Electronic Colloquium on Computational Complexity, Report No. 159 (2025)

Efficiently Batching Unambiguous Interactive Proofs

Bonnie Berger, Rohan Goyal, Matthew M. Hong, and Yael Tauman Kalai

MIT

Abstract

We show that if a language $\mathcal{L}$ admits a public-coin unambiguous interactive proof (UIP) with round complexity ℓ, where a bits are communicated per round, then the *batch language* $\mathcal{L}^{\otimes k}$, i.e. the set of k-tuples of statements all belonging to $\mathcal{L}$, has an unambiguous interactive proof with round complexity $\ell \cdot \mathsf{polylog}(k)$, per-round communication of $a \cdot \ell \cdot \mathsf{polylog}(k) + \mathrm{poly}(\ell)$ bits, assuming the verifier in the UIP has depth bounded by $\mathsf{polylog}(k)$. Prior to this work, the best known batch UIP for $\mathcal{L}^{\otimes k}$ required communication complexity at least $(\mathsf{poly}(a) \cdot k^{\epsilon} + k) \cdot \ell^{1/\epsilon}$ for any arbitrarily small constant $\epsilon > 0$ (Reingold-Rothblum-Rothblum, STOC 2016).

As a corollary of our result, we obtain a *doubly efficient proof system*, that is, a proof system whose proving overhead is polynomial in the time of the underlying computation, for any language computable in polynomial space and in time at most $n^{O\left(\sqrt{\frac{\log n}{\log \log n}}\right)}$. This expands the state of the art of doubly efficient proof systems: prior to our work, such systems were known for languages computable in polynomial space and in time $n^{(\log n)^{\delta}}$ for a small $\delta > 0$ significantly smaller than $1/2$ (Reingold-Rothblum-Rothblum, STOC 2016).

ISSN 1433-8092

Contents

1 Introduction

Verification is one of the most fundamental concepts in computer science and is the basis for the definition of the complexity class NP. In the mid-eighties, a flurry of works expanded the notion of proofs beyond the classical notion of a mathematical proof to interactive proofs [GMR89, BM88], multi-prover interactive proofs [BGKW88, BFL90], probabilistically checkable proofs (PCPs) [BFLS91, FGL+91, AS92, ALM+92], interactive PCPs [KR08], and interactive oracle proofs [BCS16, RRR16]. This line of work has proven instrumental in cryptography and complexity theory, leading to breakthroughs in hardness of approximation and to the emergence of fundamental concepts in cryptography, such as zero-knowledge proofs [GMR89], which underlie many cryptographic primitives today. It is also instrumental in constructing "succinct proofs" used in many blockchain applications.

Interactive proofs Interactive proof systems were shown to be extremely powerful in the celebrated IP = PSPACE theorem [LFKN92, Sha92] Specifically, it was proven that the correctness of any time-T space-S computation could be verified by a $\text{poly}(S, n)$ time verifier, via an interactive proof. The main drawback of this result is that the time required by the prover to convince the verifier of the correctness is $2^{O(S \cdot \log S)}$, rendering it impractical for real-world applications.[1]

Doubly efficient interactive proofs. The work of [GKR15] initiated the study of *doubly efficient* interactive proofs, in which the prover !S runtime is required to be at most $p(T)$, where T is the time needed to carry out the underlying computation and p is a polynomial independent of T, while the verifier runs in time significantly less than T. They showed that the correctness of any computation that can be performed by a (log-space uniform) circuit of depth D and size T can be proven via an interactive proof whose communication complexity is $D \cdot \text{polylog}(T)$, the verifier runs in time $D \cdot \text{polylog}(T) + \tilde{O}(n)$, and the prover runs in time $\text{poly}(T)$.[2] In a breakthrough result, Reingold, Rothblum, and Rothblum [RRR16] constructed a doubly efficient (constant round) interactive proof for every language in time $T = n^{(\log n)^{\delta}}$ and polynomial space, for a sufficiently small constant $\delta > 0$. Since this work which was posted nearly 10 years ago, no improvements have been made to this fundamental problem. One of the building blocks in [RRR16] is a doubly efficient *batch unambiguous interactive proof*, which is the focus of our work.

Doubly efficient batch interactive proofs. Another important line of work is the one that focuses on doubly efficient *batch interactive proofs* [RRR16, RRR18, RR20], where the goal is to prove many statements !Hat the price of one. !IConcretely, suppose we want to verify statements $x_1, \ldots, x_k$, each of which belongs to a language that admits an interactive proof (i.e., a language in PSPACE). Observing that the total space needed to run k computations grows only by an (additive) $\text{polylog}(k)$ factor, it follows, from IP = PSPACE, that all k statements can also be proven via a single (batch) interactive proof with $\text{polylog}(k)$ overhead in communication complexity. This naturally raises the question:

[1] We note that in those works (and at that time) the primary focus was on the power of the proof systems themselves, where the verifier was assumed to run in polynomial time, while the prover was thought of as being all-powerful. In fact, in [BM88] the all-powerful prover was even named after the famous wizard Merlin.

[2] In particular, this implies an improved IP = PSPACE theorem where the communication complexity is $\text{poly}(S)$, the verifier runs in time $\text{poly}(S) + \tilde{O}(n)$, and the prover runs in time $2^{O(S)}$.

> Which languages in IP can be batched *doubly-efficiently*, with $\text{polylog}(k)$ overhead in communication complexity, while preserving the efficiency of the prover and verifier?

The work of [BKP+24] proved that if a language $\mathcal{L} \in \mathsf{NP}$ has a batch interactive proof with polylogarithmic overhead, where the prover runs in polynomial time given the witnesses (we refer to such proof as a *doubly efficient batch interactive proof*), then $\mathcal{L}$ also has a *statistical witness indistinguishable* (WI) proof system. This result can be viewed as an indication that not all of NP has doubly efficient batch interactive proofs, since it seems likely that not all NP languages admit statistical WI proofs.

Indeed, all known positive results in this regime focus either on UP, which is the class of NP languages for which each instance $x \in \mathcal{L}$ has a *unique* witness, or more generally, on the class UIP, consisting of all languages $\mathcal{L}$ that have an *unambiguous* interactive proofs.

An unambiguous interactive proof for a language $\mathcal{L}$ is an interactive proof system in which, for every $x \in \mathcal{L}$ and for every possible verifier's message there is only one answer that a prover can send that will later allow it convince the verifier to accept conditioned on this answer (with non-negligible probability). In other words, even though the prover may be all-powerful, there is at most one way for it to produce an accepting proof. We refer the reader to Section 3.3 for the formal definition.

For the case of UP, it was shown in [RR20] that there exists a doubly efficient batch interactive proof for any language $\mathcal{L} \in \mathsf{UP}$, where proving $x_1, \ldots, x_k \in \mathcal{L}$ requires communication complexity $m \cdot \text{polylog}(k)$, where m is the length of a single witness, and the prover runs in polynomial time given all the witnesses.

It was shown by [RRR16] that for every language $\mathcal{L}$ that has a public-coin UIP consisting of ℓ rounds, where in each round the prover and verifier send a bits, there exists a batch unambiguous interactive proof for proving that $x_1, \ldots, x_k \in \mathcal{L}$, where the communication complexity is at least $(\mathsf{poly}(a) \cdot k^{\epsilon} + k) \cdot \ell^{1/\epsilon}$, the number of rounds increase only by a constant factor (that grows exponentially with $1/\epsilon$), and the prover's and verifier's runtime increase by $\text{poly}(k, n, \ell)$, where $n = |x_i|$. It is critical that the underlying protocol being batched is public-coin[3], i.e. that the verifier's messages are simply uniformly random coins.

1.1 Our Results

We construct a batch unambiguous interactive proof for any language $\mathcal{L}$ that has an ℓ-round public-coin unambiguous interactive proof, where the communication complexity of the batch interactive proof for proving $x_1, \ldots, x_k \in \mathcal{L}$ grows by a factor of $\ell \cdot \text{polylog}(k)$, the round complexity grows by a factor of $\text{polylog}(k)$, and the runtime of the prover and verifier grow polynomially in ℓ, n, k where $n = |x_i|$.

Theorem 1 (Our Batch-UIP, Informal)**.** *Let $\mathcal{L}$ be a language with a public-coin* unambiguous interactive proof *system that on input x of length n (and where the prover may have an additional input w) runs in $\ell = \ell(n)$ rounds, in each round $a = a(n)$ bits are sent, and the verifier's verification circuit is a log-space uniform boolean circuit of depth at most* $\text{polylog}(k)$*. Then there is a public-coin unambiguous interactive proof $(\mathsf{P}_{\textit{Batch-UIP}}, \mathsf{V}_{\textit{Batch-UIP}})$ for $\mathcal{L}^{\otimes k}$, that on input $(x_1, \ldots, x_k)$, each of length n (and where the prover may have additional input $w_1, \ldots, w_k$), runs in $\ell \cdot \text{polylog}(k)$ rounds,*

[3]Note that the Goldwasser-Sipser transformation from any interactive proof to a public-coin interactive proof does not preserve prover efficiency [GS86, Vad00].

where in each round $a \cdot \ell \cdot \mathrm{polylog}(k) + \mathrm{poly}(\ell)$ *bits are sent. Moreover, the prover's computational complexity increases multiplicatively by* $\mathrm{poly}(k, \ell, n)$ *and the verifier's computational complexity becomes* $(\ell \cdot \mathsf{Vtime} + kna^2 \cdot \mathrm{poly}(\ell)) \cdot \mathrm{polylog}(k, n, a, \ell)$*, where* Vtime *is the verifier runtime of the base protocol.*

Remark 1. The assumption that the verification circuit is log-space uniform is without loss of generality by the Cook-Levin reduction [Coo71, Lev73] since any efficient verifier, as a Turing machine, can be simulated by a log-space uniform boolean circuit of a comparable size. The additional assumption that the verifier's verdict circuit is of low depth is relatively mild, since we can generically flatten out this circuit by having the prover send the values of all the wires in the verdict circuit. This increases the communication complexity (in a single round).

We provide a high-level overview of our batch UIP protocol in Section 2. We note that it uses a component of the batch UP protocol of [RR20], which we refer to as the Instance Reduction protocol. We need to generalize this protocol to hold with respect a distance measure, which we call execution-wise distance. We elaborate on this in Section 2.2.

Applications to doubly efficient interactive proofs We show how one can use Theorem 1 to improve the state-of-the-art on doubly efficient interactive proofs.

Theorem 2 (Our Doubly-efficient Interactive Proof, Informal)**.** *Every language computable in time* $T(n) \leq n^{O\left(\sqrt{\frac{\log n}{\log\log n}}\right)}$ *and space* $S(n) = \mathrm{poly}(n)$ *has a (public-coin) doubly efficient interactive proof, with verifier runtime and communication complexity as* $\mathrm{poly}(n) \cdot S(n))$ *and prover runtime* $T \cdot \mathrm{poly}(n)$*.*

Corollary 1 (Our Doubly Efficient IP, Informal)**.** *Every language decidable in time* $T(n) = n^{O\left(\sqrt{\frac{\log n}{\log\log n}}\right)}$ *and polynomial space has a (public-coin) doubly efficient interactive proof.*

For more details, see Theorem 8 and corollary 2. This improves upon the doubly efficient interactive proof from [RRR16], which is only for languages computed in polynomial space and time $T = n^{(\log n)^{\delta}}$ for a small constant $\delta > 0$ (which is significantly smaller than $1/2$).

1.2 Additional Related Work

The area of proof systems has become a popular area of research with numerous papers published every year. Most of these works are in the computational setting, where soundness is guaranteed to hold only against computationally bounded cheating provers. We do not elaborate on these works here, since they are less relevant for our result. We emphasize that our work is in the information theoretic setting, where soundness holds against any (even computationally unbounded) cheating prover. Most of the work in this regime is focused on guaranteeing privacy, such as zero-knowledge or witness indistinguishability. Again, we do not elaborate on these works since these are less relevant for our result.

A proof model which turns out to be useful for us, is that of *interactive proofs of proximity* (IPP) [RVW13], where a verifier has oracle access to a long input, and it interacts with a prover to check whether the input is close to one that satisfies a given property, while running in sublinear time. The non-interactive variant was explored in [GR15], and the significance of the round complexity of IPPs was studied in [GR17]. Although we do not focus on sublinear verification in our work, such proofs serve as a stepping stone to obtain our result.

2 Technical Overview

Our batch UIP protocol takes inspiration from the Batch-UP protocols of [RRR18, RR20].

The batch UP protocol from [RRR18, RR20]. The intuition underlying these protocols stems from the simple observation that if we are given k instances $x_1, \ldots, x_k$ that are d-far from being in $\mathcal{L}^{\otimes k}$, in the sense that at least d of the instances are not in $\mathcal{L}$, then by choosing $k \cdot \text{polylog}(k)/d$ of the instances at random, we have the guarantee that at least one of the selected instances is not in $\mathcal{L}$ with probability $1 - \mathsf{negl}(k)$. In other words, distance from $\mathcal{L}^{\otimes k}$ allows us to shrink the number of instances. Indeed, in the batch UP protocol of [RR20], they create distance and then shrink the number of instances, and they do this iteratively. To create distance, they use the GKR protocol.

The GKR protocol The GKR protocol [GKR15] is an unambiguous interactive protocol between a prover who wishes to prove to the verifier that a circuit C on input x outputs a bit b. The communication complexity and the number of rounds grow linearly with the depth of C, denoted by D, and poly-logarithmically with the size of C, denoted by S. It has the desired properties of being public-coin and the verdict function requires access to the input x at a single point in the low-degree extension of x.[4] In particular, if the verifier is given oracle access to the low-degree extension of the input x, then its runtime is $D \cdot \text{polylog}(S)$, which may be poly-logarithmic in the input length.

The GKR protocol can be thought of as a reduction from verifying that $C(x) = b$ to verifying that the low-degree extension of x at point j is equal to a value v, where j is a uniformly random point determined by the random messages sent by the verifier in the GKR protocol, and v is determined by the answers sent by the prover in GKR protocol. This is also denoted by $x \in \mathsf{PVAL}(j, v)$, a notation that originated in [RVW13]. The GKR protocol has the guarantee that if $C(x) \neq b$ then with high probability $x \notin \mathsf{PVAL}(j, v)$, where we can make this probability as close as we want to 1, and obtain probability $1 - \mu$ at the price of the communication complexity growing polynomially with $\log(\mu^{-1})$. Importantly, the GKR protocol is run in the *holographic* mode, namely that the verifier never access the input x directly — it just outputs claims about the low-degree extension of x.

The batch UP protocol of [RR20] starts by running the GKR protocol on the batch verification circuit that has the instance $\boldsymbol{x} = (x_1, \ldots, x_k)$ hardwired (supposedly in $\mathcal{L}^{\otimes k}$), and takes as input a witness $\boldsymbol{w} = (w_1, \ldots, w_k)$, and outputs 1 if and only if $\boldsymbol{w}$ is a valid witness for $\boldsymbol{x} \in \mathcal{L}^{\otimes k}$. Note that the depth of the batch verification circuit grows logarithmically with k, and hence the GKR protocol has communication complexity that grows only logarithmically in k (as desired). It has the guarantee that if $\boldsymbol{x} \notin \mathcal{L}^{\otimes k}$ then for every $\boldsymbol{w} = (w_1, \ldots, w_k)$, with high probability $\boldsymbol{w} \notin \mathsf{PVAL}(j, v)$. The batch UP protocol runs this GKR protocol $T = d \cdot m \cdot \text{polylog}(k)$ times, where $d = \text{polylog}(k)$ is the desired distance, in the sense that d witnesses will need to change in order to satisfy the resulting PVAL constraint, and m is the length of a single witness. Running the protocol T times in parallel reduces the soundness error to be exponentially small in T. They obtain the guarantee that if $\boldsymbol{x} \notin \mathcal{L}^{\otimes k}$ then for any $\boldsymbol{w} = (w_1, \ldots, w_k)$, with high probability, which is exponentially (in

[4] The low-degree extension is a specific linear error correcting code (similar to the Reed-Muller encoding). We define it formally in Section 3.1 but the details can be ignored here.

T) close to 1, it holds that $\boldsymbol{w} \notin \cap_{i=1}^{T} \mathsf{PVAL}(j_i, v_i)$. From now on, we use the condensed notation

$$\mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v}) = \cap_{i=1}^{T} \mathsf{PVAL}(j_i, v_i),$$

where $\boldsymbol{j} = (j_1, \ldots, j_T)$ and $\boldsymbol{v} = (v_1, \ldots, v_T)$. They then rely on the union bound to argue that even if a single x_i is not in $\mathcal{L}$, with high probability, every $\boldsymbol{w}$ that is d-close to the unambiguous witness,[5] in the sense that it differs from the unambiguous one on at most d witnesses, satisfies that $\boldsymbol{w} \notin \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$, and hence obtain the desired distance guarantee!

We emphasize that if $\boldsymbol{x} \notin \mathcal{L}^{\otimes k}$ then the batch verification circuit computes the identically zero function, and in particular outputs zero for every witness vector that is close to the unambiguous one. This is precisely what allows us to obtain distance from running the GKR protocol. As we shall see shortly, this will no longer be the case in the UIP setting, which brings with it new technical challenges.

Note that after creating distance by running the GKR protocol T times, the new instances are no longer batch instances, rather are instances of the form $\boldsymbol{x} = (x_1, \ldots, x_k)$ together with a joint PVAL constraint $\Phi = \Phi_{\boldsymbol{j},\boldsymbol{v}}$ on the corresponding witnesses $\boldsymbol{w} = (w_1, \ldots, w_k)$. The guarantee they obtain is that (with high probability) the unambiguous witness $\boldsymbol{w}$ corresponding to $\boldsymbol{x}$ is d-far from satisfying the constraint Φ, where distance is defined in terms of the minimum number of witnesses that need to be modified in order for the constraint Φ to be satisfied. In other words, they consider a new UP language $\mathcal{L}'$, where $(\boldsymbol{x}, \Phi) \in \mathcal{L}'$ if and only if the unambiguous witness $\boldsymbol{w}$ corresponding to $\boldsymbol{x} \in \mathcal{L}^{\otimes k}$ satisfies the PVAL constraint $\Phi(\boldsymbol{w}) = 1$. They obtain the guarantee that $(\boldsymbol{x}, \Phi)$ is d-far from $\mathcal{L}'$ in the sense that $(\boldsymbol{x}', \Phi) \notin \mathcal{L}'$, for every $\boldsymbol{x}'$ that differs from $\boldsymbol{x}$ on at most d instances.

They then show how to convert any such instance $(\boldsymbol{x}, \Phi)$ that is d-far from $\mathcal{L}'$ into a new instance $(x_{i_1}, \ldots, x_{i_{k/d}}, \Phi')$ that is not in $\mathcal{L}'$, but without any distance guarantee.[6] Recall that d was chosen so that $d = \mathrm{polylog}(k)$, and hence the number of instances decreased by a factor of $\mathrm{polylog}(k)$.

They continue as above, and use the GKR protocol to convert this instance into a new instance of the form $(x_{i_1}, \ldots, x_{i_{k/d}}, \Phi'')$, where Φ'' is a new PVAL instance, with the guarantee that if $(x_{i_1}, \ldots, x_{i_{k/d}}, \Phi') \notin \mathcal{L}'$ then $(x_{i_1}, \ldots, x_{i_{k/d}}, \Phi'')$ is d-far from $\mathcal{L}'$. To obtain this reduction, they run the GKR protocol (repeated in parallel T times) with respect to the circuit that has $(x_{i_1}, \ldots, x_{i_{k/d}}, \Phi')$ hardwired and on input $(w_{i_1}, \ldots, w_{i_{k/d}})$ outputs 1 if and only if these are valid witnesses of $(x_{i_1}, \ldots, x_{i_{k/d}})$ and they satisfy the constraint Φ'. They continue to iterate between creating distance and shrinking the number of instances until they are left with $\mathrm{polylog}(k)$ number of instances, at which point the prover simply sends them over.

Our batch UIP protocol We refer to the verifier messages in the UIP protocol as *queries* and the prover messages as *answers*. Our batch UIP protocol follows a similar blueprint to the batch UP protocol of [RRR18, RR20], but for UIP as opposed to UP, which brings with it several challenges that we elaborate on below. The protocol starts by creating distance.

[5] If $x_i \notin \mathcal{L}$, we define the unambiguous witness to be $w_i = 0^m$.

[6] We note that the instance reduction protocol from [RR20] needs the stronger distance guarantee that at least $d \cdot m$ bits of the unambiguous witness of $\boldsymbol{x}$ need to change in order to satisfy the constraint Φ. This distance guarantee is indeed promised by the GKR protocol. We chose to present distance in terms of the number of witnesses that need to be changed since it is consistent with our approach. See Section 2.1.

2.1 Creating Distance

In the UIP setting, the instances $x_1, \ldots, x_k$ may lack witnesses certifying membership in $\mathcal{L}$, since $\mathcal{L}$ need not belong to NP. Hence, we cannot apply the GKR protocol on the batch verification circuit that takes as input the witnesses (as such witnesses do not exist). Instead, in our UIP setting, distance is created by running a checksum over the k UIP protocols.

Checksum over the UIPs We begin by performing a checksum over the k UIP instances, following the batch UIP construction of [RRR16].[7] Namely, we run the k UIP protocols in parallel, where the verifier uses the same queries in all the executions and the prover sends its answers via a checksum.

We assume that the underlying UIP protocol is public-coin and hence it can be run in the following way. Denoting by ℓ the number of rounds in the underlying UIP protocol, in each round $r \in [\ell]$ of the batch UIP protocol, the verifier sends a random message $q_r \in \{0,1\}^a$ as in a single UIP protocol. The prover generates the k answers $\boldsymbol{a}_r = (a_{r,1}, \ldots, a_{r,k}) \in (\{0,1\}^a)^k$, but rather than sending $\boldsymbol{a}_r$ (which is too long), it sends a checksum of $\boldsymbol{a}_r$ (which should be thought of as a "digest" of $\boldsymbol{a}_r$). Formally, the prover sends the low-degree extension of $\boldsymbol{a}_r$ at $T = d \cdot a \cdot \text{polylog}(k)$ random points, where $d = \text{polylog}(k)$, and a is the length of each of the prover's messages in a single execution. These points are sampled by the verifier at the beginning of the protocol.

