Electronic Colloquium on Computational Complexity, Revision 1 of Report No. 157 (2020)

Batch Verification and Proofs of Proximity with Polylog Overhead*

Guy N. Rothblum† Weizmann Institute

Ron D. Rothblum‡ Technion

June 18, 2026

In Memoriam: Uriel G. Rothblum

Abstract

Suppose Alice wants to convince Bob of the correctness of k NP statements. Alice could send k witnesses to Bob, but as k grows the communication becomes prohibitive. Is it possible to convince Bob using smaller communication (without making cryptographic assumptions or bounding the computational power of a malicious Alice)? This is the question of *batch verification* for NP statements. Our main result is a new interactive proof protocol for verifying the correctness of k UP statements (NP statements with a unique witness) using communication that is *poly-logarithmic* in k (and a fixed polynomial in the length of a single witness).

This result is obtained by making progress on a different question in the study of interactive proofs. Suppose Alice wants to convince Bob that a huge dataset has some property. Can this be done if Bob can't even read the entire input? In other words, what properties can be verified in *sublinear* time? An Interactive Proof of Proximity guarantees that Bob accepts if the input has the property, and rejects if the input is far (say in Hamming distance) from having the property. Two central complexity measures of such a protocol are the query and communication complexities (which should both be sublinear). For every query parameter q, and for every language in logspace uniform NC, we construct an interactive proof of proximity with query complexity q and communication complexity $(n/q) \cdot \mathsf{polylog}(n)$.

Both results are optimal up to poly-logarithmic factors, under reasonable complexity-theoretic or cryptographic assumptions. The second result, which is our main technical contribution, builds on a distance amplification technique due to Ben-Sasson, Kopparty and Saraf [CCC 2018].

*The original version of this paper [RR20] contained an error. This was discovered and communicated to us by Goyal, Hong and Kalai [GHK25]. The current version corrects this error, which required an adjustment to our main protocol.

†Email: rothblum@alum.mit.edu. This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No. 819702). Research also supported by the Israel Science Foundation (grant number 5219/17) and an Amazon Research Award.

‡Email: rothblum@cs.technion.ac.il. Supported in part by a Milgrom family grant, by the Israeli Science Foundation (Grants No. 1262/18 and 2137/19), and the Technion Hiroshi Fujiwara cyber security research center and Israel cyber directorate.

ISSN 1433-8092

Contents

1 Introduction

The power of efficiently verifiable proof-systems is a central question in the study of computation. It has been the focus of a rich literature spanning cryptography and complexity theory. This literature has put forth and studied different notions of proof systems and different notions of efficient verification. Interactive proofs, introduced in the seminal work of Goldwasser, Micali and Rackoff [GMR89], are one of the most fundamental notions in this field. An interactive proof is an interactive protocol between a randomized verifier and an untrusted prover. The prover convinces the verifier of the validity of a computational statement, usually framed as membership of an input x in a language $\mathcal{L}$. Soundness is unconditional. Namely, if the input is not in the language, then no matter what (unbounded and adaptive) strategy a cheating prover might employ, the verifier should reject with high probability over its own coin tosses. Interactive proofs have had a dramatic impact on complexity theory and on cryptography. Opening the door to randomized and interactive verification led to revolutionary notions of proof verification, such as zero knowledge interactive proofs [GMR89, GMW91] and probabilistically checkable proofs (PCPs) [BGKW88, FRS94, BFL91, BFLS91, FGL+96, AS92, ALM+98]. Interactive proof-systems also allow for more efficient verification of larger classes of computations (compared with NP proof systems), as demonstrated in the celebrated $\mathsf{IP} = \mathsf{PSPACE}$ Theorem [LFKN92, Sha92].

Still, foundational questions about the power of interactive proof systems have remained open. Our work studies two such questions:

1.1 Batch Verification

Can interactive proofs allow for more efficient *batch verification* of a collection of NP statements?

Question 1:

How efficiently can an untrusted prover convince a verifier of the correctness of k NP statements?

A naive solution is sending the k witnesses in their entirety. An honest prover, who knows the witnesses, runs in polynomial time, but the communication grows linearly with k. For the case of UP statements — NP statements with a unique witness — we show a protocol where the communication complexity grows poly-logarithmically with k (and the honest prover remains efficient):

Theorem 1 (Informally Stated, see Theorem 4 and Corollary 5)**.** *Let $\mathcal{L} \in \mathsf{UP}$ with witnesses of length $m = m(n)$. There exists an interactive proof for verifying that k instances $x_1, \ldots, x_k$, each of length n, all belong to $\mathcal{L}$. The communication complexity is $\mathsf{poly}(\log(k), m)$, where poly refers to a fixed polynomial that depends only on the language $\mathcal{L}$. The number of rounds is $\mathsf{polylog}(k, m)$. The verifier runs in time $\tilde{O}(k \cdot n) + \mathsf{polylog}(k) \cdot \mathsf{poly}(m)$, where n is the length of each of the instances. The honest prover runs in time $\mathsf{poly}(k, n, m)$ given the k unique witnesses.*

This resolves the communication complexity of batch verification for UP up to $\mathsf{poly}(\log(k), m)$ factors: under complexity-theoretic assumptions, even for $k = 1$ there are UP languages (e.g. unique SAT) for which every interactive proof system requires communication complexity $\Omega(m)$ [GH98, GVW02]. When the number of instances k is large, this is a significant improvement over the naive solution in which the prover sends over all k witnesses.

We note that for UP relations that are checkable in log-space uniform NC, we can reduce the communication complexity to $m \cdot \mathsf{polylog}(k, m)$. As discussed above, this is tight up to $\mathsf{polylog}(k, m)$ factors (under complexity assumptions). We also note that, assuming the existence of one-way functions, our batch verification protocol (which is public coin) can be made (computational) zero-knowledge using standard techniques [BGG+88] .

Comparison to prior work. A different solution can be obtained via the $\mathsf{IP} = \mathsf{PSPACE}$ theorem, by observing that the membership of k inputs in an NP language can be decided in space $O(\log k + m \cdot \mathsf{poly}(n))$, where n is the length of a single input and m is the length of a single NP witness. Thus, by the $\mathsf{IP} = \mathsf{PSPACE}$ Theorem, there is an interactive proof for batch verification with communication complexity $\mathsf{poly}(\log k, n, m)$. A major caveat, however, is that the complexity of proving correctness (the running time of the *honest* prover) is *exponential in* $\mathsf{poly}(n, m)$. We, on the other hand, focus on batch verification where the honest prover runs in *polynomial time* given the k NP witnesses. We refer to such an interactive proof as having an *efficient prover*.[1] Another significant drawback of this solution is that the number of rounds becomes $\mathsf{poly}(m, \log k)$.

Two recent works have constructed protocols for efficient batch verification of UP statements. Reingold, Rothblum and Rothblum [RRR16] gave a protocol with communication complexity $\mathsf{polylog}(k) \cdot \mathsf{poly}(m) + k \cdot \mathsf{polylog}(m)$. In a subsequent work [RRR18] they eliminated the additive k factor but increased the multiplicative factor, by showing a (constant-round) protocol with communication complexity $k^{\varepsilon} \cdot \mathsf{poly}(m)$, for any $\varepsilon > 0$. Our main result achieves the best of both worlds: eliminating the additive linear factor while preserving the poly-logarithmic multiplicative factor (although our protocol has a larger number of rounds than that of [RRR18]).

1.2 Interactive Proofs of Proximity

A different question (which turns out to be related) asks which statements can be verified in *sublinear* time, i.e. without even reading the entire input. This immediately raises the question of what computational model is used to capture "sublinear time". Drawing inspiration from the literature on sublinear *algorithms*, a natural choice is to adopt the perspective of property testing, a study initiated by Rubinfeld and Sudan [RS96] and Goldreich, Goldwasser and Ron [GGR98], which considers highly-efficient randomized algorithms that solve approximate decision problems, while only inspecting a small fraction of the input. Such algorithms, commonly referred to as property testers for a set S (say the set of objects with some property), are given query access to an input, and are required to determine whether the input is in S (has the property), or is far (say, in Hamming distance) from every string in S (far from having the property). A rich literature has put forward property testers for many natural properties.

Analogously, in the proof verification setting, Interactive Proofs of Proximity (IPPs) aim to verify that a given input is close to a set (or a property). Given a desired proximity parameter $\delta \in (0, 1]$, the soundness condition of standard interactive proofs is relaxed: it should be impossible to convince the verifier to accept statements that are δ-far (in fractional Hamming distance) from true statements (except with small probability). Such proof-systems were first introduced by Ergün, Kumar and Rubinfeld [EKR04] and were more recently further studied by Rothblum, Vadhan and Wigderson [RVW13] and by Gur and Rothblum [GR13]. The verifier's query complexity and running time, as well as the communication, should all be sublinear in the input length. Other parameters of interest include the (honest) prover's running time and the number of rounds.

[1] Efficiency of the honest prover (given an NP witness) has been central in the study of zero-knowledge interactive proofs [GMR89, GMW91]. It has also been central to the study of efficient batch verification in recent works [RRR16, RRR18].

The hope is that IPPs can overcome inherent limitations of property testing: for example, demonstrating specific properties where verifying proximity can be significantly faster than the time needed to test (without a prover). Another goal is showing that sublinear-time verification is possible for much richer families of properties than those for which property testers exist. In particular, research on property testing has focused on constructing testers for languages based on their combinatorial or algebraic structure. This limitation seems inherent, because there exist simple and natural languages for which (provably) no sublinear time property testers exist. In contrast, it is known that highly non-trivial IPPs exist for every language that can be decided in bounded-polynomial depth or space [RVW13, RRR16]. However, the optimal tradeoffs between the query and communication complexities needed for proof verification were not known, and this is the second foundational question we study:

Question 2:

What are the possible tradeoffs between the query and communication complexities in interactive proofs of proximity, and for which statements?

For the case of languages in (uniform) NC—languages that can be decided by polynomial-sized circuits of polylogarithmic depth—we show that the product of the query and communication complexities can be quasi-linear.

Theorem 2 (Informally Stated, see Theorem 3)**.** *Let $\mathcal{L}$ in log-space uniform* NC*. For every proximity parameter $\delta \in (0,1)$ and parameter $t = t(n) \in [n]$, there exists a δ-*IPP *for $\mathcal{L}$, with communication complexity $t \cdot \mathsf{polylog}(n)$ and query complexity $q(n, \delta) = O\big(\frac{1}{\delta} + \frac{n \cdot \mathsf{polylog}(n)}{t}\big)$. The verifier runs in time $\tilde{O}\big(\frac{1}{\delta} + t + n/t\big)$ and the prover runs in time $\mathsf{poly}(n)$.*

For example, by setting $t(n) = \sqrt{n}$ we obtain an IPP for NC with query, communication and verification complexity all $\tilde{O}(\sqrt{n})$. This result resolves the question for such languages, up to poly-logarithmic factors, as Kalai and Rothblum [KR15] showed that (under a reasonable cryptographic assumption) there exists a language in NC^1 for which the product of the query and communication complexities cannot be sublinear.

Setting $t(n) = \mathsf{polylog}(n)$ we obtain an IPP with poly-logarithmic communication and sublinear query complexity. This setting of parameters will be useful towards constructing UP batch verification protocols and establishing Theorem 1.

Comparison and relationship to [RVW13]. Theorem 2 shows that the product of the query complexity and the communication can be $\tilde{O}(n)$. Rothblum, Vadhan and Wigderson [RVW13] showed a similar statement, but the product of the query and communication complexities was $n^{1+o(1)}$ (see also the recent digest [Gol26]).

Our protocol builds on the framework developed in their work, introducing several new ideas and using a distance amplification technique from the work of Ben-Sasson, Kopparty and Saraf [BKS18]. We find the improvement from $n^{1+o(1)}$ to $\tilde{O}(n)$ to be significant: beyond the fact that it provides a nearly-optimal (up to $\mathsf{polylog}(n)$ factors) trade-off for a foundational problem, it allows for IPPs with $\mathsf{polylog}(n)$ communication and sublinear query complexity. In prior work, achieving sublinear query complexity (for NC) required $n^{o(1)}$ communication. The importance of this distinction is exemplified in the application of IPPs towards batch verification for UP [RRR18]. That construction repeatedly uses IPPs with slightly-sublinear query complexity. The communication of the resulting batch verification protocol is dominated by the communication complexity of the IPPs. Indeed, the improved IPP of Theorem 2 is the key component behind the improved UP batch verification protocol of Theorem 1.

1.3 Related Works

Batch Verification. Batch verification has been used extensively in the context of cryptography, especially for fast verification of digital signatures (see, e.g., [NMVR94, BGR98, CHP12]). Ishai, Kushilevitz and Ostrovsky [IKO07] raised the question of using efficient interactive proof-systems to perform *general-purpose* batch verification. Meir [Mei16] uses batch verification of PCPs in order to construct short PCPs for NP, whereas [RRR16] use batch verification of doubly-efficient interactive proofs in order to construct doubly-efficient interactive proofs for bounded space computations.

Batch Verification with Computational Soundness. If one is willing to settle for *computational* soundness (i.e., soundness holds only against polynomial-time cheating strategies) and to use cryptographic assumptions, then efficient batch verification is possible for all of NP. In particular, Kilian [Kil92] gave an interactive argument-system for all of NP based on collision-resistant hash functions with only poly-logarithmic communication complexity. Since verifying the membership of k instances in an NP language is itself an NP problem, we immediately obtain a batch verification protocol with communication complexity $\mathsf{poly}(\log(n), \log(k), \kappa)$, where κ is a cryptographic security parameter.

More recently, Brakerski, Holmgren and Kalai [BHK17] obtained an efficient *non-interactive* batch-verification protocol assuming the existence of a computational private information retrieval scheme. Non-interactive batch verification protocols also follow from the existence of *succinct non-interactive zero-knowledge arguments (zkSNARGs)*, which are known to exist under certain strong, and non-falsifiable, assumptions (see, e.g. [Ish], for a recent survey).

We emphasize that the batch verification protocols of both [Kil92] and [BHK17] only provide computational soundness and are based on unproven cryptographic assumptions. In contrast, the result of Theorem 1 offers statistical soundness and is unconditional.

Interactive Proofs of Proximity. Beyond the works [EKR04, RVW13, GR13] that were mentioned above, IPPs have drawn considerable attention recently [FGL14, GGR15, KR15, RRR16, GR17, BRV18, RRR18, CG18, GLR18, RR19, GRSY20]. See also Goldreich's [Gol26] recent digest of [RVW13].

In particular, we mention that a recent work of Ron-Zewi and Rothblum [RR19, Theorem 3, see also Remark 1.3] shows that for every constant ϵ, every language computable in polynomial-time and bounded polynomial space has an IPP with communication complexity $\epsilon \cdot n$ and constant query complexity. Note that the product between the query and communication complexity in their result is $O(n)$, rather than $n \cdot \mathsf{polylog}(n)$ as in Theorem 2. However, in constrast to Theorem 2, their result is restricted to the regime of constant query complexity and only yields communication complexity that is smaller by a constant factor than that of the trivial solution (see Proposition 3.8).

1.4 Subsequent Developments

We detail subsequent developments since the original publication of this work. As noted above, Goyal, Hong and Kalai [GHK25] discovered an error in the conference version of this work [RR20].

The current version fixes this error with a small change to the protocol, see Remark 2.1 for a high-level discussion. An alternative fix was proposed by Berger, Goyal, Hong and Kalai [BGHK25], see Remark 2.1.

Subsequently to our work, Bitansky *et al.* [BKP+24] show that any language that has a proof system for batch verification also has an unconditionally statistically witness indistinguishable proof system. This can be viewed as a barrier to extending the result of Theorem 1 to NP languages beyond UP (where witness indistinguishability is trivial).

IPPs and their connection to batch verification have played a central role in a diverse set of recent protocols. Amit and Rothblum [AR24] build on our work to construct *constant-round* arguments for batch-verification of UP statements, based on the existence of one-way functions. The nearly-optimal IPP of Theorem 2 is a central building block in building sample-based IPPs [GR22], in proof-systems for verifying general distribution properties [HR24], and in the interactive proofs for bounded-space computations of [BGHK25].

1.5 Organization

Section 2 contains a technical overview of our techniques. In Section 3 we provide preliminaries and our main results are stated in Section 4. In Section 5 we introduce the PVAL problem and show how to amplify its distance. Our efficient PVAL IPP is in Section 6. Lastly, in Section 7 we use the results established in the prior sections to prove Theorem 1 and Theorem 2.

2 Technical Overview

To prove Theorem 1 we rely on a recent result of Reingold *et al.* [RRR18] who showed how to reduce the construction of UP batch verification protocol to that of constructing efficient IPPs. In particular, via the connection established in [RRR18], in order to prove Theorem 1, it suffices to prove Theorem 2 with respect to $cc = \mathsf{polylog}(n)$.

Thus, in this overview we focus on proving Theorem 2. Our starting point for the proof of Theorem 2 is the IPP construction for NC from [RVW13] (which achieves weaker parameters than those of Theorem 2).

The [RVW13] protocol is centered around a parameterized problem called PVAL, which stands for "Polynomial eVALuation" and is defined next. A key step in the [RVW13] proof is showing that PVAL is "complete" for constructing IPPs for NC. In more detail, for every language $\mathcal{L} \in \mathsf{NC}$, [RVW13] show an *interactive* reduction, in which the verifier *makes no queries to its input.* At the end of the reduction, the verifier generates a "parameterization" of the PVAL problem so that if the original input x belonged to $\mathcal{L}$ then x belongs to PVAL, whereas if x was *far* from $\mathcal{L}$ then, with high probability, x is also far from PVAL.

In this work we follow the same strategy. We do not modify the interactive reduction step from [RVW13] (which builds on the protocol of [GKR15]). Our improved efficiency stems from a more efficient IPP for PVAL (than that of [RVW13]), which suffices to obtain our main results.

We start by defining a specific variant[2] of the PVAL problem that suffices for our purposes.

[2]In particular, for simplicity and since it is sufficient for our results we consider a variant of PVAL with respect to the *multi-linear* extension rather than a more general low degree extension considered in [RVW13].

