [n] \to [n]$ and $r \colon [n] \to [n]$ the functions such that $g_{\ell(i)}$ and $g_{r(i)}$ are the gates whose outputs are the left and right inputs of g_i; for $i \in [s]$, the value of $\ell(i)$ and $r(i)$ is arbitrary, e.g., 1. We say that $w \in \{0,1\}^n$ is a* satisfying assignment *for ϕ if $w_n = 0$ and, for every $i = s+1, \dots, n$, w_i is the output of g_i when the input to the circuit is $w_1 \cdots w_s$.*

The relation $\mathscr{R}_{\mathrm{CSAT}}$ comprises all instance-witness pairs (ϕ, w) such that w is a satisfying assignment to ϕ.

In the statement and proof below we use the notion of a systematic code family, described in Section 2.4, and the notion of an *evading set*, described in Section 2.5. Also, given a systematic code family $\mathscr{C} = \{C_n\}_{n \in \mathbb{N}}$, we denote by:
- $\mathrm{Rel}(\mathscr{C})$ the relation of instance-witness pairs (n, w) such that $n \in \mathbb{N}$ and $w \in C_n$; and
- $\mathrm{Rel}(\mathrm{SC}, \mathscr{C})$ the relation of instance-witness pairs (n, w) such that $n \in \mathbb{N}$, $w \in C_n$, and $\sum_{i \in [n]} w(i) = 0$.

Theorem 6.2 (From CSAT to Sumcheck — formal statement of Lemma 1.6)**.** *Suppose that*
- *$\mathscr{C} = \{C_n\}_{n \in \mathbb{N}}$ is a $T_{\mathscr{C}}(\cdot)$-efficient systematic code family with alphabet $\mathbb{F}(\cdot)$,*
- *$\mathscr{D} = \{D_n\}_{n \in \mathbb{N}}$ is a $T_{\mathscr{D}}(\cdot)$-efficient systematic code family with relative distance $\tau_{\mathscr{D}}(\cdot)$ and is a degree-3 closure of $\mathscr{C}$,*
- *$\mathscr{S} = \{S_n\}_{n \in \mathbb{N}}$ is a $T_{\mathscr{S}}(\cdot)$-efficient $\gamma(\cdot)$-evading set family for $\mathbb{F}(\cdot)$ with $\frac{1}{|\mathbb{F}(n)|} \leq \gamma(n)$.*

Suppose further that the following two conditions hold with $\delta_{\mathscr{C}}(n) < \tau_{\mathscr{D}}(n)/16$:

***(1)** there exists an IOPP system $(P_{\mathscr{C}}, V_{\mathscr{C}})$ that puts $\mathrm{Rel}(\mathscr{C})$ in the complexity class*

$$\textbf{IOPP}\left[\begin{array}{ll} \text{rounds} & \mathsf{k}_{\mathscr{C}} \\ \text{answer alphabet} & \mathsf{a} \\ \text{proof length} & \mathsf{l}_{\mathscr{C}} \\ \text{randomness} & \mathsf{r}_{\mathscr{C}} \\ \text{query complexity} & \mathsf{q}_{\mathscr{C}} \\ \text{soundness error} & \varepsilon_{\mathscr{C}} \\ \text{proximity parameter} & \delta_{\mathscr{C}} \\ \text{prover time} & \mathsf{tp}_{\mathscr{C}} \\ \text{verifier time} & \mathsf{tv}_{\mathscr{C}} \end{array}\right]$$

and

***(2)** there exists an IOPP system $(P_{\mathscr{D}}, V_{\mathscr{D}})$ that puts $\mathrm{Rel}(\mathrm{SC}, \mathscr{D})$ in the complexity class*

$$\textbf{IOPP}\left[\begin{array}{ll} \text{rounds} & \mathsf{k}_{\mathscr{D}} \\ \text{answer alphabet} & \mathsf{a} \\ \text{proof length} & \mathsf{l}_{\mathscr{D}} \\ \text{randomness} & \mathsf{r}_{\mathscr{D}} \\ \text{query complexity} & \mathsf{q}_{\mathscr{D}} \\ \text{soundness error} & \varepsilon_{\mathscr{D}} \\ \text{proximity parameter} & \delta_{\mathscr{D}} \\ \text{prover time} & \mathsf{tp}_{\mathscr{D}} \\ \text{verifier time} & \mathsf{tv}_{\mathscr{D}} \end{array}\right]$$

Then there exists an IOP system (P, V) that puts the relation $\mathscr{R}_{\mathrm{CSAT}}$ in the complexity class