Now we show that distance is created: if the checksummed transcript of answers only deviate from the unambiguous one by at most d executions, then at least one execution in the transcript will be rejecting with overwhelming probability. Suppose the cheating prover sends checksums that correspond to answers deviating in at most d executions, then for every $r \in [\ell]$, by the checksum's error-correcting property, it is possible to extract from the r-th round checksums the entire set of r-th round answers. By unambiguity, at least one of the extracted answers will lead to a rejecting transcript with overwhelming probability when $(x_1, \ldots, x_k) \notin \mathcal{L}^{\otimes k}$.

However, note that the verifier cannot verify the validity of the answers given by the prover, since the answers were not sent in the clear, but rather in the form of a checksum. To this end, we use the GKR protocol, this time not to create distance, but rather to allow the verifier to verify the validity of the answers. Specifically, as we describe below, the GKR protocol is used to convert the check-summed transcript with the distance guarantee mentioned above, into a PVAL claim with an analogous distance guarantee.

Remark 2. We note that this is where our approach differs substantially from [RRR16], which verified correctness by having the prover open some of the bits in *all* k instances (as opposed to using a succinct interactive proof, such as GKR, to verify correctness). As a result, denoting the number of iterations in their protocol as $1/\epsilon$ (for some $\epsilon > 0$), their batch UIP incurs an additive factor of k as well as a multiplicative factor of $(1/\epsilon)^{\text{poly}(1/\epsilon)}$ to the communication complexity. We elaborate on this difference in Section 2.3.

The GKR protocol In order to verify the check-summed UIPs, the prover and verifier run the GKR protocol, this time on the batch verification circuit that has hardwired into it the instances $\boldsymbol{x} = (x_1, \ldots, x_k)$, the verifier's queries $\boldsymbol{q} = (q_1, \ldots, q_\ell)$, and a constraint Φ on the prover's answers (as defined by the checksum). The circuit takes as input all the prover's answers $\boldsymbol{a} = (\boldsymbol{a}_1, \ldots, \boldsymbol{a}_\ell)$

[7] We note that in [RRR16] the checksum was not done directly on the UIP but rather the UIP was converted to an IOP and the checksum was done on the IOP. We elaborate on this in Section 2.3.

and outputs 1 if and only if for every $j \in [k]$ the transcript $(q_1, a_{1,j}, \ldots q_\ell, a_{\ell,j})$ is accepted with respect to x_j, and the answers $\boldsymbol{a}$ satisfy Φ.

The distance guarantee of the check-summed protocol implies that with high probability, every transcript of answers $\boldsymbol{a} = (\boldsymbol{a}_1, \ldots, \boldsymbol{a}_\ell)$ that is d-close to the unambiguous one is rejected by the batch verification circuit, where here (and throughout our work) we define d-closeness to mean that it differs in at most d executions.

Remark 3. We emphasize that our definition of d-closeness is not the standard Hamming distance (which is bit-wise distance), but rather it is an execution-wise distance. Since our definition of distance deviates from the one used in [RR20] (which is Hamming distance) we need to prove that the instance reduction protocol from [RR20] works with our notion of execution-wise distance. We elaborate on this in Section 2.2.

After running the GKR protocol (in parallel) the parties obtain $\boldsymbol{j}$ and $\boldsymbol{v}$ such that for every $\boldsymbol{a}$ that is d-close to the unambiguous one, with high probability, $\boldsymbol{a} \notin \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$. We then rely on the union bound, following the approach in the UP setting, to argue that with high probability for every $\boldsymbol{a}$ that is d-close to the unambiguous one it holds that $\boldsymbol{a} \notin \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$. Therefore, we can apply an instance reduction protocol (similar to the one from [RR20], which we elaborate on in Section 2.2), which reduces the problem to verifying that $(x_{i_1}, \ldots, x_{i_{k/d}}, \boldsymbol{q}, \Phi') \in \mathcal{L}'$ for some Φ', where the language $\mathcal{L}'$ is defined so that $(x_{i_1}, \ldots, x_{i_{k/d}}, \boldsymbol{q}, \Phi') \in \mathcal{L}'$ if and only if the unambiguous answers, corresponding to the instances $x_{i_1}, \ldots, x_{i_{k/d}}$ and queries $\boldsymbol{q}$, satisfy Φ'.[8]

The guarantee we have is that if $(x_1, \ldots, x_k, \boldsymbol{q}, \Phi)$ is d-far from $\mathcal{L}'$, in the sense that every answer $\boldsymbol{a}'$ that is d-close to the unambiguous one, does not satisfy Φ, then with high probability $(x_{i_1}, \ldots, x_{i_{k/d}}, \boldsymbol{q}, \Phi') \notin \mathcal{L}'$. We note that this instance does not have a distance guarantee, and generating distance for instances in $\mathcal{L}'$ introduces new challenges that we elaborate on below.

Obtaining distance for $\mathcal{L}'$ It is tempting to try and continue as in the UP setting, and apply the GKR protocol to create distance. The problem is that at this point the cheating prover knows all the verifier's queries $\boldsymbol{q}$ in advance, and thus can easily generate accepting answers $\boldsymbol{a}$. Moreover, a cheating prover can generate answers $\boldsymbol{a}$ that are close to the unambiguous ones and which cause the GKR circuit to accept! Hence, the GKR protocol cannot be used to create distance.

We emphasize that the problem stems from the fact that the prover knows all the queries $\boldsymbol{q}$ ahead of time, and thus can easily generate accepting answers even for $x \notin \mathcal{L}$. We get around this problem by adding the following random continuation protocol from [RRR16].

Random continuation to the rescue In the random continuation protocol, the verifier chooses fresh queries $q'_1, \ldots, q'_\ell \leftarrow \{0,1\}^a$, and the prover and verifier run a checksum of $\ell \cdot k/d$ UIP protocols, where in the $(i \cdot k/d + j)$'th protocol, where $i \in \{0, 1, \ldots, \ell - 1\}$ and $j \in \{1, \ldots, k/d\}$, the verifier sends messages $(q_1, \ldots, q_i, q'_{i+1}, \ldots, q'_\ell)$ and the prover replies with answers corresponding to the j'th instance. Thus, the prover and verifier run ℓ UIP protocols per instance, and thus a total of $\ell \cdot k/d$ UIP protocols, all run in a checksum.

Each of the prover's checksums consist of $T = d' \cdot a \cdot \mathrm{polylog}(k)$ points in the low-degree extension of each vector of answers, where here we take $d' = \ell \cdot d$.

Remark 4. The reason for this choice of distance d' is that the instance reduction protocol, applied after the distance is generated, reduces the number of instances by a factor of d'. However, the

[8] We note that $\mathcal{L}'$ is not necessarily in NP (as was not $\mathcal{L}$).

random continuation increases the number of protocols by a factor of ℓ, and hence to ensure that the number of instances goes down by a factor of d, we need to take $d' = \ell \cdot d$.

One can argue (as above) that if the prover deviates from the (unambiguous) random continuation protocol on at most d' of the executions (where in each execution it sends a message of length a) then one can efficiently extract the messages of the prover from the check-summed transcript. Therefore, if there exists $i \in [\ell]$ such that the i'th round check-summed answers sent by the prover, corresponding to $(q_1, \ldots, q_i)$, were not the unambiguous ones, then by the unambiguity property the prover will fail to provide accepting answers for random continuation $(q'_{i+1}, \ldots, q'_\ell)$, even if the corresponding instance is in $\mathcal{L}$. This implies that if there exists $i \in [\ell]$ such that the i'th check-summed answers corresponding to $(q_1, \ldots, q_i)$, are not the unambiguous ones, and the prover deviates from the unambiguous answers in at most d' executions, then we can efficiently extract from the check-summed transcript a rejecting transcript of answers sent by the prover. This gives us the desired distance guarantee, and we can now use the GKR protocol, as above, to reduce this to a PVAL instance $\mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$ with the guarantee for every $\boldsymbol{w}$ that is d'-close to the unambiguous one, $\boldsymbol{w} \notin \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$. Hence, we can use the instance reduction protocol (similar to the one given in [RR20]) to reduce the number of instances from $\ell \cdot k/d$ to $\ell \cdot k/d \cdot 1/d' = k/(d^2)$.

Remark 5. We note that in our actual batch UIP protocol, for the sake of simplicity, we use distance $d' = \ell \cdot \mathrm{polylog}(k)$ from the beginning of the protocol, and do not treat the initial reduction, from a batch instance $(x_1, \ldots, x_k) \in \mathcal{L}^{\otimes k}$ to an instance in $\mathcal{L}'$, differently.

2.2 Instance Reduction

The Need for Execution-wise Distance Recall that the distance guarantee provided by our distance generation protocol, described in Section 2.1, is that if we start with a false statement on say k' instances, then after the distance generation protocol (which includes the checksum protocol followed by the GKR protocol), we obtain a PVAL constraint $\mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$ that is not satisfied by any transcript of answers that differs on at most d' executions from the prescribed transcript of answers (with respect to fixed verifier's queries). For the next step, we would like to use the instance reduction protocol from [RR20] to further reduce this to a false claim on a subset of k'/d' instances (at the cost of losing our distance guarantee).

Unfortunately, the [RR20] protocol is guaranteed to produce a false claim on a subset of k'/d' instances, only if the original PVAL constraint $\mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$ is not satisfied by any transcript that differs from the prescribed transcript in at most $(d' \cdot \ell \cdot a)$-bits! Specifically, the issue is that these two protocols use a different distance measure, and these are not compatible: One guarantees execution-wise distance while the other requires Hamming distance.

We note that being d'-far in execution distance trivially implies being d'-far in Hamming distance. However, to reduce the number of instances by a d'-factor, the [RR20] protocol requires the Hamming distance to be $d' \cdot \ell \cdot a$, where $\ell \cdot a$ is the length of a single transcript. This requirement is much stronger than requiring the execution distance to be d'. In other words, if we only have the guarantee that the execution distance is d', then applying [RR20] will result with the number of instances being $k' \cdot \frac{d'}{\ell \cdot a}$, which offers no decrease in the number of instances, unless we take d' to be larger than $\ell \cdot a$, in which case we invest $d' \cdot \ell \cdot a \cdot \mathrm{polylog}(k)$ in communication to decrease the number of instances by a factor of d'. This will result in a suboptimal batch UIP result. While further optimizations are possible here, it seems that any purely Hamming distance based approach will result in a larger overhead than what we achieve.

Instead, we choose to make the two distance measures compatible by changing the instance reduction protocol to guarantee that if the PVAL constraint is not satisfied by any transcript that is d'-close in execution distance to the prescribed one, then it results with a false claim on a subset of k'/d' instances. We next elaborate on how this is done.

Overview of the original instance reduction protocol for PVAL from [RR20] The verifier in the instance reduction protocol of [RR20] takes as input $\boldsymbol{j}, \boldsymbol{v}$ that defines the PVAL constraint. In the YES case, the claim $\boldsymbol{w} \in \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$ is true, where $\boldsymbol{w}$ is the implicit vector of unique witnesses $\boldsymbol{w} = (w_1, \ldots, w_k) \in (\{0,1\}^m)^k$. In the NO case, $\boldsymbol{w}$ as a string in $\{0,1\}^{k \cdot m}$ is d-far in Hamming distance from $\mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$ with $d = m \cdot \mathrm{polylog}(k)$.

They show how to subsample a $\frac{1}{d}$ fraction of the instances together with a new false PVAL claim on the corresponding witnesses. This is not straightforward since the initial constraint is a global constraint, and cannot be applied to each witness w_i separately. The authors in [RR20] propose to transform the global constraint into a bunch of localized constraints, each defined over $\frac{1}{d}$ fraction of the instances. In order to obtain these localized claims, the [RR20] protocol proceeds in at most $\log k$ iterations as follows:

1. The multi-linearity of $\mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$ allows the prover to break the PVAL claim into two halves: PVAL_0 and PVAL_1, where each claim is over $k/2$ witnesses. Specifically, the claim PVAL_b is over $\boldsymbol{w}_b$ where $\boldsymbol{w}_0$ and $\boldsymbol{w}_1$ are defined by:

$$\boldsymbol{w}_0 = (w_1, \ldots, w_{k/2}) \quad \text{and} \quad \boldsymbol{w}_1 = (w_{k/2+1}, \ldots, w_k).$$

   This results in $(\boldsymbol{w}_0, \boldsymbol{w}_1)$ being d-far from the new claim $\mathsf{PVAL}_0 \times \mathsf{PVAL}_1$. For this overview, assume that these w_0 and w_1 are both $d/2$ far from the respective PVAL instances.

2. The next step would be to try to fold these two claims into a single claim of "size $k/2$" *while keeping the absolute distance to be d.*

   A naive folding by taking a random linear combination of $\boldsymbol{w}_0$ and $\boldsymbol{w}_1$ along with the corresponding linear combination of PVAL_0 and PVAL_1 may reduce the absolute distance to $d/2$ since the errors in $\boldsymbol{w}_0$ and $\boldsymbol{w}_1$ may overlap. Instead [RR20] applies a more sophisticated folding, as follows.

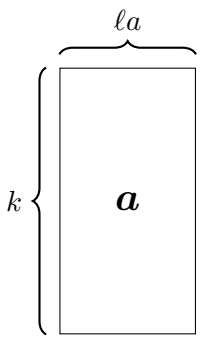
3. The verifier uses a *pairwise-independent* family π to transform $\boldsymbol{w}_1$ and PVAL_1 into a new claim PVAL'_1 about $\pi(\boldsymbol{w}_1)$. The purpose is to scramble the $d/2$ errors that prevent $\boldsymbol{w}_1$ from satisfying PVAL_1.

4. The verifier "folds" the two claims into one, by taking a random linear combination of $\boldsymbol{w}_0$ and $\pi(\boldsymbol{w}_1)$, and a random linear combination of PVAL_0 and PVAL'_1 to get a new claim that is also roughly d-far.

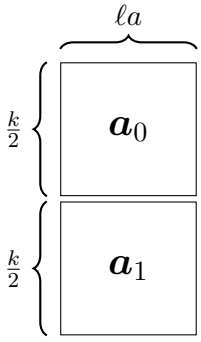
Each iteration allows us to reduce a d-far claim to another d-far claim but on a "folded" instance of half the size. The verifier repeats this until it is left with a d-far claim over roughly d "folded" witnesses. Then the verifier kindly asks the prover to send the batch of "folded" witnesses $\boldsymbol{w}^* = (\boldsymbol{w}^*_1, \ldots, \boldsymbol{w}^*_d)$ in the clear, which requires $d \cdot m$ communication. Then it checks whether the batch of the "folded" witnesses $\boldsymbol{w}^*$ satisfies the final "folded" PVAL claim. If not, it rejects immediately. Otherwise, in the NO case, $\boldsymbol{w}^* = (\boldsymbol{w}^*_1, \ldots, \boldsymbol{w}^*_d)$ is still d-far from any $\boldsymbol{w}^*$ that satisfies the final

PVAL claim, so the sent $\boldsymbol{w}^*$ must differ from $\boldsymbol{w}^*$ on at least d instances. Therefore, the verifier can indeed subsample constantly many of these instances $\boldsymbol{w}_i^*$ to detect an inconsistency with high probability, and the inconsistency depends only on the (k/d) witnesses folded in $\boldsymbol{w}_i^*$.

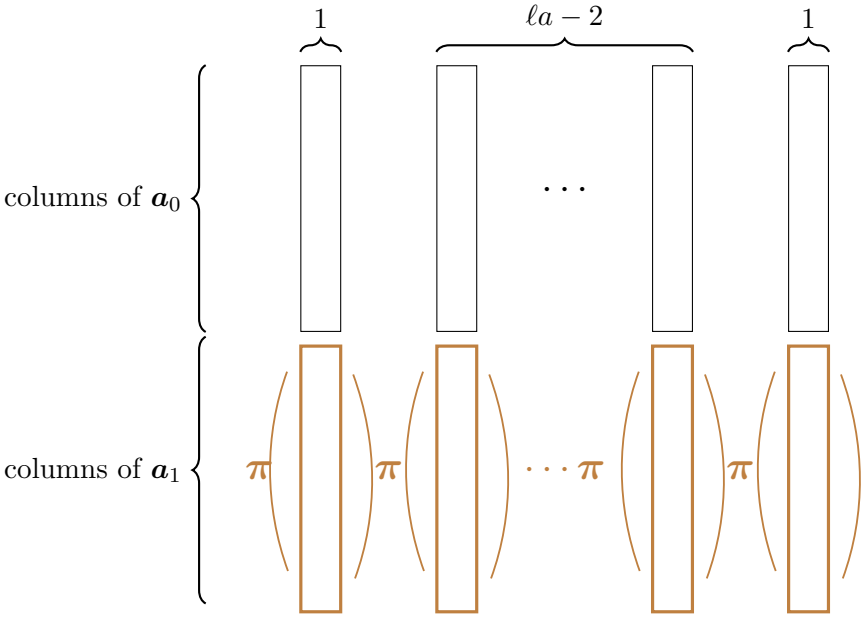
Working with execution-wise distance We generalize (and tweak) the protocol from [RR20] for Hamming distance to work for the execution-wise distance. The modification of the folding step is illustrated in Figure 1. To this end, we view the batch of transcripts $\boldsymbol{a}$ as a matrix where each row corresponds to a transcript of a single instance (Figure 1a). We split $\boldsymbol{a}$ into its top half submatrix $\boldsymbol{a}_0$ and bottom half submatrix $\boldsymbol{a}_1$ (Figure 1b). Execution-wise distance is preserved in the folding step by choosing a sufficiently random permutation π to scramble each column of $\boldsymbol{a}_1$ independently (Figure 1c). (Note that this is different from scrambling the entire $\boldsymbol{a}_1$, which moves errors across columns, as in the original protocol from [RR20].)



(a) $\boldsymbol{a}$'s rows are the transcripts of the instances.



(b) $\boldsymbol{a}$ split into two halves $\boldsymbol{a}_0$ and $\boldsymbol{a}_1$.



(c) The same permutation π: $\{0,1\}^{\frac{k}{2}} \to \{0,1\}^{\frac{k}{2}}$ is applied to the columns of $\boldsymbol{a}_1$.

Figure 1: Our folding step works with execution-wise distance by permuting each column randomly.

We show that this protocol (as well as the [RR20] protocol) also has unambiguous soundness when the underlying field is large. Thus, it can also be used as a building block for a UIP protocol, leading to many of our applications.

2.3 Efficiency Gain over the RRR16 Protocol

The Batch-UIP protocol of [RRR16] has communication complexity at least

$$k \cdot (\ell/\epsilon)^{\mathrm{poly}(1/\epsilon)} + \mathrm{poly}(a) \cdot (\ell/\epsilon)^{\mathrm{poly}(1/\epsilon)} \cdot k^{\epsilon}$$

for an arbitrarily small constant $\epsilon > 0$. On a closer inspection, their protocol allows the choice of a slightly sub-constant $\epsilon = \epsilon(k)$, at the cost of creating a super-polynomial dependency on ℓ (i.e. the terms involving $\ell^{1/\epsilon}$) in the communication complexity. In contrast, our Batch-UIP always achieves a communication complexity of $O(a \cdot \ell^2 \cdot \mathrm{polylog}(k) + \mathrm{poly}(\ell))$.

Both Batch-UIP protocols use the idea of running many protocols in a checksum and running the random continuation protocol (also in a checksum). These are used to enforce that the cheating prover will be caught if it deviates from the protocol only in a few instances. Nevertheless, our protocol is very different in how the verifier catches the prover if it indeed only deviated in a few instances. While the verifier in [RRR16] checks explicitly on its own that this is the case, by asking the prover to reveal some of these check-summed messages, the verifier in our protocol delegates these checks back to the prover (via the GKR protocol). This reduces the overall costs significantly.

Specifically, in the protocol of [RRR16], the prover first encodes the base UIP into an (unambiguous) *Interactive Oracle Proof* (IOP). In such a system, the prover and verifier interact, with the verifier sending random coins and receiving answers from the prover. The verifier does not read the prover's messages immediately, and only after the interaction is complete, it makes its verdict based on a set of random queries into very few locations of the stored transcript. Similar to our usage, the prover and verifier run the k instances of the IOP implicitly, with the prover only sending checksums of the underlying messages.

Importantly, after the interaction, in [RRR16], the verifier sends the set of query locations that it would like to check in the IOP to the prover, and the prover responds with *all* the messages on these locations. In addition to checking that all k instances of the IOP are accepting, the verifier also confirms that the checksums on those messages agree with what the prover committed to during the interaction. Note that this requires communicating $\Omega(k)$ bits.

The guarantee they obtain is that if originally there was at least one false statement, then if the verifier accepts the openings then it must be the case that the prover deviated on many instances of the IOP, so the verifier can subsample the instances to reduce the number of instances.

At this point, the random queries of the verifier in the original IOP have been revealed through the interaction, and to remedy this, the verifier in [RRR16] runs the IOP corresponding to a random continuation of the base IOP (constructed from the base UIP) in the next distance-creation (checksum) phase. The random continuation always incurs a multiplicative factor of ℓ blowup in the IOP, so if this process is iterated for $1/\epsilon$ times, the transcript length of the final IOP is $\mathrm{poly}(a) \cdot \ell^{1/\epsilon}$.