The PVAL Problem. Let $\mathbb{F}$ be a (sufficiently large) finite field. The PVAL problem is parameterized by an integer $t \in \mathbb{N}$, which we refer to as the *arity*, and a *dimension* $m \in \mathbb{N}$. In addition the problem is parameterized by t vectors $\mathbf{j} = (\mathbf{j}_1, \ldots, \mathbf{j}_t) \in (\mathbb{F}^m)^t$ and t scalars $\mathbf{v} = (v_1, \ldots, v_t) \in \mathbb{F}^t$. The main input to $\mathsf{PVAL}(t, \mathbf{j}, \mathbf{v})$ is the truth table of a function $f : \{0,1\}^m \to \mathbb{F}$. We say that $f \in \mathsf{PVAL}(t, \mathbf{j}, \mathbf{v})$ if it holds that $\hat{f}(\mathbf{j}_i) = v_i$, for every $i \in [t]$, where $\hat{f} : \mathbb{F}^m \to \mathbb{F}$ is the multi-linear extension of f.[3] Thus, the goal of the PVAL verifier is to distinguish the case that (1) the multilinear extension $\hat{f}$ of the input function f is equal, at t given points, to t corresponding values, or (2) the input function f is far from any such function. Note that the verifier is only allowed to make a sub-linear (i.e., $\ll 2^m$) number of queries to f, but is allowed to communicate with the (untrusted) prover who has full access to f.

Our main technical contribution is an IPP for checking δ-proximity to $\mathsf{PVAL}(t, \mathbf{j}, \mathbf{v})$ with communication complexity roughly $t \cdot \mathsf{poly}(m)$ and query complexity $O(1/\delta)$ (see Theorem 6.1 for the formal statement). (Note that setting $\delta = 2^m \cdot \mathsf{poly}(m)/t$ results in the product of the query and communication complexities being $\tilde{O}(2^m)$, which is quasi-linear in the input length.) We proceed to describe the new IPP for PVAL.

Attempt 1: Divide and Conquer. Fix a parameterization $(t, \mathbf{j}, \mathbf{v})$ for PVAL, where $\mathbf{j} = (\mathbf{j}_1, \ldots, \mathbf{j}_t)$ and $\mathbf{v} = (v_1, \ldots, v_t)$, and consider a given input $f : \{0,1\}^m \to \mathbb{F}$. Following [RVW13], we would like to first decompose the t claims that we are given about f into claims about the underlying functions $f_0, f_1 : \{0,1\}^{m-1} \to \mathbb{F}$, where $f_0(\cdot) \equiv f(0, \cdot)$ and $f_1(\cdot) \equiv f(1, \cdot)$. To do so, the verifier asks the prover to provide the contributions of f_0 and f_1 to the linear claims $\hat{f}(\mathbf{j}_i) = v_i$, for all $i \in [t]$. In more detail, let us view each vector $\mathbf{j}_i$ as $\mathbf{j}_i = (\chi_i, \mathbf{j}'_i)$ where $\chi_i \in \mathbb{F}$ and $\mathbf{j}'_i \in \mathbb{F}^{m-1}$ (i.e., we isolate the first component of $\mathbf{j}_i$ as χ_i and the remaining components as an $(m-1)$-dimensional vector $\mathbf{j}'_i$). The prover sends the vectors $\mathbf{v}_0 = \left(\hat{f}_0(\mathbf{j}'_1), \ldots, \hat{f}_0(\mathbf{j}'_t)\right)$ and $\mathbf{v}_1 = \left(\hat{f}_1(\mathbf{j}'_1), \ldots, \hat{f}_1(\mathbf{j}'_t)\right)$. Note that the prover cannot send arbitrary vectors since the verifier can check (and indeed *does* check) that $\mathbf{v}_0$ and $\mathbf{v}_1$ are consistent with $\mathbf{v}$. (I.e., that $\mathbf{v} = (1 - \bar{\chi}) \cdot \mathbf{v}_0 + \bar{\chi} \cdot \mathbf{v}_1$, where $\bar{\chi} = (\chi_1, \ldots, \chi_t)$ and the multiplication is pointwise.) See Fig. 1 for an illustration.

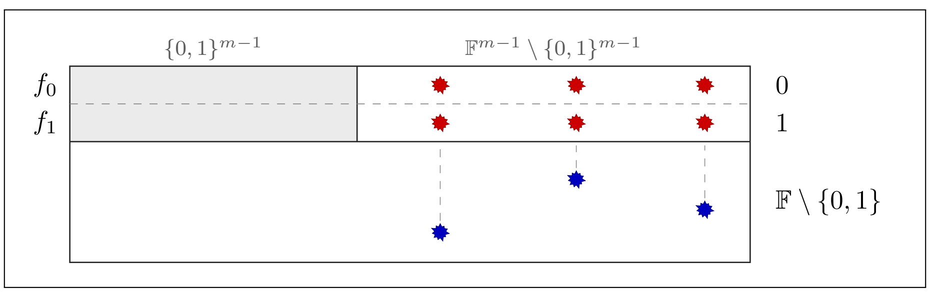

Figure 1: Decomposing the claims $f|_{\mathbf{j}} = \mathbf{v}$. The function f splits into $f_0(\cdot) = f(0, \cdot)$ and $f_1(\cdot) = f(1, \cdot)$; the prover sends the values $\hat{f}_b(\mathbf{j}'_i)$ at the projected query points $\mathbf{j}'_i$ (red starbursts in the $\mathbb{F}^{m-1}$ extension of each f_b). Blue starbursts mark the original query points $\mathbf{j}_i \in \mathbb{F}^m$.

A natural idea at this point, is to try to combine f_0 and f_1 (and the corresponding claims that

[3] Recall that the multilinear extension $\hat{f} : \mathbb{F}^m \to \mathbb{F}$ of $f : \{0,1\}^m \to \mathbb{F}$ is the unique multilinear polynomial that agrees with f on $\{0,1\}^m$. Note that the truth-table of the multilinear extension of a function is super-polynomial in its domain size, but will never be fully materialized or computed in our context (even by the prover). See Section 3.1 for details.

we have about them) into a single $m-1$ variate function on which we can recurse. For example, we can take a random linear combination of the two functions as follows: the verifier chooses a random coefficient $c \in \mathbb{F}$, and sends it to the prover. The two parties then recurse on the input $f' = f_0 + c \cdot f_1$ wrt the claims $\hat{f}'|_{\mathbf{j}'} = \mathbf{v}'$, with $\mathbf{v}' = \mathbf{v}_0 + c \cdot \mathbf{v}_1$ and where $\hat{f}'|_{\mathbf{j}'}$ denotes the restriction of $\hat{f}'$ to the inputs in $\mathbf{j}' = (\mathbf{j}'_1, \ldots, \mathbf{j}'_t)$.[4]

Note that the input has shrunk by a factor of 2 and it is not too difficult to argue that if f was δ-far from $\mathsf{PVAL}(t, \mathbf{j}, \mathbf{v})$ then (whp over c) f' is δ-far from $\mathsf{PVAL}(t, \mathbf{j}', \mathbf{v}')$. Thus, with very little communication (i.e., $O(t \cdot \log(|\mathbb{F}|))$ we have reduced the input size by half and preserved the distance. The issue is that while the input size has shrunk by half, each location in f' depends on two location in f. Thus, while the input length has shrunk by half, this benefit is counteracted by the fact that the effective query complexity has doubled.

Doubling the Distance. In the argument above, we used the fact that the random linear combination f' is (typically) at least δ-far from the relevant PVAL instance. While this is actually tight in the worst-case, we will show that if f_0 and f_1 are "nicely behaved" (see below) then the bound can be improved and f' is roughly 2δ-far from PVAL. We further show that randomly permuting the instances causes the desired nice behavior, resulting in the improved distance bound, details follow.

For every $b \in \{0,1\}$, let δ_b be the distance of f_b from $\mathsf{PVAL}(t, \mathbf{j}', \mathbf{v}_b)$ and let $P_b \in \mathsf{PVAL}(t, \mathbf{j}', \mathbf{v}_b)$ such that $\Delta(f_b, P_b) = \delta_b$ (without getting into the details we remark that P_0 and P_1 will be unique in our regime of parameters). If f is δ-far from PVAL then $\delta_0 + \delta_1 \geq 2\delta$, since otherwise f is δ-close to the function $P \in \mathsf{PVAL}(t, \mathbf{j}, \mathbf{v})$ defined as $P(\sigma, \mathbf{x}) = (1-\sigma) \cdot P_0(\mathbf{x}) + \sigma \cdot P_1(\mathbf{x})$.

For every $b \in \{0,1\}$, let $I_b \subseteq \{0,1\}^m$ be the set of $\delta_b \cdot 2^m$ points on which P_b and f_b disagree (we refer to these as the "error pattern"). Suppose momentarily that I_0 and I_1 have a small intersection, or are even disjoint (this is the "nice behavior" alluded to above). In this case, f' is roughly $\delta_0 + \delta_1 \geq 2\delta$ far from $P_0 + c \cdot P_1$, which is the closest vector to f' in $\mathsf{PVAL}(t, \mathbf{j}', \mathbf{v}')$, and thus 2δ-far from $\mathsf{PVAL}(t, \mathbf{j}', \mathbf{v}')$.

Note that if we can ensure that f' is 2δ-far from the corresponding PVAL instance, then we have improved two parameters: both the distance and the input size, while only paying in the query complexity. This enables an efficient recursion, as described below.

Unfortunately, the above analysis hinged on the assumption that I_0 and I_1 have a small intersection, which we cannot justify. As a matter of fact, for all we know, the two sets could very well be *identical*. In such a case, the distance of f' from PVAL will indeed be (roughly) δ and we are back to square one. See Fig. 2 for an illustration for the possible "error patterns" of f_0 and f_1 and how they affect the "error patterns" of f'.

We pause here for a detour, recalling the approach of [RVW13] (this is not essential for understanding our construction and can be skipped). They observe that if δ_0 and δ_1 are roughly equal, then the verifier can simply recurse on one of them. This roughly maintains the distance, while avoiding doubling the query complexity. On the other hand, if say $\delta_0 \gg \delta_1$, they show that the random linear combination technique described above does increase the distance (intuitively, the row with smaller distance cannot "cancel out" the error pattern of the row with larger distance). They lose a constant multiplicative factor in this argument. Moreover, since the verifier does not

[4] Intuitively, the reason to use a *random* linear combination rather than some fixed combination such as $f_0 + f_1$ is avoiding (w.h.p) the possibility that the differences of f_0 and f_1 from their corresponding PVAL instances (i.e. the 0/1 vectors that can be added to f_0 and f_1 to reach vectors in PVAL) cancel each other out.

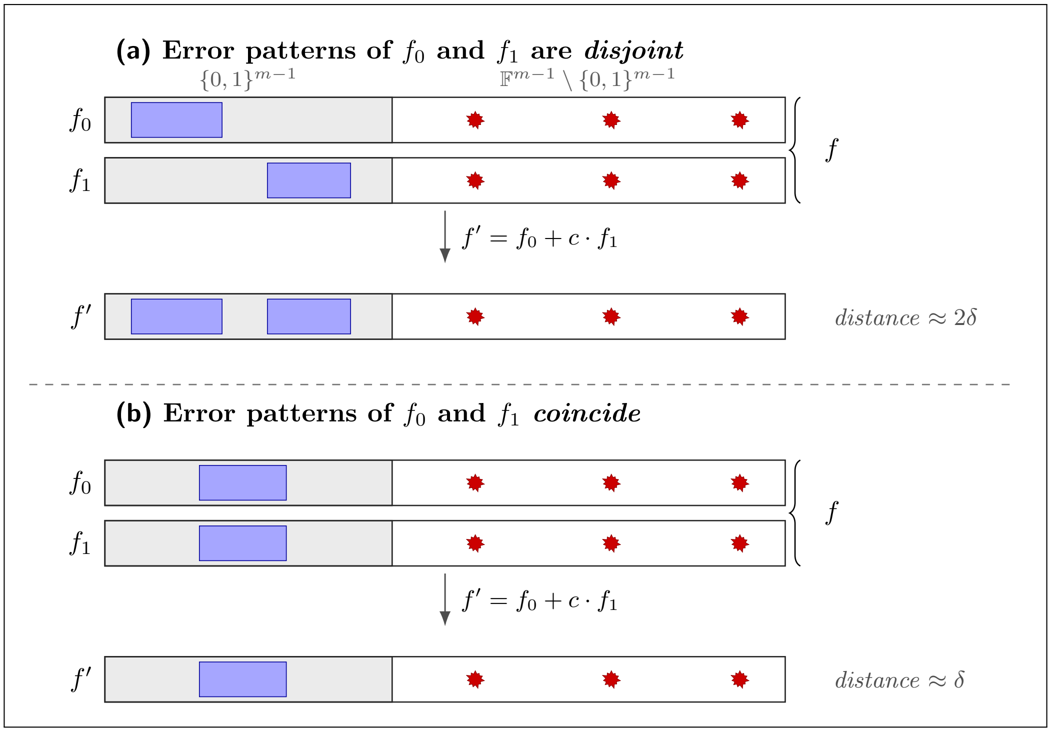

Figure 2: Possible alignments of the "noise". Blue regions mark the error pattern I_b (the positions on which f_b disagrees with the closest function $P_b \in \mathsf{PVAL}(t, \mathbf{j}', \mathbf{v}_b)$); red stars mark the t evaluation positions $\mathbf{j}'$. In (a) the patterns are disjoint, so the combined function $f' = f_0 + c \cdot f_1$ is roughly 2δ-far from PVAL. In (b) the patterns coincide, so f' remains only δ-far.

know which case it is in (i.e., whether $\delta_0 \approx \delta_1$ or $\delta_0 \gg \delta_1$) they employ a more complex argument. Overall this results in a roughly $2^{\log n/\log\log n} = n^{o(1)}$ overhead in the product of the final query and communication complexities.

Reducing the Intersection Size. Roughly speaking, we resolve the above difficulty by randomly permuting the truth table of the function f_1, in order to make the set I_1 (pseudo-)random, and therefore likely to have a small intersection with I_0. This is inspired by a result of Ben Sasson, Kopparty and Saraf [BKS18] on amplifying distances from the Reed-Solomon code. More precisely, the verifier chooses a permutation $\pi : \{0,1\}^m \to \{0,1\}^m$ at random from a suitable family of permutations (to be discussed below). We consider the new function $f_1 \circ \pi$. Intuitively, the entropy induced by the permutation will make the error patterns in f_0 and in $f_1 \circ \pi$ have a small intersection. Then, rather than recursing on $f_0 + c \cdot f_1$, we recurse on $f' = f_0 + c \cdot (f_1 \circ \pi)$.

To make this approach work we have to overcome several difficulties. First, we need to ensure that we can translate the claims that we have about $\hat{f}_1$ into claims about $\widehat{f_1 \circ \pi}$. We do so by choosing π as a random shift, aka a translation map. That is, we choose a random $b \in \{0,1\}^m$ and set $\pi_b(x) = x + b$. Note that π_b is indeed a permutation over $\{0,1\}^m$. Observe that π_b is also well defined over $\mathbb{F}^m$, and since each of the m outputs of π_b is an affine function of the corresponding input, this choice ensures that:

$$\widehat{f_1 \circ \pi} \equiv \hat{f}_1 \circ \pi. \tag{1}$$

To see that Eq. (1) holds, observe both sides of the equation are multilinear polynomials that agree on $\{0,1\}^m$. Therefore they must also agree on $\mathbb{F}^m$.

Eq. (1) implies that the claims that we have about $f_1 \circ \pi$ are simply permutations of the claims about f_1. A second difficulty that arises at this point is that the claims that we have about f_0 and $f_1 \circ \pi$ are not "aligned". The former claims are about the positions $\mathbf{j}'$ and the latter about $\pi^{-1}(\mathbf{j}')$ (in the multi-linear extensions of f_0 and of $f_1 \circ \pi$, respectively). Since the claims are not aligned, it is unclear how to combine them to get t claims about the input f'.

As our first step toward resolving this difficulty, we have the prover "complete the picture" by providing the verifier also with the (alleged) values of $\hat{f}_0$ at positions $\pi^{-1}(\mathbf{j}')$ and those of $\hat{f}_1 \circ \pi$ at positions $\mathbf{j}'$.

Note that the prover can cheat to its heart's desire about these additional claims, but the point is that we now have a single set $\mathbf{j}'' = \mathbf{j}' \cup \pi^{-1}(\mathbf{j}')$ so that each function f_b is still δ_b far from the claims that we have about $f_b|_{\mathbf{j}''}$. Since the claims are now properly aligned, we can derive a new sequence of claims about f'. More importantly, we prove a technical lemma (building on the result of Ben Sasson *et al.* [BKS18]), showing that if f is δ-far from $\mathsf{PVAL}(t, \mathbf{j}, \mathbf{v})$ then, with high probability, f' is roughly 2δ-far from the corresponding PVAL instance (induced by the prover's new claims).

To summarize, the approach so far lets us double the distance in each iteration as we desired. Unfortunately, it also raises a new problem: the arity of the new PVAL instance that we generated has doubled — rather than having just t claims, we now have roughly $2t$ claims (corresponding to the size of the set $\mathbf{j}''$). See Fig. 3 for an illustration.

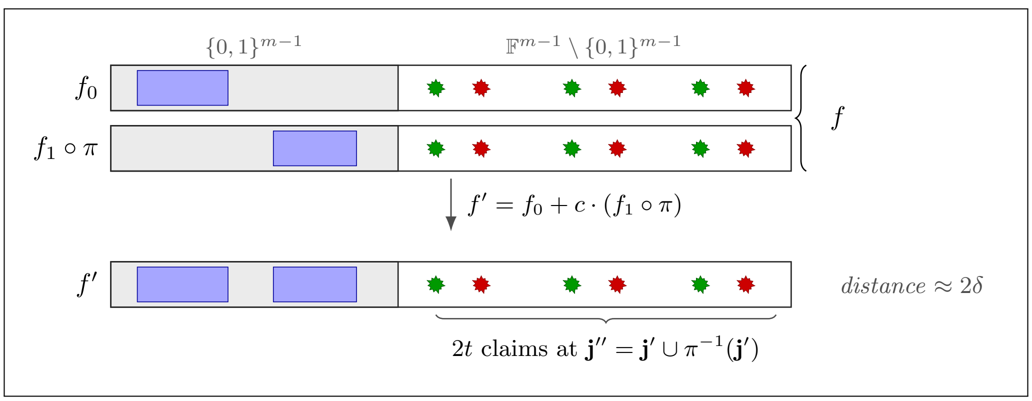

Figure 3: Permuting the inputs and the resulting arity growth. After applying a random shift permutation π to f_1, the error patterns of f_0 and $f_1 \circ \pi$ are (w.h.p.) nearly disjoint, so $f' = f_0 + c \cdot (f_1 \circ \pi)$ is roughly 2δ-far from PVAL. The price is that the verifier now holds claims at the larger set $\mathbf{j}'' = \mathbf{j}' \cup \pi^{-1}(\mathbf{j}')$, so the arity has grown from t to $2t$: the red starbursts mark the original t positions $\mathbf{j}'$ and the green starbursts mark the t added positions $\pi^{-1}(\mathbf{j}')$.