$$\textbf{IOP}\left[\begin{array}{llcl} \text{rounds} & \mathsf{k}(n) & = & 1 + \max\{\mathsf{k}_{\mathscr{C}}(n), \mathsf{k}_{\mathscr{D}}(n)\} \\ \text{answer alphabet} & \mathsf{a}(n) & & \\ \text{proof length} & \mathsf{l}(n) & = & 3 \cdot \mathsf{l}_{\mathscr{C}}(n) + \mathsf{l}_{\mathscr{D}}(n) + 3 \cdot \ell_{\mathscr{C}}(n) \\ \text{randomness} & \mathsf{r}(n) & = & 3 \cdot \mathsf{r}_{\mathscr{C}}(n) + \mathsf{r}_{\mathscr{D}}(n) + 4 \cdot \log|S_n| + \log|\mathbb{F}(n)| \\ \text{query complexity} & \mathsf{q}(n) & = & 3 \cdot \mathsf{q}_{\mathscr{C}}(n) + \mathsf{q}_{\mathscr{D}}(n) \\ \text{soundness error} & \varepsilon(n) & = & \max\{\varepsilon_{\mathscr{C}}(n), \varepsilon_{\mathscr{D}}(n) + \gamma(n)\} \\ \text{prover time} & \mathsf{tp}(n) & = & 3 \cdot \mathsf{tp}_{\mathscr{C}}(n) + \mathsf{tp}_{\mathscr{D}}(n) + 8 \cdot T_{\mathscr{C}}(n) + 4 \cdot T_{\mathscr{S}}(n) + T_{\mathscr{D}}(n) + O(n \cdot \log|\mathbb{F}(n)|) \\ \text{verifier time} & \mathsf{tv}(n) & = & 3 \cdot \mathsf{tv}_{\mathscr{C}}(n) + \mathsf{tv}_{\mathscr{D}}(n) + 5 \cdot T_{\mathscr{C}}(n) + 4 \cdot T_{\mathscr{S}}(n) \end{array}\right]$$

Moreover, if $V_{\mathscr{C}}$'s and $V_{\mathscr{D}}$'s queries are non-adaptive so are V's queries; also, if $V_{\mathscr{C}}$ and $V_{\mathscr{D}}$ is public coin so is V.

We first give a simple lemma that says that a circuit's satisfiability can be represented as a set of low-degree constraints on three codewords; these codewords represent encodings of all gates' outputs, left inputs, and right inputs. Similar statements appear in several prior works that encode computation via, e.g., low-degree polynomials.

Lemma 6.3. *Let ϕ be a boolean circuit with n gates and s inputs, C an n-systematic code with alphabet $\mathbb{F}$, and $P_{\text{NAND}} \colon \mathbb{F}^3 \to \mathbb{F}$ the polynomial of total degree 2 that describes a NAND gate (for every $x, y, z \in \{0,1\}$, $P_{\text{NAND}}(x,y,z) = 0$ if and only if the NAND of x and y equals z). There exists a satisfying assignment for ϕ if and only if there exist codewords $W, L, R \in C$ that satisfy the following:*

- (wiring constraints) *for every $i \in [n]$, $L(i) = W(\ell(i))$ and $R(i) = W(r(i))$;*
- (satisfiability constraints) *$W(n) = 0$ and, for every $i \in [n] \setminus [s]$, $P_{\text{NAND}}(L(i), R(i), W(i)) = 0$;*
- (booleanity constraints) *for every $i \in [n]$, $W(i)^2 = W(i)$.*

Proof. For every $w \in \{0,1\}^n \subseteq \mathbb{F}^n$, w is a satisfying assignment for ϕ if and only if $w_n = 0$ and $P_{\text{NAND}}(w_{\ell(i)}, w_{r(i)}, w_i) = 0$ for every $i \in [n] \setminus [s]$. With this in mind, we can argue the two sides.

Let $w \in \{0,1\}^n$ be a satisfying assignment for ϕ. Let W, L, R be the codewords in C such that, for every $i \in [n]$, $W(i) = w_i$, $L(i) = w_{\ell(i)}$, and $R(i) = w_{r(i)}$; such codes exist because C is n-systematic. One can verify that this choice of codewords fulfills the constraints in the statement.

Conversely, let W, L, R be codewords in C that satisfy the constraints in the statement. Let w be the assignment that equals the codeword W restricted to $[n]$. One can verify that this choice of assignment is satisfying for ϕ. □

We now return to the proof of the theorem.

Proof of Theorem 6.2. We construct the IOPP system (P, V), then analyze its completeness, its soundness, and efficiency parameters.

Construction. Construct the IOPP system (P, V) as follows. Let (ϕ, w) be an instance-witness pair in the circuit-satisfiability relation $\mathscr{R}_{\text{CSAT}}$; the prover P receives the instance and witness as input, while the verifier V receives the instance as input and the witness as an oracle.

- In the first round, the verifier V sends an empty message to P; next, the prover P proceeds as follows: (i) compute $W, L, R \in C_n$ from the assignment $w \in \{0,1\}^n$ so that, for every $i \in [n]$, $W(i) = w_i$, $L(i) = w_{\ell(i)}$, and $R(i) = w_{r(i)}$ (as in the proof of Lemma 6.3); (ii) send the proof string (W, L, R) to the verifier V.