In our protocol, the verifier avoids this accumulation of the factor ℓ in the transcript length by instead "flattening" the transcripts produced by the random continuation. Namely, instead of treating the implicit transcripts as k transcripts of the random continuation protocol (where in each round the prover sends a message of length $a \cdot \ell$), it treats them as $k \cdot \ell$ instances of the *base UIP*. This is desirable because if we generate a distance of $d \geq \ell \cdot \mathrm{polylog}(k)$ by the checksum, we

can offset this factor of ℓ in the number of instances in the instance reduction step. A caveat is that the verifier should additionally check a joint constraint over each of the ℓ hybrid transcripts derived from the same instance by the random continuation, namely that they have shared prefixes. This check can be delegated to the prover via the GKR protocol. This optimization is not possible under the framework of IOP of [RRR16], since a priori the IOP of the random continuation protocol cannot be flattened into ℓ independent IOP's.

2.4 Doubly Efficient Interactive Proofs

Our Batch-UIP protocol can be used to obtain new (and more powerful) doubly-efficient interactive proofs for bounded-space computations, by applying a similar technique as that presented in [RRR16]. Intuitively, any Batch-UIP protocol can be used to obtain a doubly-efficient interactive proof for poly-space computations.

Formally, let $\mathcal{L} \in \mathsf{DTISP}(T, S)$ be a language that is decidable by a Turing machine $\mathcal{M}$ in time $T(n)$ and space $S(n)$. Let $t = t(n) \leq T(n)$ be a parameter, and let $\mathcal{L}_t$ be the set of all tuples $(x, w_1, w_2) \in \{0,1\}^n \times \{0,1\}^{O(S(n))} \times \{0,1\}^{O(S(n))}$ such that the Turing machine $\mathcal{M}$ transitions from configuration w_1 to configuration w_2 on input x in exactly t steps.

Crucially, for every $k = k(n) \in \mathbb{N}$, the language $\mathcal{L}_{k \cdot t}$ can be expressed as a batch of $\mathcal{L}_t$. Specifically, for input $(x, w_0, w_{k \cdot t})$:

- The prover runs $\mathcal{M}$ on input x for $k \cdot t$ steps, and obtains the intermediate configurations $w_1, w_2, \ldots, w_{k \cdot t}$. (Note this step can be skipped if the prover already has the intermediate configurations.)
- The prover sends the k intermediate configurations $w_t, w_{2t}, \ldots, w_{k \cdot t}$ to the verifier.
- Both parties run the Batch-UIP protocol on the k instances $\{(x, w_{(i-1) \cdot t}, w_{i \cdot t})\}_{i=1}^{k}$ of $\mathcal{L}_t$, and the verifier accepts if and only the Batch-UIP protocol accepts.

Therefore, an efficient batching protocol such as our protocol allows the verifier to verify $(k \cdot t)$-step transitions by batch-verifying t-step transitions, at the cost of a small (polylogarithmic) increase in complexity. By repeating this process with a careful balancing of the parameters, the verifier gradually increases the transition length t that it can verify, until it reaches $\mathcal{L}_{T(n)}$.

We show that with $k = \mathrm{poly}(n)$, this process can be repeated for $O(\sqrt{\frac{\log n}{\log \log n}})$ times before the communication complexity becomes superpolynomial, obtaining a doubly-efficient interactive proof for languages in $\mathsf{DTISP}(T, S)$ where $T = n^{O\left(\sqrt{\frac{\log n}{\log \log n}}\right)}$ and $S = \mathrm{poly}(n)$, where the prover runtime is $T \cdot \mathrm{poly}(n)$, and the verifier runtime is $\mathrm{poly}(n)$.

3 Preliminaries

We adhere to the following convention when possible.

- Lower case letters a, b mean scalars, while bolded lower case letters $\boldsymbol{a}, \boldsymbol{b}$ mean vectors. Upper case bold letters like $\boldsymbol{A}, \boldsymbol{B}$ are matrices.
- For an integer n, we denote by $[n]$ the set $\{1, 2, \ldots, n\}$. We let $\mathbb{F}$ denote a finite field. $\mathbb{GF}(2)$ denotes the finite field with 2 elements.

- For a vector $\boldsymbol{u} \in \mathbb{F}^n$, and a subset $\mathcal{S} \subset [n]$, $\boldsymbol{u}|_{\mathcal{S}} \in \mathbb{F}^{|\mathcal{S}|}$ denotes the subvector of $\boldsymbol{u}$ indexed by $\mathcal{S}$. For a sequence of vectors $\{\boldsymbol{u}_i\}_{i\in\mathcal{I}}$, where each $\boldsymbol{u}_i \in \mathbb{F}^n$, the notation $(\boldsymbol{u}_i)_{i\in\mathcal{I}} \in \mathbb{F}^{n\times|\mathcal{I}|}$ represents the matrix whose columns are the vectors $\boldsymbol{u}_i$ for every $i \in \mathcal{I}$.

- Given a matrix $\boldsymbol{A} = (\boldsymbol{a}_1, \ldots, \boldsymbol{a}_n) \in \mathbb{F}^{m\times n}$, $\boldsymbol{a}_j \in \mathbb{F}^m$ denotes its j-th column. We also use $\boldsymbol{A}[i,:]$ to denote the i-th row of A. For a subset $\mathcal{S} \subset [m]$, $\boldsymbol{A}[\mathcal{S},:]$ denotes the submatrix of $\boldsymbol{A}$ consisting of rows indexed by $\mathcal{S}$. We also consider matrices $A \subset \mathbb{F}^{k\times n}$ whose rows are indexed by an ordered set of k pairs, $\mathcal{T} = \{(u^1, v^1), \ldots, (u^k, v^k)\}$, and we denote by $\boldsymbol{A}[(u^i, v^i),:]$ the row of $\boldsymbol{A}$ indexed by $(u^i, v^i) \in \mathcal{T}$, and for $\mathcal{S} \subset \mathcal{T}$, $\boldsymbol{A}[\mathcal{S},:]$ denotes the submatrix of $\boldsymbol{A}$ consisting of rows indexed by $\mathcal{S}$.

- $\Delta(\boldsymbol{u}, \boldsymbol{v})$ is the (absolute) Hamming distance between the vectors $\boldsymbol{u}$ and $\boldsymbol{v}$. Denote by $\Delta(\boldsymbol{u})$ the vector $\boldsymbol{u}$'s *Hamming weight*, defined to be the number of non-zero elements in $\boldsymbol{u}$. For any $d \in \mathbb{N}$, two vectors $\boldsymbol{u}, \boldsymbol{v}$ are d-close (in Hamming distance) if $\Delta(\boldsymbol{u}, \boldsymbol{v}) \leq d$. On a linear space $\mathcal{U} \subset \mathbb{F}^a$, let $\Delta(\mathcal{U}) \coloneqq \min_{\boldsymbol{u}\in\mathcal{U}, \boldsymbol{u}\neq\boldsymbol{0}} \Delta(\boldsymbol{u})$.

- Given a Boolean circuit V, $\mathsf{size}(\mathsf{V})$ is the number of gates in the circuit, and $\mathsf{depth}(\mathsf{V})$ is the depth of the circuit.

An important distance notion that we consider is the "column distance," denoted by Δ_c, between matrices.

Definition 1 (Δ_c-distance)**.** Let $\mathbb{F}$ be a finite field. For matrices $\boldsymbol{A} = (\boldsymbol{a}_1, \ldots, \boldsymbol{a}_L) \in \mathbb{F}^{M\times L}$ and $\boldsymbol{B} = (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_L) \in \mathbb{F}^{M\times L}$, the Δ_c-*distance* between $\boldsymbol{A}$ and $\boldsymbol{B}$, denoted by $\Delta_c(\boldsymbol{A}, \boldsymbol{B})$, is the maximum Hamming distance between corresponding columns of $\boldsymbol{A}$ and $\boldsymbol{B}$, i.e.

$$\Delta_c(\boldsymbol{A}, \boldsymbol{B}) = \max_{i\in[L]} \Delta(\boldsymbol{a}_i, \boldsymbol{b}_i).$$

If $\Delta_c(\boldsymbol{A}, \boldsymbol{B}) \leq d$, we say $\boldsymbol{A}$ and $\boldsymbol{B}$ are Δ_c-d-*close*,

Furthermore, let $d \in \mathbb{N}$ be some distance parameter, then given $\boldsymbol{A} \in \mathbb{F}^{M\times L}$, its Δ_c-d-*ball* is defined to be

$$\mathcal{B}_{d,\mathbb{F}}(\boldsymbol{A}) \coloneqq \left\{\boldsymbol{A}' \in \mathbb{F}^{M\times L} : \Delta_c(\boldsymbol{A}, \boldsymbol{A}') \leq d\right\}.$$

Remark 6. Observe that Δ_c is a (non-tight) lower bound on how many rows have to be modified to transform one matrix into another.

3.1 Low Degree Extensions and Polynomial Valuation (PVAL)

All finite fields $\mathbb{F}$ considered in this work are always *constructible* in the following sense.

Definition 2 (Constructible Field Ensemble)**.** We say a field ensemble $\mathbb{F} = (\mathbb{F}_n)_{n\in\mathbb{N}}$ is *constructible* if every element in $\mathbb{F}_n$ has a $O(\log|\mathbb{F}_n|)$-bit representation, and addition, multiplication, and inverses can be computed in $\mathrm{polylog}(|\mathbb{F}_n|)$ time given the representations.

It is well known that for every $S = S(n)$, constructible field ensembles $\mathbb{F} = (\mathbb{F}_n)$ with $|\mathbb{F}_n| = \Theta(S)$ exist. Moreover, we make the (mild) assumption that addition, multiplication and inverses can be computed in $\tilde{O}(\log|\mathbb{F}_n|)$ time, where $\tilde{O}$ omits $\mathrm{polylog}\log(|\mathbb{F}_n|)$ factors. This implies that degree-d

polynomials in $\mathbb{F}[X]$ can be evaluated in $\tilde{O}(d \log|\mathbb{F}_n|)$ time (with $\tilde{O}$ omitting $\text{polylog}(d, \log|\mathbb{F}_n|)$ factors)[9]. These properties are satisfied in fields that support FFT. (See table 8.6 in [vzGG13])

We utilize the Schwartz-Zippel lemma, a basic fact about low-degree polynomials over finite fields.

Lemma 1 (Schwartz-Zippel Lemma, [Zip79, Sch80])**.** *Let $\mathbb{F}$ be a field, let $m \in \mathbb{N}$, and let $f \in \mathbb{F}[X_1, \ldots, X_m]$ be a non-zero polynomial of total degree d. Then*

$$\Pr_{\boldsymbol{r} \leftarrow \mathbb{F}^m}[f(\boldsymbol{r}) = 0] \leq \frac{d}{|\mathbb{F}|}.$$

Given a subset $H \subset \mathbb{F}$ and an integer $m \in \mathbb{N}$, the *low-degree extension* (LDE) of a function $f : H^m \to \mathbb{F}$ is the unique $(|H| - 1)$-individual-degree polynomial $\hat{f} : \mathbb{F}^m \to \mathbb{F}$ such that for all $\boldsymbol{s} \in H^m$, $\hat{f}(\boldsymbol{s}) = f(\boldsymbol{s})$. In this work we focus on the special case when $H = \{0, 1\}$, where $\hat{f}$ is multilinear over $\mathbb{F}^m$, and is called the multilinear extension of f. Abusing the notation, given a string $x \in \mathbb{F}^{2^m}$, $\hat{x} : \mathbb{F}^m \to \mathbb{F}$ denotes the multilinear extension of the function $f_x : \{0, 1\}^m \to \mathbb{F}$ whose function table is x. On a sequence $\boldsymbol{j} = (\boldsymbol{j}_1, \ldots, \boldsymbol{j}_T) \in (\mathbb{F}^m)^T$ of length T, we use the shorthand $\hat{x}(\boldsymbol{j}) \coloneqq (\hat{x}(\boldsymbol{j}_1), \ldots, \hat{x}(\boldsymbol{j}_T)) \in \mathbb{F}^T$.

Consider the following *Polynomial Valuation* (PVAL) set, which is an affine subspace over $\mathbb{F}$, first defined in [RVW13].

Definition 3 (The PVAL set)**.** Let $m, T \in \mathbb{N}$. The set $\mathsf{PVAL}_{\mathbb{F}}(\boldsymbol{j}, \boldsymbol{v}) \subset \mathbb{F}^{2^m}$ is parameterized by the sequences $\boldsymbol{j} = (\boldsymbol{j}_1, \ldots, \boldsymbol{j}_T) \in (\mathbb{F}^m)^T$ and $\boldsymbol{v} = (v_1, \ldots, v_T) \in \mathbb{F}^T$. It consists of all strings $x \in \mathbb{F}^{2^m}$ whose corresponding multilinear extension $\hat{x} : \mathbb{F}^m \to \mathbb{F}$ satisfies $\hat{x}(\boldsymbol{j}) = \boldsymbol{v}$.

Note that $\mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{0})$ is a linear subspace of $\mathbb{F}^{2^m}$, so we can define $\Delta(\mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{0}))$ to be the minimum Hamming distance of a non-zero vector in $\mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{0})$, and the following proposition holds.

Proposition 1 (Random PVAL kernel has large distance)**.** *Let $\sigma, d, m \in \mathbb{N}$. Suppose $\mathbb{F}$ is a field of characteristic 2, $|\mathbb{F}| \geq 2 \cdot m$, and $T \geq 2d(m + \log|\mathbb{F}|) + \sigma$. Then,*

$$\Pr_{\boldsymbol{j} \leftarrow (\mathbb{F}^m)^T}[\Delta(\mathit{PVAL}(\boldsymbol{j}, \boldsymbol{0})) \leq 2d] \leq 2^{-\sigma}.$$

Proof. Fix a non-zero $\boldsymbol{x} \in \mathbb{F}^{2^m}$ with Hamming weight at most $2d$. Since $\hat{\boldsymbol{x}}$ is a multilinear polynomial with total-degree m in $\mathbb{F}[X_1, \ldots, X_m]$, by Lemma 1, the probability that $\hat{x}(\boldsymbol{j}_i) = 0$ is at most $m/|\mathbb{F}|$. As $\boldsymbol{j}$ contains T independent samples, all of these are zero with probability at most $(m/|\mathbb{F}|)^T$. Therefore, $\boldsymbol{x} \in \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{0})$ with probability at most $(m/|\mathbb{F}|)^T$.

There are at most $\binom{2^m}{2d}|\mathbb{F}|^{2d}$ choices for $\boldsymbol{x} \in \mathbb{F}^{2^m}$ with Hamming weight at most $2d$, so by a union-bound and the assumption $T \geq 2d(m + \log|\mathbb{F}|) + \sigma$, the probability that some of them are in $\mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{0})$ is at most

$$\binom{2^m}{2d}|\mathbb{F}|^{2d}\left(\frac{m}{|\mathbb{F}|}\right)^T \leq 2^{2dm} \cdot |\mathbb{F}|^{2d} \cdot 2^{-T} \leq 2^{-\sigma}. \qquad \square$$

[9]Both these assumptions can also be relaxed by further delegating operations over large fields. For simplicity, we do not deal with that here.

3.2 Succinct Descriptions of Sets and Predicates

Definition 4 (Uniform Arithmetic Circuits)**.** Let $\mathbb{F}$ be a field. Let $C = \{C_n : \mathbb{F}^n \to \mathbb{F}^m\}_{n\in\mathbb{N}}$ be a family of arithmetic circuits, consisting of fan-in 2 ADD and MULT gates over a field $\mathbb{F}$. For any $f = f(n)$, we say that C is f-space uniform if there exists a fixed $O(f(n))$-space Turing machine $\mathcal{M}$ that on input 1^n outputs the full description of the circuit C_n. When n is clear from the context, we omit the subscript n and write C instead of C_n.

An important special case is when $f(n) = \log(n)$, in which case we say that C is log-space uniform.

Remark 7. Any f-space uniform Boolean circuit C can be trivially extended to by a f-space uniform arithmetic circuit C' of the same size and depth over any field $\mathbb{F}$.

The following *succinct descriptions* of sets can be used to recover the entire set.

Definition 5 (Descriptions of Sets)**.** A bit string $\langle\mathcal{S}\rangle \in \{0,1\}^B$ is a description of a set $\mathcal{S} = \{s_1, \ldots, s_k\} \subset \{0,1\}^p$ if there exists a (multi-output) p-space uniform $G : [k] \times \{0,1\}^B \to \{0,1\}^p$ of fan-in 2, such that $G(i, \langle\mathcal{S}\rangle) = s_i$ for all $i \in [k]$. The description is succinct if $|\langle\mathcal{S}\rangle| = B < k \cdot p$.

Similarly, *succinct descriptions* of functions can be used to configure a uniform circuit family to implement the function.

Definition 6 (Descriptions of Functions)**.** Let $\mathbb{F}$ be a field. We say that $\langle\Phi\rangle \in \{0,1\}^B$ is a description of a function $\Phi : \mathbb{F}^n \to \mathbb{F}$ if there exists a log-space uniform circuit $C : \mathbb{F}^{n+B} \to \mathbb{F}$ of fan-in 2 such that $C(x, \langle\Phi\rangle) = \Phi(x)$ for all $x \in \mathbb{F}^n$. The description is succinct if $|\langle\Phi\rangle| = B < \mathsf{size}(\Phi)$.

The Turing machines that generate the uniform G and C in Definitions 5 and 6 take in their "shape parameters" $(k, 1^B, 1^p)$ and $(1^n, 1^B)$ as input, respectively.

3.3 Unambiguous Interactive Proof (UIP)

An $(\ell, a, b, \mathsf{Ptime}, \mathsf{Vtime}, \Sigma)$-protocol is a *public-coin, ℓ-round interactive protocol, with alphabet Σ, per-round prover message length a and per-round verifier message length b, and prover runtime* Ptime *and verifier runtime* Vtime. Specifically, such a protocol consists of a pair of interacting Turing machines (P, V) each having as input a string x of length $n = |x|$. The machines may also take in other parameters as additional input, such as parameters controlling the soundness error,[10] but we omit them from the notation for simplicity. The machine P is deterministic and runs in time Ptime, and is called *the prover*, while V is probabilistic, runs in time Vtime, and is called *the verifier*. P and V are the *two parties* of the protocol. We omit the specification of Σ when $\Sigma = \{0,1\}$.

In each round $j \in [\ell]$ of the protocol:

1. V sends a random message $q_j \leftarrow_R \Sigma^b$ to P, referred to as *queries*.
2. P responds with a message $a_j \in \Sigma^a$ determined by the prescribed next-message function, referred to as *answers*, which (abusing notation) is denoted by

$$a_j \coloneqq \mathsf{P}(x, j, (q_1, \ldots, q_j)) \in \Sigma^a.$$

[10] In particular, the prover might have some extra information that helps make it more efficient, for example the UP witnesses in case the language was in UP.

We make the simplifying assumption that V never rejects in the middle of an execution. We abuse notation and denote the verdict circuit of V, that reads the entire transcript and decides if to accept or reject, also by V:

$$\mathsf{V}(x, (q_1, \ldots, q_\ell), (a_1, \ldots, a_\ell)) \in \{0, 1\}.$$

We denote the sequence of verifier random coins by $\boldsymbol{q} \coloneqq (q_1, \ldots, q_\ell)$, and for $j \in [\ell]$ we denote by

$$\boldsymbol{q}_{\leq j} \coloneqq (q_1, \ldots, q_j) \quad \text{and} \quad \boldsymbol{q}_{>j} \coloneqq (q_{j+1}, \ldots, q_\ell).$$

Finally, let $\mathsf{P}(x, \boldsymbol{q}) \coloneqq (a_1, \ldots, a_\ell)$, where $a_j = \mathsf{P}(x, j, \boldsymbol{q}_{\leq j})$ are the prescribed messages.

The *total communication complexity* of the protocol is the number of bits exchanged between the prover and the verifier, i.e. $(a + b)\ell \cdot \log(|\Sigma|)$.

Let $\mathcal{L}$ be a language. We next define the notion of an unambiguous interactive proof of $\mathcal{L}$.

Definition 7 (ϵ-unambiguous $(\ell, a, b, \mathsf{Ptime}, \mathsf{Vtime}, \Sigma)$-$\mathsf{UIP}$)**.** An $(\ell, a, b, \mathsf{Ptime}, \mathsf{Vtime}, \Sigma)$-protocol (P, V) is an *unambiguous interactive proof* (UIP) with an unambiguity error ϵ for a language $\mathcal{L}$, if it satisfies the following completeness and unambiguity conditions.

- **Prescribed Completeness:** For any $x \in \{0, 1\}^n$ and any verifier coins $\boldsymbol{q} = (q_1, \ldots, q_\ell) \in (\Sigma^b)^\ell$, the answers $\boldsymbol{a} = \mathsf{P}(x, \boldsymbol{q}) \in (\Sigma^a)^\ell$ satisfy that $\mathsf{V}(x, \boldsymbol{q}, \boldsymbol{a}) = 1$ iff $x \in \mathcal{L}$.

  Remark 8. Completeness is usually defined only for $x \in \mathcal{L}$. *Prescribed completeness* also requires the prescribed prover to send rejected messages on $x \notin \mathcal{L}$. We assume that when $x \notin L$, P sends the default string $\mathbf{0} \in \Sigma^a$ in every round, and V rejects this transcript.

- **ϵ-Unambiguity**: For any $x \in \{0, 1\}^n$, any $j^* \in [\ell]$, any prefix $\boldsymbol{q}_{\leq j^*} = (q_1, \ldots, q_{j^*})$, and any (computationally unbounded) prover strategy P^* that deviates from P firstly in round $j^* \in [\ell]$, namely

  $$\mathsf{P}^*(x, j^*, \boldsymbol{q}_{\leq j^*}) \neq \mathsf{P}(x, j^*, \boldsymbol{q}_{\leq j^*}) \quad \text{and} \quad \mathsf{P}^*(x, j, \boldsymbol{q}_{\leq j}) = \mathsf{P}(x, j, \boldsymbol{q}_{\leq j}) \text{ for all } j < j^*,$$

  we have that P^* is accepted with probability at most $\epsilon = \epsilon(n)$ over the remaining random coins of V. Namely,

  $$\Pr[\mathsf{V}(x, (\boldsymbol{q}_{\leq j^*}, \boldsymbol{q}_{>j^*}), \boldsymbol{a}) = 1] \leq \epsilon,$$

  where the probability is taken over $\boldsymbol{q}_{>j^*} = (q_{j^*+1}, \ldots, q_\ell)$, and where $\boldsymbol{a} = \mathsf{P}^*(x, (\boldsymbol{q}_{\leq j^*}, \boldsymbol{q}_{>j^*}))$.