Arity Reduction Step. We resolve this final difficulty by once more employing interaction, and using the prover in order to reduce the $2t$ claims that we have about f' to just t (aligned) claims each, while preserving the distance.

The idea here is to consider a degree $O(t)$ curve $\mathcal{C} : \mathbb{F} \to \mathbb{F}^m$ passing through the set of points $\mathbf{j}''$. The prover sends to the verifier the values of $\hat{f}'|_{\mathcal{C}}$. The verifier checks that the provided values lie on a degree $O(t)$ univariate polynomial (since $\hat{f}' \circ \mathcal{C}$ has low degree). The verifier also checks that the values that correspond to points in the set $\mathbf{j}''$, are consistent with the claims that it has. The verifier now chooses a set of t random points $\rho = (\rho_1, \ldots, \rho_t)$ on the curve. The new claims about $\hat{f}'$ are those that correspond to the set of points in ρ. In particular, this lets us reduce the number of claims from $2t$ to t.

We want to argue that this arity-reduction sub-protocol preserves the distance. This is accomplished by taking a union bound over all inputs that are (roughly) 2δ-close to f', and showing that for each of them, the probability that it satisfies the new claim is tiny. We conclude that f' is indeed (roughly) 2δ-far from the resulting PVAL instance.

Recursion. Putting the previous components together, one iteration of our protocol takes a $\mathsf{PVAL}(t, \mathbf{j}, \mathbf{v})$ instance over $\{0,1\}^m$ at distance δ (that is not too large[5]), and produces a new $\mathsf{PVAL}(t, \mathbf{j}', \mathbf{v}')$ instance over $\{0,1\}^{m-1}$ at distance roughly 2δ, while preserving the arity t. Each query the verifier makes to the new (implicit) input f' is emulated by two queries to the original input f.

We invoke this sub-protocol recursively, halving the dimension and (roughly) doubling the distance in each step, until the dimension shrinks to $\sim \log(t)$, so that the implicit input has size $\sim t$. At this point the prover simply sends an alleged copy of the implicit input to the verifier; the verifier checks consistency with the claims about the final implicit input from the last level of the recursion and additionally spot-checks the final implicit input vs. the one sent by the prover at a small number of random positions.

Tallying the costs across the $m - \log(t)$ levels of recursion, the verifier's query count to the original input f doubles per recursion level, but the distance also doubles, and the two effects offset: the verifier ultimately makes only $\approx 1/\delta$ queries to f. This holds as long as the original distance δ was at most $\sim t/2^m$ (which ensures that the distance is not too large in each step of the recursion). This yields an IPP for PVAL in which the communication complexity is quasi-linear in t and the query complexity is roughly $1/\delta \approx 2^m/t$. Thus, the product of the query and communication complexities is quasi-linear in the input length 2^m, as desired.

Remark 2.1 (Changes from the conference version of this work [RR20])**.** *The conference version of this work [RR20] has the following error, pointed out by Goyal, Hong and Kalai [GHK25]: the function $\pi : \mathbb{F}^m \to \mathbb{F}^m$ used there is of the form $\pi_{A,b}(x) = Ax + b$ (for a specific random choice of a matrix A and vector b). This permutation ensures that the composed function $\hat{f} \circ \pi$ has* total *degree m, but it may not be multilinear.[6] Thus it was not true that $\widehat{f_1 \circ \pi} \equiv \hat{f}_1 \circ \pi$.*

To resolve this issue, we choose π to be the simpler permutation $\pi(x) = x + b$. Notice that $f \circ \pi(x) = f(x_1 + b_1, \ldots, x_m + b_m)$ is indeed multilinear, as long as f is multilinear. Moreover, this family is 1-wise independent: each input in $\{0,1\}^m$ is mapped to a random input in the same set. This suffices for proving the technical lemma showing that if f is δ-far from $\mathsf{PVAL}(t, \mathbf{j}, \mathbf{v})$ then, with high probability, f' is roughly 2δ-far from the corresponding PVAL instance (induced by the prover's new claims). The new construction incurs a larger soundness error because the permutation family

[5] If the distance is close to 1, then, needless to say, we cannot expect for it to double.

[6] For example, taking $\pi(x_1, x_2) = (x_1, x_1 + x_2)$ and $\hat{f}(x_1, x_2) = x_1 \cdot x_2$, the composed function $x_1 \cdot (x_1 + x_2)$ is quadratic in x_1.

is only 1-wise independent (rather than pairwise independent in the prior version of this work), but this overhead is not significant for our end result.

Berger, Goyal, Hong and Kalai [BGHK25] suggested an alternative modification to the protocol, where there is no change to the permutation family. Omitting many details, the high level idea is to use the GKR protocol [GKR15] to translate claims about the composed function $\hat{f}_1 \circ \pi$ to claims about the multilinear extension of $\widehat{f_1 \circ \pi}$. This retains the small soundness error associated with using a pairwise-independent permutation family, but incurs overhead in the additional complexity of running the GKR protocol for each "divide and conquer" step.

Finally, we remark that in the original version of this work we applied a permutation to both f_0 and f_1. This revision applies the permutation only to f_1, which slightly simplifies the exposition (this change has no effect on soundness).

3 Preliminaries

For a string $x \in \Sigma^n$ and an index $i \in [n]$, we denote by $x_i \in \Sigma$ the i^{th} entry in x. If $I \subseteq [n]$ is a set then we denote by $x|_I$ the sequence of entries in x corresponding to coordinates in I.

Let $x, y \in \Sigma^n$ be two strings of length $n \in \mathbb{N}$ over a (finite) alphabet Σ. We define the (relative Hamming) distance of x and y as $\Delta(x, y) \stackrel{\text{def}}{=} |\{x_i \neq y_i : i \in [n]\}| / n$. If $\Delta(x, y) \leq \varepsilon$, then we say that x is ε-close to y, and otherwise we say that x is ε-far from y. We define the distance of x from a (non-empty) set $S \subseteq \Sigma^n$ as $\Delta(x, S) \stackrel{\text{def}}{=} \min_{y \in S} \Delta(x, y)$. If $\Delta(x, S) \leq \varepsilon$, then we say that x is ε-close to S and otherwise we say that x is ε-far from S. We extend these definitions from strings to functions by identifying a function with its truth table. For a set S, of size $|S| \geq 2$, take its minimum distance to be the minimum, over all distinct vectors $x, y \in S$ of $\Delta(x, y)$. We use $\Delta(S)$ to denote the minimum distance of S. Fixing a vector space, for a set S and a vector x, we denote $(x + S) = \{x + y : y \in S\}$. For a scalar c, we denote $(c \cdot S) = \{c \cdot y : y \in S\}$.

3.1 Multivariate Polynomials and Low Degree Extensions

We recall some important facts on multivariate polynomials (see [Sud95] for a far more detailed introduction). A basic fact, captured by the Schwartz-Zippel lemma is that low degree polynomials cannot have too many roots.

Lemma 3.1 (Schwartz-Zippel Lemma). *Let $P : \mathbb{F}^m \to \mathbb{F}$ be a* non-zero *polynomial of total degree d. Then,*

$$\Pr_{x \in \mathbb{F}^m} \left[P(x) = 0 \right] \leq \frac{d}{|\mathbb{F}|}.$$

An immediate corollary of the Schwartz-Zippel Lemma is that two distinct polynomials $P, Q : \mathbb{F}^m \to \mathbb{F}$ of total degree d may agree on at most a $\frac{d}{|\mathbb{F}|}$-fraction of their domain $\mathbb{F}^m$.

Throughout this work we consider fields in which operations can be implemented efficiently (i.e., in poly-logarithmic time in the field size). Formally we define such fields as follows.

Definition 3.2. *We say that an ensemble of finite fields $\mathbb{F} = (\mathbb{F}_n)_{n \in \mathbb{N}}$ is* constructible *if elements in $\mathbb{F}_n$ can be represented by $O(\log(|\mathbb{F}_n|))$ bits and field operations (i.e., addition, subtraction, multiplication, inversion and sampling random elements) can all be performed in $\mathsf{polylog}(|\mathbb{F}_n|)$ time given this representation.*

A well known fact is that for every $S = S(n)$, there exists a *constructible* field ensemble of size $O(S)$ and its representation can be found in $\mathsf{polylog}(S)$ time (see, e.g., [Gol08, Appendix G.3] for details).

Let $\mathbb{H}$ be a finite field and $\mathbb{F} \supseteq \mathbb{H}$ be an extension field of $\mathbb{H}$. Fix an integer $m \in \mathbb{N}$. A basic fact is that for every function $\phi : \mathbb{H}^m \to \mathbb{F}$, there exists a unique extension of ϕ into a function $\hat{\phi} : \mathbb{F}^m \to \mathbb{F}$ (which agrees with ϕ on $\mathbb{H}^m$; i.e., $\hat{\phi}|_{\mathbb{H}^m} \equiv \phi$), such that $\hat{\phi}$ is an m-variate polynomial of individual degree at most $|\mathbb{H}| - 1$. Moreover, there exists a collection of $|\mathbb{H}|^m$ functions $\{\hat{\tau}_x\}_{x\in\mathbb{H}^m}$ such that each $\hat{\tau}_x : \mathbb{F}^m \to \mathbb{F}$ is the m-variate polynomial of degree $|\mathbb{H}| - 1$ in each variable defined as:

$$\hat{\tau}_x(z) \stackrel{\text{def}}{=} \prod_{i\in[m]} \prod_{h\in\mathbb{H}\setminus\{x_i\}} \frac{z_i - h}{x_i - h}.$$

and for every function $\phi : \mathbb{H}^m \to \mathbb{F}$ it holds that

$$\hat{\phi}(z_1, \ldots, z_m) = \sum_{x\in\mathbb{H}^m} \hat{\tau}_x(z_1, \ldots, z_m) \cdot \phi(x).$$

The function $\hat{\phi}$ is called the *low degree extension* of ϕ (with respect to $\mathbb{F}$, $\mathbb{H}$ and m). In the special case in which $\mathbb{H} = \mathbb{GF}(2)$, the function $\hat{\phi}$ (which has individual degree 1) is called the *multilinear extension* of ϕ (with respect to $\mathbb{F}$ and m).

3.2 Shift Permutations

Let $m \in \mathbb{N}$ and let $\mathbb{F}$ be a finite field that is an extension field of $\mathbb{GF}(2)$. For $b \in (\mathbb{GF}(2))^m$ define the function $\pi_b : \mathbb{F}^m \to \mathbb{F}^m$ as: $\pi_b(x) = x + b$. Denote by $\pi_b|_{(\mathbb{GF}(2))^m}$ the restriction of π_b to the domain $(\mathbb{GF}(2))^m$ and let $\Pi_m = \{\pi_b \; : \; b \in (\mathbb{GF}(2))^m\}$.

Fact 3.3. *For every $b \in (\mathbb{GF}(2))^m$:*

1. *The function π_b is a permutation over $\mathbb{F}^m$ and $\pi_b|_{(\mathbb{GF}(2))^m}$ is a permutation over $(\mathbb{GF}(2))^m$.*
2. *The function family $\{\pi_b|_{(\mathbb{GF}(2))^m}\}_{a,b\in(\mathbb{GF}(2))^m}$ is 1-wise independent.*
3. *If $\mathbb{F}$ is constructible, then given $b \in (\mathbb{GF}(2))^m$ and $x \in \mathbb{F}^m$ it is possible to compute $\pi_b(x)$ in time $\mathsf{poly}(m, \log(|\mathbb{F}|))$.*

Proposition 3.4. *Let $\phi : (\mathbb{GF}(2))^m \to \mathbb{F}$. Let $b \in (\mathbb{GF}(2))^m$ and $\psi : (\mathbb{GF}(2))^m \to \mathbb{F}$ be defined as $\psi(x) = \phi(x + b)$. Let $\hat{\phi}$ and $\hat{\psi}$ denote the multilinear extensions of ϕ and ψ, respectively. Then:*

$$\forall x \in \mathbb{F}^m, \; (\hat{\phi} \circ \pi_b)(x) = \hat{\psi}(x).$$

Proof. For $x \in (\mathbb{GF}(2))^m$,

$$\hat{\phi} \circ \pi_b(x) = \hat{\phi}(x + b) = \phi(x + b) = \psi(x)$$

The claim follows by observing that the function $\hat{\phi} \circ \pi_b(x)$ multilinear, and two multilinear functions that agree on $\{0, 1\}^m$ must also agree on $\mathbb{F}^m$. $\square$

3.3 Succinct Descriptions

Throughout this work we use NC^1 to refer to the class of logspace uniform Boolean circuits of logarithmic depth and constant fan-in. Namely, $\mathcal{L} \in \mathsf{NC}^1$ if there exists a logspace Turing machine M that on input 1^n outputs a full description of a logarithmic depth circuit $C : \{0,1\}^n \to \{0,1\}$ such that for every $x \in \{0,1\}^n$ it holds that $C(x) = 1$ if and only if $x \in \mathcal{L}$.

We next define a notion of succinct representation of circuits. Loosely speaking, a function $f : \{0,1\}^n \to \{0,1\}$ has a succinct representation if there is a short string $\langle f \rangle$, of poly-logarithmic length, that describes f. That is, $\langle f \rangle$ can be expanded to a full description of f. The actual technical definition is slightly more involved and in particular requires that the full description of f be an NC^1 (i.e., logarithmic depth) circuit:

Definition 3.5 (Succinct Description of Functions)**.** *We say that a function $f : \{0,1\}^n \to \{0,1\}$ of size s has a* succinct description *if there exists a string $\langle f \rangle$ of length $\mathsf{polylog}(n)$ and a logspace Turing machine M (of constant size, independent of n) such that on input 1^n, the machine M outputs a full description of an NC^1 circuit C such that for every $x \in \{0,1\}^n$ it holds that $C(\langle f \rangle, x) = f(x)$. We refer to $\langle f \rangle$ as the succinct description of f.*

We also define succinct representation for sets $S \subseteq [k]$. Roughly speaking this means that the set can be described by a string of length $\mathsf{polylog}(k)$. The formal definition is somewhat more involved:

Definition 3.6 (Succinct Description of Sets)**.** *We say that a set $S \subseteq [k]$ of size s has a succinct description if there exists a string $\langle S \rangle$ of length $\mathsf{polylog}(k)$ and a logspace Turing machine M such that on input 1^k, the machine M outputs a full description of a depth $\mathsf{polylog}(k)$ and size $\mathsf{poly}(s, \log k)$ circuit (of constant fan-in) that on input $\langle S \rangle$ outputs all the elements of S as a list (of length $s \cdot \log(k)$).*

We emphasize that the size of the circuit that M outputs is proportional to the actual size of the set S, rather than the universe size k.

3.4 Interactive Proofs of Proximity

Loosely speaking, IPPs are interactive proofs in which the verifier runs in sub-linear time in the input length, where the soundness requirement is relaxed to rejecting inputs that are *far* from the language w.h.p. (for inputs that are not in the language, but are close to it, no requirement is made). Actually, we will think of the input of the verifier as being composed of two parts: an *explicit* input $x \in \{0,1\}^n$ to which the verifier has direct access, and an *implicit* (longer) input $y \in \{0,1\}^m$ to which the verifier has oracle access. The goal is for the verifier to run in time that is sub-linear in m and to verify that y is far from any y' such that the pair (x, y') are in the language. Since such languages are composed of input pairs, we refer to them as *pair languages*.

Definition 3.7 (Interactive Proof of Proximity (IPP) [EKR04, RVW13])**.** *An* interactive proof of proximity *(IPP) for the pair language $\mathcal{L}$ is an interactive protocol with two parties: a (computationally unbounded) prover $\mathcal{P}$ and a computationally bounded verifier $\mathcal{V}$. Both parties get as input $x \in \{0,1\}^n$ and a proximity parameter $\varepsilon > 0$. The verifier also gets oracle access to $y \in \{0,1\}^m$ whereas the prover has full access to y. At the end of the interaction, the following two conditions are satisfied:*

1. **Completeness:** *For every pair* $(x, y) \in \mathcal{L}$*, and proximity parameter* $\varepsilon > 0$ *it holds that*

$$\Pr\left[\left(\mathcal{P}(y), \mathcal{V}^y\right)(x, |y|, \varepsilon) = 1\right] = 1.$$

2. **Soundness:** *For every* $\varepsilon > 0$*,* $x \in \{0,1\}^n$ *and* y *that is* ε*-far from the set* $\{y' : (x, y') \in \mathcal{L}\}$*, and for every computationally unbounded (cheating) prover* $\mathcal{P}^*$ *it holds that*

$$\Pr\left[\left(\mathcal{P}^*(y), \mathcal{V}^y\right)(x, |y|, \varepsilon) = 1\right] \leq 1/2.$$

An IPP for $\mathcal{L}$ is said to have query complexity $q = q(n, m, \varepsilon)$ if, for every $\varepsilon > 0$ and $(x, y) \in \mathcal{L}$, the verifier $\mathcal{V}$ makes at most $q(|x|, |y|, \varepsilon)$ queries to y when interacting with $\mathcal{P}$. The IPP is said to have communication complexity $\mathsf{cc} = \mathsf{cc}(n, m, \varepsilon)$ if, for every $\varepsilon > 0$ and pair $(x, y) \in \mathcal{L}$, the communication between $\mathcal{V}$ and $\mathcal{P}$ consists of at most $\mathsf{cc}(|x|, |y|, \varepsilon)$ bits. If the honest prover's running time is polynomial in n and m, then we way that the IPP is *doubly-efficient*.