- In the second round, the verifier chooses uniformly (and independently) at random $r, r', t, t' \in S_n$ and $\alpha \in \mathbb{F}(n)$, and sends these to P; here S_n is the $\gamma(n)$-evading set for $\mathbb{F}(n)^n$ in the family $\mathscr{S}$. This randomness induces the codewords $R_1, R_2, R_3, T_1, T_2 \in C_n$ defined as follows:

$$\begin{aligned}
\forall\, i \in [n-1]\,, &\quad R_1(i) = -\left(\textstyle\sum_{j:\ell(j)=i} r_j + \sum_{j:r(j)=i} r'_j\right) \quad \text{and} \quad R_1(n) = \alpha \\
\forall\, i \in [n]\,, &\quad R_2(i) = r_i \\
\forall\, i \in [n]\,, &\quad R_3(i) = r'_i \\
\forall\, i \in [n] \setminus [s]\,, &\quad T_1(i) = t_i \quad \text{and} \quad \forall\, i \in [s] \quad T_1(i) = 0 \\
\forall\, i \in [n]\,, &\quad T_2(i) = t'_i
\end{aligned}$$

The prover P and verifier V may compute these codewords, which in turn induce the codeword H in D_n defined as

$$H := R_1 \cdot W + R_2 \cdot L + R_3 \cdot R + T_1 \cdot P_{\text{NAND}}(L, R, W) + T_2 \cdot (W^2 + W)$$

Indeed, note that H equals $Q(R_1, R_2, R_3, T_1, T_2, W, L, R)$ for a polynomial Q of total degree 3.

In parallel, the prover P and verifier V engage in several interactive oracle proofs:

- an IOPP $(P_{\mathscr{C}}, V_{\mathscr{C}})$ to prove proximity of (n, W) to $\text{Rel}(\mathscr{C})$;
- an IOPP $(P_{\mathscr{C}}, V_{\mathscr{C}})$ to prove proximity of (n, L) to $\text{Rel}(\mathscr{C})$;
- an IOPP $(P_{\mathscr{C}}, V_{\mathscr{C}})$ to prove proximity of (n, R) to $\text{Rel}(\mathscr{C})$;
- an IOPP $(P_{\mathscr{D}}, V_{\mathscr{D}})$ to prove proximity of (n, H) to $\text{Rel}(\text{SC}, \mathscr{D})$.

Completeness. Completeness of (P, V) follows from that of $(P_{\mathscr{C}}, V_{\mathscr{C}})$ and $(P_{\mathscr{D}}, V_{\mathscr{D}})$, and the fact that the sum of H over $[n]$ equals 0. Namely, for every instance-witness pair (ϕ, w) in the relation $\mathscr{R}_{\text{CSAT}}$ it holds that:

- In the first round, the prover P sends three codewords W, L, R that are in the code C_n; by construction, and since w is a satisfying assignment for ϕ, we know that W, L, R satisfy the conditions in Lemma 6.3.
- In the second round, for any choice of verifier randomness, the codewords R_1, R_2, R_3, T_1, T_2 are in the code C_n.
- The codeword H is derived from the above codewords via a polynomial Q of total degree 3 so that H is in the code D_n, because D_n is a degree-3 closure of C_n. Moreover, H sums to 0 on $[n]$ because:

$$\begin{aligned}\sum_{i\in[n]} H(i) = &\sum_{i\in[n]} R_2(i)\cdot\Big(L(i)-W(\ell(i))\Big)+\sum_{i\in[n]} R_3(i)\cdot\Big(R(i)-W(r(i))\Big)\\ &+\sum_{i\in[n]\setminus[s]} T_1(i)\cdot P_{\text{NAND}}\Big(L(i),R(i),W(i)\Big)+\sum_{i\in[n]} T_2(i)\cdot\Big(W(i)^2-W(i)\Big)+\alpha\cdot W(n)\\ =&\sum_{i\in[n]} r_i\cdot\Big(L(i)-W(\ell(i))\Big)+\sum_{i\in[n]} r'_i\cdot\Big(R(i)-W(r(i))\Big)\\ &+\sum_{i\in[n]\setminus[s]} t_i\cdot P_{\text{NAND}}\Big(L(i),R(i),W(i)\Big)+\sum_{i\in[n]} t'_i\cdot\Big(W(i)^2-W(i)\Big)+\alpha\cdot W(n)\\ =&\sum_{i\in[n]} r_i\cdot 0+\sum_{i\in[n]} r'_i\cdot 0+\sum_{i\in[n]\setminus[s]} t_i\cdot 0+\sum_{i\in[n]} t'_i\cdot 0+\alpha\cdot 0=0\ .\end{aligned}$$

- Hence, also in the second round (and any later rounds): the use of the IOPP system $(P_{\mathscr{C}}, V_{\mathscr{C}})$ to separately prove proximity of W, L, R to C_n and the use of the IOPP system $(P_{\mathscr{D}}, V_{\mathscr{D}})$ to prove proximity of H to the subcode of D_n of codewords that sum to 0 on $[n]$ results in the verifier V accepting with probability 1.

Soundness. Consider any unsatisfiable boolean circuit ϕ, and unbounded malicious prover $\tilde{P}$. We distinguish among the following cases:

- *Case 1: one of W, L, R sent by $\tilde{P}$ in the first round is $\delta_{\mathscr{C}}(n)$-far from C_n.*

  In this case, the IOPP verifier $V_{\mathscr{C}}$ accepts with probability at most $\varepsilon_{\mathscr{C}}(n)$.

- *Case 2: the above case does not hold.*

  Observe that H is a random variable that depends on the verifier randomness $\chi := (r, r', t, t', \alpha)$. Let A be the set of χ for which H is $\delta_{\mathscr{D}}(n)$-far from D_n.

  For any $\chi \in A$, the IOPP verifier $V_{\mathscr{D}}$ accepts with probability at most $\varepsilon_{\mathscr{D}}(n)$.