  Note that by Remark 8, unambiguity implies standard soundness.

3.3.1 The GKR protocol

We restate the main result from [GKR15] (in the language of PVAL)[11]. We note that unambiguity is proven in [JKKZ21].

Theorem 3 (The GKR protocol)**.** *Let $\mathbb{F}$ be a field, and let $\Phi : \mathbb{F}^n \to \mathbb{F}$ be an arithmetic circuit with addition and multiplication gates of fan-in 2 over $\mathbb{F}$, with description $\langle \Phi \rangle$. Let G be the log-space uniform circuit that outputs $\Phi(x)$ on input $x \in \mathbb{F}^n$ and $\langle \Phi \rangle \in \{0, 1\}^{|\langle \Phi \rangle|}$. Denote the depth and size of G by $D = D(n) \geq \log n$, and $S = S(n) \geq n$.*

[11] In what follows we formulate the GKR protocol as a reduction (as opposed to a proof system). Specifically, at the end of the protocol the verifier either rejects or outputs a PVAL claim.

There exists an $(\ell, a, b, \mathsf{Ptime}, \mathsf{Vtime})$*-protocol* $(\mathsf{P}_{GKR}, \mathsf{V}_{GKR})$ *such that the following holds. The prover and verifier get the description* $\langle \Phi \rangle$*, and the prover additionally gets the input* $x \in \mathbb{F}^n$*. Let* $m = \lceil \log n \rceil$*. At the end, either* V_{GKR} *rejects, or both parties output a point* $\boldsymbol{j} \in \mathbb{F}^m$ *and a value* $v \in \mathbb{F}$*, such that:*

- ***Prescribed Completeness:*** *If both parties follow the protocol then* $\hat{x}(\boldsymbol{j}) = v$ *iff* $\Phi(x) = 1$*.*
- ϵ_{GKR}***-Unambiguity:*** *For any (unbounded) cheating prover strategy* P^* *that deviates from* P_{GKR} *firstly in round* $j^* \in [\ell]$*,*

$$\epsilon_{GKR}(D, \log S, |\mathbb{F}|) \coloneqq \Pr[\mathsf{V}_{GKR} \text{ accepts} \quad \wedge \quad \hat{x}(\boldsymbol{j}) = v] = O\left(\frac{D \log S}{|\mathbb{F}|}\right)$$

  where the probability is taken over the verifier's remaining coins (after round j^**). In other words, with probability at least* $1 - \epsilon_{GKR}(D, \log S, |\mathbb{F}|)$*, either* V_{GKR} *rejects or* $x \notin \mathit{PVAL}(\boldsymbol{j}, v)$*.*

- *Regardless of the prover's strategy,* $\boldsymbol{j}$ *is uniformly distributed in* $\mathbb{F}^m$*, and only depends on the random coins of* V_{GKR}*.*

The complexity of the protocol is as follows (with $\tilde{O}$ *ignoring polylogarithmic factors in* $D, \log S$*):*

- $\ell = O(D \cdot \log S)$*.*
- $a = O(\log |\mathbb{F}|)$*.*
- $b = O(\log |\mathbb{F}|)$*.*
- $\mathsf{Ptime} = \tilde{O}(\mathrm{poly}(S) \cdot \mathrm{polylog}|\mathbb{F}|)$*.*
- $\mathsf{Vtime} = \tilde{O}(D \log S \cdot \log |\mathbb{F}| + |\langle \Phi \rangle| \cdot \log |\mathbb{F}|)$ *(and* V *does not access* x*).*

Furthermore, the verifier's verdict (on whether to output the strings $\boldsymbol{j}, \boldsymbol{v}$ *or reject at the end) has the following complexities (with* $\tilde{O}$ *ignoring polylogarithmic factors in* $D, \log S, \log|\mathbb{F}|$*):*

- $\mathsf{size}(\mathsf{V}) = \tilde{O}(D \log S \cdot \log |\mathbb{F}| + |\langle \Phi \rangle| \cdot \log |\mathbb{F}|)$*.*
- $\mathsf{depth}(\mathsf{V}) = \tilde{O}(1)$*.*

Remark 9 (The role of the circuit description $\langle \Phi \rangle$). After running the original GKR protocol from [GKR15] to the circuit $G : \mathbb{F}^{n+|\langle \Phi \rangle|} \to \mathbb{F}$ and input $(x, \langle \Phi \rangle)$ (which verifies the claim $\Phi(x) = 1$), the verifier obtains $\boldsymbol{j}' \in \mathbb{F}^{\lceil \log(n+|\langle \Phi \rangle|) \rceil}$, $v' \in \mathbb{F}$ and has to verify the claim $\widehat{x \| \langle \Phi \rangle}(\boldsymbol{j}') = v'$ (where $\widehat{x \| \langle \Phi \rangle}$ denotes the LDE of the pair $(x, \langle \Phi \rangle)$ as a string under some appropriate encoding). The protocol in Theorem 3 additionally reduces this to verifying $\hat{x}(\boldsymbol{j}) = v$ for some $\boldsymbol{j}, v$. This is without additional communication as follows: using $\langle \Phi \rangle$, the verifier first computes $v_0 = \widehat{\mathbf{0}^n \| \langle \Phi \rangle}(\boldsymbol{j}')$, and subtracts it from v', which yields the claim $\widehat{x \| \mathbf{0}^{|\langle \Phi \rangle|}}(\boldsymbol{j}') = v' - v_0$. This is equivalent to checking that $\hat{x}(\boldsymbol{j}) = \frac{v' - v_0}{\chi(\boldsymbol{j}')} \in \mathbb{F}$ where $\chi(\boldsymbol{j}')$ is a factor that depends only on $\boldsymbol{j}'$ and computable by the verifier on its own. The post-processing procedure amounts to the additional $O(|\langle \Phi \rangle| \cdot \log |\mathbb{F}|)$ verifier runtime.

The GKR protocol is Δ_c-distance-preserving Let $(\mathsf{P}_{\mathsf{GKR}}, \mathsf{V}_{\mathsf{GKR}})$ be the GKR protocol from Theorem 3. In [RVW13], the authors observed that when $(\mathsf{P}_{\mathsf{GKR}}, \mathsf{V}_{\mathsf{GKR}})$ is parallel-repeated T times for a large enough T, it is Δ_c-distance-preserving in the following sense: suppose the input x is d-far from satisfying Φ, and $\boldsymbol{j} = (\boldsymbol{j}_1, \ldots, \boldsymbol{j}_T)$ and $\boldsymbol{v} = (v_1, \ldots, v_T)$ are the outputs of T parallel repetitions of $(\mathsf{P}_{\mathsf{GKR}}, \mathsf{V}_{\mathsf{GKR}})$, then x is d-far from the set $\mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$. Note that this increases the overall cost of the protocol by a factor of T.

This observation generalizes to Δ_c-distance, captured in the following lemma.

Lemma 2 (GKR is Δ_c-Distance-Preserving)**.** *Let $M, L, d \in \mathbb{N}$ and assume $M = 2^m$ for some $m \in \mathbb{N}$. Let $\mathbb{F}$ be a field of characteristic 2. Let $\Phi : \mathbb{F}^{M \times L} \to \{0,1\}$ be a log-space uniform arithmetic circuit over $\mathbb{F}$, with addition and multiplication gates of fan-in 2, of size S and depth D, and with description $\langle \Phi \rangle$.*

Let $\boldsymbol{a} \in \mathbb{F}^{M \times L}$. Denote by $\mathcal{S} \subset \mathbb{F}^{M \times L}$ the set of matrices accepted by Φ, and suppose that

$$|\mathcal{S} \cap \mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a})| \leq 1,$$

i.e. at most one element in $\mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a})$ satisfies the circuit. (See Definition 1 for the definition of $\mathcal{B}_{d,\mathbb{F}}$.)

When applying $(\mathsf{P}_{\mathsf{GKR}}, \mathsf{V}_{\mathsf{GKR}})$ to the claim $\Phi(\boldsymbol{a}) = 1$[12] with $T \in \mathbb{N}$ parallel repetitions, we obtain $\boldsymbol{j} \in (\mathbb{F}^{m+\log L})^T, \boldsymbol{v} \in \mathbb{F}^T$, with the following guarantee (in addition to the ones stated in Theorem 3, with all complexities but the round complexity multiplied by T).

Unambiguous Distance Preservation: *For every prover P^*, the distance $\Delta_c(\boldsymbol{a}, \mathcal{S})$ is preserved by the interaction $(\mathsf{P}^*, \mathsf{V}_{\mathsf{GKR}})$ in the following sense:*

* *If $\Delta_c(\boldsymbol{a}, \mathcal{S}) \leq d$ (i.e. $\mathcal{S} \cap \mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a}) \neq \varnothing$) and P^* answers according to the prescribed strategy $\mathsf{P}_{\mathsf{GKR}}$ corresponding to the unique $\boldsymbol{a}^* \in \mathcal{S} \cap \mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a})$, then the output $(\boldsymbol{j}, \boldsymbol{v})$ satisfies*

$$\begin{aligned}&\Pr[\Delta_c(\boldsymbol{a}, \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})) \leq d \text{ and } \boldsymbol{a}^* \text{ remains the unique element in } \mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a}) \cap \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})] \\ &\geq 1 - (\epsilon_{\mathsf{GKR}}(D, \log S, |\mathbb{F}|))^T \cdot \left(\binom{M}{d} |\mathbb{F}|^d \right)^L.\end{aligned}$$

* *On the other hand, if $\Delta_c(\boldsymbol{a}, \mathcal{S}) > d$ (i.e. $\mathcal{S} \cap \mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a}) = \varnothing$) or P^* does not answer according to $\mathsf{P}_{\mathsf{GKR}}$ corresponding to the unique $\boldsymbol{a}^* \in \mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a})$, then the output $(\boldsymbol{j}, \boldsymbol{v})$ satisfies*

$$\begin{aligned}&\Pr[\Delta_c(\boldsymbol{a}, \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})) > d] \\ &\geq 1 - \left(\epsilon_{\mathsf{GKR}}(D, \log S, |\mathbb{F}|) + (\epsilon_{\mathsf{GKR}}(D, \log S, |\mathbb{F}|))^T \cdot \left(\binom{M}{d} |\mathbb{F}|^d \right)^L \right).\end{aligned}$$

Here $\epsilon_{\mathsf{GKR}}(D, \log S, |\mathbb{F}|) = O(\frac{D \log S}{|\mathbb{F}|})$ is the soundness error of one GKR protocol.

Proof. Fix $\boldsymbol{a}' \in \mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a})$, and consider the following two cases:

Case 1 $\Phi(\boldsymbol{a}') = 0$.

The claim $\Phi(\boldsymbol{a}') = 1$ is false. For any prover strategy P^*, by GKR's standard soundness amplification, $\Pr[\boldsymbol{a}' \in \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})] \leq (\epsilon_{\mathsf{GKR}}(D, \log S, |\mathbb{F}|))^T$.

[12] Recall that both $\mathsf{P}_{\mathsf{GKR}}$ and $\mathsf{V}_{\mathsf{GKR}}$ get the description $\langle \Phi \rangle$. Additionally, $\mathsf{P}_{\mathsf{GKR}}$ gets the input $\boldsymbol{a}$.

Case 2 $\Phi(\boldsymbol{a}') = 1$.

If the prover P^* deviates from the prescribed prover P with respect to $\boldsymbol{a}^*$ in some round j^*, then it must be deviating in one of the T parallel executions of GKR. By the unambiguity of GKR[13], $\Pr[\boldsymbol{a}' \in \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})] \leq \epsilon_{\mathsf{GKR}}(D, \log S, |\mathbb{F}|)$.

Crucially, note that there are at most $(\binom{M}{d}|\mathbb{F}|^d)^L$ points $\boldsymbol{a}'$ in $\mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a})$. Let E denote the event that $\exists \boldsymbol{a}' \in \mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a})$ s.t. $\Phi(\boldsymbol{a}') = 0$ and $\boldsymbol{a}' \in \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$. By Case 1 and a union bound,

$$\Pr[E] \leq \epsilon' \coloneqq \epsilon_{\mathsf{GKR}}(D, \log S, |\mathbb{F}|)^T \cdot \left(\binom{M}{d}|\mathbb{F}|^d\right)^L.$$

Using this bound on E, we have

* When $\mathcal{S} \cap \mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a}) = \{\boldsymbol{a}^*\}$ and P^* answers according to the prescribed strategy $\mathsf{P}_{\mathsf{GKR}}$ corresponding to $\boldsymbol{a}^*$:

  In this case, any other $\boldsymbol{a}' \in \mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a})$ satisfies $\Phi(\boldsymbol{a}') = 0$. Additionally, by the prescribed completeness of the GKR protocol, $\boldsymbol{a}^* \in \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$. Therefore,

$$\begin{aligned}
&\Pr\begin{bmatrix}\Delta_c(\boldsymbol{a}, \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})) \leq d \text{ and} \\ \boldsymbol{a}^* \text{ remains the unique element} \\ \text{in } \mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a}) \cap \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})\end{bmatrix} \\
&\geq \Pr\left[\boldsymbol{a} \in \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v}) \text{ and } \overline{E}\right] \\
&= 1 - \epsilon'.
\end{aligned}$$

* When $\mathcal{S} \cap \mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a}) = \{\boldsymbol{a}^*\}$ but P^* does not answer according to $\mathsf{P}_{\mathsf{GKR}}$ corresponding to $\boldsymbol{a}^*$:

  By Case 2, the probability that $\boldsymbol{a}^* \in \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$ is at most $\epsilon_{\mathsf{GKR}}(D, \log S, |\mathbb{F}|)$. Therefore,

$$\begin{aligned}
&\Pr[\mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a}) \cap \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v}) = \varnothing] \\
&\geq 1 - (\Pr[\boldsymbol{a}^* \in \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})] + \Pr[E]) \\
&\geq 1 - \left(\epsilon_{\mathsf{GKR}}(D, \log S, |\mathbb{F}|) + \epsilon'\right).
\end{aligned}$$

* When $\mathcal{S} \cap \mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a}) = \varnothing$, no matter what P^* does,

$$\begin{aligned}
\Pr[\mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a}) \cap \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v}) = \varnothing] &\geq 1 - \Pr[E] \\
&\geq 1 - \epsilon'.
\end{aligned}$$

□

3.4 Almost d-wise independent permutations

We begin by defining *almost d-wise independent permutations*:

Definition 8 (Almost d-wise Independent Permutations)**.** A distribution Π on S_N[14] is η-almost d-wise independent if for all $1 \leq x_1 < x_2 < \ldots < x_d \leq N$, the distribution $(\pi(x_1), \ldots, \pi(x_d))_{\pi \sim \Pi}$ is at most η far from the uniform distribution on distinct d-tuples over $[N]$ in TV distance.

[13] Note that P^* does not necessarily deviate in all T executions, so the error term is not necessarily amplified.
[14] The symmetric group on N elements i.e. the set of all permutations

We will be using a family of almost $d-$wise independent permutations on $\{0,1\}^m$ in Section 6, which has a *pairwise independence* property and has succinct descriptions. The properties we will desire from our permutation family are:

- Almost d-wise independence.
- $\pi \sim \Pi$ have succinct descriptions, short seeds, and can be efficiently implemented.
- π^{-1} for $\pi \sim \Pi$ have succinct descriptions, short seeds, and can be efficiently implemented.

The question of sampling from almost d-wise independent permutations has been well studied [Gow96, HMMR04, BH08, KNR05, GHKO25, GHP25]. From this line of work, we get the following result:

Theorem 4 (Theorem 2 in [GHP25])**.** *For any m, and $d \leq 2^{m/50}$, a random reversible circuit with $\tilde{O}(dm \cdot \log(1/\eta))$ width-2 gates computes an η-almost d-wise independent permutation where $\tilde{O}$ hides $\mathrm{polylog}(d,m)$ factors.*

Observe that since the sampled circuit is a random reversible circuit, it is always a permutation, the inverse is easy to compute, and the seed is just the set of gates which can also be described with $\tilde{O}(nk \cdot \log(1/\eta))$ bits and the permutation is also efficient to compute.

Definition 9 ($\Pi_{m,d,\eta}$)**.** We let $\Pi_{m,d,\eta}$ be the distribution from Theorem 4.

Definition 10 (Uniform distributions)**.** Let $\mathcal{U}_{m,d}$ be the uniform distribution of distinct $d-$tuples over $[2^m]$. Let $\mathcal{U}'_{m,d}$ be the uniform distribution of $d-$tuples over $[2^m]$.

Theorem 5. *Let $A_0, A_1 \subseteq \{0,1\}^m$ such that $|A_0|, |A_1| \leq d$ such that $d \leq 2^{m/50}$. Then,*

$$\Pr_{\pi \sim \Pi_{m,d,\eta}}[|A_0 \cap \pi(A_1)| \geq \epsilon d] \leq \eta + \exp(-\epsilon d/6)$$

if $\epsilon d \geq 2$.

Proof. Observe that since the indicator random variable for $|A_0 \cap \pi(A_1)| \geq \epsilon d$ is Bernoulli, and the TV distance, we have that

$$\Pr_{\pi \sim \Pi_{m,d,\eta}}[|A_0 \cap \pi(A_1)| \geq \epsilon d] \leq \eta + \Pr_{\pi \sim \mathcal{U}_{m,d}}[|A_0 \cap \pi(A_1)| \geq 1 + \epsilon d].$$

For any $a \in A_1$, let X_a, X'_a be the indicator random variables for $\pi(a) \in A_0$ where $\pi \sim \mathcal{U}_{m,d}$ and $\mathcal{U}'_{m,d}$ respectively. Additionally, let $X = \sum X_a$ and $X' = \sum X'_a$. Observe that for any $A \subset A_0$,

$$\mathbb{E}[\Pi_{a \in A} X_a] = \Pr[\forall a \in A, X_a = 1] = \prod_{j=0}^{|A|-1} \frac{|A_0| - j}{2^m - j} \leq \left(\frac{|A_0|}{2^m}\right)^{|A|} \leq \mathbb{E}[\Pi_{a \in A} X'_a].$$

Thus, for any integral $c \geq 0$, we have that $\mathbb{E}[X^c] \geq \mathbb{E}[X'^c]$. Now, since we get Chernoff bound by applying Markov Inequality on the moment generating function, we must have that

$$\Pr[X \geq \mu(1 + \frac{\epsilon d}{\mu} - 1)] \leq e^{(1-\epsilon d) \cdot \frac{\frac{\epsilon d}{\mu} - 1}{1 + \frac{\epsilon d}{\mu}}} \leq e^{-\epsilon d/6}$$

where $\mu = \mathbb{E}[X] = \frac{|A_0| \cdot |A_1|}{2^m} \leq \frac{d^2}{2^m} \leq 1$. □

3.5 Unambiguous Interactive Proofs of Proximity (IPP)

Defined in [EKR04, RVW13], Interactive Proofs of Proximity (IPP) for the PVAL problem are a key ingredient in several prior works (for example, [RVW13, RRR18, RR20]). They are also used in our protocols. Here, we define a slight variant of the standard IPP with the added unambiguity property.

Definition 11 ([EKR04, RVW13] (Unambiguous) Interactive Proof of Proximity)**.** Let $d \in \mathbb{N}$. A pair of interacting Turing machines (P, V), where P is deterministic and takes as input a string $x \in \{0,1\}^n$ and V has oracle access to x, is an *unambiguous interactive proof of proximity with distance d* (d-IPP) for a language $\mathcal{L}$, under distance dist, with perfect completeness and unambiguity error $\epsilon = \epsilon(n)$ if:

- **Prescribed Completeness:** For all $x \in \{0,1\}^n$, the verifier accepts in the interaction $(\mathsf{P}(x), \mathsf{V}^x(1^n))$ iff $\mathsf{dist}(x, \mathcal{L}) < d$.
- **ϵ-Unambiguity:** For all $x \in \{0,1\}^n$, and all provers P^* that deviates from P in some round j^*,
$$\Pr[\mathsf{V} \text{ accepts the interaction with } \mathsf{P}^*] \leq \epsilon(n),$$
where the probability is taken over the verifier's random coins after round j^*.

The query complexity $q(n)$ is the number of queries the verifier makes to x, the communication and round complexity of the protocol are as usual.

4 Our Batch-UIP

Let $k \in \mathbb{N}$. For a language $\mathcal{L}$, let its batched version $\mathcal{L}^{\otimes k}$ be

$$\mathcal{L}^{\otimes k} \coloneqq \{(x_1, \ldots, x_k) : \forall i \in [k], x_i \in \mathcal{L} \text{ and } |x_1| = \ldots = |x_k|\}.$$

If $\mathcal{L}$ has a UIP (Definition 7), then $\mathcal{L}^{\otimes k}$ has a UIP obtained by running the underlying UIP for $\mathcal{L}$ for each of the k instances in parallel. This introduces a factor of k overhead. Our main result is a better batching protocol. For simplicity, assume the alphabet is $\Sigma = \{0,1\}$.