The special case of IPPs in which the entire interaction consists of a single message sent from the prover to the verifier is called MAPs (in analogy to the complexity class MA) and was studied in [GR17, GGR15]. We will use the following simple observation:

Proposition 3.8 (See, e.g., [GR17])**.** *Every* $\mathcal{L} \in \mathsf{DTIME}(t)$ *has an* MAP *with respect to proximity parameter* $\delta \in (0, 1)$ *with communication complexity* n *and query complexity* $O(1/\delta)$*. The verifier runs in time* $t + n + O(\log(n)/\delta)$*. The prover runs in time* $O(n)$

Proof Sketch. The prover sends to the verifier a full description of the input x (i.e., an n bit string). Given the message x' received from the prover (allegedly equal to the input x), the verifier first checks that $x' \in \mathcal{L}$ (this step requires no queries to x). The verifier further checks that x and x' agree on a random set of $O(1/\delta)$ coordinates.

Completeness is immediate, whereas to see that soundness holds, observe that the prover must send $x' \in \mathcal{L}$, since otherwise the verifier rejects. If x is δ-far from $\mathcal{L}$ then x and x' disagree on at least a δ fraction of their coordinates and so the verifier accepts with probability at most $(1 - \delta)^{O(1/\delta)} = 1/2$. ☐

4 Our Results

Our first main result is an IPP for any language in NC with optimal query/communication tradeoff (up to poly-logarithmic factors).

Theorem 3. *Let* $\delta = \delta(n) \in (0, 1)$ *be a proximity parameter and let* $\mathcal{L}$ *be a pair language that is computable by logspace-uniform Boolean circuits of depth* $D = D(n) \geq \log n$ *and size* $S = S(n) \geq n$ *with fan-in 2 (where* n *denotes the implicit input and* n_{exp} *denotes the explicit input). Then,* $\mathcal{L}$ *has a public-coin* IPP *for* δ*-proximity with perfect completeness and the following parameters:*

- *Soundness Error:* $1/2$.
- *Query complexity:* $q = O(1/\delta)$.
- *Communication Complexity:* $cc = \delta \cdot n \cdot D \cdot \mathsf{polylog}(S)$.

- *Round Complexity:* $D \cdot \mathsf{polylog}(S)$.
- *Verifier Running Time:* $\delta \cdot n \cdot n_{exp} \cdot \mathsf{poly}(D, \log(S)) + (1/\delta) \cdot \mathsf{polylog}(n)$.
- *Prover Running Time:* $\mathsf{poly}(S)$.

Furthermore, the verification procedure can be described succinctly as follows. At the end of the interaction either the verifier rejects or in time $\delta \cdot n \cdot \mathsf{poly}(D, \log(S))$ *it outputs a succinct description* $\langle Q \rangle$ *of a set* $Q \subseteq [n]$ *of size* q *and a succinct description* $\langle \phi \rangle$ *of a predicate* $\phi : \{0,1\}^q \to \{0,1\}$ *so that its decision predicate given an input function* f *is equal to* $\phi(f|_Q)$.

Our second main result (which relies on Theorem 3) is an interactive proof for batch verification of any UP language, with communication complexity that is optimal up to poly-logarithmic factors.

Theorem 4. *For every* UP *language* $\mathcal{L}$ *with witness length* $m = m(n)$*, whose witness relation can be computed in logspace-uniform* NC*, there exists a public-coin interactive proof (with perfect completeness) for verifying that* k *instances* $x_1, \ldots, x_k$*, each of length* $n \leq \mathsf{poly}(m)$*, are all in* $\mathcal{L}$*. The complexity of the protocol is as follows:*

- *Communication complexity:* $m \cdot \mathsf{polylog}(k, m)$.
- *Number of rounds:* $\mathsf{polylog}(k, m)$.
- *Verifier runtime:* $(n \cdot k + m) \cdot \mathsf{polylog}(k, m)$.
- *The honest prover, given the* k *unique witnesses, runs in time* $\mathsf{poly}(m, k)$.

Using the Cook-Levin reduction, any UP language can be reduced to Unique-SAT which is a UP language whose witness relation can be computed in logspace-uniform NC, with only a $\mathsf{poly}(n, m)$ blowup to the witness size. Hence, Theorem 4 yields the following corollary.

Corollary 5. *For every* UP *language* $\mathcal{L}$ *with witness length* $m = m(n)$*, there exists a public-coin interactive proof (with perfect completeness) for verifying that* k *instances* $x_1, \ldots, x_k$*, each of length* $n \leq \mathsf{poly}(m)$*, are all in* $\mathcal{L}$*. The complexity of the protocol is as follows:*

- *Communication complexity:* $\mathsf{poly}(m, \log(k))$.
- *Number of rounds:* $\mathsf{polylog}(m, k)$.
- *Verifier runtime:* $(n \cdot k) \cdot \mathsf{polylog}(m, k) + \mathsf{poly}(m, \log k)$.
- *The honest prover, given the* k *unique witnesses, runs in time* $\mathsf{poly}(m, k)$.

5 The PVAL Problem

Let $\mathbb{F}$ be a finite field, $\mathbb{H} \subseteq \mathbb{F}$ and $m \in \mathbb{N}$ be an integer.

Definition 5.1. *The* PVAL *problem is parameterized by an ensemble* $(\mathbb{F}, \mathbb{H}, m)_n$*. The explicit input to the problem is* $(n, t, \mathbf{j}, \mathbf{v})$*, where* $t \in \mathbb{N}$*,* $\mathbf{j} = (j_1, \ldots, j_t) \in (\mathbb{F}^m)^t$ *and* $\mathbf{v} = v_1, \ldots, v_t \in \mathbb{F}^t$*. The implicit input is a function* $f : \mathbb{H}^m \to \mathbb{F}$*. YES instances of the problems are all functions* $f : \mathbb{H}^m \to \mathbb{F}$ *such that for every* $i \in [t]$ *it holds that* $\hat{f}(j_i) = v_i$*, where* $\hat{f}$ *is the low degree extension of* f*.*

The following two claims follow directly from the definition of PVAL:

Claim 5.2. *Let $\mathbb{H} \subseteq \mathbb{F}$ be finite fields, and let m, t be integers. For any set of locations $\mathbf{j} \in (\mathbb{F}^m)^t$, the set $\mathsf{PVAL}(t, \mathbf{j}, \mathbf{0})$ is a linear space over functions from $\mathbb{H}^m$ to $\mathbb{F}$. For non-zero vectors $\mathbf{v} \in \mathbb{F}^t$, the set $\mathsf{PVAL}(t, \mathbf{j}, \mathbf{v})$ is an affine space.*

Claim 5.3. *Let $\mathbb{H} \subseteq \mathbb{F}$ be finite fields, and let m, t be integers. For any set of locations $\mathbf{j} \in (\mathbb{F}^m)^t$, and for any set of values $\mathbf{v} \in \mathbb{F}^t$ such that $\mathsf{PVAL}(t, \mathbf{j}, \mathbf{v})$ is not empty, $\Delta(\mathsf{PVAL}(t, \mathbf{j}, \mathbf{0})) = \Delta(\mathsf{PVAL}(t, \mathbf{j}, \mathbf{v}))$.*

Since the low-degree extension is an error correcting code with high distance, for sufficiently large randomly chosen location sets $\mathbf{j}$, the induced PVAL problem has large minimum distance:

Proposition 5.4 (PVAL on random locations has large distance)**.** *Let $\mathbb{H} \subseteq \mathbb{F}$ be finite fields, and let m, d be integers s.t. $|\mathbb{F}| \geq 2m|\mathbb{H}|$. For $t \geq (d \cdot \log(|\mathbb{H}|^m \cdot |\mathbb{F}|) + \kappa)$ it is the case that:*

$$\Pr_{\mathbf{j} \in (\mathbb{F}^m)^t} \left[\Delta(\mathsf{PVAL}(t, \mathbf{j}, \mathbf{0})) \leq \frac{d}{|\mathbb{H}|^m} \right] < 2^{-\kappa}$$

Proof. By Claim 5.2, $\mathsf{PVAL}(t, \mathbf{j}, \mathbf{0})$ is a linear space over functions from $\mathbb{H}^m$ to $\mathbb{F}$. Thus, its minimal distance equals the minimal weight of a non-zero vector in the set. Fix a function $f : \mathbb{H}^m \to \mathbb{F}$ of weight at most d. Since $\hat{f}$ is a non-zero low-degree function, by the Schwartz-Zippel Lemma:

$$\begin{aligned}
\Pr_{\mathbf{j}}[f \in \mathsf{PVAL}(t, \mathbf{j}, \mathbf{0})] &\leq \left(\frac{m \cdot (|\mathbb{H}| - 1)}{|\mathbb{F}|} \right)^t \\
&\leq \left(\frac{1}{2} \right)^t \\
&\leq \frac{1}{2^\kappa \cdot |\mathbb{H}|^{m \cdot d} \cdot |\mathbb{F}|^d}.
\end{aligned}$$

The number of functions $f : \mathbb{H}^m \to \mathbb{F}$ of weight at most d is smaller than $(|\mathbb{H}|^m \cdot |\mathbb{F}|)^d$, and the proposition follows by taking a union bound over all such functions. ☐

We conclude with a technical claim, showing that if we change a PVAL instance by permuting the set of locations $\mathbf{j}$ using a permutation from the useful family introduced in Section 3.2 the minimum distance does not change.

Proposition 5.5. *Let $\mathbb{F}$ be an extension field of $\mathbb{GF}(2)$, let $m, t > 0$ be integers, and let Π be the family of shift permutations defined in Section 3.2. Taking $\mathbb{H} = \mathbb{GF}(2)$, for every $\mathbf{j} \in (\mathbb{F}^m)^t$ and $\mathbf{v} \in \mathbb{F}^t$, and for every (non-zero) permutation $\pi \in \Pi$:*

$$\Delta\left(\mathsf{PVAL}(t, \mathbf{j}, \mathbf{v})\right) = \Delta\left(\mathsf{PVAL}(t, \pi^{-1}(\mathbf{j}), \mathbf{v})\right).$$

Proof. Let f, f' be distinct functions in $\mathsf{PVAL}(t, \mathbf{j}, \mathbf{v})$ that minimize the distance. By Proposition 3.4, $(f \circ \pi)$ and $(f' \circ \pi)$ are functions in $\mathsf{PVAL}(t, \pi^{-1}(\mathbf{j}), \mathbf{v})$. Moreover, since π is a permutation:

$$\Delta\left(f, f'\right) = \Delta\left((f \circ \pi), (f' \circ \pi)\right).$$

We conclude that $\Delta\left(\mathsf{PVAL}(t, \mathbf{j}, \mathbf{v})\right) \geq \Delta\left(\mathsf{PVAL}(t, \pi^{-1}(\mathbf{j}), \mathbf{v})\right)$. The reverse inequality follows similarly. ☐

5.1 Interactive Distance-Preserving Reduction to PVAL

Recall the definitions of pair languages and IPPs for those languages (see Section 3.4). We restate the distance-preserving reduction of [RVW13] from any language computable by bounded-depth circuits to PVAL:

Theorem 5.6 (See Theorem 1.3 in [RVW13], restated for pair languages and LDE)**.** *Let $\delta = \delta(n) \in [0,1)$ be a distance bound and let $\mathcal{L}$ be a pair language that is computable by logspace-uniform Boolean circuits of depth $D = D(n) \geq \log n$ and size $S = S(n) \geq n$ with fan-in 2. There exists a uniform ensemble $(\mathbb{F}, m)_n$ where $|\mathbb{F}| = \mathsf{poly}(D, \log S)$ and $m = O(\log S)$, such that taking $t = t(n) = O(\delta \cdot n \log n)$, there exists an interactive protocol between a prover and a verifier with the following properties:*

- *The prover's input is $x \in \{0,1\}^{n_{exp}}, y \in \{0,1\}^{n_{imp}}$, the verifier's input is x (in particular, the verifier does not access y). The verifier's output is either "reject" or a PVAL instance over the fields $\mathbb{F}, \mathbb{H} = \{0,1\}$, comprised of $\mathbf{j} \in (\mathbb{F}^m)^t$ and $\mathbf{v} \in \mathbb{F}^t$.*
- *If $(x,y) \in \mathcal{L}$ and the prover follows the protocol honestly, then $y \in \mathsf{PVAL}(t, \mathbf{j}, \mathbf{v})$.*
- *If y is δ-far from $\mathcal{L}(x)$, then for any cheating prover, with all but $1/2$ probability over the verifier's coin tosses, y is also δ-far from $\mathsf{PVAL}(t, \mathbf{j}, \mathbf{v})$.*
- *For any cheating prover, the vector $\mathbf{j}$, viewed as a random variable over the verifier's coin tosses, is uniformly random in $(\mathbb{F}^m)^t$.*

The prover's running time is $\mathsf{poly}(S)$, the verifier's running time is $O(\delta \cdot n_{imp} \cdot D \cdot \mathsf{polylog} S + n_{exp} \cdot \log|S|)$, the communication is $(\delta \cdot n_{imp} \cdot D \cdot \mathsf{polylog} S)$, and the number of rounds is $O(D \log S)$.

We remark that the condition on the distribution of $\mathbf{j}$ follows immediately from the protocol, which runs t parallel and independent invocations of the GKR interactive proof [GKR15]. Each of these invocation outputs a claim about the value of the input's LDE at a uniformly random coordinate chosen by the verifier.

5.2 Approximate Distance Preservation for PVAL Instances

Lemma 5.7 below, due to [RVW13], will be used in the *simple* case of the distance-amplification analysis: when the average distance is already large enough that no amplification is needed.

Lemma 5.7 ([RVW13], Lemma 1.6. See also Claim 3.11)**.** *Fix a finite field $\mathbb{F}$ of characteristic 2 and integers $m, t > 0$. For $\mathbf{j} \in (\mathbb{F}^m)^t$, suppose that $\mathsf{PVAL}(t, \mathbf{j}, \mathbf{0})$ has size strictly larger than 1. Let $\mathbf{v}', \mathbf{v}'' \in \mathbb{F}^t$ be vectors s.t. both $\mathsf{PVAL}(t, \mathbf{j}, \mathbf{v}')$ and $\mathsf{PVAL}(t, \mathbf{j}, \mathbf{v}'')$ are non-empty. Let $f' : \{0,1\}^m \to \mathbb{F}$ be at distance δ' from $\mathsf{PVAL}(t, \mathbf{j}, \mathbf{v}')$, and let $f'' : \{0,1\}^m \to \mathbb{F}$ be at distance δ'' from $\mathsf{PVAL}(t, \mathbf{j}, \mathbf{v}'')$.*

For scalars $c', c'' \in \mathbb{F}$, define:

$$f \triangleq c' \cdot f' + c'' \cdot f''.$$

Then:

$$\Pr_{c', c'' \leftarrow \mathbb{F}} \left[\Delta \left(f, \mathsf{PVAL}(t, \mathbf{j}, (c' \cdot \mathbf{v}' + c'' \cdot \mathbf{v}'')) \right) < \max(\delta, \delta')/2 \right] \leq \frac{1}{|\mathbb{F}| - 1}.$$

5.3 Distance Amplification for PVAL Instances

In this section we describe the distance-amplification procedure for PVAL instances. We begin by stating technical claims about distances from linear and affine spaces from prior works, and then proceed to prove the distance amplification lemma (Lemma 5.11).

RVW Lemma. The following lemma, due to Rothblum, Vadhan and Wigderon [RVW13], asserts that if there is even a single point on a line that is far from a linear subspace, then almost every point on the line is (a bit less) far form the linear space. The lemma loses a multiplicative factor of 2 in the conclusion about nearly all points on the line. Here we state and prove a statement tailored to our usage of the lemma, which has to do with affine spaces.

Lemma 5.8 (Lemma 1.6 in [RVW13], affine variant)**.** *Let $V \subseteq \mathbb{F}^n$ be a non-empty linear space over a finite field $\mathbb{F}$. Suppose there are vectors $\mathbf{x}, \mathbf{x}^*, \mathbf{a}, \mathbf{a}^* \in \mathbb{F}^n$ s.t. $\mathbf{x}$ is at distance at least δ from the affine space $(V + \mathbf{a})$ and $\mathbf{x}^*$ is at distance at least δ^* from the affine space $(V + \mathbf{a}^*)$, then:*

$$\Pr_{\alpha \in \mathbb{F}}\left[\Delta\left((\mathbf{x} + \alpha \cdot \mathbf{x}^*), (V + \mathbf{a} + \alpha \cdot \mathbf{a}^*)\right) < \frac{\max(\delta, \delta^*)}{2}\right] \leq \frac{1}{|\mathbb{F}| - 1}. \tag{2}$$

Proof. If Eq. (2) doesn't hold there exist distinct $\alpha', \alpha'' \in \mathbb{F}$ and $\mathbf{v}', \mathbf{v}'' \in V$ s.t.:

$$\Delta\left((\mathbf{x} - \mathbf{a}) + \alpha'(\mathbf{x}^* - \mathbf{a}^*), \mathbf{v}'\right) < \max(\delta, \delta^*)/2,$$
$$\Delta\left((\mathbf{x} - \mathbf{a}) + \alpha''(\mathbf{x}^* - \mathbf{a}^*), \mathbf{v}''\right) < \max(\delta, \delta^*)/2.$$

Take $I', I'' \subseteq [n]$ to be the sets of indices where $((\mathbf{x} - \mathbf{a}) + \alpha'(\mathbf{x}^* - \mathbf{a}^*))$ differs from $\mathbf{v}'$ and where $((\mathbf{x} - \mathbf{a}) + \alpha''(\mathbf{x}^* - \mathbf{a}^*))$ differs from $\mathbf{v}''$ (respectively), and take $I = I_1 \cup I_2$ to be their union, a set of size smaller than $\max(\delta, \delta^*)$.

Similarly to the proof in [RVW13], for any linear combination $\mathbf{y}_{\beta,\gamma} = (\beta \cdot (\mathbf{x} - \mathbf{a}) + \gamma(\mathbf{x}^* - \mathbf{a}^*))$, there is a vector $\mathbf{v}_{\beta,\gamma} \in V$ s.t. $\mathbf{y}_{\beta,\gamma}$ and $\mathbf{v}_{\beta,\gamma}$ only differ on indices in I (and in particular they are $\max(\delta, \delta^*)$-close). In particular, this is true for $\mathbf{y}_{1,0} = (\mathbf{x} - \mathbf{a})$, and we conclude that $\mathbf{x}$ is at distance smaller than $\max(\delta, \delta^*)$ from $(V + \mathbf{a})$. Similarly, the foregoing statement is also true for the linear combination $\mathbf{y}_{0,1} = (\mathbf{x}^* - \mathbf{a}^*)$, and we also have that $\mathbf{x}^*$ is at distance smaller than $\max(\delta, \delta^*)$ from $(V + \mathbf{a}^*)$. But one of these two conclusions must be false, leading to a contradiction. ☐

BKS Lemma. The following lemma, due to Ben-Sasson, Kopparty and Saraf [BKS18], asserts that for any linear code V, and any line passing through a point that is δ-far from the code there cannot be "too many" points on the line that are at distance significantly smaller than δ from the code. This is true for any δ that is significantly smaller than the code's distance. Taking the counterpositive: if there are "enough" points on the line that are "close enough" to the code, then it must be the case that *every* point on the line is close to the code. Moreover, the set of locations on which *any* point on the line differs from its nearest codeword is small.