  For any $\chi \notin A$, let $\hat{W}, \hat{L}, \hat{R}$ be the unique codewords in C_n that are closest to W, L, R (respectively); also, let $\hat{H}$ be the unique codeword in D_n that is closest to H. (Recall that proximity parameters are less than the unique-decoding radius, so such codewords exist; see Section 2.2.) By hypothesis, $\delta_{\mathscr{C}}(n) < \tau(D_n)/16$; hence, by Claim 2.2 (invoked for C_n, D_n, and $m = 8$), we deduce that $\hat{H} = Q(R_1, R_2, R_3, T_1, T_2, \hat{W}, \hat{L}, \hat{R})$. Denote by $\varepsilon_{\hat{H}}$ the probability over χ, conditioned on $\chi \notin A$, that $\hat{H}$ sums to 0 on $[n]$.

  If ε is the probability that the verifier accepts in this case, we can write

$$\begin{aligned}\varepsilon &= \Pr[\chi\in A]\cdot\varepsilon_{\mathscr{D}}(n)+\Pr[\chi\notin A]\cdot\big(\varepsilon_{\hat{H}}\cdot 1+(1-\varepsilon_{\hat{H}})\cdot\varepsilon_{\mathscr{D}}(n)\big)\\ &\le \max\big\{\varepsilon_{\mathscr{D}}(n),\varepsilon_{\hat{H}}\cdot 1+(1-\varepsilon_{\hat{H}})\cdot\varepsilon_{\mathscr{D}}(n)\big\}\\ &\le \max\big\{\varepsilon_{\mathscr{D}}(n),\varepsilon_{\mathscr{D}}(n)+\varepsilon_{\hat{H}}\big\}=\varepsilon_{\mathscr{D}}(n)+\varepsilon_{\hat{H}}\end{aligned}$$

  so we are left to upper bound $\varepsilon_{\hat{H}}$.

  Since ϕ is unsatisfiable, $\hat{W}, \hat{L}, \hat{R}$ do not satisfy the conditions in Lemma 6.3. Therefore

$$\begin{aligned}\sum_{i\in[n]} \hat{H}(i) = &\sum_{i\in[n]} r_i\cdot\Big(\hat{L}(i)-\hat{W}(\ell(i))\Big)+\sum_{i\in[n]} r'_i\cdot\Big(\hat{R}(i)-\hat{W}(r(i))\Big)\\ &+\sum_{i\in[n]\setminus[s]} t_i\cdot P_{\text{NAND}}\Big(\hat{L}(i),\hat{R}(i),\hat{W}(i)\Big)+\sum_{i\in[n]} t'_i\cdot\Big(\hat{W}(i)^2-\hat{W}(i)\Big)+\alpha\cdot\hat{W}(n)\end{aligned}$$

  is zero with probability at most $\gamma(n)$, as we now explain. Consider two sub-cases:

- *Case 2.a:* $\hat{W}(n) \neq 0$. In this case $\varepsilon_{\hat{H}} \leq \frac{1}{|\mathbb{F}(n)|}$ because α is uniformly random in $\mathbb{F}(n)$.
- *Case 2.b:* $\hat{W}(n) = 0$. One of the four sums is an inner product of a non-zero vector with a uniformly random element in the $\gamma(n)$-evading set S_n; hence, $\varepsilon_{\hat{H}} \leq \gamma(n)$.

Recalling that $\frac{1}{|\mathbb{F}(n)|} \leq \gamma(n)$ by hypothesis, we deduce that $\varepsilon_{\hat{H}} \leq \gamma(n)$.

We deduce that the verifier accepts with probability that is at most the maximum of the acceptance probability across the two cases, namely, $\max\{\varepsilon_{\mathscr{C}}(n), \varepsilon_{\mathscr{D}}(n) + \gamma(n)\}$.

Efficiency parameters. The constructed IOP system has $\mathsf{k}(n) := 1 + \max\{\mathsf{k}_{\mathscr{C}}(n), \mathsf{k}_{\mathscr{D}}(n)\}$ rounds. The proof length is $\mathsf{l}(n) := 3 \cdot \mathsf{l}_{\mathscr{C}}(n) + \mathsf{l}_{\mathscr{D}}(n) + 3 \cdot \ell_{\mathscr{C}}(n)$ because the prover sends the three codewords W, L, R in C_n, and also runs three invocations of $P_{\mathscr{C}}$ and one invocation of $P_{\mathscr{D}}$. The randomness complexity is $\mathsf{r}(n) := 3 \cdot \mathsf{r}_{\mathscr{C}}(n) + \mathsf{r}_{\mathscr{D}}(n) + 4 \cdot \log|S_n| + \log|\mathbb{F}(n)|$ because the verifier runs three invocations of $V_{\mathscr{C}}$ and one invocation of $V_{\mathscr{D}}$, samples four elements from the evading set S_n, and also one element from $\mathbb{F}(n)$. The query complexity is $\mathsf{q}(n) := 3 \cdot \mathsf{q}_{\mathscr{C}}(n) + \mathsf{q}_{\mathscr{D}}(n)$ because the verifier runs three invocations of $V_{\mathscr{C}}$ and one invocation of $V_{\mathscr{D}}$, and makes no other queries otherwise. The prover running time is $\mathsf{tp}(n) := 3 \cdot \mathsf{tp}_{\mathscr{C}}(n) + \mathsf{tp}_{\mathscr{D}}(n) + 8 \cdot T_{\mathscr{C}}(n) + T_{\mathscr{D}}(n) + O(n \cdot \log|\mathbb{F}(n)|)$ because the prover encodes the circuit assignment to obtain the three codewords W, L, R in C_n, computes the four vectors sampled by the verifier from the evading set, encodes these four vectors, along with a fifth derived from them, to obtain the five codewords R_1, R_2, R_3, T_1, T_2 in C_n, coordinate-wise computes a message and encodes it to obtain the codeword H in D_n, and then runs three invocations of $P_{\mathscr{C}}$ and one invocation of $P_{\mathscr{D}}$. The verifier running time is $\mathsf{tv}(n) := 3 \cdot \mathsf{tv}_{\mathscr{C}}(n) + \mathsf{tv}_{\mathscr{D}}(n) + 5 \cdot T_{\mathscr{C}}(n) + 4 \cdot T_{\mathscr{S}}(n)$ because the verifier samples the four vectors from the evading set, also computes the codewords R_1, R_2, R_3, T_1, T_2, and then runs three invocations of $V_{\mathscr{C}}$ and one invocation of $V_{\mathscr{D}}$. $\square$