Theorem 6 (Formal version of Theorem 1)**.** *Let $\mathcal{L}$ be a language with a (public-coin) ϵ-unambiguous $(\ell, a, b, \mathsf{Ptime}, \mathsf{Vtime})$-UIP (P, V), where the verifier verdict V is a log-space uniform boolean circuit with size S and depth D. We also make the simplifying assumptions that the verifier never rejects in the middle of the UIP, and that $b \leq a$.*

Let $\boldsymbol{x} = (x_1, \ldots, x_k) \in (\{0,1\}^n)^k$ be a batch of statements in $\mathcal{L}^{\otimes k}$ to be verified, let $\sigma = \sigma(k) \in \mathbb{N}$ be an unambiguity parameter. Then there exists an ϵ'-unambiguous $(\ell', a', b', \mathsf{Ptime}', \mathsf{Vtime}', \{0,1\})$-UIP for the language $\mathcal{L}^{\otimes k}$, denoted by $(\mathsf{P}', \mathsf{V}')$ (Protocol 1), with the following properties, where $\tilde{O}$ hides $\mathrm{poly}(\sigma) \cdot \mathrm{polylog}(k, n, a, \ell)$ factors:

- *The unambiguity error is $\epsilon' = \tilde{O}(\epsilon + 2^{-\sigma})$.*
- *The round complexity is $\ell' = \tilde{O}(\ell \cdot D \log S)$.*
- *The prover and verifier message length per round is $a' = b' = \tilde{O}(a\ell + \mathrm{poly}(\ell))$.*

- *The prover runtime is* $\mathsf{Ptime}' = \tilde{O}(k \cdot \ell \cdot \mathsf{Ptime} + \mathrm{poly}(k, a, \ell, S))$.
- *The verifier runtime is* $\mathsf{Vtime}' = \tilde{O}(\ell \cdot \mathsf{Vtime} + a\ell^2(kn + D \log S) + a^2 \cdot \mathrm{poly}(\ell))$.

Furthermore, the verifier's verdict circuit is log-space uniform and has the following properties:

- $\mathsf{size}(\mathsf{V}') = \tilde{O}(\ell \cdot S + a\ell^2(kn + D \log S) + a^2 \cdot \mathrm{poly}(\ell))$.
- $\mathsf{depth}(\mathsf{V}') = D + \tilde{O}(1)$.

The unambiguity parameter σ controls the unambiguity error ϵ', which we show to be concretely upper-bounded by $2 \log k \cdot (\epsilon + 2^{-\sigma})$ in Section 4.2. Increasing σ imposes an overhead factor of $O(2^\sigma)$ in the prover's message length and the prover's and verifier's runtime.

If $D \log S = \mathrm{polylog}(k, n)$, then Theorem 1 follows by choosing $\sigma = \log \frac{1}{\epsilon}$. With $\tilde{O}$ hiding $\mathrm{polylog}(\sigma^{-1}, k, n, a, \ell)$ factors,

- $\epsilon' = O(\epsilon \log k)$.[15]
- $\ell' = \tilde{O}(\ell)$,
- $a' = b' = \tilde{O}(a\ell)$,
- $\mathsf{Ptime}' = \tilde{O}(k \cdot \ell \cdot \mathsf{Ptime} + \mathrm{poly}(k, a, \ell, S))$,
- $\mathsf{Vtime}' = \tilde{O}(\ell \cdot \mathsf{Vtime} + kna\ell^2 + a^2 \cdot \mathrm{poly}(\ell))$. Note that Theorem 1 uses the looser bound $kna\mathrm{poly}(\ell)$ for the second and third term.

4.1 Our Ingredients

Our Construction Our Batch-UIP protocol consists of two main ingredients: a *Distance Generation Protocol* and an *Instance Reduction Protocol*. It iteratively applies these two protocols, where in each iteration, the *Distance Generation Protocol* $(\mathsf{P}_{\mathsf{Dist}}, \mathsf{V}_{\mathsf{Dist}})$ is applied, followed by the application of *Instance Reduction Protocol* $(\mathsf{P}_{\mathsf{Reduce}}, \mathsf{V}_{\mathsf{Reduce}})$.

The Distance Generation Protocol (Lemma 3) ensures that if the k-fold input instance was not in the language, then after running it, the prover and verifier obtain a related instance that is far from the language. While the Reduction Protocol (Lemma 4) has the guarantee that as long as the given k-fold input instance is far from the language, it produces a significantly smaller k'-fold instance not in the language (but without any distance guarantees). Therefore, as long as the overall shrinkage is significant, say larger than a factor of 2, the parties can repeat the two sub-protocols $\rho = O(\log k)$ times, and end up having roughly $k \cdot 2^{-\rho} = \mathrm{polylog}(k)$ instances that can be explicitly checked.

In the initial stage, the verifier samples a random $\boldsymbol{q} \leftarrow \{0,1\}^{b\ell}$, and sends the entire $\boldsymbol{q}$ to the prover. This $\boldsymbol{q}$, along with the statements $(x_1, \ldots, x_k)$, implicitly define a set of k "prescribed answers": For each i, denote by $\boldsymbol{a}^{x_i,\boldsymbol{q}} = \mathsf{P}(x_i, \boldsymbol{q}) \in \{0,1\}^{a\ell}$ the answers the base UIP's prescribed prover P would send, on input statement x_i upon receiving $\boldsymbol{q}$ from the verifier.

We denote by $\Phi^{(0)}$ the boolean circuit that has $\boldsymbol{q}$ and $(x_1, \ldots, x_k)$ hardwired. It takes the input $\boldsymbol{a}^{\boldsymbol{q}} := \{\boldsymbol{a}^{x_i,\boldsymbol{q}}\}_{i \in [k]}$ and outputs 1 iff $\mathsf{V}(x_i, \boldsymbol{q}, \boldsymbol{a}^{x_i,\boldsymbol{q}}) = 1$ for all $i \in [k]$. We treat $\boldsymbol{a}^{\boldsymbol{q}}$ as a matrix in $\{0,1\}^{k \times (a\ell)}$ whose i'th row is $\boldsymbol{a}^{x_i,\boldsymbol{q}}$, and call $\boldsymbol{a}^{\boldsymbol{q}}$ the *prescribed matrix* corresponding to $\boldsymbol{q}$.

[15] We require $\epsilon = O(1/\log k)$ for the unambiguity error to be non-trivially small.

Let $\mathcal{L}'_{\boldsymbol{x}}$ be the *associated language* that consists of all instances $(\mathcal{S}, \Phi)$, where

$$\mathcal{S} = \left\{(i^1, \boldsymbol{q}^1), \ldots, (i^g, \boldsymbol{q}^g)\right\}$$

is a set of g pairs in $[k] \times \{0,1\}^{b\ell}$, such that the prescribed matrix $\boldsymbol{a}^{\mathcal{S}} \in \{0,1\}^{g \times (a\ell)}$, consisting of rows $\left\{\boldsymbol{a}^{x_{i^j}, \boldsymbol{q}^j}\right\}_{j \in [g]}$ where $\boldsymbol{a}^{x_{i^j}, \boldsymbol{q}^j} = \mathsf{P}(x_{i^j}, \boldsymbol{q}^j)$, satisfies the constraint Φ. Note that the language $\mathcal{L}'_{\boldsymbol{x}}$ is not necessarily in NP, but as we shall see in Section 4.5.2, there is a UIP for explicitly checking it. We mention that the prescribed matrix $\boldsymbol{a}^{\mathcal{S}}$ replaces the role of the implicit vector of unique witnesses in the case of $\mathsf{Batch\text{-}UP}$.

It follows from the prescribed completeness of the base UIP that for $\mathcal{S}^{(0)} \coloneqq \{(i, \boldsymbol{q})\}_{i \in [k]}$,

$$(\mathcal{S}^{(0)}, \Phi^{(0)}) \in \mathcal{L}'_{\boldsymbol{x}} \text{ iff } \boldsymbol{x} \in \mathcal{L}^{\otimes k}.$$

In each iteration, the following two sub-protocols are run:

1. The Distance Generation Protocol is applied to $(\mathcal{S}^{(0)}, \Phi^{(0)}) \in \mathcal{L}'_{\boldsymbol{x}}$ with distance parameter $d = \ell \cdot \mathrm{polylog}(k, \ell) \in \mathbb{N}$. At the end of this protocol, the verifier obtains a set $\mathcal{S}_{\mathsf{Dist}}$ containing $M = |\mathcal{S}^{(0)}| \cdot \ell$ pairs, where initially $|\mathcal{S}^{(0)}| = |\{(i, \boldsymbol{q})\}_{i \in k}| = k$, as well as a predicate Φ_{Dist} that is evaluated over the prescribed matrix $\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}$. The guarantee of our Distance Generation Protocol is that when $(\mathcal{S}^{(0)}, \Phi^{(0)}) \notin \mathcal{L}'_{\boldsymbol{x}}$, the prescribed matrix $\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}$ is Δ_c-d-far (Definition 1) from satisfying the predicate Φ_{Dist}.

2. Then, the Instance Reduction protocol with shrinkage parameter $d = \ell \cdot \mathrm{polylog}(k, \ell) \in \mathbb{N}$ transforms the claim $(\mathcal{S}_{\mathsf{Dist}}, \Phi_{\mathsf{Dist}}) \in \mathcal{L}'_{\boldsymbol{x}}$ to a new claim $(\mathcal{S}^{(1)}, \Phi^{(1)}) \in \mathcal{L}'_{\boldsymbol{x}}$, where $\mathcal{S}^{(1)}$ is a subset of $\mathcal{S}_{\mathsf{Dist}}$ with size $M' \leq M/d$. The Instance Reduction Protocol guarantees that $(\mathcal{S}^{(1)}, \Phi^{(1)}) \notin \mathcal{L}'_{\boldsymbol{x}}$ if the prescribed matrix $\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}$ is Δ_c-d-far (Definition 1) from satisfying Φ_{Dist}. Importantly, the verifier never needs to explicitly access the prescribed matrix $\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}$.

We set the distance/shrinkage parameter to be $d > 2\ell$ to offset the intermediate factor of ℓ. In other words,

$$|\mathcal{S}_{\mathsf{Dist}}| \leq \ell \cdot \left|\mathcal{S}^{(0)}\right| \quad \text{and} \quad \left|\mathcal{S}^{(1)}\right| \leq \frac{|\mathcal{S}_{\mathsf{Dist}}|}{d} < \frac{\left|\mathcal{S}^{(0)}\right|}{2},$$

and thus each iteration shrinks the number of instances at least by a factor of 2, which implies that the parties need to iterate at most $\rho = \lfloor \log k \rfloor$ times, until they can check the final claim $(\mathcal{S}^{(\rho)}, \Phi^{(\rho)}) \in \mathcal{L}'_{\boldsymbol{x}}$, defined over $\left|\mathcal{S}^{(\rho)}\right| \leq k/2^{\rho}$ pairs. The final check can be done by explicitly running the UIP for verifying $(\mathcal{S}^{(\rho)}, \Phi^{(\rho)}) \in \mathcal{L}'_{\boldsymbol{x}}$ (as formally proven in Lemma 5 below).

The associated language $\mathcal{L}'_{\boldsymbol{x}}$ We formally set up the notations to define the language $\mathcal{L}'_{\boldsymbol{x}}$.

Definition 12 (Prescribed transcript relation $\mathcal{R}$)**.** Let $\mathcal{R}$ be the binary relation containing pairs $((x, \boldsymbol{q}), \boldsymbol{a}) \in (\{0,1\}^n \times \{0,1\}^{b\ell}) \times \{0,1\}^{a\ell}$, such that

- $\mathsf{V}(x, \boldsymbol{q}, \boldsymbol{a}) = 1$, and
- $\boldsymbol{a}$ is the *prescribed prover transcript*, i.e. $\boldsymbol{a} = \mathsf{P}(x, \boldsymbol{q}) \in \{0,1\}^{a\ell}$ is the prescribed prover messages $(\boldsymbol{a}_1^{x,\boldsymbol{q}}, \ldots, \boldsymbol{a}_{\ell}^{x,\boldsymbol{q}})$ sent by P on input x when interacting with V that uses $\boldsymbol{q}$ as its random coins.[16]

[16] Recall that if $x \notin \mathcal{L}$, $\boldsymbol{a}^{x,\boldsymbol{q}} = \mathsf{P}(x, \boldsymbol{q})$ is the default all-zero string, and $\mathsf{V}(x, \boldsymbol{q}, \boldsymbol{a}^{x,\boldsymbol{q}}) = 0$.

Notations (Prescribed prover messages $\boldsymbol{a}^{x,\boldsymbol{q}}$, $\boldsymbol{a}_j^{x,\boldsymbol{q}}$, and matrices $\boldsymbol{a}^{\mathcal{S}}, \boldsymbol{a}_j^{\mathcal{S}}$) For any $x \in \{0,1\}^n$ and any $\boldsymbol{q} \in (\{0,1\}^b)^\ell$, the shorthand $\boldsymbol{a}^{x,\boldsymbol{q}} \in \{0,1\}^{a\ell}$ stands for the (unique) prescribed prover messages corresponding to $(x, \boldsymbol{q})$. Namely, $\boldsymbol{a}^{x,\boldsymbol{q}} := \mathsf{P}(x, \boldsymbol{q})$. Similarly, for every $j \in [\ell]$, the shorthand $\boldsymbol{a}_j^{x,\boldsymbol{q}}$ means the j-th round prover message in $\boldsymbol{a}^{x,\boldsymbol{q}} := (\boldsymbol{a}_1^{x,\boldsymbol{q}}, \ldots, \boldsymbol{a}_\ell^{x,\boldsymbol{q}})$. More generally, given a batch of statements $\boldsymbol{x} = (x_1, \ldots, x_k) \in (\{0,1\}^n)^k$, and a subset of g pairs $\mathcal{S} \subset [k] \times (\{0,1\}^b)^\ell$, we denote by $\boldsymbol{a}^{\mathcal{S}} := \left((\boldsymbol{a}^{x_i,\boldsymbol{q}})_{(i,\boldsymbol{q})\in\mathcal{S}}\right)^\top$ the matrix whose rows are $\boldsymbol{a}^{x_i,\boldsymbol{q}} \in \{0,1\}^{a\ell}$. For every $j \in [\ell]$, $\boldsymbol{a}_j^{\mathcal{S}} := \left((\boldsymbol{a}_j^{x_i,\boldsymbol{q}})_{(i,\boldsymbol{q})\in\mathcal{S}}\right)^\top$ is the matrix whose rows are $\boldsymbol{a}_j^{x_i,\boldsymbol{q}} \in \{0,1\}^a$, the j-th round prescribed message in $\boldsymbol{a}^{x_i,\boldsymbol{q}}$.

Note that $\mathcal{R}$ is not necessary a UP (or even NP) relation, but for every $x \in \mathcal{L}$ and every $\boldsymbol{q} \in \{0,1\}^{b\ell}$, there is a single $\boldsymbol{a}$ that satisfies $((x, \boldsymbol{q}), \boldsymbol{a}) \in \mathcal{R}$. As we shall see in Proposition 2, when $x \in \mathcal{L}$, there exists a UIP for certifying that $\boldsymbol{a}$ is indeed the (unique) $\boldsymbol{a}^{x,\boldsymbol{q}}$ for which $((x, \boldsymbol{q}), \boldsymbol{a}) \in \mathcal{R}$. Therefore, in this sense, we can treat $\boldsymbol{a}^{x,\boldsymbol{q}}$ as the unique witness for the relation $((x, \boldsymbol{q}), \boldsymbol{a}) \in \mathcal{R}$.

We now formally define the associated language $\mathcal{L}'_{\boldsymbol{x}}$. For $(\mathcal{S}, \Phi) \in \mathcal{L}'_{\boldsymbol{x}}$ to hold, not only should all remaining x_i's be in $\mathcal{L}$, a joint constraint Φ should also be satisfied on the prescribed matrix $\boldsymbol{a}^{\mathcal{S}}$ associated with $\mathcal{S}$.

Definition 13 (The associated language $\mathcal{L}'_{\boldsymbol{x}}$)**.** Let $\boldsymbol{x} \in (\{0,1\}^n)^k$, $\mathcal{S} \subset [k] \times \{0,1\}^{b\ell}$ be a set of $g := |\mathcal{S}|$ pairs, and Φ be a boolean circuit that takes $\boldsymbol{a}^{\mathcal{S}}$ as input. Let $\mathcal{I} := \{i : (i, \boldsymbol{q}) \in \mathcal{S}\}$. Then

$$(\mathcal{S}, \Phi) \in \mathcal{L}'_{\boldsymbol{x}} \quad \text{iff} \quad \Phi(\boldsymbol{a}^{\mathcal{S}}) = 1 \quad \wedge \quad \boldsymbol{x}|_{\mathcal{I}} \in \mathcal{L}^{\otimes|\mathcal{I}|}.$$

4.1.1 The Distance Generation Protocol

We give the full specification of $(\mathsf{P}_{\mathsf{Dist}}, \mathsf{V}_{\mathsf{Dist}})$ and its analysis in Section 4.3, which has the guarantees stated in Lemma 3 below.

The protocol takes as input $\langle \mathcal{S} \rangle, \langle \Phi \rangle$, an unambiguity parameter $\sigma \in \mathbb{N}$, a distance parameter $d \in \mathbb{N}$ and a field parameter $\mathbb{F}$ with $|\mathbb{F}| \geq 2^\sigma \cdot \mathrm{polylog}(k, n, a, \ell)$, and outputs $\langle \mathcal{S}_{\mathsf{Dist}} \rangle, \langle \Phi_{\mathsf{Dist}} \rangle$ such that $(\mathcal{S}_{\mathsf{Dist}}, \Phi_{\mathsf{Dist}}) \in \mathcal{L}'_{\boldsymbol{x}}$ iff $(\mathcal{S}, \Phi) \in \mathcal{L}'_{\boldsymbol{x}}$. It has a slightly increased instance size $|\mathcal{S}_{\mathsf{Dist}}| = |\mathcal{S}| \cdot \ell$, but has the guarantee that if $\Phi(\boldsymbol{a}^{\mathcal{S}}) = 0$ then $\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}$ is Δ_c-d-far from satisfying the circuit Φ_{Dist},[17] where $\boldsymbol{a}^{\mathcal{S}}$ and $\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}$ are the prescribed matrices corresponding to $\mathcal{S}$ and $\mathcal{S}_{\mathsf{Dist}}$, respectively.

Lemma 3 (Δ_c-Distance Generation Protocol)**.** *Let $\mathcal{L}$ be a language with an ϵ-unambiguous $(\ell, a, b, \mathsf{Ptime}, \mathsf{Vtime})$-UIP (P, V). Let S and D be the size and depth of the decision circuit V.*

There exists a protocol $(\mathsf{P}_{Dist}, \mathsf{V}_{Dist})$ (Protocol 2) that takes as input $\boldsymbol{x}, \langle \mathcal{S} \rangle, \langle \Phi \rangle$, where $\boldsymbol{x} \in (\{0,1\}^n)^k$, $\mathcal{S} \subset [k] \times \{0,1\}^{b\ell}$, and $\Phi : \{0,1\}^{|\mathcal{S}| \times a\ell} \to \{0,1\}$, an unambiguity parameter $\sigma \in \mathbb{N}$, a distance parameter $d \in \mathbb{N}$, a field $\mathbb{F}$ with $|\mathbb{F}| \geq 2^\sigma \cdot \mathrm{polylog}(k, n, a, \ell)$, and outputs $\boldsymbol{x}, \langle \mathcal{S}_{Dist} \rangle, \langle \Phi_{Dist} \rangle$ such that $|\mathcal{S}_{Dist}| = |\mathcal{S}| \cdot \ell$, and the following holds.

- ***Prescribed Completeness:***

  In an honest execution, $(\mathcal{S}_{Dist}, \Phi_{Dist}) \in \mathcal{L}'_{\boldsymbol{x}}$ iff $(\mathcal{S}, \Phi) \in \mathcal{L}'_{\boldsymbol{x}}$.

- $(\epsilon + 2^{-\sigma})$***-Unambiguous Generation of Δ_c-Distance:*** *For every cheating prover P^* that deviates from P_{Dist} first in round j^*, conditioned on the first j^* rounds of the protocol,*

$$\Pr\left[\exists \boldsymbol{a}^* \in \mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a}^{\mathcal{S}_{Dist}}) \text{ s.t. } \Phi_{Dist}(\boldsymbol{a}^*) = 1\right] \leq \epsilon + 2^{-\sigma},$$

[17] Recall that $\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}$ is Δ_c-d-far from satisfying Φ_{Dist}, if for every $\boldsymbol{a}^* \in \mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}})$, it holds that $\Phi_{\mathsf{Dist}}(\boldsymbol{a}^*) = 0$.

where $\langle \mathcal{S}_{Dist} \rangle, \langle \Phi_{Dist} \rangle$ *denotes the output of the interaction between* V_{Dist} *and* P^**, and where the probability is over the remaining coins of* V_{Dist}*.*

Furthermore, for any (honest or cheating) prover P^**,* $\boldsymbol{a}^{\mathcal{S}_{Dist}}$ *is likely the only input in* $\mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a}^{\mathcal{S}_{Dist}})$ *that satisfies* Φ_{Dist}*. Formally,*

$$\Pr\left[\exists \boldsymbol{a}^* \in \mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a}^{\mathcal{S}_{Dist}}) : \ \boldsymbol{a}^* \neq \boldsymbol{a}^{\mathcal{S}_{Dist}} \ \wedge \ \Phi_{Dist}(\boldsymbol{a}^*) = 1\right] \leq \epsilon + 2^{-\sigma},$$

where the probability is over the random coins of V_{Dist}*.*

Let $\tilde{O}$ *be hiding* $\mathrm{polylog}(|\mathbb{F}|, k, n, a, \ell)$ *factors. Denote by* $|\mathcal{S}| = g$*, then*

- $\ell_{Dist} = O(\ell)$.
- $a_{Dist} = \tilde{O}(da)$.
- $b_{Dist} = \tilde{O}(\max(b, d)) = \tilde{O}(da)$*, with the simplifying assumption that* $b \leq a$*.*
- $\mathsf{Ptime}_{Dist} = \tilde{O}(g\ell \cdot \mathsf{Ptime} + dga\ell^2 + |\langle \Phi \rangle| + |\langle \mathcal{S} \rangle| + k \cdot n)$.
- $\mathsf{Vtime}_{Dist} = \tilde{O}(da\ell + |\langle \Phi \rangle| + |\langle \mathcal{S} \rangle| + k \cdot n)$.