Lemma 5.9 (Lemma 4.2 in [BKS18])**.** *Let $V \subseteq \mathbb{F}^n$ be a non-empty linear space over a finite field $\mathbb{F}$, such that $\Delta(V) = \lambda$. For $\varepsilon, \delta > 0$ satisfying $\delta - \varepsilon < \lambda/3$, suppose there are two points $\mathbf{u}, \mathbf{u}^* \in \mathbb{F}^n$ such that:*

$$|\{\alpha \in \mathbb{F} : \Delta(\mathbf{u}^* + \alpha\mathbf{u}, V) < \delta - \varepsilon\}| > 1/\varepsilon,$$

then there exist $\mathbf{v}, \mathbf{v}^* \in V$ such that:

$$|\{i \in [n] : (\mathbf{u}_i = \mathbf{v}_i) \wedge (\mathbf{u}^*_i = \mathbf{v}^*_i)\}| \geq (1 - \delta) \cdot n.$$

The following straightforward corollary extends the lemma to affine spaces:

Corollary 5.10 (Affine variant of Lemma 5.9)**.** *Let $V \subseteq \mathbb{F}^n$ be a non-empty linear space over a finite field $\mathbb{F}$, such that $\Delta(V) = \lambda$. For $\varepsilon, \delta > 0$ satisfying $\delta - \varepsilon < \lambda/3$, suppose there are vectors $\mathbf{x}, \mathbf{x}^*, \mathbf{a}, \mathbf{a}^* \in \mathbb{F}^n$ such that:*

$$|\{\alpha \in \mathbb{F} : \Delta(\mathbf{x} + \alpha\mathbf{x}^*, (V + \mathbf{a} + \alpha\mathbf{a}^*)) < \delta - \varepsilon\}| > 1/\varepsilon,$$

then there exist $\mathbf{y} \in (V + \mathbf{a})$ and $\mathbf{y}^* \in (V + \mathbf{a}^*)$ such that:

$$|\{i \in [n] : (\mathbf{x}_i = \mathbf{y}_i) \wedge (\mathbf{x}^*_i = \mathbf{y}^*_i)\}| \geq (1 - \delta) \cdot n.$$

Proof. Consider the vectors $\mathbf{u} = (\mathbf{x} - \mathbf{a})$ and $\mathbf{u}^* = (\mathbf{x}^* - \mathbf{a}^*)$. The conditions of Lemma 5.9 hold w.r.t the linear space V and the vectors $\mathbf{u}, \mathbf{u}^*$, and so there exist $\mathbf{v}, \mathbf{v}^* \in V$ s.t.

$$|\{i \in [n] : (\mathbf{u}_i = \mathbf{v}_i) \wedge (\mathbf{u}^*_i = \mathbf{v}^*_i)\}| \geq (1 - \delta) \cdot n.$$

The corollary follows by taking $\mathbf{y} = (\mathbf{v} + \mathbf{a})$ and $\mathbf{y}^* = (\mathbf{v}^* + \mathbf{a}^*)$. □

Distance amplification. The following lemma draws inspiration from the distance amplification theorem for Reed Solomon codes of Ben-Sasson, Kopparty and Saraf [BKS18] (see the discussion in Section 2 and Remark 2.1).

Lemma 5.11. *Fix a finite field $\mathbb{F}$ of characteristic 2 and integers $m, t > 0$. For $\mathbf{j} \in (\mathbb{F}^m)^t$, suppose that $\mathsf{PVAL}(t, \mathbf{j}, \mathbf{0})$ has size strictly larger than 1 and minimal distance λ. Let $\mathbf{v}', \mathbf{v}'' \in \mathbb{F}^t$ be vectors s.t. both $\mathsf{PVAL}(t, \mathbf{j}, \mathbf{v}')$ and $\mathsf{PVAL}(t, \mathbf{j}, \mathbf{v}'')$ are non-empty. Let $f' : \{0,1\}^m \to \mathbb{F}$ be at distance at least δ' from $\mathsf{PVAL}(t, \mathbf{j}, \mathbf{v}')$, and let $f'' : \{0,1\}^m \to \mathbb{F}$ be at distance at least δ'' from $\mathsf{PVAL}(t, \mathbf{j}, \mathbf{v}'')$. Consider a permutation $\pi \in \Pi$, where Π is the collection of shift-permutations over $\mathbb{F}^m$ defined in Section 3.2. For a scalar $c \in \mathbb{F}$, define:*

$$f \triangleq f' + c \cdot (f'' \circ \pi),$$

and let $S_\pi \subseteq \mathbb{F}^{2t}$ be the set of pairs of vectors $(\mathbf{u}, \mathbf{w})$ s.t. the sets $\mathsf{PVAL}\left(2t, (\mathbf{j}, \pi^{-1}(\mathbf{j})), (\mathbf{v}', \mathbf{u})\right)$ and $\mathsf{PVAL}\left(2t, (\mathbf{j}, \pi^{-1}(\mathbf{j})), (\mathbf{w}, \mathbf{v}'')\right)$ are non-empty. For $(\mathbf{u}, \mathbf{w}) \in S_\pi$, define:

$$\delta_{\pi,c,\mathbf{u},\mathbf{w}} = \Delta\left(f, \mathsf{PVAL}\left(2t, (\mathbf{j}, \pi^{-1}(\mathbf{j})), ((\mathbf{v}', \mathbf{u}) + c \cdot (\mathbf{w}, \mathbf{v}''))\right)\right).$$

Then for $\varepsilon, \gamma \in (0, 1/2]$, taking

$$\delta = \max\left(\frac{\delta' + \delta''}{2}, \min(\delta' + \delta'' - \frac{\delta'\delta''}{\gamma} - \varepsilon, \lambda/3 - \varepsilon)\right),$$

and assuming further ε, γ are chosen so that the above quantities are all non-negative, it is the case that:

$$\Pr_{\pi \leftarrow \Pi}\left[\exists (\mathbf{u}, \mathbf{w}) \in S_\pi \text{ s.t. } \Pr_{c \leftarrow \mathbb{F}}[\delta_{\pi,c,\mathbf{u},\mathbf{w}} < \delta] > \frac{1}{\varepsilon|\mathbb{F}|}\right] < \gamma \tag{3}$$

Proof of Lemma 5.11. We proceed with a case analysis. The simpler case is when $(\delta' + \delta'')/2 \geq \min(\delta' + \delta'' - \delta'\delta''/\gamma - \varepsilon, \lambda/3 - \varepsilon)$. In this case the lemma follows from the approximate distance preservation lemma [RVW13] (Lemma 5.7). By that lemma, for any $\pi \in \Pi$, and for any $(\mathbf{u}, \mathbf{w}) \in S_\pi$:

$$\Pr_{c \leftarrow \mathbb{F}}\left[\delta_{\pi,c,\mathbf{u},\mathbf{w}} < \frac{\max(\delta', \delta'')}{2}\right] \leq \frac{1}{|\mathbb{F}| - 1} \leq \frac{1}{\varepsilon|\mathbb{F}|},$$

where the second inequality above is because $\varepsilon \in (0, 1/2]$ and $|\mathbb{F}| \geq 2$. The bound follows by observing that $\max(\delta', \delta'') < \delta' + \delta''$.

We proceed to analyze the more involved case, where $(\delta' + \delta'')/2 < \min(\delta' + \delta'' - \delta'\delta''/\gamma - \varepsilon, \lambda/3 - \varepsilon)$. Note that in this case we have $\delta \leq \lambda/3 - \varepsilon$. Assume for contradiction that Eq. (3) doesn't hold: i.e., with probability at least γ over the choice of π there exist vectors $(\mathbf{u}, \mathbf{w}) \in S_\pi$ s.t.

$$\Pr_{c \leftarrow \mathbb{F}}\left[\delta_{\pi,c,\mathbf{u},\mathbf{w}} < \delta\right] > \frac{1}{\varepsilon|\mathbb{F}|}. \tag{4}$$

Analysis for fixed π. We begin by considering any fixed choice of π for which Eq. (4) holds, we define the linear space:

$$V = \mathsf{PVAL}\left(2t, (\mathbf{j}, \pi^{-1}(\mathbf{j})), \mathbf{0}\right),$$

and observe that (since V is a subcode of $\mathsf{PVAL}(t, \mathbf{j}, \mathbf{0})$) V's minimal distance is at least λ

Let z' and z'' be the closest vectors to f' and f'' in the corresponding PVAL set:

- $z' \in \mathsf{PVAL}(t, \mathbf{j}, \mathbf{v}')$ is at distance at least δ' from f',
- $z'' \in \mathsf{PVAL}(t, \mathbf{j}, \mathbf{v}'')$ is at distance at least δ'' from f'',

where if there is more than one minimal-distance vector in the relevant PVAL set then we take the lexicographically first one. Note that z', z'' exist because we assumed the aforementioned PVAL sets are non-empty.

Let $\mathbf{u}$ and $\mathbf{w}$ be the vectors promised in Eq. (4). Intuitively, if $\mathbf{u} = z'[\pi^{-1}(\mathbf{j})]$ and $\mathbf{w} = z''[\pi(\mathbf{j})]$, then the distances of f' and $\pi(f'')$ from the "further constrained" PVAL sets $\mathsf{PVAL}\left(2t, (\mathbf{j}, \pi^{-1}(\mathbf{j})), (\mathbf{v}', \mathbf{u})\right)$ and $\mathsf{PVAL}\left(2t, (\mathbf{j}, \pi^{-1}(\mathbf{j})), (\mathbf{w}, \mathbf{v}'')\right)$ remain δ' and δ'' (respectively). This is the challenging case to analyze: Otherwise, either f' or $\pi(f'')$ (or both) are very far from their "further constrained" PVAL sets and Eq. (4) cannot hold. This is shown in Claim 5.12.

Claim 5.12. *For $(\mathbf{u}, \mathbf{w}) \in S_\pi$ if either $\mathbf{u} \neq z'[\pi^{-1}(\mathbf{j})]$ or $\mathbf{w} \neq z''[\pi(\mathbf{j})]$, then*

$$\Pr_{c \in \mathbb{F}}\left[\delta_{\pi,c,\mathbf{u},\mathbf{w}} < \frac{\lambda}{3} - \varepsilon\right] \leq \frac{1}{\varepsilon|\mathbb{F}|}$$

Proof of Claim 5.12. Suppose that $\mathbf{u} \neq z'[\pi^{-1}(\mathbf{j})]$. Let $g' \in \mathsf{PVAL}\left(2t, (\mathbf{j}, \pi^{-1}(\mathbf{j})), (\mathbf{v}', \mathbf{u})\right)$ be the closest vector to f' in that affine set (the set is not empty because $(\mathbf{u}, \mathbf{w}) \in S_\pi$). We have:

$$g' \in \mathsf{PVAL}\left(2t, (\mathbf{j}, \pi^{-1}(\mathbf{j})), (\mathbf{v}', \mathbf{u})\right) \subseteq \mathsf{PVAL}\left(t, \mathbf{j}, \mathbf{v}'\right),$$

and moreover $g' \neq z'$ (because $g'[\pi^{-1}(\mathbf{j})] = \mathbf{u} \neq z'[\pi^{-1}(\mathbf{j})]$). Thus z' and g' are distinct vectors in $\mathsf{PVAL}\,(t, \mathbf{j}, \mathbf{v}'))$, an affine set with minimal distance λ, and so their distance is at least λ. The closest vector to f' in $\mathsf{PVAL}\,(t, \mathbf{j}, \mathbf{v}')$ is z', so the distance between g' and f' is at least $\lambda/2$. Thus, since g' is the closest vector in the affine set, we have that

$$\Delta(f', \mathsf{PVAL}\left(2t, (\mathbf{j}, \pi^{-1}(\mathbf{j})), (\mathbf{v}', \mathbf{u})\right) \geq \lambda/2.$$

On the other hand, if we had that

$$\Pr_{c \in \mathbb{F}}\left[\delta_{\pi,c,\mathbf{u},\mathbf{w}} < \frac{\lambda}{3} - \varepsilon\right] > \frac{1}{\varepsilon|\mathbb{F}|},$$

then by Corollary 5.10 we would have that $\Delta(f', \mathsf{PVAL}\left(2t, (\mathbf{j}, \pi^{-1}(\mathbf{j})), (\mathbf{v}', \mathbf{u})\right) < \lambda/3$.

The analysis for the case that $\mathbf{w} \neq z''[\pi(\mathbf{j})]$ is similar and the claim follows. $\square$

Consistent $\mathbf{u}, \mathbf{w}$. By Claim 5.12 we can restrict our attention to the case that $\mathbf{u} = z'[\pi^{-1}(\mathbf{j})]$ and $\mathbf{w} = z''[\pi(\mathbf{j})]$. When this is the case we have that:

$$z' \in \mathsf{PVAL}\left(2t, (\mathbf{j}, \pi^{-1}(\mathbf{j})), (\mathbf{v}', \mathbf{u})\right) = V + z'$$
$$\pi(z'') \in \mathsf{PVAL}\left(2t, (\mathbf{j}, \pi^{-1}(\mathbf{j})), (\mathbf{w}, \mathbf{v}'')\right) = V + \pi(z'').$$

Morover, z' is the closest vector to f' in the affine set $(V + z')$ (at distance at least δ'), and $\pi(z'')$ is the closest vector to $\pi(f'')$ in the affine set $(V + \pi(z''))$ (at distance at least δ'').

If Eq. (4) holds, then by Corollary 5.10 (using the fact that the minimal distance of V is λ) we conclude that there exist $y'_\pi \in (V + z')$ and $y''_\pi \in (V + \pi(z''))$ such that:

$$\left|\left\{i \in \{0,1\}^m : \left(f'(i) = y'_\pi(i)\right) \wedge \left(f''(\pi(i)) = y''_\pi(i)\right)\right\}\right| \geq (1 - \delta - \varepsilon) \cdot 2^m. \tag{5}$$

Since the minimal distance of the affine space $(V + z')$ is λ (see Claim 5.3), there is a unique vector in $(V + z')$ that is at distance smaller than $(\delta + \varepsilon)$ from f' (recall from above that we can assume $\delta \leq \lambda/3 - \varepsilon$), and this vector is z'. Thus, for the choices of π for which Eq. (5) holds we have $y'_\pi = z'$. Similarly, there is a unique vector in $(V + \pi(z''))$ that is at distance smaller than $(\delta + \varepsilon)$ from f'', and for the choices of π for which Eq. (5) holds we have $y''_\pi = \pi(z'')$.

Enumerating over the (many) choices of π for which Eq. (4) holds, we have:

$$\Pr_{\pi \leftarrow \Pi}\left[\left|\left\{i \in \{0,1\}^m : \left(f'(i) = z'(i)\right) \wedge \left(f''(\pi(i)) = z''(\pi(i))\right)\right\}\right| \geq (1 - \delta - \varepsilon) \cdot 2^m\right] > \gamma. \tag{6}$$

Let B' be the set of points on which f' and z' *disagree* (of relative size δ'), and let B'' be the set of points on which f'' and z'' *disagree* (of relative size δ''). Eq. (6) implies that w.h.p. over the choice of π, the size of the union $\left(B' \cup \pi^{-1}(B'')\right)$ is small (of relative size at most $\delta+\varepsilon$): significantly smaller than $|B'| + |B''|$. This can only happen if the size of the intersection $\left(B' \cap \pi^{-1}(B'')\right)$ is large. We show that this cannot be the case. For each $i \in \{0,1\}^m$, let X_i be the indicator random variable for the event that $i \in B''$ and $\pi^{-1}(i) \in B'$. Taking $X = \sum_i X_i$:

$$X = \left|B' \cap \pi^{-1}(B'')\right|.$$

By the 1-wise independence of Π (Fact 3.3, item 2), $\pi^{-1}(i)$ is uniformly distributed in $\{0,1\}^m$ for each fixed i, so $\Pr[\pi^{-1}(i) \in B'] = |B'|/2^m$. Thus, by linearity of expectation:

$$\mathbf{E}[X] = \frac{|B'||B''|}{2^m}.$$

By Markov's inequality for every $\gamma \in (0,1]$:

$$\Pr\left[X \geq \frac{|B'||B''|}{\gamma \cdot 2^m}\right] \leq \gamma.$$

And so we have that:

$$\begin{aligned}
&\Pr_{\pi \leftarrow \Pi}\left[\left|\left\{i \in \{0,1\}^m : \left(f'(i) \neq z'(i)\right) \vee \left(f''(\pi(i)) \neq z''(\pi(i))\right)\right\}\right| \leq \left(\delta' + \delta'' - \frac{\delta' \cdot \delta''}{\gamma}\right) \cdot 2^m\right] \\
&= \Pr_{\pi \leftarrow \Pi}\left[\left|B' \cap \pi^{-1}(B'')\right| \geq \frac{\delta' \cdot \delta''}{\gamma} \cdot 2^m\right] \\
&= \Pr_{\pi \leftarrow \Pi}\left[X \geq \frac{|B'||B''|}{\gamma \cdot 2^m}\right] \\
&\leq \gamma,
\end{aligned}$$

contradicting Eq. (6). □

5.4 Interactive Proof for PVAL Non-Triviality

Our PVAL IPP will also utilize the following (standard) interactive proof for checking whether a given PVAL instance (specified by the vector sequence $\mathbf{j}$) is non-trivial.

Lemma 5.13. *Let $t, m \in \mathbb{N}$ and $\mathbb{F}$ a finite field. There is a public-coin interactive proof for the language $\mathcal{L} = \left\{\mathbf{j} \in (\mathbb{F}^m)^t : |\mathsf{PVAL}(t, \mathbf{j}, \mathbf{0})| > 1\right\}$ with perfect completeness and the following parameters:*

- *Communication complexity:* $\mathsf{poly}(m, \log(t))$.
- *Round Complexity:* $\mathsf{poly}(m, \log(t))$.
- *Verifier running time:* $t \cdot \mathsf{poly}(m, \log(|\mathbb{F}|))$.
- *Prover running time:* $\mathsf{poly}(2^m, t)$.