7 IOP for circuit satisfiability

We show that for circuit satisfiability we can obtain IOPs with linear proof length and constant query complexity; moreover, 3 rounds of interaction and public coins suffice.

Theorem 7.1 (formal statement of Theorem 1.1)**.** *Let $\mathscr{R}_{\mathrm{CSAT}}$ be the relation consisting of instance-witness pairs (ϕ, w) where ϕ is a boolean circuit (of two-input NAND gates) and w is a binary input that satisfies ϕ; we use n to denote the number of gates in ϕ. There exists a public-coin IOP system that puts $\mathscr{R}_{\mathrm{CSAT}}$ is in the complexity class*

$$\mathbf{IOP}\left[\begin{array}{lrcl} \text{rounds} & \mathsf{k}(n) & = & 3 \\ \text{answer alphabet} & \mathsf{a}(n) & = & \mathbb{F}_{O(1)} \\ \text{proof length} & \mathsf{l}(n) & = & O(n) \\ \text{randomness} & \mathsf{r}(n) & = & \mathrm{polylog}(n) \\ \text{query complexity} & \mathsf{q}(n) & = & O(1) \\ \text{soundness error} & \varepsilon(n) & = & 1/2 \\ \text{prover time} & \mathsf{tp}(n) & = & O(n^3 \log n) \\ \text{verifier time} & \mathsf{tv}(n) & = & O(n^3 \log n) \end{array}\right].$$

By now we have proved all the key ingredients for the above theorem. We now explain how to combine them. First, we introduce the notion of an *efficient* IOPP, that will simplify dealing with the various parameters.

Efficient IOPPs: The idea of the definition an efficient IOPP is that only the number of rounds and the proof length need to be stated explicitly, while all other parameters are assumed to attain the "desirable value". Specifically, in an efficient IOPP we assume (i) the soundness error, and proximity error can be any constant of our choosing; (ii) the query complexity and alphabet size are constant (depending on our choice of the later); (iii) the verifier runtime and randomness complexity are polylogarithmic in the instance size and proof length (iv) the prover runtime is quasilinear in the proof length. A technicality is that to enable *any* constant soundness error and proximity parameter, we allow the proof length to differ by a multiplicative constant from the explicitly stated value.

We proceed with the definition. Fix a relation $\mathscr{R}$, integer k, and function $\mathsf{l} : \mathbb{N} \to \mathbb{N}$. We say $\mathscr{R}$ has a k*-round* l*-efficient* IOPP if for every $\epsilon \in (0,1)$ and $\delta \in (0, \delta_{\mathscr{R}})$ there exists $c > 0$ such that $\mathscr{R}$ is in the complexity class :

$$\mathbf{IOPP}\left[\begin{array}{llcl} \text{rounds} & \mathsf{k} & & \\ \text{answer alphabet} & \mathsf{a}(n) & = & \mathbb{F}_{O(1)} \\ \text{proof length} & c \cdot \mathsf{l}(n) & & \\ \text{randomness} & \mathsf{r}(n) & = & \mathrm{polylog}(n + \mathsf{l}(n)) \\ \text{query complexity} & \Theta(1) & & \\ \text{soundness error} & \epsilon & & \\ \text{proximity parameter} & \delta & & \\ \text{prover time} & \tilde{O}(\mathsf{l}(n)) & & \\ \text{verifier time} & c \cdot \mathrm{polylog}(n + \mathsf{l}(n) & & \end{array}\right].$$

One convenient use of the efficient IOPP notion that will be used in this section is the following: Theorem 2.9 implies that any relation in $\mathscr{R} \in \mathbf{NTIME}(T)$ has a 1-round $\tilde{O}(T)$-efficient IOPP .

We proceed by stating a corollary of the main theorem of Section 6, using the terminology of efficient IOPPs. Informally, the corollary states that if we have an efficient IOPP for a systematic code and for the sumcheck relation of its closure, we can obtain an IOP system for $\mathscr{R}_{\mathrm{CSAT}}$ where most parameters will essentially equal the sum of corresponding parameters in the IOPPs. However, we will have the following notable exceptions:

1. The proof length will contain a factor corresponding to the encoding length of the code - for which reason, we will want codes of constant rate and constant alphabet size.