The bit lengths of the descriptions are $|\langle \mathcal{S}_{Dist} \rangle| = \tilde{O}(|\langle \mathcal{S} \rangle| + a\ell)$, $|\langle \Phi_{Dist} \rangle| = \tilde{O}(|\langle \Phi \rangle| + |\langle \mathcal{S} \rangle| + da\ell + k \cdot n)$*. Let* $G(i, \langle \mathcal{S} \rangle)$ *be the circuit that returns the* i*-th element in* $\mathcal{S}$*, and* $C(x, \langle \Phi \rangle)$ *be the circuit that returns* $\Phi(x)$*, then the new circuits* $G_{Dist}(i, \langle \mathcal{S}_{Dist} \rangle)$ *and* $C_{Dist}(x, \langle \Phi_{Dist} \rangle)$*, satisfy the following.*

- $\mathsf{size}(G_{Dist}) = \mathsf{size}(G) + b\ell + \tilde{O}(1)$.
- $\mathsf{depth}(G_{Dist}) = \mathsf{depth}(G) + \tilde{O}(1)$.
- $\mathsf{size}(C_{Dist}) = \tilde{O}(g\ell \cdot \mathsf{size}(G) + \mathsf{size}(C) + g \cdot S + gda\ell^2)$.
- $\mathsf{depth}(C_{Dist}) \leq \max(\mathsf{depth}(C), \mathsf{depth}(G) + D) + \tilde{O}(1)$.

The verifier never rejects this protocol, i.e. V_{Dist} *always outputs 1.*

4.1.2 The Instance Reduction Protocol

The full specification of $(\mathsf{P}_{\mathsf{Reduce}}, \mathsf{V}_{\mathsf{Reduce}})$ and its analysis are in Section 4.4. It has the following properties.

The protocol takes as input $\langle \mathcal{S}_{\mathsf{Dist}} \rangle, \langle \Phi_{\mathsf{Dist}} \rangle$, as well the parameters $\sigma, d \in \mathbb{N}$ and the field $\mathbb{F}$ (i.e. the same parameters as $(\mathsf{P}_{\mathsf{Dist}}, \mathsf{V}_{\mathsf{Dist}})$), and outputs $\langle \mathcal{S}' \rangle, \langle \Phi' \rangle$ such that $|\mathcal{S}'| \approx \frac{|\mathcal{S}_{\mathsf{Dist}}|}{d}$, with the guarantee that $(\mathcal{S}', \Phi') \in \mathcal{L}'_{\boldsymbol{x}}$ only if $\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}$ is Δ_c-d-close to satisfying Φ_{Dist}. Importantly, $\mathsf{V}_{\mathsf{Reduce}}$ never accesses $\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}$ explicitly during this protocol.

Lemma 4 (Instance Reduction Protocol for Δ_c-Distance)**.** *Let* $\mathcal{L}$ *be a language with an* ϵ*-unambiguous* $(\ell, a, b, \mathsf{Ptime}, \mathsf{Vtime})$*-UIP* (P, V)*, where the verifier's decision* V *is a log-space uniform boolean circuit of size* S *and depth* D*.*

There exists a constant C*, and a protocol* $(\mathsf{P}_{Reduce}, \mathsf{V}_{Reduce})$ *(Protocol 3) that takes as input* $\boldsymbol{x}, \langle \mathcal{S}_{Dist} \rangle, \langle \Phi_{Dist} \rangle$*, where* $\boldsymbol{x} \in (\{0,1\}^n)^k$*,* $\mathcal{S}_{Dist} \subset [k] \times \{0,1\}^{b\ell}$*,* $\Phi_{Dist} : \{0,1\}^{|\mathcal{S}_{Dist}| \times a\ell} \to \{0,1\}$*, an unambiguity parameter* $\sigma \in \mathbb{N}$*, a distance parameter* $d \in \mathbb{N}$*, a field* $\mathbb{F}$ *that satisfies* $|\mathbb{F}| \geq 32 \cdot 2^{\sigma}(da\ell m)^C$*, and outputs* $\langle \mathcal{S}' \rangle, \langle \Phi' \rangle$*, with* $|\mathcal{S}'| = \tilde{O}(\frac{|\mathcal{S}_{Dist}|}{d})$ *(where* $\tilde{O}$ *hides* $\mathrm{polylog}(|\mathbb{F}|, k, n, a, \ell)$ *factors). Let* $M := |\mathcal{S}_{Dist}| \leq k \cdot \ell$*, and suppose* $M = 2^m$ *for* $m \in \mathbb{N}$*. If additionally, the technical condition* $d \geq 16m\sigma$ *holds, then the protocol has the following guarantees.*

- ***Prescribed Completeness:***
  If $\mathsf{V}_{\mathit{Reduce}}$ *interacts with* $\mathsf{P}_{\mathit{Reduce}}$*, then* $(\mathcal{S}', \Phi') \in \mathcal{L}'_{\boldsymbol{x}}$ *iff* $(\mathcal{S}_{\mathit{Dist}}, \Phi_{\mathit{Dist}}) \in \mathcal{L}'_{\boldsymbol{x}}$.

- $2^{-\sigma}$***-Unambiguous Distance Preservation:*** *Suppose for every* $\boldsymbol{a}^* \in \mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a}^{\mathcal{S}_{\mathit{Dist}}}) \setminus \{\boldsymbol{a}^{\mathcal{S}_{\mathit{Dist}}}\}$, $\Phi_{\mathit{Dist}}(\boldsymbol{a}^*) = 0$*, then for every cheating prover* P^* *that deviates from* $\mathsf{P}_{\mathit{Reduce}}$ *first in round* j^**, let the output of the interaction between* $\mathsf{V}_{\mathit{Reduce}}$ *and* P^* *be* $\langle \mathcal{S}' \rangle, \langle \Phi' \rangle$*, then*

$$\Pr[\mathsf{V}_{\mathit{Reduce}} \textit{ accepts } \wedge \ \Phi'(\boldsymbol{a}^{\mathcal{S}'}) = 1] \leq 2^{-\sigma},$$

  where the probability is over $\mathsf{V}_{\mathit{Reduce}}$*'s remaining coins.*

The protocol has the following complexities.

- $\ell_{\mathit{Reduce}} = \tilde{O}(\ell \cdot \mathsf{depth}(C_{\mathit{Dist}}) \log \mathsf{size}(C_{\mathit{Dist}}))$.
- $a_{\mathit{Reduce}} = b_{\mathit{Reduce}} = \tilde{O}(da + \mathrm{poly}(d))$.
- $\mathsf{Ptime}_{\mathit{Reduce}} = \mathrm{poly}(\log |\mathbb{F}|, k, a, \ell, d, \mathsf{size}(C_{\mathit{Dist}}), |\langle \mathcal{S}_{\mathit{Dist}} \rangle|, |\langle \Phi_{\mathit{Dist}} \rangle|, n)$.
- $\mathsf{Vtime}_{\mathit{Reduce}} = \tilde{O}(da\ell \cdot (\mathsf{depth}(C_{\mathit{Dist}}) \log \mathsf{size}(C_{\mathit{Dist}}) + |\langle \Phi_{\mathit{Dist}} \rangle|) + \mathrm{poly}(d) + |\langle \mathcal{S}_{\mathit{Dist}} \rangle| + k \cdot n)$.

The bit lengths of the descriptions are $|\langle \mathcal{S}' \rangle| = \tilde{O}(|\langle \mathcal{S}_{\mathit{Dist}} \rangle| + \mathrm{poly}(d))$, $|\langle \Phi' \rangle| = \tilde{O}(da\ell + \mathrm{poly}(d))$. *Furthermore, suppose:*

- $G_{\mathit{Dist}}(i, \langle \mathcal{S}_{\mathit{Dist}} \rangle)$ *is the circuit that returns the* i*-th element in* $\mathcal{S}_{\mathit{Dist}}$*, and*
- $C_{\mathit{Dist}}(x, \langle \Phi_{\mathit{Dist}} \rangle)$ *is the circuit that returns* $\Phi_{\mathit{Dist}}(x)$.

The new circuits $G'(i, \langle \mathcal{S}' \rangle)$ *and* $C'(x, \langle \Phi' \rangle)$ *satisfy the following.*

- $\mathsf{size}(G') = \mathsf{size}(G_{\mathit{Dist}}) + \tilde{O}(\mathrm{poly}(d))$.
- $\mathsf{depth}(G') = \mathsf{depth}(G_{\mathit{Dist}}) + \tilde{O}(1)$.
- $\mathsf{size}(C') = \tilde{O}(\frac{M}{d} \cdot a\ell)$.
- $\mathsf{depth}(C') = \tilde{O}(1)$.

And the verifier's verdict circuit (which outputs 0 iff it rejects amidst the protocol) satisfies:

- $\mathsf{size}(\mathsf{V}_{\mathit{Reduce}}) = \tilde{O}(da\ell \cdot (\mathsf{depth}(C_{\mathit{Dist}}) \log \mathsf{size}(C_{\mathit{Dist}}) + |\langle \Phi_{\mathit{Dist}} \rangle|) + \mathrm{poly}(d) + |\langle \mathcal{S}_{\mathit{Dist}} \rangle| + k \cdot n)$.
- $\mathsf{depth}(\mathsf{V}_{\mathit{Reduce}}) = \tilde{O}(1)$.

The number of instances is concretely $|\mathcal{S}'| \leq \left\lceil 8\sigma \frac{|\mathcal{S}_{\mathrm{Dist}}|}{d} \right\rceil$.

4.1.3 The Explicit UIP for the associated language

We give the full specification of the UIP $(\mathsf{P}_{\mathcal{L}'_{\boldsymbol{x}}}, \mathsf{V}_{\mathcal{L}'_{\boldsymbol{x}}})$ that explicitly checks $(\mathcal{S}, \Phi) \in \mathcal{L}'_{\boldsymbol{x}}$ in Section 4.5.2. This protocol has a cost proportional to $|\mathcal{S}|$. Therefore, it is only run at the end of the Batch-UIP protocol, when $|\mathcal{S}|$ is small.

Lemma 5 (The UIP for $\mathcal{L}'_{\boldsymbol{x}}$)**.** *Let $\mathcal{L}$ be a language with an ϵ-unambiguous $(\ell, a, b, \mathsf{Ptime}, \mathsf{Vtime})$-UIP (P, V), where the verifier verdict V is a log-space uniform boolean circuit of size S and depth D.*

There exists an ϵ-unambiguous $(\ell', a', b', \mathsf{Ptime}', \mathsf{Vtime}')$-UIP $(\mathsf{P}_{\mathcal{L}'_{\boldsymbol{x}}}, \mathsf{V}_{\mathcal{L}'_{\boldsymbol{x}}})$ for verifying $(\mathcal{S}, \Phi) \in \mathcal{L}'_{\boldsymbol{x}}$. Both parties have input $\boldsymbol{x}$ and $\langle \mathcal{S} \rangle, \langle \Phi \rangle$, where $|\mathcal{S}| = g$, and in the end $\mathsf{V}_{\mathcal{L}'_{\boldsymbol{x}}}$ either accepts or rejects.

Let $G(i, \langle \mathcal{S} \rangle)$ be the circuit that returns the i-th element in $\mathcal{S}$, and $C(x, \langle \Phi \rangle)$ be the circuit that returns $\Phi(x)$. The protocol has the following complexities.

- $\ell_{\mathcal{L}'_{\boldsymbol{x}}} = \ell + 1$.
- $a_{\mathcal{L}'_{\boldsymbol{x}}} = g\ell \cdot a$.
- $b_{\mathcal{L}'_{\boldsymbol{x}}} = b$.
- $\mathsf{Ptime}_{\mathcal{L}'_{\boldsymbol{x}}} = O(g\ell \cdot \mathsf{Ptime})$.
- $\mathsf{Vtime}_{\mathcal{L}'_{\boldsymbol{x}}} = O(g\ell \cdot \mathsf{Vtime} + \mathsf{size}(C) + g\mathsf{size}(G))$.

Let $\tilde{O}$ hide $\mathrm{polylog}(g, \ell)$ *factors. The verifier's decision circuit $\mathsf{V}_{\mathcal{L}'_{\boldsymbol{x}}}$ has the following complexities:*

- $\mathsf{size}(\mathsf{V}_{\mathcal{L}'_{\boldsymbol{x}}}) = \tilde{O}(g\ell \cdot S + \mathsf{size}(C) + g\mathsf{size}(G))$.
- $\mathsf{depth}(\mathsf{V}_{\mathcal{L}'_{\boldsymbol{x}}}) = \max(\mathsf{depth}(G) + D, \mathsf{depth}(C)) + \tilde{O}(1)$.

4.2 Our Construction of Batch-UIP

We show how to prove Theorem 6 assuming Lemmas 3 to 5.

Proof of Theorem 6. Denote the sub-protocols as $(\mathsf{P}_{\mathsf{Dist}}, \mathsf{V}_{\mathsf{Dist}})$, $(\mathsf{P}_{\mathsf{Reduce}}, \mathsf{V}_{\mathsf{Reduce}})$, and $(\mathsf{P}_{\mathcal{L}'_{\boldsymbol{x}}}, \mathsf{V}_{\mathcal{L}'_{\boldsymbol{x}}})$. The full batching protocol $(\mathsf{P}', \mathsf{V}')$ is given in Protocol 1.

Prescribed Completeness. Note that $(\mathcal{S}^{(0)}, \Phi^{(0)}) \in \mathcal{L}'_{\boldsymbol{x}}$ iff $\Phi^{(0)}(\boldsymbol{a}^{\mathcal{S}^{(0)}}) = 1$ and $\boldsymbol{x} \in \mathcal{L}^{\otimes k}$. The former condition is true iff $\mathsf{V}(x_i, \boldsymbol{q}, \boldsymbol{a}^{x_i, \boldsymbol{q}}) = 1$ for every $(i, \boldsymbol{q}) \in \mathcal{S}^{(0)}$, which is true iff $\boldsymbol{x} \in \mathcal{L}^{\otimes k}$ because of the prescribed completeness of (P, V). By invoking the prescribed completeness of $(\mathsf{P}_{\mathsf{Dist}}, \mathsf{V}_{\mathsf{Dist}})$ and $(\mathsf{P}_{\mathsf{Reduce}}, \mathsf{V}_{\mathsf{Reduce}})$ and $(\mathsf{P}_{\mathcal{L}'_{\boldsymbol{x}}}, \mathsf{V}_{\mathcal{L}'_{\boldsymbol{x}}})$, we get that V' accepts iff $\boldsymbol{x} \in \mathcal{L}^{\otimes k}$.

Unambiguity. Let P^* be a prover strategy that deviates from P' in some round, then it must be deviating in one of the ρ applications of $(\mathsf{P}_{\mathsf{Dist}}, \mathsf{V}_{\mathsf{Dist}})$, one of the ρ applications of $(\mathsf{P}_{\mathsf{Reduce}}, \mathsf{V}_{\mathsf{Reduce}})$, or in the final $(\mathsf{P}_{\mathcal{L}'_{\boldsymbol{x}}}, \mathsf{V}_{\mathcal{L}'_{\boldsymbol{x}}})$. The probability that V' accepts is upper bounded by the probability that unambiguity breaks in any of the subsequent rounds, which, by a union bound, is at most $\epsilon' = \rho \cdot (\epsilon + 2^{-\sigma} + 2^{-\sigma}) + \epsilon < (\rho + 1)\epsilon + 2 \cdot \rho 2^{-\rho} < 2 \log k \cdot (\epsilon + 2^{-\sigma})$.

Protocol 1 Our Batch-UIP $(\mathsf{P}', \mathsf{V}')$.

Input Parameters: $\sigma \in \mathbb{N}$.

Input: $k \in \mathbb{N}$, and $\boldsymbol{x} = (x_1, \ldots, x_k)$ are statements.

Other Parameters: $d = \lceil 16 \cdot \log(k\ell) \cdot \sigma \rceil \cdot \ell$ is the distance parameter, $\mathbb{F}$ is a characteristic-2 finite field of size $|\mathbb{F}| \geq 32 \cdot 2^{\sigma} \cdot (16 \log(k) \cdot \sigma a \ell m)^{C_0}$, (where C_0 is some universal constant), and $\rho = \lfloor \log k \rfloor$ is the number of iterations.

Ingredients:

- The *Distance Generation Protocol*, $(\mathsf{P}_{\mathsf{Dist}}, \mathsf{V}_{\mathsf{Dist}})$ (Lemma 3).

  It takes in a claim $\langle \mathcal{S} \rangle, \langle \Phi \rangle$ over $|\mathcal{S}|$ active pairs and output an intermediate claim $\langle \mathcal{S}_{\mathsf{Dist}} \rangle, \langle \Phi_{\mathsf{Dist}} \rangle$ over $|\mathcal{S}_{\mathsf{Dist}}| = M = g\ell$ active pairs.

- The *Instance Reduction Protocol*, $(\mathsf{P}_{\mathsf{Reduce}}, \mathsf{V}_{\mathsf{Reduce}})$ (Lemma 4).

  It takes in the claim $\langle \mathcal{S}_{\mathsf{Dist}} \rangle, \langle \Phi_{\mathsf{Dist}} \rangle$ over M active pairs and output a reduced claim $\langle \mathcal{S}' \rangle, \langle \Phi' \rangle$ over $|\mathcal{S}'| \leq M/d$ active pairs.

- The UIP $(\mathsf{P}', \mathsf{V}')$ for explicitly checking the final claim $(\mathcal{S}^{(\rho)}, \Phi^{(\rho)}) \in \mathcal{L}'_{\boldsymbol{x}}$ (Lemma 5).

1: V' samples a random $\boldsymbol{q} \leftarrow \{0,1\}^{b\ell}$, and send it entirely to P'.

2: Both parties let $\mathcal{S}^{(0)} \leftarrow \{(i, \boldsymbol{q})\}_{i \in [k]}$, and $\Phi^{(0)}$ be the predicate that checks every row $\boldsymbol{a}^{x_i, \boldsymbol{q}}$ in $\boldsymbol{a}^{\mathcal{S}^{(0)}}$ are accepted, i.e. $\mathsf{V}(x_i, \boldsymbol{q}, \boldsymbol{a}^{x_i, \boldsymbol{q}}) = 1$ for every $i \in [k]$.

3: Define $\langle \mathcal{S}^{(0)} \rangle = (\boldsymbol{q})$, and $G^{(0)}(\langle \mathcal{S}^{(0)} \rangle)$ which enumerates $\mathcal{S}^{(0)}$ is the circuit that outputs $(i, \boldsymbol{q})$ for every $i \in [k]$.

4: Define $\langle \Phi^{(0)} \rangle = (\boldsymbol{x}, \boldsymbol{q})$, and $C^{(0)}(x, \langle \Phi^{(0)} \rangle)$ be the circuit that checks $\Phi^{(0)}(x) = 1$.

5: **for** $j = 1, \ldots, \rho$ **do**

6: Run $(\mathsf{P}_{\mathsf{Dist}}, \mathsf{V}_{\mathsf{Dist}})$ on $\langle \mathcal{S}^{(j-1)} \rangle, \langle \Phi^{(j-1)} \rangle$ with parameters σ, d, and $\mathbb{F}$, obtaining $\langle {\mathcal{S}_{\mathsf{Dist}}}^{(j-1)} \rangle$, $\langle {\Phi_{\mathsf{Dist}}}^{(j-1)} \rangle$.

7: Run $(\mathsf{P}_{\mathsf{Reduce}}, \mathsf{V}_{\mathsf{Reduce}})$ on $\langle {\mathcal{S}_{\mathsf{Dist}}}^{(j-1)} \rangle$ and $\langle {\Phi_{\mathsf{Dist}}}^{(j-1)} \rangle$ with parameters σ, d, and $\mathbb{F}$, obtaining $\langle \mathcal{S}^{(j)} \rangle, \langle \Phi^{(j)} \rangle$.

8: Run the UIP $(\mathsf{P}_{\mathcal{L}'_{\boldsymbol{x}}}, \mathsf{V}_{\mathcal{L}'_{\boldsymbol{x}}})$ explicitly on $\boldsymbol{x}$ and $\langle \mathcal{S}^{(\rho)} \rangle, \langle \Phi^{(\rho)} \rangle$ to check $(\mathcal{S}^{(\rho)}, \Phi^{(\rho)}) \in \mathcal{L}'_{\boldsymbol{x}}$.

9: V' accepts iff $\mathsf{V}_{\mathcal{L}'_{\boldsymbol{x}}}$ accepts.

Number of Intermediate Instances We show by induction that $\left|\mathcal{S}^{(j)}\right| \leq \lceil k \cdot 2^{-j} \rceil$, for $j \in [\rho]$. For the base case, $\left|\mathcal{S}^{(0)}\right| = k$ by definition. Assuming the inductive hypothesis, by the guarantees of $(\mathsf{P}_{\mathsf{Dist}}, \mathsf{V}_{\mathsf{Dist}})$, $\left|{\mathcal{S}_{\mathsf{Dist}}}^{(j)}\right| = \left|\mathcal{S}^{(j)}\right| \cdot \ell \leq \lceil k \cdot 2^{-j} \rceil \cdot \ell$. We conclude by the guarantees of $(\mathsf{P}_{\mathsf{Reduce}}, \mathsf{V}_{\mathsf{Reduce}})$, $\left|\mathcal{S}^{(j+1)}\right| \leq \left\lceil 8\sigma \left|{\mathcal{S}_{\mathsf{Dist}}}^{(j)}\right| / d \right\rceil \leq \frac{\lceil 8\sigma \cdot k \cdot 2^{-j} \cdot \ell \rceil}{\lceil 16 \cdot \log(k\ell) \cdot \sigma \rceil \cdot \ell} < \lceil k \cdot 2^{-j-1} \rceil$.