Proof Sketch. We first argue that $\mathcal{L}$ can be decided by a logspace uniform NC circuit. Lemma 5.13 then follows by relying on the [GKR15] doubly-efficient interactive proof.

Since $\mathsf{PVAL}(t, \mathbf{j}, 0)$ is a linear space, it suffices to check whether its kernel is trivial. To do so the circuit explicitly generates the parity check matrix (of dimension $t \cdot 2^m$) of this space and computes its rank. Note that each entry of the matrix can be generated in depth $\log(m) + \mathsf{polylog}(|\mathbb{F}|)$ (and size $m \cdot \mathsf{polylog}(|\mathbb{F}|)$). The rank of a matrix can be computed in NC^2 (i.e., in depth $O(\log^2(t \cdot 2^m)) = m^2 \cdot \mathsf{polylog}(t)$ and size $\mathsf{poly}(2^m, t)$). □

6 Efficient IPP for PVAL

In this section we show our efficient IPP protocol for the PVAL problem.

Theorem 6.1 (IPP for PVAL). *Let $m, t \in \mathbb{N}$ and let $\mathbb{F}$ be a constructible finite field ensemble of characteristic 2 such that $|\mathbb{F}| = \Theta\left(2^m \cdot t^2 \cdot m^2\right)$. Let $\mathbf{j} = (\mathbf{j}_1, \dots, \mathbf{j}_t) \in (\mathbb{F}^m)^t$ and $\mathbf{v} = (v_1, \dots, v_t) \in \mathbb{F}^t$ such that $\Delta(\mathsf{PVAL}(t, \mathbf{j}, \mathbf{0})) \geq (t/2^m) \cdot \frac{1}{14m^2}$.*

Then, for every proximity parameter $\delta \geq 2^{-m}$ the set $\mathsf{PVAL}(t, \mathbf{j}, \mathbf{v})$ has a public-coin IPP with respect to proximity parameter δ, with perfect completeness and the following parameters:

- *Soundness Error:* $1/2$.
- *Query complexity:* $q = O\Big(\max\left(1/\delta, \frac{2^m}{t} \cdot \mathsf{poly}(m)\right)\Big)$.
- *Round complexity:* $\mathsf{poly}(m)$.
- *Communication Complexity:* $cc = t \cdot \mathsf{poly}(m)$.
- *Verifier Running Time:* $(t + q) \cdot \mathsf{poly}(m)$.
- *Prover Running Time:* $\mathsf{poly}(2^m)$.

Furthermore, if $\delta > (t/2^m) \cdot \frac{1}{\mathsf{poly}(m)}$ then, the entire verification procedure can be described succinctly as follows. At the end of the interaction either the verifier rejects or in time $\mathsf{poly}(m)$ it outputs a succinct description $\langle Q \rangle$ of a set $Q \subseteq [|2^m|]$ of size q and a succinct description $\langle \phi \rangle$ of a predicate $\phi : \{0,1\}^q \to \{0,1\}$ so that its decision predicate given an input function f is equal to $\phi(f|Q)$.

The rest of this section is devoted to the proof of Theorem 6.1.

The IPP protocol for PVAL is recursive. In each step we reduce the dimension m by 1 (which shrinks the input size by half), while simultaneously (roughly) doubling the distance of the problem from the relevant PVAL instance but also doubling the query complexity.

We denote the starting dimension by m_0 whereas the current dimension (within the recursion) is denoted by m (initially we set $m = m_0$. With that notation, the efficient IPP protocol for PVAL is presented in Fig. 4. Its completeness, soundness and complexity are analyzed in the subsequent subsections.

6.1 Completeness

We prove that completeness holds by induction on m. The base case (i.e., $m \leq \log(t)$) follows from Step 1 in the protocol (while relying on Proposition 3.8). We proceed to analyze the case $m > \log(t)$ (under the inductive hypothesis that the protocol is complete for dimension $m - 1$).

Let $\mathbf{j} = (\mathbf{j}_1, \dots, \mathbf{j}_t) \in (\mathbb{F}^m)^t$ and $\mathbf{v} = (v_1, \dots, v_t) \in \mathbb{F}^t$. Suppose that $f \in \mathsf{PVAL}(t, \mathbf{j}, \mathbf{v})$. As in the protocol, for every $i \in [t]$, decompose $\mathbf{j}_i$ into $\mathbf{j}_i = (\chi_i, \mathbf{j}'_i) \in \mathbb{F} \times \mathbb{F}^{m-1}$. Let $\mathbf{j}' \stackrel{\text{def}}{=} (\mathbf{j}'_1, \dots, \mathbf{j}'_t) \in (\mathbb{F}^{m-1})^t$.

We show that all the checks made by the verifier in the protocol pass (when interacting with the honest prover):

<u>**Efficient IPP for PVAL**</u>

Fixed parameters (unchanged in the recursion): PVAL arity parameter $t \in \mathbb{N}$ and a maximal dimension m_0. A finite field $\mathbb{F}$ of characteristic 2 such that $|\mathbb{F}| = \Theta\left(2^{m_0} \cdot t^2 \cdot m_0^2\right)$.

Parameters (modified in the recursion): dimension $m \in [\log(t), m_0]$, proximity parameter $\delta \in (2^{-m_0}, 1]$. A sequence of vectors $\mathbf{j} = (\mathbf{j}_1, \dots, \mathbf{j}_t) \in (\mathbb{F}^m)^t$ and field elements $\mathbf{v} = (v_1, \dots, v_t) \in \mathbb{F}^t$. A parameter λ such that $\frac{(t/2^m)}{14(m_0)^2} \leq \lambda \leq \frac{1}{(m_0)^2}$.

Invariants: $\lambda \geq \Delta(\mathsf{PVAL}(t, \mathbf{j}, \mathbf{0}))$.

Verifier Input: oracle access to $f : \{0,1\}^m \to \mathbb{F}$.

Prover Input: direct access to f.

Goal: verify that $\forall i \in [t]$, $\hat{f}(\mathbf{j}_i) = v_i$ (recall that $\hat{f} : \mathbb{F}^m \to \mathbb{F}$ is the multilinear extension of f).

The Protocol:

1. (Base Case:) If $m \leq \log(t)$, then the prover and verifier run the trivial MAP protocol of Proposition 3.8, with soundness error $\frac{m}{2m_0}$. The verifier accepts if the underlying MAP verifier accepts and otherwise it rejects.
2. Otherwise (i.e., if $m > \log(t)$), the protocol proceeds as follows.
3. For every $i \in [t]$, decompose $\mathbf{j}_i$ into $\mathbf{j}_i = (\chi_i, \mathbf{j}'_i) \in \mathbb{F} \times \mathbb{F}^{m-1}$.
4. For every $i \in [t]$ and $b \in \{0,1\}$, the prover computes and sends to the verifier $v_i^{(b)} = \hat{f}(b, \mathbf{j}'_i)$.
5. The verifier receives $\left(\tilde{v}_i^{(b)}\right)_{b \in \{0,1\}, i \in [t]}$ and checks that $v_i = (1-\chi_i) \cdot \tilde{v}_i^{(0)} + \chi_i \cdot \tilde{v}_i^{(1)}$, for every $i \in [t]$.
6. The verifier samples a random permutation $\pi \leftarrow \Pi_{m-1}$, where Π_{m-1} is the family of shift permutations over $\mathbb{F}^{m-1}$ defined in Section 3.2. The verifier sends π to the prover.
7. The verifier chooses t random points $\rho_1, \dots, \rho_t \in \mathbb{F}^{m-1}$. Let $\mathcal{C} : \mathbb{F} \to \mathbb{F}^{m-1}$ be a low degree curve passing through the set of $3t$ points $\cup_{i \in [t]} \{\mathbf{j}'_i, \pi^{-1}(\mathbf{j}'_i), \rho_i\}$. In more detail, fix a canonical set of $3t$ distinct field elements $\{\lambda_i^{(b)}\}_{b \in \{0,1,2\}, i \in [t]} \subset \mathbb{F}$. Let $\mathcal{C} : \mathbb{F} \to \mathbb{F}^{m-1}$ be the unique degree $3t-1$ curve such that $\mathcal{C}(\lambda_i^{(0)}) = \mathbf{j}'_i$, $\mathcal{C}(\lambda_i^{(1)}) = \pi^{-1}(\mathbf{j}'_i)$ and $\mathcal{C}(\lambda_i^{(2)}) = \rho_i$, for every $i \in [t]$ (such a curve can be found by interpolation). The verifier sends the values $\rho_1, \dots, \rho_t$ (which determine $\mathcal{C}$) to the prover.
8. The prover sends to the verifier the degree $O(m \cdot t)$ univariate polynomials $g^{(0)}$ and $g^{(1)}$, where $g^{(0)}(\cdot) = \hat{f}\Big(0, \big(\mathcal{C}(\cdot)\big)\Big)$ and $g^{(1)}(\cdot) = \hat{f}\Big(1, \pi\big(\mathcal{C}(\cdot)\big)\Big)$ (by sending a list of coefficients).
9. For every $b \in \{0,1\}$, the verifier receives $\tilde{g}^{(b)}$ from the prover and checks that for every $i \in [t]$, it holds that $\tilde{g}^{(b)}(\lambda_i^{(b)}) = \tilde{v}_i^{(b)}$. The prover and verifier also run the interactive proof of Lemma 5.13, with soundness error $\frac{1}{10m_0}$ to check that $\mathsf{PVAL}\Big(2t, \{\lambda_i^{(b)}\}_{b \in \{0,1\}, i \in [t]}, \{\tilde{g}^{(b')}(\lambda_i^{(b)})\}_{b \in \{0,1\}, i \in [t]}\Big) \neq \emptyset$ has positive dimension, for both $b' \in \{0,1\}$.
10. The verifier chooses at random $\xi_1, \dots, \xi_t \in \mathbb{F}$ and $c \in \mathbb{F}$.
11. The parties recurse on the implicit input function $f' : \{0,1\}^{m-1} \to \mathbb{F}$ defined as $f'(\mathbf{x}) = f(0, \mathbf{x}) + c \cdot f(1, \pi(\mathbf{x}))$ relative to the claim that $\forall i \in [t]$, $\hat{f}'(\mathcal{C}(\xi_i)) = \tilde{g}^{(0)}(\xi_i) + c \cdot \tilde{g}^{(1)}(\xi_i)$ (a PVAL instance of dimension $m-1$). Each of the verifier's queries to f' in the recursion are emulated by making 2 queries to f. The proximity parameter in the recursion is set to be $\delta' = \min\left(2\delta \cdot (1 - \frac{1}{m_0}), (t/2^m) \cdot \frac{1}{140m_0^2}\right)$ and relative to $\lambda' = \frac{(t/2^{m-1})}{14(m_0)^2}$.
12. If any of the verifier's checks failed then it rejects, otherwise it accepts.

Figure 4: Efficient IPP for PVAL

1. In Step 5, for every $i \in [t]$:

$$(1-\chi_i)\cdot v_i^{(0)} + \chi_i \cdot v_i^{(1)} = (1-\chi_i)\cdot \hat{f}(0,\mathbf{j}'_i) + \chi_i \cdot \hat{f}(1,\mathbf{j}'_i) = \hat{f}(\mathbf{j}_i) = v_i,$$

as required.

2. In Step 9, for every $i \in [t]$:

$$\begin{aligned} g^{(0)}\left(\lambda_i^{(0)}\right) &= \hat{f}\left(0, \mathcal{C}\left(\lambda_i^{(0)}\right)\right) = \hat{f}(0,\mathbf{j}'_i) = v_i^{(0)}, \\ g^{(1)}\left(\lambda_i^{(1)}\right) &= \hat{f}\left(1, \pi\left(\mathcal{C}\left(\lambda_i^{(1)}\right)\right)\right) = \hat{f}\left(1, \pi\left(\pi^{-1}\left(\mathbf{j}'_i\right)\right)\right) = \hat{f}(1,\mathbf{j}'_i) = v_i^{(1)}, \end{aligned}$$

as required.

3. For Step 11, for every $i \in [t]$, it holds that:

$$\hat{f}'(\mathcal{C}(\xi_i)) = \hat{f}\big(0, \mathcal{C}(\xi_i)\big) + c \cdot \hat{f}\big(1, \pi(\mathcal{C}(\xi_i))\big) = g^{(0)}(\xi_i) + c \cdot g^{(1)}(\xi_i),$$

where the first equality follows from Proposition 3.4.

Since all the verifier's checks pass, it accepts, and completeness follows.

6.2 Soundness

We prove, by induction on m, that the soundness error of the protocol is at most $\frac{m}{2m_0}$. Let $\mathbf{j} = (\mathbf{j}_1, \ldots, \mathbf{j}_t) \in (\mathbb{F}^m)^t$ and $\mathbf{v} = (v_1, \ldots, v_t) \in \mathbb{F}^t$. Let $f : \{0,1\}^m \to \mathbb{F}$ be such that f is δ-far from $\mathsf{PVAL}(t, \mathbf{j}, \mathbf{v})$. Fix a cheating prover strategy $\tilde{P}$. Assume without loss of generality that $\tilde{P}$ is deterministic (otherwise fix its best choice of randomness).

The base case (i.e., $m \le \log(t)$) is immediate from Step 1 (while relying on Proposition 3.8). We proceed to analyze the case that $m > \log(t)$ (under the inductive hypothesis that the protocol has soundness error at most $\frac{m-1}{2m_0}$ for dimension $m-1$).

We start by defining several important values that will be used in the analysis. Let $\left(\tilde{v}_i^{(b)}\right)_{b \in \{0,1\}, i \in [t]}$ be the (fixed) values sent by $\tilde{P}$ as its first message (i.e., in Step 4). We may assume that

$$(1-\chi_i)\cdot \tilde{v}_i^{(0)} + \chi_i \cdot \tilde{v}_i^{(1)} = v_i, \tag{7}$$

for every $i \in [t]$, since otherwise the verifier rejects in Step 5.

For $b \in \{0,1\}$, define $\mathbf{v}^{(b)} \stackrel{\text{def}}{=} \left(\tilde{v}_1^{(b)}, \ldots, \tilde{v}_t^{(b)}\right) \in \mathbb{F}^t$. For every $i \in [t]$, decompose $\mathbf{j}_i$ (i.e., the i-th component of $\mathbf{j}$) into $\mathbf{j}_i = (\chi_i, \mathbf{j}'_i) \in \mathbb{F} \times \mathbb{F}^{m-1}$. Denote by $\mathbf{j}' \stackrel{\text{def}}{=} (\mathbf{j}'_1, \ldots, \mathbf{j}'_t) \in (\mathbb{F}^{m-1})^t$. Lastly, for every $b \in \{0,1\}$, let $f^{(b)}(\cdot) \stackrel{\text{def}}{=} f(b, \cdot)$ and let $\delta^{(b)} \stackrel{\text{def}}{=} \Delta\left(f^{(b)}, \mathsf{PVAL}\left(t, \mathbf{j}', \mathbf{v}^{(b)}\right)\right)$.

We first argue that both $\mathsf{PVAL}\left(t, \mathbf{j}', \mathbf{v}^{(0)}\right)$ and $\mathsf{PVAL}\left(t, \mathbf{j}', \mathbf{v}^{(1)}\right)$ are non-empty. Indeed, if one of them were empty, the verifier would reject in Step 9 with probability at least $\frac{1}{10m_0}$. Thus, we may assume that both sets are non-empty and so $\delta^{(0)}, \delta^{(1)} \le 1$.

Proximity Parameter Doubles. Our goal will be to show that the input f' constructed for the recursive step (i.e., Step 11) has distance roughly 2δ from the corresponding PVAL instance (i.e., the distance doubles). This is done in two steps: (1) showing that f' has distance not much less than $\delta^{(0)} + \delta^{(1)}$, and (2) that this quantity is at least 2δ. We start with the second step.

Claim 6.2.

$$\delta^{(0)} + \delta^{(1)} \geq 2\delta.$$

Proof. For every $b \in \{0,1\}$, let $P^{(b)} : \{0,1\}^{m-1} \to \mathbb{F}$ such that $P^{(b)} \in \mathsf{PVAL}\left(t, \mathbf{j}', \mathbf{v}^{(b)}\right)$ and $\Delta\left(P^{(b)}, f^{(b)}\right) = \delta^{(b)}$ (such a $P^{(b)}$ exists since $\mathsf{PVAL}\left(t, \mathbf{j}', \mathbf{v}^{(b)}\right) \neq \emptyset$).

Consider the function $P : \{0,1\}^m \to \mathbb{F}$ defined as $P(b, \mathbf{x}) = P^{(b)}(\mathbf{x})$. Observe that $\Delta(P, f) = \frac{\delta^{(0)}+\delta^{(1)}}{2}$. On the other hand, for every $i \in [t]$, $b \in \{0,1\}$ and $(j_1, \ldots, j_m) \in \mathbb{F}^m$:

$$\begin{aligned} \hat{P}(j_1, \ldots, j_m) &= (1 - j_1) \cdot \hat{P}(0, j_2, \ldots, j_m) + j_1 \cdot \hat{P}(1, j_2, \ldots, j_m) \\ &= (1 - j_1) \cdot \hat{P}^{(0)}(j_2, \ldots, j_m) + j_1 \cdot \hat{P}^{(1)}(j_2, \ldots, j_m), \end{aligned} \tag{8}$$

where both equalities can be verified by observing that they hold for all $(j_1, \ldots, j_m) \in \{0,1\}^m$, and therefore hold also for all $(j_1, \ldots, j_m) \in \mathbb{F}^m$ (since two multilinear polynomials that agree on the Boolean inputs agree everywhere).

Thus, for every $i \in [t]$, it holds that

$$\begin{aligned} \hat{P}(\mathbf{j}_i) &= (1 - \chi_i) \cdot \hat{P}^{(0)}(\mathbf{j}'_i) + \chi_i \cdot \hat{P}^{(1)}(\mathbf{j}'_i) \\ &= (1 - \chi_i) \cdot \tilde{v}_i^{(0)} + \chi_i \cdot \tilde{v}_i^{(1)} \\ &= v_i, \end{aligned}$$

where the first equality is by Eq. (8), the second equality follows from the fact that $P^{(b)} \in \mathsf{PVAL}\left(t, \mathbf{j}', \mathbf{v}^{(b)}\right)$ and the third equality from Eq. (7).