2. The verifier and prover runtime will contain a factor corresponding to the encoding time of the code.

For simplicity, rather than taking the evading set as parameter as done in Theorem 6.2, the corollary uses the evading set of Lemma 2.5 with parameter $\gamma = 2/5$. We state the corollary.

Corollary 7.2. *Let $\mathbb{F}$ be a finite field with $|\mathbb{F}| \geq 3$.*

1. Let $\mathscr{C}$ be a systematic ensemble of codes over $\mathbb{F}$ with $\tau(\mathscr{C}) > 0$.

2. Let $\mathscr{D}$ be a degree 3-closure of $\mathscr{C}$ with $\tau(\mathscr{D}) > 0$.

3. Assume we have a k_1-round l_1-efficient IOPP *for $\mathscr{R}_{\mathscr{C}}$; and a k_2-round l_2-efficient* IOPP *for* $\mathrm{Rel}(\mathrm{SC}, \mathscr{D})$.

Then we have an IOP *system putting $\mathscr{R}_{\mathrm{CSAT}}$ in the complexity class*

$$\textbf{IOP}\left[\begin{array}{ll} \text{rounds} & \mathsf{k} = \max\{\mathsf{k}_1, \mathsf{k}_2\} + 1 \\ \text{answer alphabet} & \mathsf{a} = \mathbb{F}_{O(1)} \\ \text{proof length} & \mathsf{l}(n) = O(\mathsf{l}_1(n) + \mathsf{l}_2(n) + \ell(C_n)) \\ \text{randomness} & \mathsf{r}(n) = \mathrm{polylog}(n + \mathsf{l}_1(n) + \mathsf{l}_2(n)) \\ \text{query complexity} & \mathsf{q} = O(1) \\ \text{soundness error} & \varepsilon = 1/2 \\ \text{prover time} & \mathsf{tp}(n) = \tilde{O}(\mathsf{l}_1(n) + \mathsf{l}_2(n) + n \cdot |\mathbb{F}|) + O(T_{\mathscr{C}}(n) + T_{\mathscr{D}}(n)) \\ \text{verifier time} & \mathsf{tv}(n) = \mathrm{polylog}(n + \mathsf{l}_1(n) + \mathsf{l}_2(n)) + O(T_{\mathscr{C}}(n)) \end{array}\right]$$

Let us focus on the proof length parameter in Corollary 7.2. It depends linearly on 1. the proof length for the code relation $\mathscr{R}_{\mathscr{C}}$; 2. the proof length for the sumcheck relation of the closure $\mathrm{Rel}(\mathrm{SC}, \mathscr{D})$; and 3. the block length $\ell(C_n)$. It follows that if we had an efficient IOPP for both relations with linear proof length, where the code family $\mathscr{C}$ has constant rate over a constant alphabet size, we would attain our goal: An IOP for $\mathscr{R}_{\mathrm{CSAT}}$ with constant query complexity and linear proof length. The algebraic-geometry codes of Theorem 2.3 indeed have constant rate and constant alphabet size. However, we do not have an efficient IOPP for the corresponding relations of these codes. Thus, we will use high enough (constant) tensors of these codes instead. We can do this, as a constant degree tensor of a constant rate code, still has constant rate.

We first need a statement about efficient sumchecks for tensor codes in the terminology of efficient IOPPs. This is done in Corollary 7.3 and Theorem 7.4. Corollary 7.3 simply restates the result of Theorem 4.1, for the special case of a degree two tensor, with the simplified parameter list of an efficient IOPP. Theorem 7.4 will then tell us roughly the following. Starting with any code family $\mathscr{C}$ with polynomial time encoding, and taking an appropriate (constant) tensor $\mathscr{C}^{\otimes m}$, we get a code whose sumcheck relation has efficient IOPPs with sublinear proof length. We first note a techincality. In the relations used in Theorem 4.1 the instance lengths corresponded to blocklengths of codes; while in relations of the form $\mathrm{Rel}(\mathrm{SC}, \mathscr{C})$ and $\mathscr{R}_{\mathscr{C}}$, as defined in Section 6 for systematic code families, we take the index n, measuring "how systematic" C_n is, as the instance. We note that for a family $\mathscr{C}$ with $\rho(\mathscr{C}) > 0$ (see Section 2.4 for a definition) the blocklength $\ell(C_n) = O(n)$ for all $n \in \mathbb{N}$. Thus, in the context of efficient IOPPs, it does not matter if we use n as instance length rather than the block length.

Corollary 7.3. *Let $m \in \mathbb{N}$ and $\mathscr{C}$ be a family of systematic linear codes of constant rate $\rho(\mathscr{C}) > 0$. Suppose that*
1. there is a k_1-round l_1-efficient IOPP *for* $\mathrm{Rel}(\mathrm{SC}, \mathscr{C})$*;*
2. there is a k_2-round l_2-efficient IOPP *for* $\mathrm{Rel}(\mathscr{C}^{\otimes 2})$.
Then $\mathrm{Rel}(\mathrm{SC}, \mathscr{C}^{\otimes 2})$ *has a* $\max\{2 \cdot \mathsf{k}_1, \mathsf{k}_2\}$*-round l-efficient* IOPP *where*
1. $\mathsf{l}(s^2) = \mathsf{l}_1(s^2) + 2 \cdot \mathsf{l}_2(s) + 2 \cdot s$
2. $\mathsf{tp}(s^2) = \tilde{O}(\mathsf{l}_1(s^2) + \mathsf{l}_2(s)) + 2 \cdot s^2$