Description Lengths We first bound the bit lengths of all descriptions.
With $\tilde{O}$ hiding $\mathrm{polylog}(|\mathbb{F}|, k, n, a, \ell)$ factors, for all $1 \leq j \leq \rho < \log k = \tilde{O}(1)$, and

- $$\begin{aligned}\left|\langle \mathcal{S}^{(j)} \rangle\right| &= \tilde{O}(\left|\langle {\mathcal{S}_{\mathsf{Dist}}}^{(j-1)} \rangle\right| + \mathrm{poly}(d)) \\ &= \tilde{O}(\left|\langle \mathcal{S} \rangle^{(j-1)}\right| + a\ell + \mathrm{poly}(d)) \\ &= \ldots = \tilde{O}(j \cdot (a\ell + \mathrm{poly}(d))) = \tilde{O}(a\ell + \mathrm{poly}(d)), \text{ because } j = \tilde{O}(1).\end{aligned}$$
- $\left|\langle \Phi^{(j)} \rangle\right| = \tilde{O}(a\ell + \mathrm{poly}(d))$.
- $\left|\langle {\Phi_{\mathsf{Dist}}}^{(j)} \rangle\right| = \tilde{O}(\left|\langle \Phi \rangle^{(j)}\right| + \left|\langle \mathcal{S}^{(j)} \rangle\right| + da\ell + k \cdot n) = \tilde{O}(da\ell + \mathrm{poly}(d) + k \cdot n)$.

Note that $\left|\langle \Phi^{(0)} \rangle\right| = \tilde{O}(k \cdot n + a\ell)$, because $\langle \Phi^{(0)} \rangle = (\boldsymbol{x}, \boldsymbol{q})$.

Intermediate Circuit Sizes and Depths For all $1 \leq j \leq \rho \leq \log k = \tilde{O}(1)$, let $G^{(j)}(s, \langle \mathcal{S}^{(j)} \rangle)$ be the circuit that returns the s-th element in $\mathcal{S}^{(j)}$, and let $C^{(j)}(x, \langle \Phi^{(j)} \rangle)$ be the circuit that returns $\Phi^{(j)}(x)$. For the base case, we already have $\mathsf{size}(G^{(0)}) = \tilde{O}(k + a\ell)$ and $\mathsf{depth}(G^{(0)}) = O(1)$, and $\mathsf{size}(C^{(0)}) = \tilde{O}(k \cdot S)$ and $\mathsf{depth}(C^{(0)}) = D + \tilde{O}(1)$. By the guarantees of Lemmas 3 and 4, we calculate the following:

- $\mathsf{size}(G^{(j)}) = \mathsf{size}(G^{(j-1)}_{\mathsf{Dist}}) + \tilde{O}(\mathrm{poly}(d)) = \mathsf{size}(G^{(j-1)}) + \tilde{O}(a\ell + \mathrm{poly}(d)) = \ldots = \tilde{O}(k + a\ell + \mathrm{poly}(d))$.
- $\mathsf{depth}(G^{(j)}) = \mathsf{depth}(G^{(j-1)}_{\mathsf{Dist}}) + \tilde{O}(1) = \ldots = \tilde{O}(1)$.
- $\mathsf{size}(C^{(j)}) = \tilde{O}(\left|\mathcal{S}^{(j)}\right| \cdot a\ell)$.
- $\mathsf{depth}(C^{(j)}) = \tilde{O}(1)$.
- $$\begin{aligned}\mathsf{size}(C^{(j)}_{\mathsf{Dist}}) &= \tilde{O}(\left|\mathcal{S}^{(j)}\right| \cdot \ell \cdot \mathsf{size}(G^{(j)}) + \mathsf{size}(C^{(j)}) + \left|\mathcal{S}^{(j)}\right| \cdot S + gda\ell^2) \\ &= \tilde{O}(\left|\mathcal{S}^{(j)}\right| \cdot \ell \cdot (k + a\ell + \mathrm{poly}(d)) + \left|\mathcal{S}^{(j)}\right| \cdot (a\ell + S) + gda\ell^2) \\ &= \tilde{O}(k^2\ell + ka\ell^2 + ka\ell + k \cdot \ell \cdot \mathrm{poly}(d) + kS + gda\ell^2).\end{aligned}$$
- $\mathsf{depth}(C^{(j)}_{\mathsf{Dist}}) = \max(\mathsf{depth}(C^{(j-1)}), \mathsf{depth}(G^{(j-1)}) + D) + \tilde{O}(1) = \ldots = D + \tilde{O}(1)$.

Therefore, $\mathsf{depth}(C^{(j)}_{\mathsf{Dist}}) \log(\mathsf{size}(C^{(j)}_{\mathsf{Dist}}) = (D + \tilde{O}(1))(\log S + \tilde{O}(1)) = D \log S + \tilde{O}(1)$.

The technical conditions stated in Lemma 4 are satisfied by the choice of parameters:

1. Since we choose $|\mathbb{F}|$ to be at least $32 \cdot 2^{\sigma} \cdot (16 \log(k) \cdot \sigma a\ell m)^{C_0}$, it satisfies the field size requirement of $|\mathbb{F}| \geq 32 \cdot 2^{\sigma} (da\ell m)^{C}$ as long as $C_0 \geq 2C$, where C is the constant in Lemma 4.

2. Fix $j \in [\rho]$, by the above analysis, $\left|\mathcal{S}_{\mathsf{Dist}}{}^{(j-1)}\right| = \left|\mathcal{S}^{(j-1)}\right| \cdot \ell \leq \lceil k \cdot 2^{1-j} \cdot \ell \rceil$. To apply $(\mathsf{P}_{\mathsf{Reduce}}, \mathsf{V}_{\mathsf{Reduce}})$, the condition $d \geq 16m\sigma$ must hold, where $m \coloneqq \log\left|\mathcal{S}_{\mathsf{Dist}}{}^{(j-1)}\right| \leq \log(k\ell)$. Indeed, $d = \lceil 16 \cdot \log(k\ell) \cdot \sigma \rceil \cdot \ell > 16m\sigma$.

We rely on $d = \lceil 16 \cdot \log(k\ell) \cdot \sigma \rceil \cdot \ell = \tilde{O}(\ell)$ and $\rho \leq \log k = \tilde{O}(1)$ to simplify the complexities as follows.

The overall cost equals the cost of running the two sub-protocols Dist and Reduce for ρ times, plus the cost of running the final UIP $(\mathsf{P}_{\mathcal{L}'_{\boldsymbol{x}}}, \mathsf{V}_{\mathcal{L}'_{\boldsymbol{x}}})$ once. Note that the size of the final $\left|\mathcal{S}^{(\rho)}\right|$ is at most $k \cdot 2^{-\rho} = \tilde{O}(1)$. We use notations such as $\mathsf{Ptime}_{\mathsf{Dist},j}$ to denote the time complexity of running Dist in the j-th round (and similarly for Reduce and $\mathcal{L}'_{\boldsymbol{x}}$).

- $\ell' = \sum_{j=1}^{\rho}(\ell_{\mathsf{Dist},j} + \ell_{\mathsf{Reduce},j}) + \ell_{\mathcal{L}'_{\boldsymbol{x}}}) = \tilde{O}(\rho \cdot \ell D \log S) = \tilde{O}(\ell D \log S)$.
- $a' = b' = \max(a_{\mathsf{Dist}}, a_{\mathsf{Reduce}}, a_{\mathcal{L}'_{\boldsymbol{x}}}) = \tilde{O}(\max(da, da + \mathrm{poly}(d), \left|\mathcal{S}^{(\rho)}\right|\ell \cdot a)) = \tilde{O}(a\ell + \mathrm{poly}(\ell))$.
- $$\begin{aligned}\mathsf{Ptime}' &= \tilde{O}(\sum_{j=1}^{\rho}((k \cdot 2^{1-j}) \cdot \ell \cdot \mathsf{Ptime} + d(k \cdot 2^{1-j})a\ell^2 + \left|\langle \Phi^{(j)} \rangle\right| + |\langle \mathcal{S}_{\mathsf{Dist}} \rangle| + k \cdot n)) \\ &\quad + \tilde{O}(\sum_{j=1}^{\rho} \mathrm{poly}(d, k \cdot 2^{1-j}, a, \ell, \mathsf{size}(C_{\mathsf{Dist}}), \left|\langle \Phi^{(j)} \rangle\right|, |\langle \mathcal{S}_{\mathsf{Dist}} \rangle|), n) \\ &\quad + \tilde{O}(\ell \cdot \mathsf{Ptime}) \\ &= \tilde{O}(k \cdot \ell \cdot \mathsf{Ptime} + \mathrm{poly}(k, a, \ell, S)).\end{aligned}$$
- $$\begin{aligned}\mathsf{Vtime}' &= \sum_{j=1}^{\rho} \mathsf{Vtime}_{\mathsf{Dist},j} + \sum_{j=1}^{\rho} \mathsf{Vtime}_{\mathsf{Reduce},j} + \mathsf{Vtime}_{\mathcal{L}'_{\boldsymbol{x}}} \\ &= \tilde{O}(\rho \cdot da\ell + \left|\langle \mathcal{S}^{(j)} \rangle\right| + \left|\langle \Phi^{(j)} \rangle\right| + k \cdot n) \\ &\quad + \tilde{O}\left(\sum_{j=0}^{\rho-1}(da\ell(\mathsf{depth}(C_{\mathsf{Dist}}) \log \mathsf{size}(C_{\mathsf{Dist}}) + \left|\langle \Phi_{\mathsf{Dist}}{}^{(j)} \rangle\right|) + \mathrm{poly}(d) + \left|\langle \mathcal{S}_{\mathsf{Dist}}{}^{(j)} \rangle\right| \right. \\ &\qquad \left. + k \cdot n\right) \\ &\quad + \tilde{O}(\left|\mathcal{S}^{(\rho)}\right|\ell \cdot \mathsf{Vtime} + \mathsf{size}(\Phi^{(\rho)}) + \left|\mathcal{S}^{(\rho)}\right|\mathsf{size}(G^{(\rho)})) \\ &= \tilde{O}\left(\rho da\ell + k \cdot n + \rho da\ell(D \log S + da\ell + \mathrm{poly}(d) + k \cdot n) + \ell\mathsf{Vtime} + k + a\ell\right) \\ &= \tilde{O}(\ell \cdot \mathsf{Vtime} + a\ell^2(kn + D \log S) + a^2 \cdot \mathrm{poly}(\ell)).\end{aligned}$$

Finally, the verifier's verdict circuit V' rejects iff any of the sub-protocols rejects. (Note that $\mathsf{V}_{\mathsf{Dist}}$ never rejects, so there are no terms corresponding to these sub-protocols.)

- $$\begin{aligned}\mathsf{size}(\mathsf{V}') &\leq \sum_{j=1}^{\rho} \mathsf{size}(\mathsf{V}_{\mathsf{Reduce},j}) + \mathsf{size}(\mathsf{V}_{\mathcal{L}'_{\boldsymbol{x}}}) \\ &= \tilde{O}((\ell \cdot S + a\ell^2(kn + D \log S)) + a^2 \cdot \mathrm{poly}(\ell)).\end{aligned}$$

- $\mathsf{depth}(\mathsf{V}') = \max(\{\mathsf{depth}(\mathsf{V}_{\mathsf{Reduce},j})\}_{j\in[\rho]}, \mathsf{depth}(\mathsf{V}_{\mathcal{L}'_{\boldsymbol{x}}})) + O(\log \rho)$
  $= \max(\{\tilde{O}(1)\}_{j\in[\rho]}, \max(\mathsf{depth}(G^{(\rho)}) + D, \mathsf{depth}(C^{(\rho)}))) + O(\log \rho)$
  $= D + \tilde{O}(1).$

And moreover, V' is log-space uniform because V' is simply the AND's of the ρ $\mathsf{V}_{\mathsf{Reduce}}$ and the final $\mathsf{V}_{\mathcal{L}'_{\boldsymbol{x}}}$, and all of these are themselves log-space uniform. □

4.3 Proof of Lemma 3: Generating Δ_c-Distance via Checksums.

4.3.1 Additional Ingredients

A helper UIP for verifying $\boldsymbol{a} = \boldsymbol{a}^{x,\boldsymbol{q}}$ Recall that $\boldsymbol{a}^{x,\boldsymbol{q}}$ is the prescribed transcript of (P,V) on the statement x when verifier uses random coin $\boldsymbol{q}$ (Definition 12). There exists a UIP, which we call a *Transcript Checker* protocol, that allows a verifier to certify that some transcript $\boldsymbol{a} \in \{0,1\}^{a\ell}$ indeed equals $\boldsymbol{a}^{x,\boldsymbol{q}}$.

The *Random Continuation Protocol* is such a Transcript Checker. In this protocol, the verifier samples a *fresh* random coin sequence $\boldsymbol{q}' = (q'_1, \ldots, q'_\ell) \in (\{0,1\}^b)^\ell$, and challenges the prover to succeed in answering ℓ hybrid runs of (P,V) on the input x_i, where the j-th hybrid run uses the hybrid coin sequence $\boldsymbol{q}_j^{\mathsf{Hyb}} := (\boldsymbol{q}_{\le j}, \boldsymbol{q}'_{>j})$. The proof of Proposition 2 is straightforward and deferred to Section 4.5.1.

Proposition 2 (Transaction Checker; c.f. *Random Continuation* in Lemma 6.3 of [RRR16])**.** *Let $\mathcal{L}$ be a language with an ϵ-unambiguous $(\ell, a, b, \mathsf{Ptime}, \mathsf{Vtime})$-UIP (P,V). There exists an ϵ-unambiguous $(\ell, \ell\cdot a, b, \ell\cdot\mathsf{Ptime}, \ell\cdot\mathsf{Vtime})$-UIP $(\mathsf{P}^{tc},\mathsf{V}^{tc})$, where both parties take $((x,\boldsymbol{q}),\boldsymbol{a})$ as input, and the verifier decides whether to accept or reject. We call $(\mathsf{P}^{tc},\mathsf{V}^{tc})$ the* Transcript Checker*, and it satisfies the following properties:*

- *As a UIP for checking $((x,\boldsymbol{q}),\boldsymbol{a}) \in \mathcal{R}$, it satisfies prescribed completeness and ϵ-unambiguity.*
- ***Transcript Structure.*** *The messages of P^{tc} can be exactly broken down into ℓ prescribed messages of the base UIP (P,V). Namely, the r-th round message of P^{tc} on input $((x,\boldsymbol{q}),\boldsymbol{a})$ is $\boldsymbol{m}_r = \left((\boldsymbol{a}_r^{x,\boldsymbol{q}_j^{\mathsf{Hyb}}})_{j\in[\ell]}\right)^\top$, where $\boldsymbol{a}_r^{x,\boldsymbol{q}_j^{\mathsf{Hyb}}} = (\mathsf{P}(x,(\boldsymbol{q}_j^{\mathsf{Hyb}})_{<r}))$ is the r-th round message of P. Therefore, the entire transcript of P^{tc} equals $(\boldsymbol{m}_1,\ldots,\boldsymbol{m}_\ell) = \left((\boldsymbol{a}^{x,\boldsymbol{q}_j^{\mathsf{Hyb}}})_{j\in[\ell]}\right)^\top$.*
- ***Check Structure.*** *The verifier V^{tc} on a given transcript $\boldsymbol{a} = \left((\boldsymbol{a}^{x,\boldsymbol{q}_j^{\mathsf{Hyb}}})_{j\in[\ell]}\right)^\top$ outputs 1 iff (1) for every $j \in [\ell]$, the corresponding $\mathsf{V}(x,\boldsymbol{q}_j^{\mathsf{Hyb}},\boldsymbol{a}^{x,\boldsymbol{q}_j^{\mathsf{Hyb}}}) = 1$, and (2) for each $j \in [\ell]$, the $(j\cdot a)$-th prefix of $\boldsymbol{a}^{x,\boldsymbol{q}_j^{\mathsf{Hyb}}}$ is the same as the $(j\cdot a)$-th prefix of $\boldsymbol{a}^{x,\boldsymbol{q}}$.*

Remark 10. When a batch of inputs $\boldsymbol{x} = (x_1,\ldots,x_k) \in (\{0,1\}^n)^k$ is given, for any sequence of g pairs $\mathcal{S} = ((i^1,\boldsymbol{q}^1),\ldots,(i^g,\boldsymbol{q}^g)) \subset [k] \times (\{0,1\}^b)^\ell$, and g transcripts $\boldsymbol{a}^1,\ldots,\boldsymbol{a}^g \in \{0,1\}^{\ell\times(a\ell)}$, we consider a batch run of g instances of $(\mathsf{P}^{\mathsf{tc}},\mathsf{V}^{\mathsf{tc}})$ on $((x_{i^1},\boldsymbol{q}^1),\boldsymbol{a}^1),\ldots,((x_{i^g},\boldsymbol{q}^g),\boldsymbol{a}^g)$ where V^{tc} uses a shared random coin $\boldsymbol{q}'$ for all the g instances. Let

$$\mathcal{S}^{\mathsf{Hyb}} := ((i^1,(\boldsymbol{q}^1)_1^{\mathsf{Hyb}}),\ldots,(i^1,(\boldsymbol{q}^1)_\ell^{\mathsf{Hyb}}),\ldots,(i^g,(\boldsymbol{q}^g)_1^{\mathsf{Hyb}}),\ldots,(i^g,(\boldsymbol{q}^g)_\ell^{\mathsf{Hyb}}))$$

be the set of $g\ell$ hybrid pairs constructed from every pair in $\mathcal{S}$ using the new $\boldsymbol{q}'$. By the transcript structure of $(\mathsf{P}^{\mathsf{tc}}, \mathsf{V}^{\mathsf{tc}})$, the r-th round messages of the g instances of $(\mathsf{P}^{\mathsf{tc}}, \mathsf{V}^{\mathsf{tc}})$ constitute exactly the rows of $\boldsymbol{a}_r^{\mathcal{S}^{\mathsf{Hyb}}} = \left((\boldsymbol{a}_r^{x_i,\boldsymbol{q}})_{(i,\boldsymbol{q})\in\mathcal{S}^{\mathsf{Hyb}}}\right)^\top$, and the full transcripts of the g instances is $\boldsymbol{a}^{\mathcal{S}^{\mathsf{Hyb}}}$.

Checksum Let $M = 2^m$ for some $m \in \mathbb{N}$, and $\mathbb{F}$ be a constructible field ensemble of characteristic 2, such that $\log|\mathbb{F}| = \tilde{O}(\sigma)$.

Let $d \in \mathbb{N}$ be some *distance parameter* and $T_{\mathsf{cksum}} \in \mathbb{N}$ be some parameter be specified. We utilize the checksum of a systematic error correcting code with message space $\mathbb{F}^M$ of distance $2d$, which we denoted by $\mathsf{cksum}_{\boldsymbol{U}}(\cdot)$, where the parameter $\boldsymbol{U} = (\boldsymbol{U}_1, \ldots, \boldsymbol{U}_{T_{\mathsf{cksum}}}) \in (\mathbb{F}^m)^{T_{\mathsf{cksum}}}$ is a sequence of points in $\mathbb{F}^m$.

Proposition 3 (LDE-instantiated checksum (c.f. Proposition 6.6 of [RRR16]))**.** *Let $M = 2^m$ and $m \in \mathbb{N}$, $\mathbb{F}$ be a field of characteristic 2. Let $\sigma \in \mathbb{N}$ be an unambiguity parameter, and $T_{\mathsf{cksum}} \coloneqq 2d(m + \log|\mathbb{F}|) + \sigma$. There exists a function $\mathsf{cksum}_{\boldsymbol{U}} : \mathbb{F}^M \to \mathbb{F}^{T_{\mathsf{cksum}}}$, such that with probability $1 - 2^{-\sigma}$ over uniform $\boldsymbol{U} = (\boldsymbol{U}_1, \ldots, \boldsymbol{U}_{T_{\mathsf{cksum}}})$, $\mathsf{cksum}_{\boldsymbol{U}}(\cdot)$ is the checksum of a systematic linear error-correcting code of distance $2d$ on the message space $\mathbb{F}^M$. In other words, for any $\boldsymbol{m} \in \mathbb{F}^M$, for any $\boldsymbol{m}', \boldsymbol{m}'' \in \mathbb{F}^M$ that are both d-close to $\boldsymbol{m}$ and $\boldsymbol{m}' \neq \boldsymbol{m}''$. $\mathsf{cksum}_{\boldsymbol{U}}(\boldsymbol{m}') \neq \mathsf{cksum}_{\boldsymbol{U}}(\boldsymbol{m}'')$.*

This allows unique decoding *in the following sense: if we know $\boldsymbol{m}'$ is d-close to some (fixed) $\boldsymbol{m}$, then we can uniquely determine $\boldsymbol{m}'$ given $\mathsf{cksum}_{\boldsymbol{U}}(\boldsymbol{m}')$.*

Proof. Recall that $\hat{\boldsymbol{m}} : \mathbb{F}^m \to \mathbb{F}^{T_{\mathsf{cksum}}}$ is the multilinear extension of $\boldsymbol{m}$ (Section 3.1). We can let $\mathsf{cksum}_{\boldsymbol{U}}(\boldsymbol{m}) \coloneqq \hat{\boldsymbol{m}}(\boldsymbol{U}) = (\hat{\boldsymbol{m}}(\boldsymbol{U}_1), \ldots, \hat{\boldsymbol{m}}(\boldsymbol{U}_{T_{\mathsf{cksum}}})) \in \mathbb{F}^{T_{\mathsf{cksum}}}$.

To show unique decoding, assume that on the contrary, $\mathsf{cksum}_{\boldsymbol{U}}(\boldsymbol{m}') = \mathsf{cksum}_{\boldsymbol{U}}(\boldsymbol{m}'')$, Then $\widehat{\boldsymbol{m}' - \boldsymbol{m}''}(\boldsymbol{U}) = 0$ (by linearity). Note $\boldsymbol{m}' - \boldsymbol{m}'' \in \Delta(\mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{0}))$ has Hamming weight at most $2d$, so $\Delta(\mathsf{PVAL}(\boldsymbol{U}, \boldsymbol{0})) \leq 2d$, but this happens with probability at most $2^{-\sigma}$ by Proposition 1. □

Abusing the notation, given a matrix $\boldsymbol{M} \in \mathbb{F}^{M\times a}$ with columns $(\boldsymbol{m}_1, \ldots, \boldsymbol{m}_a)$, we use $\mathsf{cksum}_{\boldsymbol{U}}(\boldsymbol{M})$ to denote $\boldsymbol{S} = (\mathsf{cksum}_{\boldsymbol{U}}(\boldsymbol{m}_1), \ldots, \mathsf{cksum}_{\boldsymbol{U}}(\boldsymbol{m}_a)) \in \mathbb{F}^{T_{\mathsf{cksum}}\times a}$.