We conclude that f is $\left(\frac{\delta^{(0)}+\delta^{(1)}}{2}\right)$-close to $\mathsf{PVAL}(t, \mathbf{j}, \mathbf{v})$ and so $\frac{\delta^{(0)}+\delta^{(1)}}{2} \geq \delta$. $\square$

Let $\pi \leftarrow \Pi_{m-1}$ be the permutation sampled randomly by the verifier in Step 6 and let $\rho_1, \ldots, \rho_t \in \mathbb{F}^{m-1}$ be the random points sampled in Step 7. As in the protocol, let $\mathcal{C} : \mathbb{F} \to \mathbb{F}^{m-1}$ be the unique degree $3t-1$ curve such that $\mathcal{C}(\lambda_i^{(0)}) = \mathbf{j}'_i$, $\mathcal{C}(\lambda_i^{(1)}) = \pi^{-1}(\mathbf{j}'_i)$ and $\mathcal{C}(\lambda_i^{(2)}) = \rho_i$, for every $i \in [t]$. Let $\tilde{g}^{(0)}$ and $\tilde{g}^{(1)}$ be the degree $O(m \cdot t)$ univariate polynomials sent by $\tilde{P}$ in Step 8. Note that $\tilde{g}^{(0)}$ and $\tilde{g}^{(1)}$ are random variables that depend on π and $\rho_1, \ldots, \rho_t$.

We may assume without loss of generality that for every choice of π and $\rho_1, \ldots, \rho_t$ made by the verifier it holds that

$$\forall i \in [t],\ b \in \{0,1\} : \quad \tilde{g}_i^{(b)}(\lambda_i^{(b)}) = \tilde{v}_i^{(b)}, \tag{9}$$

since otherwise the verifier immediately rejects in Step 9. Thus, we can modify the prover $\tilde{P}$ to *always* send polynomials satisfying Eq. (9) without decreasing $\tilde{P}$'s success probability.

For every $c \in \mathbb{F}$, define the function $f'_{\pi,c} : \{0,1\}^{m-1} \to \mathbb{F}$ as $f'_{\pi,c}(\mathbf{x}) = f(0, \mathbf{x}) + c \cdot f(1, \pi(\mathbf{x}))$.

Recall that $\delta^{(b)} = \Delta\left(f^{(b)}, \mathsf{PVAL}\left(t, \mathbf{j}', \mathbf{v}^{(b)}\right)\right)$ and let $\delta_{\mathsf{avg}} = (\delta^{(0)} + \delta^{(1)})/2$. We now invoke Lemma 5.11 on $f^{(0)}$ and $f^{(1)}$ wrt $\gamma = \frac{1}{10m_0}$ and $\varepsilon \stackrel{\text{def}}{=} \delta_{\mathsf{avg}}/m_0$. We obtain that:

$$\Pr_{\pi \leftarrow \Pi}\left[\exists (\mathbf{u}, \mathbf{w}) \in S_\pi \text{ s.t. } \Pr_{c \in \mathbb{F}}\left[\delta_{\pi,c,\mathbf{u},\mathbf{w}} < \delta^*\right] > \frac{1}{\varepsilon|\mathbb{F}|}\right] < \gamma \tag{10}$$

where $\delta_{\pi,c,\mathbf{u},\mathbf{w}}$ is defined as the distance of $f'_{\pi,c}$ from

$$\mathsf{PVAL}\Big(2t, \big(\mathbf{j}', \pi^{-1}(\mathbf{j}'\big), \big((\mathbf{v}^{(0)}, \mathbf{u}) + c \cdot (\mathbf{w}, \mathbf{v}^{(1)})\big)\Big),$$

and

$$\delta^* \stackrel{\text{def}}{=} \max\left(\delta_{\mathsf{avg}}, \min\left(\delta^{(0)} + \delta^{(1)} - \delta^{(0)} \cdot \delta^{(1)}/\gamma - \varepsilon, \lambda/3 - 3\varepsilon\right)\right), \tag{11}$$

and $S_\pi \subseteq \mathbb{F}^{2t}$ is the set of pairs of vectors $(\mathbf{u}, \mathbf{w})$ s.t. both sets $\mathsf{PVAL}\left(2t, (\mathbf{j}', \pi^{-1}(\mathbf{j}')), (\mathbf{v}^{(0)}, \mathbf{u})\right)$ and $\mathsf{PVAL}\left(2t, (\mathbf{j}', \pi^{-1}(\mathbf{j}')), (\mathbf{w}, \mathbf{v}^{(1)})\right)$ are non-empty.

We proceed to show that δ^* is lower bounded by roughly 2δ.

Proposition 6.3. $\delta^* \geq \min\left(2\delta \cdot (1 - \frac{1}{m_0}), \lambda/10\right)$.

Proof. Suppose that $\delta_{\mathsf{avg}} > \frac{1}{10(m_0)^2}$. By Eq. (11) it holds that $\delta^* \geq \delta_{\mathsf{avg}}$, and by our setting of λ it holds that $\frac{1}{10(m_0)^2} \geq \frac{\lambda}{10}$ and so in this case the claim holds trivially. Thus, we may assume that

$$\delta_{\mathsf{avg}} \leq \frac{1}{10(m_0)^2} \tag{12}$$

Now, suppose that $\delta^* \geq \delta^{(0)} + \delta^{(1)} - \delta^{(0)} \cdot \delta^{(1)}/\gamma - \varepsilon$. In this case the proposition follows from the following claim:

Claim 6.4. $\delta^{(0)} + \delta^{(1)} - \delta^{(0)} \cdot \delta^{(1)}/\gamma - \varepsilon \geq 2\delta \cdot (1 - \frac{1}{m_0})$.

Proof. By the AM-GM inequality $\delta^{(0)} \cdot \delta^{(1)} \leq \delta_{\mathsf{avg}}^2$ and so:

$$\delta^{(0)} + \delta^{(1)} - \delta^{(0)} \cdot \delta^{(1)}/\gamma \geq 2\delta_{\mathsf{avg}} - \delta_{\mathsf{avg}}^2/\gamma.$$

Suppose that $\delta_{\mathsf{avg}} \leq \frac{1}{10(m_0)^2}$. Then, using our setting of ε and γ, it holds that:

$$2\delta_{\mathsf{avg}} - \delta_{\mathsf{avg}}^2/\gamma - \varepsilon = 2\delta_{\mathsf{avg}} - 10m_0 \cdot \delta_{\mathsf{avg}}^2 - \frac{\delta_{\mathsf{avg}}}{m_0} = 2\delta_{\mathsf{avg}} \cdot \left(1 - 5m_0 \cdot \delta_{\mathsf{avg}} - \frac{1}{2m_0}\right) \geq 2\delta \cdot \left(1 - \frac{1}{m_0}\right)$$

where the inequality is based on Claim 6.2 and our upper bound on δ_{avg} (see Eq. (12)).

□

Given Claim 6.4, we may assume that $\delta^* < \delta^{(0)} + \delta^{(1)} - \delta^{(0)} \cdot \delta^{(1)}/\gamma - \varepsilon$. Eq. (11) now implies that $\delta^* \geq \lambda/3 - 3\varepsilon$.

Consider the case that ε is large. Concretely, suppose $\varepsilon > \lambda/30$ then, by definition of ε we have that $\delta_{\mathsf{avg}}/m_0 > \lambda/30$. Thus, $\delta_{\mathsf{avg}} > \frac{m_0}{30}\lambda$ and since $\delta^* \geq \delta_{\mathsf{avg}}$, for sufficiently large m_0 the claim follows.

Thus, we may assume that $\varepsilon \leq \frac{\lambda}{30}$. In this case:

$$\delta^* \geq \lambda/3 - 3\varepsilon \geq \lambda/3 - 3 \cdot \frac{\lambda}{30} \geq \lambda/10.$$

This concludes the proof of Proposition 6.3. □

We say that $\pi \in \Pi_{m-1}$ is *good* if the the event specified in Eq. (10) does not hold. Thus, π is good with all but γ probability. Fix such a good π. Then, for every $(\mathbf{u}, \mathbf{w}) \in S_\pi$ it holds that

$$\Pr_{c \in \mathbb{F}} [\delta_{\pi,c,\mathbf{u},\mathbf{w}} < \delta^*] \leq \frac{1}{\varepsilon |\mathbb{F}|} = \frac{m_0}{\delta_{\mathsf{avg}} \cdot |\mathbb{F}|} \leq \frac{m_0}{\delta \cdot (10 \cdot 2^{m_0} (m_0)^2)} \leq \frac{1}{10 m_0} \tag{13}$$

where the penultimate inequality relies on Claim 6.2, the fact that $|\mathbb{F}| \geq 10 \cdot 2^{m_0} \cdot (m_0)^2$ and that $\delta \geq 2^{-m_0}$.

Let $\mathbf{u} = \left(\tilde{g}^{(0)}(\lambda_1^{(1)}), \ldots, \tilde{g}^{(0)}(\lambda_t^{(1)})\right)$ and $\mathbf{w} = \left(\tilde{g}^{(1)}(\lambda_1^{(0)}), \ldots, \tilde{g}^{(1)}(\lambda_t^{(0)})\right)$. Suppose first that $(\mathbf{u}, \mathbf{w}) \notin S_\pi$. Then $\mathsf{PVAL}\left(2t, \left\{\lambda^{(b)}\right\}_{b \in \{0,1\}, i \in [t]}, \left\{\tilde{g}^{(b')}(\lambda_i^{(b)})\right\}_{b \in \{0,1\}, i \in [t]}\right) = \emptyset$, for either $b' = 0$ or $b' = 1$. In Step 9 the verifier and prover run an interactive proof to check that this is not the case and so the verifier rejects in this case with probability at least $1 - \frac{1}{10 m_0}$. Thus, we may assume that $(\mathbf{u}, \mathbf{w}) \in S_\pi$.

Thus, by Eq. (13) for all but $\frac{1}{10 m_0}$ fraction of $c \in \mathbb{F}$, it holds that $f'_{\pi,c}$ is at distance at least δ^* from $\mathsf{PVAL}\big(2t, (\mathbf{j}', \pi^{-1}(\mathbf{j}')), (\omega_k)_{k \in [2t]}\big)$, where $\omega_{b \cdot t + i} = \tilde{g}_i^{(0)}(\lambda_i^{(b)}) + c \cdot \tilde{g}_i^{(1)}(\lambda_i^{(b)})$, for $b \in \{0, 1\}$ and $i \in [t]$. We say that such a c is *good*.

Fix such a good c and let

$$\delta' \stackrel{\text{def}}{=} \min\left(2\delta \cdot (1 - \frac{1}{m_0}), (t/2^m) \cdot \frac{1}{140 m_0^2}\right) \tag{14}$$

and observe that by Proposition 6.3 (and our lower bound on λ) it holds that $\delta' \leq \delta^*$.

Claim 6.5. *With all but $\frac{1}{10 m_0}$ probability over the choice of $\xi_1, \ldots, \xi_t \in \mathbb{F}$ it holds that $f'_{\pi,c}$ is at distance at least δ' from* $\mathsf{PVAL}\left(t, (\mathcal{C}(\xi_i))_{i \in [t]}, \left(\tilde{g}_i^{(0)}(\xi_i) + c \cdot \tilde{g}^{(1)}(\xi_i)\right)_{i \in [t]}\right)$.

Proof. Let $h : \{0,1\}^{m-1} \to \mathbb{F}$ be a function at relative distance $\leq \delta' \leq \delta^*$ from $f'_{\pi,c}$. By our assumption on c it holds that $h \notin \mathsf{PVAL}\left(2t, (\mathbf{j}', \pi^{-1}(\mathbf{j}')), (\omega_k)_{k \in [2t]}\right)$, where $\omega_{b \cdot t + i} = \tilde{g}_i^{(0)}(\lambda_i^{(b)}) + c \cdot \tilde{g}_i^{(1)}(\lambda_i^{(b)})$. In particular, this means that there exists some $b \in \{0, 1\}$ and $i \in [t]$ such that:

$$\hat{h}\left(\mathcal{C}(\lambda_i^{(b)})\right) \neq \tilde{g}_i^{(0)}(\lambda_i^{(b)}) + c \cdot \tilde{g}_i^{(1)}(\lambda_i^{(b)}).$$

The polynomials $\hat{h} \circ \mathcal{C}$ and $\tilde{g}^{(0)}(\cdot) + c \cdot \tilde{g}^{(1)}(\cdot)$ are therefore *distinct* polynomials of degree $O(m \cdot t)$. Thus, the probability over a random $\xi \in \mathbb{F}^{m-1}$ that $\hat{h}(\mathcal{C}(\xi)) = \tilde{g}^{(0)}(\xi) + c \cdot \tilde{g}^{(1)}(\xi)$ is at most $O(m \cdot t/|\mathbb{F}|) \leq 1/2$. Therefore, the probability that $h \in \mathsf{PVAL}\left(t, (\mathcal{C}(\xi_i))_{i \in [t]}, \left(\tilde{g}_i^{(0)}(\xi_i) + c \cdot \tilde{g}^{(1)}(\xi_i)\right)_{i \in [t]}\right)$ is at most 2^{-t}.

The number of functions $h : \{0,1\}^{m-1} \to |\mathbb{F}|$ that are δ'-close to $f'_{\pi,c}$ is upper bounded by $(2^{m-1} \cdot |\mathbb{F}|)^{\delta' \cdot 2^{m-1}} \leq 2^{\delta' \cdot 2^m \cdot m \cdot \log(|\mathbb{F}|)}$. Therefore, by a union bound, we have that $f'_{\pi,c}$ is δ'-far from $\mathsf{PVAL}\left(t, (\mathcal{C}(\xi_i))_{i \in [t]}, \left(\tilde{g}_i^{(0)}(\xi_i) + c \cdot \tilde{g}^{(1)}(\xi_i)\right)_{i \in [t]}\right)$, with all but $2^{\delta' \cdot 2^m \cdot m \cdot \log(|\mathbb{F}|) - t}$ probability. Since $\delta' \leq (t/2^m) \cdot \frac{1}{140 m_0^2}$, we have that this probability is upper bounded by $\frac{1}{10 m_0}$. $\square$

We say that $(\xi_1, \ldots, \xi_t)$ are *good* if the event stated in Claim 6.5 holds. In such a case the protocol is run recursively on input $f'_{\pi,c}$ that is at least δ'-far from the relevant PVAL instance. At this point we would like to argue that by the inductive hypothesis, the verifier rejects with high probability. However, to do so, we still need to argue that the recursive invocation satisfies the invariants stated in Fig. 4.

Maintaining Invariants. We start with the easier invariant:

Claim 6.6. $\delta' \geq 2^{-m_0}$.

Proof. By Eq. (14), $\delta' \stackrel{\text{def}}{=} \min\left(2\delta \cdot (1 - \frac{1}{m_0}), (t/2^m) \cdot \frac{1}{140 m_0^2}\right)$.

Suppose that $\delta' = 2\delta \cdot (1 - \frac{1}{m_0})$. Since we know that $\delta \geq 2^{-m_0}$, the claim follows from the fact that $m_0 \geq 2$.

Otherwise, $\delta' = (t/2^m) \cdot \frac{1}{140 m_0^2} \geq \frac{140 \cdot t \cdot (m_0)^2}{2^{m_0}} \geq 2^{-m_0}$. ☐

We proceed to the second invariant.

Claim 6.7. *With all but $\frac{1}{10 m_0}$ probability over the choice of $\rho_1, \ldots, \rho_t$ and $\xi_1, \ldots, \xi_t$, it holds that $\Delta(\mathsf{PVAL}(t, \mathbf{j}', \mathbf{0}) \geq (t/2^{m-1}) \cdot \frac{1}{14(m_0)^2}$.*

Proof. Observe that $\xi_1, \ldots, \xi_t \notin \{\lambda_i^{(b)}\}_{b \in \{0,1\}, i \in [t]}$ with probability $1 - \frac{2t^2}{|\mathbb{F}|} \geq 1 - \frac{1}{100 m_0}$.

Since the curve $\mathcal{C}$ passes through t random points (i.e., $\rho_1, \ldots, \rho_t$), the distribution over points through which the curve $\mathcal{C}$ passes is t-wise independent, other than at the fixed points $\{\lambda_i^{(b)}\}_{b \in \{0,1\}, i \in [t]}$. Putting the above two facts together, we obtain that with all but $\frac{1}{200 m_0}$ probability, the set of points $\mathbf{j}' = \left(\mathcal{C}(\xi_1), \ldots, \mathcal{C}(\xi_t)\right)$ is uniformly distributed in $(\mathbb{F}^{m-1})^t$.

Observe that $t \geq \frac{t}{14(m_0)^2} \cdot \log(2^m \cdot |\mathbb{F}|) + \log(100 m_0)$

$$t = \Omega\left(\frac{2^{m_0}}{m_0}\right) \geq \frac{2^{m_0}}{(m_0)^2} \cdot \log(2^{m_0} \cdot |\mathbb{F}|) + \log(100 m_0),$$

and so, by Proposition 5.4,

$$\Pr\left[\Delta(\mathsf{PVAL}(t, \mathbf{j}', \mathbf{0}) \leq \frac{1}{(m_0)^2}\right] < \frac{1}{100 m_0},$$

and the claim follows. ☐

Thus, the invariants for the recursive step are satisfied and so the verifier accepts in the recursion with probability at most $\frac{m-1}{2m_0}$. Overall, by accounting for all of the bad events in the analysis above, we get that the verifier accepts with probability at most:

$$\frac{m-1}{2m_0} + 5 \cdot \frac{1}{10 m_0} \leq \frac{m}{2m_0}$$

as required.

6.3 Complexity

Communication Complexity. We first analyze the complexity of a single iteration (i.e., excluding the recursion). The verifier only sends to the prover a specification of the permutations π (which take m bits), the values $\rho_1, \ldots, \rho_t$, $\xi_1, \ldots, \xi_t \in \mathbb{F}$ and $c \in \mathbb{F}$. Overall the verifier-to-prover communication is $m + (2t+1) \cdot \log_2(|\mathbb{F}|)$. The prover in turn sends $(v_i^{(b)})_{i \in [t], b \in \{0,1\}}$ and the polynomials $\tilde{g}^{(0)}$ and $\tilde{g}^{(1)}$ (of degree $O(t \cdot m)$). Thus, the total prover to verifier communication is $O(t \cdot m \cdot \log(|\mathbb{F}|))$.