The corollary above now allows us to derive the following theorem:

Theorem 7.4. *Let $\mathscr{C}$ be a family of systematic linear codes with $\rho(\mathscr{C}) > 0$ and $T_{\mathscr{C}}(s) = O(s^c)$ for constant c. Let $m > c$ be a power of two such that $\tau(\mathscr{C}) \geq (1 - \frac{1}{8m})$. Then,* $\mathrm{Rel}(\mathrm{SC}, \mathscr{C}^{\otimes 2m})$ *has a 2-round $o(n)$-efficient* IOPP *(where $n = s^{2m}$ denotes the instance size of* $\mathrm{Rel}(\mathrm{SC}, \mathscr{C}^{\otimes 2m})$*).*

Proof. Let $\mathscr{D} \triangleq \mathscr{C}^{\otimes m}$. So $\mathscr{C}^{\otimes 2m} = \mathscr{D}^{\otimes 2}$. Let us denote by $n = s^{2m}$ the instance size of $\mathrm{Rel}(\mathscr{C}^{\otimes 2m})$ and thus by $\sqrt{n}$ the instance size of $\mathrm{Rel}(\mathrm{SC}, \mathscr{D})$. We know
1. from Theorem 2.9 that $\mathrm{Rel}(\mathrm{SC}, \mathscr{D})$ has a 1-round $\tilde{O}(\sqrt{n})$-efficient IOPP ,
2. from Theorem 5.3 that $\mathrm{Rel}(\mathscr{C}^{\otimes 2m})$ has a 1-round $o(n)$-efficient IOPP .
The theorem follows now from Corollary 7.3, using $\tilde{O}(\sqrt{n}) = o(n)$. □

We now "plug in" AG codes (see Section 2.4.3) to the above theorem to get the following final corollary:

Corollary 7.5. *There exists a prime power q, and families $\mathscr{C}$ and $\mathscr{D}$ of systematic linear codes with alphabet $\mathbb{F}_q$ and constant rate such that*
1. $\mathscr{D}$ is a degree 3-closure of $\mathscr{C}$;
2. there is a 1-round $o(n)$-efficient IOPP *o(n) for* $\mathrm{Rel}(\mathscr{C})$*;*
3. there is a 2-round $o(n)$-efficient IOPP *o(n) for* $\mathrm{Rel}(\mathrm{SC}, \mathscr{D})$.
4. $T_{\mathscr{C}}(n), T_{\mathscr{D}}(n) = O(n^3 \cdot \log n)$.

Proof. Let $m = 8$. Take $\mathscr{C} = \mathscr{A}^{\otimes m}$, and $\mathscr{D} = \mathscr{B}^{\otimes m}$ where $\mathscr{A}$ and $\mathscr{B}$ are the ensembles from Theorem 2.3 such that $\tau(\mathscr{C}), \tau(\mathscr{D}) \geq \eta \triangleq (1 - \frac{1}{8m})$. The second item follows now from Theorem 5.3; the third item from Theorem 7.4, and the fourth from Theorem 2.3. ☐

To conclude, we note that by plugging the code family from Corollary 7.5 into Theorem 7.2 we obtain Theorem 7.1.

References

[AGHP92] Noga Alon, Oded Goldreich, Johan Håstad, and René Peralta. Simple construction of almost k-wise independent random variables. *Random Structures and Algorithms*, 3(3):289–304, 1992.

[ALM$^+$98] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *Journal of the ACM*, 45(3):501–555, 1998. Preliminary version in FOCS ’92.

[AS98] Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: a new characterization of NP. *Journal of the ACM*, 45(1):70–122, 1998. Preliminary version in FOCS ’92.

[AS03] Sanjeev Arora and Madhu Sudan. Improved low-degree testing and its applications. *Combinatorica*, 23(3):365–426, 2003. Preliminary version appeared in STOC ’97.

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

[BCGT13] Eli Ben-Sasson, Alessandro Chiesa, Daniel Genkin, and Eran Tromer. On the concrete efficiency of probabilistically-checkable proofs. In *Proceedings of the 45th ACM Symposium on the Theory of Computing*, STOC ’13, pages 585–594, 2013.

[BCGV16] Eli Ben-Sasson, Alessandro Chiesa, Ariel Gabizon, and Madars Virza. Quasilinear-size zero knowledge from linear-algebraic PCPs. In *Proceedings of the 13th Theory of Cryptography Conference*, TCC ’16, pages 33–64, 2016.

[BCS16] Eli Ben-Sasson, Alessandro Chiesa, and Nicholas Spooner. Interactive oracle proofs, 2016. Crypto ePrint 2016/116.

[BFL90] László Babai, Lance Fortnow, and Carsten Lund. Nondeterministic exponential time has two-prover interactive protocols. In *Proceedings of the 31st Annual Symposium on Foundations of Computer Science*, SFCS ’90, pages 16–25, 1990.

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

[BGH$^+$05] Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil Vadhan. Short PCPs verifiable in polylogarithmic time. In *Proceedings of the 20th Annual IEEE Conference on Computational Complexity*, CCC ’05, pages 120–134, 2005.

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

[BKK$^+$13] Eli Ben-Sasson, Yohay Kaplan, Swastik Kopparty, Or Meir, and Henning Stichtenoth. Constant rate PCPs for Circuit-SAT with sublinear query complexity. In *Proceedings of the 54th Annual IEEE Symposium on Foundations of Computer Science*, FOCS ’13, pages 320–329, 2013.

[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. Preliminary version appeared in STOC ’05.

[BSVW03] Eli Ben-Sasson, Madhu Sudan, Salil Vadhan, and Avi Wigderson. Randomness-efficient low degree tests and short PCPs via epsilon-biased sets. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing*, STOC ’03, pages 612–621, 2003.

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

[DR04] Irit Dinur and Omer Reingold. Assignment testers: Towards a combinatorial proof of the PCP theorem. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*, FOCS ’04, pages 155–164, 2004.

[For65] David G. Forney. Concatenated codes. Technical report, MIT, Cambridge, MA, USA, 1965.

[FRS88] Lance Fortnow, John Rompel, and Michael Sipser. On the power of multi-prover interactive protocols. In *Theoretical Computer Science*, pages 156–161, 1988.

[FS86] Amos Fiat and Adi Shamir. How to prove yourself: practical solutions to identification and signature problems. In *Proceedings of the 6th Annual International Cryptology Conference*, CRYPTO ’86, pages 186–194, 1986.

[GIMS10] Vipul Goyal, Yuval Ishai, Mohammad Mahmoody, and Amit Sahai. Interactive locking, zero-knowledge PCPs, and unconditional cryptography. In *Proceedings of the 30th Annual Conference on Advances in Cryptology*, CRYPTO’10, pages 173–190, 2010.

[GKR08] Shafi Goldwasser, Yael Tauman Kalai, and Guy N. Rothblum. Delegating computation: Interactive proofs for Muggles. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, STOC '08, pages 113–122, 2008.

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1):186–208, 1989. Preliminary version appeared in STOC '85.

[Gop81] Valery Denisovich Goppa. Codes on algebraic curves. *Soviet Mathematics — Doklady*, 1(24):170–172, 1981.

[GS96] Arnaldo Garcia and Henning Stichtenoth. On the asymptotic behaviour of some towers of function fields over finite fields. *Journal of Number Theory*, 61(2):248–273, 1996.

[GS06] Oded Goldreich and Madhu Sudan. Locally testable codes and PCPs of almost-linear length. *Journal of the ACM*, 53:558–655, July 2006. Preliminary version in STOC '02.

[HS00] Prahladh Harsha and Madhu Sudan. Small PCPs with low query complexity. *Computational Complexity*, 9(3–4):157–201, Dec 2000. Preliminary version in STACS '91.

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

[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.

[Mei12] Or Meir. Combinatorial PCPs with short proofs. In *Proceedings of the 26th Annual IEEE Conference on Computational Complexity*, CCC '12, 2012.

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

[Mic00] Silvio Micali. Computationally sound proofs. *SIAM Journal on Computing*, 30(4):1253–1298, 2000. Preliminary version appeared in FOCS '94.

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

[NN90] Joseph Naor and Moni Naor. Small-bias probability spaces: efficient constructions and applications. In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing*, STOC '90, pages 213–223, 1990.

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

[PS94] Alexander Polishchuk and Daniel A. Spielman. Nearly-linear size holographic proofs. In *Proceedings of the 26th Annual ACM Symposium on Theory of Computing*, STOC '94, pages 194–203, 1994.

[PS96] David Pointcheval and Jacques Stern. Security proofs for signature schemes. In *Proceedings of the 14th Annual International Conference on Theory and Application of Cryptographic Techniques*, EUROCRYPT '96, pages 387–398, 1996.

[RS97] Ran Raz and Shmuel Safra. A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing*, STOC '97, pages 475–484, 1997.

[SAK$^+$01] Kenneth W. Shum, Ilia Aleshnikov, P. Vijay Kumar, Henning Stichtenoth, and Vinay Deolalikar. A low-complexity algorithm for the construction of algebraic-geometric codes better than the Gilbert–Varshamov bound. *IEEE Transactions on Information Theory*, 47(6):2225–2241, 2001.

[Sha92] Adi Shamir. IP = PSPACE. *Journal of the ACM*, 39(4):869–877, 1992.

[Sti08] Henning Stichtenoth. *Algebraic function fields and codes*. Springer Publishing Company, 2nd edition, 2008.

[Tan81] Robert Michael Tanner. A recursive approach to low complexity codes. *IEEE Transactions on Information Theory*, 27(5):533–547, 1981.

[Val08] Paul Valiant. Incrementally verifiable computation or proofs of knowledge imply time/space efficiency. In *Proceedings of the 5th Theory of Cryptography Conference*, TCC '08, pages 1–18, 2008.

[WE63] Jack Keil Wolf and Bernard Elspas. Error-locating codes - a new concept in error control. *IEEE Transactions on Information Theory*, 9(2):113–117, 1963.

[Wol65] Jack Keil Wolf. On codes derivable from the tensor product of check matrices. *IEEE Transactions on Information Theory*, 11(2):281–284, 1965.

ECCC ISSN 1433-8092

http://eccc.hpi-web.de