Thus, the overall communication complexity is given by $cc(m)$ where $cc(m) = O(t \cdot m \cdot \log(|\mathbb{F}|)) + cc(m-1)$ if $m > \log(t)$ and $cc(m) = 2^m \cdot \log(|\mathbb{F}|)$ otherwise. Overall we have $cc(m) \leq O(m^2 \cdot t \cdot \log(|\mathbb{F}|))$.

Query Complexity. Denote the query complexity by $q(m, \delta)$. Note that if $m \leq \log(t)$ then $q(m, \delta) = O(1/\delta)$ and otherwise $q(m, \delta) = 2 \cdot q(m-1, \delta') = 2q\Big(m-1, \min\big(2\delta \cdot (1 - \frac{1}{m_0}), (t/2^m) \cdot \frac{1}{140 m_0^2}\big)\Big)$. The stated query complexity follows from the following claim.

Claim 6.8. *There exists a fixed constant c such that for every m and δ it holds that $q(m, \delta) \leq c \cdot (1 - \frac{1}{m_0})^{-m} \cdot \max\left(\frac{1}{\delta}, \frac{280 \cdot 2^m \cdot m_0^2}{t}\right)$.*

Proof. We prove by induction on m. The base case $m = \log(t)$ is immediate. Suppose that the claim holds for $m-1$. Then:

$$q(m, \delta) = 2q\Big(m-1, \min\big(2\delta \cdot (1 - \frac{1}{m_0}), (t/2^m) \cdot \frac{1}{140 m_0^2}\big)\Big)$$

Suppose first that $2\delta \cdot (1 - \frac{1}{m_0}) < (t/2^m) \cdot \frac{1}{140 m_0^2}$. Then,

$$\begin{aligned}
q(m, \delta) &= 2q\big(m-1, 2\delta \cdot (1 - \frac{1}{m_0})\big) \\
&\leq 2c \cdot (1 - \frac{1}{m_0})^{-(m-1)} \cdot \max\Big(\frac{1}{2\delta \cdot (1 - \frac{1}{m_0})}, \frac{280 \cdot 2^{m-1} \cdot m_0^2}{t}\Big) \\
&= c \cdot (1 - \frac{1}{m_0})^{-m} \cdot \max\Big(\frac{1}{\delta}, \frac{280 \cdot 2^m \cdot m_0^2}{t}\Big)
\end{aligned}$$

as required. Otherwise, $2\delta \cdot (1 - \frac{1}{m_0}) \geq (t/2^m) \cdot \frac{1}{140 m_0^2}$ and we have that:

$$\begin{aligned}
q(m, \delta) &= 2q\Big(m-1, (t/2^m) \cdot \frac{1}{140 m_0^2}\Big) \\
&\leq 2c \cdot (1 - \frac{1}{m_0})^{-(m-1)} \cdot \max\Big(\frac{140 \cdot 2^m \cdot m_0^2}{t}, \frac{280 \cdot 2^{m-1} \cdot m_0^2}{t}\Big) \\
&\leq c \cdot (1 - \frac{1}{m_0})^{-m} \cdot \frac{280 \cdot 2^m \cdot m_0^2}{t} \\
&\leq c \cdot (1 - \frac{1}{m_0})^{-m} \cdot \max\Big(\frac{1}{\delta}, \frac{280 \cdot 2^m \cdot m_0^2}{t}\Big).
\end{aligned}$$

□

Prover Runtime. In every iteration, the prover only does elementary manipulations of the truth table of f (and never needs to fully materialize the truth table of $\hat{f}$). It also runs the prover of Lemma 5.13. Overall its running time is $\mathsf{poly}(2^m, m_0, \log(|\mathbb{F}|), t) = \mathsf{poly}(2^{m_0})$.

Verifier Runtime and Succinct Description. The queries made by the verifier can be succinctly specified by the permutation π used through the recursion as well as the random locations that it queries in the base case. The total number of bits needed to describe the permutations is at most m_0^2. The number of bits needed in the base case is equal to the total number of queries divided by $2^{m_0}/t$ (since in each of the $m_0 - \log(t)$ iterations the number of queries doubled) and multiplied by $\log(2^m) = m$ (to specify the location). By the above analysis this quantity is therefore upper bounded by $O\Big(\frac{t \cdot m_0}{2^{m_0}} \cdot \max\left(1/\delta, \frac{2^{m_0}}{t} \cdot \mathsf{poly}(m)\right)\Big) = O(\mathsf{poly}(m) + \frac{t \cdot m_0}{2^{m_0}}) \cdot (1/\delta))$. If $\delta > (t/2^{m_0}) \cdot \frac{1}{\mathsf{poly}(m)}$ this string has $\mathsf{poly}(m)$ length as required.

Given the set of base points we can generate the list of q queries by repeatedly applying the two permutations that we have for each level of the recursion. Since the permutations can be computed in $\mathsf{poly}(m)$ time (see Section 3.2), we obtain that a logspace Turing machine can generate a $\mathsf{poly}(m)$ depth circuit that outputs the entire set of q query locations.

As for the succinct description of the verification predicate, observe that all of the verifier's checks that do not involve the input can be implemented in time $\mathsf{poly}(t, m_0, \log(|\mathbb{F}|)) = \mathsf{poly}(t)$. The testing of the actual input only happens in the case in which the prover sends over the alleged actual input $\tilde{f}_\perp$ (which at the end of the recursion has length $t \cdot \log(|\mathbb{F}|)$). This string $\tilde{f}_\perp$ is part of the description of the verification predicate, together also with all of the c values generated in the recursion. Using these values it is possible to construct a $q \cdot \mathsf{poly}(m_0, \log(|\mathbb{F}|))$ size depth $\mathsf{poly}(m_0)$ circuit that given the query answers checks their consistency with $\tilde{f}_\perp$.

7 Proving Theorem 3 and Theorem 4

Theorem 3 follows immediately by combining Theorems 5.6 and 6.1, while setting $t = \delta n \cdot \mathsf{polylog}(n)$.

In order to prove Theorem 4 we utilize an idea from the work of Reingold *et al.* [RRR18] who used known IPP protocols to achieve batch verification for UP languages. We restate a more general form of their reduction below. In the interest of directness, we avoid defining or using Interactive Witness Verification protocols, as they did. Instead, we use IPPs for pair languages:

Theorem 7.1 (From IPPs to UP batch verification (generalization of Theorem 3.3 in [RRR18])). *Suppose that for every query parameter $q = q(n) \in \{1, \ldots, m\}$, and for every pair languages $\mathcal{L}$ that can be computed by log-space uniform polynomial-size circuits with fan-in 2 and depth $D = D(n)$, there exists an interactive proof of proximity where the verifier is public-coin and, on input (x, y), at the end of the interaction either the verifier rejects, or it outputs a succinct description $\langle Q \rangle$ of a set $Q \subseteq [|y|]$ of size q and succinct description $\langle \phi \rangle$ of a predicate $\phi : \{0,1\}^q \to \{0,1\}$, and for every input pair (x, y):*

- **Completeness:** *If $(x, y) \in \mathcal{L}$ then*

$$\Pr\left[\mathcal{V} \text{ does not reject and } \phi(y|_Q) = 1\right] = 1.$$

- **Soundness:** *If $\mathcal{L}(x) = \emptyset$ (there is no y' s.t. $(x, y') \in \mathcal{L}$), then for every prover $\mathcal{P}^*$:*

$$\Pr\left[\mathcal{V} \text{ does not reject and } \phi(y|_Q) = 1\right] \leq 1/2.$$

Let $\mathsf{cc} = \mathsf{cc}(q, D, n, m)$ *be the communication complexity,* $r = r(q, D, n, m)$ *the number of rounds,* $\mathcal{V}\mathsf{time}(q, D, n, m)$ *the verifier's runtime, and assume that the honest prover runs in polynomial time.*

Then, for every UP *language* $\mathcal{L}$ *with witness length* $m = m(n)$*, whose witness relation can be computed in* NC*, there exists a public-coin interactive proof (with perfect completeness) for verifying that* k *instances* $x_1, \ldots, x_k$*, each of length* n*, are all in* $\mathcal{L}$*. Taking* $D' = \mathsf{polylog}(n, k)$ *and* $m' = k \cdot m$*, the complexity of the protocol is as follows:*

- *Communication complexity:* $O\left(m + \sum_{i=1}^{\log k} \mathsf{cc}(\frac{k}{2^i}, D', \frac{n'}{2^i}, \frac{m'}{2^i})\right)$.
- *Number of rounds:* $O\left(\sum_{i=1}^{\log k} r(\frac{k}{2^i}, D', \frac{n'}{2^i}, \frac{m'}{2^i})\right)$.
- *Verifier runtime:* $O\left(m \cdot \log(n) + \sum_{i=1}^{\log k} \mathcal{V}\mathsf{time}(\frac{k}{2^i}, D', \frac{n'}{2^i}, \frac{m'}{2^i})\right)$.
- *The honest prover, given the* k *unique witnesses, runs in polynomial time.*

Theorem 4 now follows from Theorem 7.1 by utilizing the efficient IPPs for NC given in Theorem 3.

Acknowledgments

We thank Oded Goldreich for many insightful conversations and helpful suggestions on the presentation. We thank Omer Reingold for illuminating discussions on these topics. We thank the TCC revewiers for their careful reading of the manuscript and useful comments. We are grateful to Rohan Goyal, Matthew Hong and Yael Tauman Kalai [GHK25], who discovered an error in our original manuscript and communicated it to us (see Section 1.4 and Remark 2.1).

References

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

[AR24] Noga Amit and Guy N. Rothblum. Constant-round arguments for batch-verification and bounded-space computations from one-way functions. In Leonid Reyzin and Douglas Stebila, editors, *Advances in Cryptology - CRYPTO 2024 - 44th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2024, Proceedings, Part X*, volume 14929 of *Lecture Notes in Computer Science*, pages 3–37. Springer, 2024.

[AS92] Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs; A new characterization of NP. In *33rd Annual Symposium on Foundations of Computer Science, Pittsburgh, Pennsylvania, USA, 24-27 October 1992*, pages 2–13, 1992.

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

[BFLS91] László Babai, Lance Fortnow, Leonid A. Levin, and Mario Szegedy. Checking computations in polylogarithmic time. In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing, May 5-8, 1991, New Orleans, Louisiana, USA*, pages 21–31, 1991.

[BGG+88] Michael Ben-Or, Oded Goldreich, Shafi Goldwasser, Johan Håstad, Joe Kilian, Silvio Micali, and Phillip Rogaway. Everything provable is provable in zero-knowledge. In *Advances in Cryptology - CRYPTO '88, 8th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1988, Proceedings*, pages 37–56, 1988.

[BGHK25] Bonnie Berger, Rohan Goyal, Matthew M. Hong, and Yael Tauman Kalai. Efficiently batching unambiguous interactive proofs. In *66th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2025, Sydney, Australia, December 14-17, 2025*, pages 818–863. IEEE, 2025.

[BGKW88] Michael Ben-Or, Shafi Goldwasser, Joe Kilian, and Avi Wigderson. Multi-prover interactive proofs: How to remove intractability assumptions. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA*, pages 113–131, 1988.

[BGR98] Mihir Bellare, Juan A. Garay, and Tal Rabin. Fast batch verification for modular exponentiation and digital signatures. In Kaisa Nyberg, editor, *Advances in Cryptology - EUROCRYPT '98, International Conference on the Theory and Application of Cryptographic Techniques, Espoo, Finland, May 31 - June 4, 1998, Proceeding*, volume 1403 of *Lecture Notes in Computer Science*, pages 236–250. Springer, 1998.

[BHK17] Zvika Brakerski, Justin Holmgren, and Yael Tauman Kalai. Non-interactive delegation and batch NP verification from standard computational assumptions. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 474–482, 2017.

[BKP+24] Nir Bitansky, Chethan Kamath, Omer Paneth, Ron D. Rothblum, and Prashant Nalini Vasudevan. Batch proofs are statistically hiding. In Bojan Mohar, Igor Shinkar, and Ryan O'Donnell, editors, *Proceedings of the 56th Annual ACM Symposium on Theory of Computing, STOC 2024, Vancouver, BC, Canada, June 24-28, 2024*, pages 435–443. ACM, 2024.

[BKS18] Eli Ben-Sasson, Swastik Kopparty, and Shubhangi Saraf. Worst-case to average case reductions for the distance to a code. In *33rd Computational Complexity Conference, CCC 2018, June 22-24, 2018, San Diego, CA, USA*, pages 24:1–24:23, 2018.

[BRV18] Itay Berman, Ron D. Rothblum, and Vinod Vaikuntanathan. Zero-knowledge proofs of proximity. In *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*, pages 19:1–19:20, 2018.

[CG18] Alessandro Chiesa and Tom Gur. Proofs of proximity for distribution testing. In *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*, pages 53:1–53:14, 2018.

[CHP12] Jan Camenisch, Susan Hohenberger, and Michael Østergaard Pedersen. Batch verification of short signatures. *J. Cryptology*, 25(4):723–747, 2012.

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

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

[FGL14] Eldar Fischer, Yonatan Goldhirsh, and Oded Lachish. Partial tests, universal tests and decomposability. In *ITCS*, pages 483–500, 2014.

[FRS94] Lance Fortnow, John Rompel, and Michael Sipser. On the power of multi-prover interactive protocols. *Theor. Comput. Sci.*, 134(2):545–557, 1994.

[GGR98] Oded Goldreich, Shafi Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *Journal of the ACM (JACM)*, 45(4):653–750, 1998.

[GGR15] Oded Goldreich, Tom Gur, and Ron D. Rothblum. Proofs of proximity for context-free languages and read-once branching programs - (extended abstract). In *Automata, Languages, and Programming - 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part I*, pages 666–677, 2015.

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

[GHK25] Rohan Goyal, Matthew Hong, and Yael Tauman Kalai. Personal communication, 2025.

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

[GLR18] Tom Gur, Yang P. Liu, and Ron D. Rothblum. An exponential separation between MA and AM proofs of proximity. In Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella, editors, *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, volume 107 of *LIPIcs*, pages 73:1–73:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM J. Comput.*, 18(1):186–208, 1989.

[GMW91] Oded Goldreich, Silvio Micali, and Avi Wigderson. Proofs that yield nothing but their validity for all languages in NP have zero-knowledge proof systems. *J. ACM*, 38(3):691–729, 1991.

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

[Gol26] Oded Goldreich. A digest of the work of Rothblum, Vadhan, and Wigderson (2013). Available at https://www.wisdom.weizmann.ac.il/~oded/p_rvw.html, 2026.

[GR13] Tom Gur and Ron D. Rothblum. Non-interactive proofs of proximity. *Electronic Colloquium on Computational Complexity (ECCC)*, 20:78, 2013.

[GR17] Tom Gur and Ron D. Rothblum. A hierarchy theorem for interactive proofs of proximity. In *8th Innovations in Theoretical Computer Science Conference, ITCS 2017, January 9-11, 2017, Berkeley, CA, USA*, pages 39:1–39:43, 2017.

[GR22] Guy Goldberg and Guy N. Rothblum. Sample-based proofs of proximity. In Mark Braverman, editor, *13th Innovations in Theoretical Computer Science Conference, ITCS 2022, January 31 - February 3, 2022, Berkeley, CA, USA*, volume 215 of *LIPIcs*, pages 77:1–77:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[GRSY20] Shafi Goldwasser, Guy N. Rothblum, Jonathan Shafer, and Amir Yehudayoff. Interactive proofs for verifying machine learning. *Electronic Colloquium on Computational Complexity (ECCC)*, 27:58, 2020.

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

[HR24] Tal Herman and Guy N. Rothblum. Interactive proofs for general distribution properties. In *65th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2024, Chicago, IL, USA, October 27-30, 2024*, pages 528–538. IEEE, 2024.

[IKO07] Yuval Ishai, Eyal Kushilevitz, and Rafail Ostrovsky. Efficient arguments without short pcps. In *22nd Annual IEEE Conference on Computational Complexity (CCC 2007), 13-16 June 2007, San Diego, California, USA*, pages 278–291. IEEE Computer Society, 2007.

[Ish] Yuval Ishai. Zero-knowledge proofs from information-theoretic proof systems. https://zkproof.org/2020/08/12/information-theoretic-proof-systems/.

[Kil92] Joe Kilian. A note on efficient zero-knowledge proofs and arguments (extended abstract). In *STOC*, pages 723–732, 1992.

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

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

[Mei16] Or Meir. Combinatorial pcps with short proofs. *Comput. Complex.*, 25(1):1–102, 2016.

[NMVR94] David Naccache, David M'Raïhi, Serge Vaudenay, and Dan Raphaeli. Can D.S.A. be improved? complexity trade-offs with the digital signature standard. In Alfredo De Santis, editor, *Advances in Cryptology - EUROCRYPT '94, Workshop on the Theory and Application of Cryptographic Techniques, Perugia, Italy, May 9-12, 1994, Proceedings*, volume 950 of *Lecture Notes in Computer Science*, pages 77–85. Springer, 1994.

[RR19] Noga Ron-Zewi and Ron Rothblum. Local proofs approaching the witness length. *Electronic Colloquium on Computational Complexity (ECCC)*, 26:127, 2019.

[RR20] Guy N. Rothblum and Ron D. Rothblum. Batch verification and proofs of proximity with polylog overhead. In Rafael Pass and Krzysztof Pietrzak, editors, *Theory of Cryptography - 18th International Conference, TCC 2020, Durham, NC, USA, November 16-19, 2020, Proceedings, Part II*, volume 12551 of *Lecture Notes in Computer Science*, pages 108–138. Springer, 2020.

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

[RRR18] Omer Reingold, Guy N. Rothblum, and Ron D. Rothblum. Efficient batch verification for UP. In *33rd Computational Complexity Conference, CCC 2018, June 22-24, 2018, San Diego, CA, USA*, pages 22:1–22:23, 2018.

[RS96] Ronitt Rubinfeld and Madhu Sudan. Robust characterizations of polynomials with applications to program testing. *SIAM J. Comput.*, 25(2):252–271, 1996.

[RVW13] Guy N. Rothblum, Salil P. Vadhan, and Avi Wigderson. Interactive proofs of proximity: delegating computation in sublinear time. In *STOC*, pages 793–802, 2013.

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

[Sud95] Madhu Sudan. *Efficient Checking of Polynomials and Proofs and the Hardness of Approximation Problems*, volume 1001 of *Lecture Notes in Computer Science*. Springer, 1995.

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il