4.3.2 Construction of the Dist Protocol

Road map of our Construction. Recall that the input claim is $(\mathcal{S}, \Phi) \in \mathcal{L}'_{\boldsymbol{x}}$, and given the parameters $\sigma, d \in \mathbb{N}$ and the field $\mathbb{F}$ where $\log|\mathbb{F}| = \tilde{O}(\sigma)$, the goal is to define $(\mathcal{S}_{\mathsf{Dist}}, \Phi_{\mathsf{Dist}})$ such that Φ_{Dist} rejects everything in the $\mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}})$ around $\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}$ if $(\mathcal{S}, \Phi) \notin \mathcal{L}'_{\boldsymbol{x}}$, except with probability at most $(\epsilon + 2^{-\sigma})$.

To this end, both parties interact for ℓ rounds, to *implicitly* run a batch of $(\mathsf{P}^{\mathsf{tc}}, \mathsf{V}^{\mathsf{tc}})$ in parallel. These protocols are run implicitly in the following sense.

- $\mathsf{V}_{\mathsf{Dist}}$ samples $T_{\mathsf{cksum}} = \tilde{O}(\sigma d)$ random points that defines the checksum function $\mathsf{cksum}_{\boldsymbol{U}} : \mathbb{F}^{g\ell} \to \mathbb{F}^{T_{\mathsf{cksum}}}$. The checksums enable unique decoding except with probability $2^{-\sigma}$ over the choice of $\boldsymbol{U}$ (Proposition 3).
- $\mathsf{V}_{\mathsf{Dist}}$ samples a new random coin sequence $\boldsymbol{q}' \in (\{0,1\}^b)^\ell$, and send the coin to the prover round by round,
- $\mathsf{P}_{\mathsf{Dist}}$ only responds by the checksums of the underlying messages.

Specifically, in round $r \in [\ell]$, the prover computes checksums $\boldsymbol{S}_r \in \mathbb{F}^{T_{\mathsf{cksum}}}$ on the columns of the matrix $\boldsymbol{a}_r^{\mathcal{S}^{\mathsf{Hyb}}} \in \{0,1\}^{(g\ell)\times a}$, which consists of the $(g\ell)$ base-UIP r-th round answers that the prover would have sent if the protocols $(\mathsf{P}^{\mathsf{tc}}, \mathsf{V}^{\mathsf{tc}})$ were run explicitly. Note that we would pick $T_{\mathsf{cksum}} = \tilde{O}(\sigma d) \ll g\ell$, so indeed the checksums can be represented using very few bits ($\tilde{O}(\sigma d \log|\mathbb{F}|) = \tilde{O}(\sigma^2 d)$ bits).

Finally, they let the resulting claim be $(\mathcal{S}_{\mathsf{Dist}}, \Phi_{\mathsf{Dist}}) \in \mathcal{L}'_{\boldsymbol{x}}$ where $\mathcal{S}_{\mathsf{Dist}} = \mathcal{S}^{\mathsf{Hyb}}$ and Φ_{Dist} is a circuit that takes in $\boldsymbol{a}$ (purported to be $\boldsymbol{a}^{\mathcal{S}^{\mathsf{Hyb}}}$) as input. In additional to checking that rows (transcripts) in $\boldsymbol{a}$ are indeed accepted by the base UIP and satisfy the original constraint Φ, the circuit also recomputes the checksums of the columns of $\boldsymbol{a}$ to make sure they are equal to the checksums committed by the prover. Due to the unique decoding guarantee of cksum (Proposition 3), if there exists $\boldsymbol{a}^*$ in $\mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}})$ with $\Phi_{\mathsf{Dist}}(\boldsymbol{a}^*) = 1$, the cheating prover would be implicitly committing to this $\boldsymbol{a}^*$. Therefore, one can actually extract (via unique decoding) an accepting but deviating prover strategy for one of the base UIP. The base UIP's unambiguity guarantees that any deviating strategy should only be accepted with probability at most ϵ. Therefore, with high probability, there is no $\boldsymbol{a}^* \in \mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}})$ that would make $\Phi_{\mathsf{Dist}}(\boldsymbol{a}^*) = 1$, i.e. distance is generated.

The Protocol. The protocol is given in Protocol 2.

4.3.3 Analysis of the Dist Protocol

Recall that the input claim $(\mathcal{S}, \Phi) \in \mathcal{L}'_{\boldsymbol{x}}$ is true iff $(\Phi(\boldsymbol{a}^{\mathcal{S}}) = 1) \wedge (\boldsymbol{x}|_{\mathcal{I}} \in \mathcal{L}^{\otimes|\mathcal{I}|})$ where $\mathcal{S}$ consists of $g \le k$ pairs of the form $(i, \boldsymbol{q}) \in [k] \times \{0,1\}^{b\ell}$, $\log|\mathbb{F}| = \tilde{O}(\sigma)$, and Φ is a log-space uniform boolean circuit of size $\tilde{O}(ga\ell)$ and depth $\tilde{O}(1)$.

Prescribed Completeness. The honest prover $\mathsf{P}_{\mathsf{Dist}}$ answers the random coin $\boldsymbol{q}'$ sampled by the verifier by running the Random Continuation UIP $(\mathsf{P}_{i,\boldsymbol{q}}, \mathsf{V}_{i,\boldsymbol{q}})$ in parallel on all $g = |\mathcal{S}|$ instances. It honestly sends the checksums $\boldsymbol{S}$ that are computed on the columns of $\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}$, where $\mathcal{S}^{\mathsf{Hyb}}$ is the set of hybrid pairs between $\mathcal{S}$ and q'. Recall that $\Phi_{\mathsf{Dist}}(\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}) = 1$ iff

1. $\Phi(\boldsymbol{a}[\mathcal{S}, :]) = 1$ is true. Note that the submatrix $\boldsymbol{a}[\mathcal{S}, :]$ is indexed by the subset $\mathcal{S}$ on $\boldsymbol{a}$. (This is valid because $\mathcal{S} \subset \mathcal{S}^{\mathsf{Hyb}}$.)
2. For all $(i, \boldsymbol{q}) \in \mathcal{S}^{\mathsf{Hyb}}$, $\mathsf{V}(x_i, \boldsymbol{q}, \boldsymbol{a}'[(i, \boldsymbol{q}), :]) = 1$, and that for every $j \in [\ell]$, the $(j \cdot a)$-th prefix of $\boldsymbol{a}[(i, \boldsymbol{q}_j^{\mathsf{Hyb}}), :]$ and $\boldsymbol{a}[(i, \boldsymbol{q}), :]$ are identical.
3. $\mathsf{cksum}_U(\boldsymbol{a}) = (\boldsymbol{S}_1, \ldots, \boldsymbol{S}_\ell)$.
4. $\boldsymbol{a} \in \{0,1\}^{M \times L}$ (i.e. all elements strictly lies in $\mathbb{GF}(2)$.)

The third condition is always true because the checksums are computed honestly on the columns of $\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}$. On the other hand, the first condition is true iff $\Phi(\boldsymbol{a}^{\mathcal{S}}) = 1$, while the second condition is true iff $\boldsymbol{x}|_{\mathcal{I}} \in \mathcal{L}^{\otimes\mathcal{I}}$, so, $\Phi_{\mathsf{Dist}}(\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}) = 1$ iff $(\mathcal{S}, \Phi) \in \mathcal{L}'_{\boldsymbol{x}}$.

Unambiguous Generation of Δ_c-Distance. The verifier $\mathsf{V}_{\mathsf{Dist}}$ sends the fresh random coins $\boldsymbol{q}' = (q'_1, q'_2, \ldots, q'_\ell)$ round by round. Let P^* be a cheating strategy that deviates from $\mathsf{P}_{\mathsf{Dist}}$. For each $r \in [\ell]$, P^*'s behavior in the r-th round is fully determined by $\boldsymbol{q}'_{\le r} = (q'_1, \ldots, q'_r)$, so let $\boldsymbol{S}^*_r$ be the checksum produced by P^* on $\boldsymbol{q}'_{\le r}$. Let $\boldsymbol{S}^* = (\boldsymbol{S}^*_1, \ldots, \boldsymbol{S}^*_\ell)$ be all the checksum sent.

Protocol 2 $(\mathsf{P}_{\mathsf{Dist}}, \mathsf{V}_{\mathsf{Dist}})$ Generating Δ_c-Distance via Checksums..

Input Parameters: $\sigma, d \in \mathbb{N}$ and a field $\mathbb{F}$.

Input: $\boldsymbol{x} = (x_1, \ldots, x_k)$ are the UIP statements. $\langle \mathcal{S} \rangle, \langle \Phi \rangle$ are the descriptions for the claim $(\mathcal{S}, \Phi) \in \mathcal{L}'$.

Other Parameters: $g = |\mathcal{S}|$, $M = g\ell = 2^m$, and $T_{\mathsf{cksum}} = 2d(m + \sigma) + \sigma \in \mathbb{N}$.

Ingredients:

- *A Transcript Checker UIP,* $(\mathsf{P}^{\mathsf{tc}}, \mathsf{V}^{\mathsf{tc}})$ *(Proposition 2), for certifying* $((x_i, \boldsymbol{q}), \boldsymbol{a}) \in \mathcal{R}$.
- *A Checksum Function,* $\mathsf{cksum}_{\boldsymbol{U}}(\cdot)$*, computing the checksum of a systematic error-correcting code with distance* $2d$, where $\boldsymbol{U} = (\boldsymbol{U}_1, \ldots, \boldsymbol{U}_{T_{\mathsf{cksum}}}) \in (\mathbb{F}^m)^{T_{\mathsf{cksum}}}$ is a sequence of points in $\mathbb{F}^m$.

Output: $\boldsymbol{x}, \langle \mathcal{S}_{\mathsf{Dist}} \rangle, \langle \Phi_{\mathsf{Dist}} \rangle$.

1: $\mathsf{V}_{\mathsf{Dist}}$ samples $\boldsymbol{q}' = (q'_1, \ldots, q'_\ell) \in (\{0,1\}^b)^\ell$.
2: $\mathsf{V}_{\mathsf{Dist}}$ samples $\boldsymbol{U} \leftarrow_R (\mathbb{F}^m)^{T_{\mathsf{cksum}}}$ and sends it to $\mathsf{P}_{\mathsf{Dist}}$.
3: **for** $r = 1, \ldots, \ell$ **do**
4: $\quad \mathsf{V}_{\mathsf{Dist}}$ sends q'_r to $\mathsf{P}_{\mathsf{Dist}}$. ▷ *Random coins are shared.*
5: $\quad \mathsf{P}_{\mathsf{Dist}}$ receives q'_r and does:
6: $\quad\quad$ 1. Compute $\boldsymbol{m}_r^{i,\boldsymbol{q}} \coloneqq \mathsf{P}^{\mathsf{tc}}((x_i, \boldsymbol{q}), (\boldsymbol{m}_1^{i,\boldsymbol{q}}, \ldots, \boldsymbol{m}_{r-1}^{i,\boldsymbol{q}}))$ for all $(i, \boldsymbol{q}) \in \mathcal{S}$,
7: $\quad\quad$ 2. By Remark 10, $\left\{\boldsymbol{m}_r^{i,\boldsymbol{q}}\right\}_{(i,\boldsymbol{q}) \in \mathcal{S}}$ exactly forms the rows of the matrix $\boldsymbol{a}_r \coloneqq \boldsymbol{a}_r^{\mathcal{S}^{\mathsf{Hyb}}} \in \{0,1\}^{(g\ell) \times a}$.
8: $\quad\quad$ 3. Compute $\boldsymbol{S}_r \leftarrow \mathsf{cksum}_{\boldsymbol{U}}(\boldsymbol{a}_r) \in \{0,1\}^{T_{\mathsf{cksum}} \times a}$, and send it to $\mathsf{V}_{\mathsf{Dist}}$.
9: Define an arithmetic circuit $\Phi_{\mathsf{Dist}}(\boldsymbol{a})$, where $\boldsymbol{a} \in \mathbb{F}^{M \times (a\ell)}$ is purported to be $\boldsymbol{a}^{\mathcal{S}^{\mathsf{Hyb}}}$, and the circuit checks the following set of conditions. ▷ *Note that* $\left|\mathcal{S}^{\mathsf{Hyb}}\right| = M = g\ell$.

1. $\Phi(\boldsymbol{a}[\mathcal{S}, :]) = 1$ is true. Note that the submatrix $\boldsymbol{a}[\mathcal{S}, :]$ is indexed by the subset $\mathcal{S}$ on $\boldsymbol{a}$. (This is valid because $\mathcal{S} \subset \mathcal{S}^{\mathsf{Hyb}}$.)
2. For all $(i, \boldsymbol{q}) \in \mathcal{S}^{\mathsf{Hyb}}$, $\mathsf{V}(x_i, \boldsymbol{q}, \boldsymbol{a}'[(i, \boldsymbol{q}), :]) = 1$, and that for every $j \in [\ell]$, the $(j \cdot a)$-th prefix of $\boldsymbol{a}[(i, \boldsymbol{q}_j^{\mathsf{Hyb}}), :]$ and $\boldsymbol{a}[(i, \boldsymbol{q}), :]$ are identical.
3. $\mathsf{cksum}_{\boldsymbol{U}}(\boldsymbol{a}) = (\boldsymbol{S}_1, \ldots, \boldsymbol{S}_\ell)$.
4. $\boldsymbol{a} \in \{0,1\}^{M \times L}$ (i.e. all elements strictly lies in $\mathbb{GF}(2)$.)

10: **return** $\langle \mathcal{S}^{\mathsf{Hyb}} \rangle, \langle \Phi_{\mathsf{Dist}} \rangle$.

Fix some $\boldsymbol{q}'_{\leq r^*}$ such that $r^* \in [\ell]$ is indeed the first round when P^* deviates, i.e. when it first produces some $\boldsymbol{S}^*_{r^*} \neq \mathsf{cksum}(\boldsymbol{a}_{r^*})$. As r^* is minimal, $\boldsymbol{S}^*_{r^*} \neq \mathsf{cksum}(\boldsymbol{a}_{r^*})$ but $\boldsymbol{S}^*_r = \mathsf{cksum}(\boldsymbol{a}_r)$ for all $r < r^*$. (Recall that $\boldsymbol{a}_r$ is the r-th round chunks of $\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}$). Note that this implies $\Phi_{\mathsf{Dist}}(\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}) = 0$ because $\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}$ does not have the checksum $\boldsymbol{S}^*_{r^*}$ committed by the prover in round r^*.

Let E be the event that after $\boldsymbol{q}'_{>r^*}$ is sampled, there exists $\boldsymbol{a}^* \in \mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}) \setminus \left\{\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}\right\}$, such that $\Phi_{\mathsf{Dist}}(\boldsymbol{a}^*) = 1$. Given that we already showed $\Phi_{\mathsf{Dist}}(\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}) = 0$, to prove generation of Δ_c-distance (with probability at least $1 - (\epsilon + 2^{-\sigma})$, for every $\boldsymbol{a}^* \in \mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}})$, $\Phi_{\mathsf{Dist}}(\boldsymbol{a}^*) = 0$) we only need to show that $\Pr[E] \leq \epsilon + 2^{-\sigma}$. If E is impossible on this $\boldsymbol{q}'_{\leq r^*}$, then we are done, so we consider the case when $\Pr[E] > 0$.

Since $\Pr[E] > 0$, there exists some continuation $\boldsymbol{q}'_{>r^*}$ and $\boldsymbol{a}^* \in \mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}) \setminus \left\{\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}\right\}$ such that $\Phi_{\mathsf{Dist}}(\boldsymbol{a}^*) = 1$. In this case, for every $r \leq r^*$, if we let $\boldsymbol{a}^*_r$ be the corresponding chunk of $\boldsymbol{a}^*$ used in round r, we have $\mathsf{cksum}_{\boldsymbol{U}}(\boldsymbol{a}^*_r) = \boldsymbol{S}^*_r$ because of Condition 3 of Φ_{Dist}.

Furthermore, since P^* is first cheating on round r^*, we have $\boldsymbol{S}^*_{r^*} = \mathsf{cksum}(\boldsymbol{a}^*_{r^*}) \neq \mathsf{cksum}(\boldsymbol{a}_{r^*})$, and thus $\boldsymbol{a}^*_{r^*} \neq \boldsymbol{a}_{r^*}$. This means there exists a row-index $(i, \boldsymbol{q}^{\mathsf{Hyb}}) \in \mathcal{S}^{\mathsf{Hyb}}$ such that $\boldsymbol{a}^*_{r^*}[i, \boldsymbol{q}^{\mathsf{Hyb}}] \neq \boldsymbol{a}_{r^*}[i, \boldsymbol{q}^{\mathsf{Hyb}}]$. By Remark 10, $(i, \boldsymbol{q}^{\mathsf{Hyb}})$ is some hybrid derived from some $(i, \boldsymbol{q}) \in \mathcal{S}$. Let $\mathcal{S}^{i,\mathsf{Hyb}} = \left\{(i, \boldsymbol{q}^{\mathsf{Hyb}}_1), \ldots, (i, \boldsymbol{q}^{\mathsf{Hyb}}_\ell)\right\}$ be the set of hybrid pairs derived from $(i, \boldsymbol{q})$.

We construct a cheating prover $\mathsf{P}^{\mathsf{tc}}_*$, that breaks the unambiguous soundness of the UIP $(\mathsf{P}^{\mathsf{tc}}, \mathsf{V}^{\mathsf{tc}})$ as follows. For each round $r \in [\ell]$, it does:

1. Sample $\boldsymbol{U} \leftarrow_R \{0,1\}^{T_{\mathsf{cksum}}}$ uniformly at random and send it to P^*.
2. Let $\boldsymbol{q}'_{\leq r} = (q'_1, \ldots, q'_r)$ be the random coins sent by V^{tc} in the previous rounds,
3. Apply P^* on $\boldsymbol{x}|_{\mathcal{I}}$ and $\boldsymbol{q}'_{\leq r}$ to obtain $\boldsymbol{S}^*_r$,
4. Try to infer $\boldsymbol{a}^*_r$, committed by P^*, by finding at most one $\boldsymbol{a}^*_r$ that is Δ_c-d-close to $\boldsymbol{a}_r$ with checksum $\boldsymbol{S}^*_r$.
5. If such an $\boldsymbol{a}^*_r$ exists and $\boldsymbol{a}^*_r \in \{0,1\}^{M \times a}$, send the message $\boldsymbol{a}^*_r[\mathcal{S}^{i,\mathsf{Hyb}}, :]$ to V^{tc}.

Claim 1. *Conditioned on unique decoding*[18] *holding for the sampled* $\boldsymbol{U}$*, when* V^{tc} *samples the random coin prefix* $\boldsymbol{q}'_{\leq r^*}$*,* $\mathsf{P}^{\mathsf{tc}}_*$ *first deviates on the* r^**-th round and convinces* V^{tc} *to accept with probability at least* $\Pr[E]$*.*

Proof. First note that by the minimality of r^* and the uniqueness of $\boldsymbol{a}^*_r$ in each round, $\mathsf{P}^{\mathsf{tc}}_*$ deviates from the prescribed P^{tc} the first time exactly in round r^*.

Whenever E occurs, there exists some completion $\boldsymbol{a}' \in \mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a})$ such that $\Phi_{\mathsf{Dist}}(\boldsymbol{a}') = 1$. Let $(\boldsymbol{a}'_1, \ldots, \boldsymbol{a}'_L)$ be the columns of $\boldsymbol{a}'$. Again by unique decoding, $\mathsf{P}^{\mathsf{tc}}_*$ are *exactly* sending those $\boldsymbol{a}'_r$ ($\boldsymbol{a}'_r = \boldsymbol{a}^*_r$) for all $r \in [\ell]$, because their checksums are agreeing with $\boldsymbol{S}^*$. Furthermore, $(\boldsymbol{a}'_1[\mathcal{S}^{i,\mathsf{Hyb}}, :], \ldots, \boldsymbol{a}'_\ell[\mathcal{S}^{i,\mathsf{Hyb}}])$ satisfies Conditions 2 and 4 of Φ_{Dist}, so it is also accepted by V^{tc} (Proposition 2). □

By the ϵ-unambiguity of $(\mathsf{P}^{\mathsf{tc}}, \mathsf{V}^{\mathsf{tc}})$, the probability of $\mathsf{P}^{\mathsf{tc}}_*$ breaking the unambiguity of $(\mathsf{P}^{\mathsf{tc}}, \mathsf{V}^{\mathsf{tc}})$ is upper-bounded by ϵ. Therefore, $\Pr[E] - 2^{-\sigma} \leq \epsilon$. Note the loss of $2^{-\sigma}$ comes from the (rare) event that unique decoding does not hold for the initially sampled $\boldsymbol{U}$ (Proposition 3).

The *furthermore* part follows from that when E does not happen, $\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}$ is the only candidate solution to $\Phi_{\mathsf{Dist}}(\boldsymbol{a}) = 1$ in $\mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}})$. This holds for the prescribed $\mathsf{P}_{\mathsf{Dist}}$ as well.

[18] Recall that unique decoding means for any pair $\boldsymbol{a}', \boldsymbol{a}'' \in \mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}})$, if $\mathsf{cksum}_{\boldsymbol{U}}(\boldsymbol{a}') = \mathsf{cksum}_{\boldsymbol{U}}(\boldsymbol{a}'')$ then $\boldsymbol{a}' = \boldsymbol{a}''$.

Complexities. With $\tilde{O}$ hiding $\mathrm{poly}(\sigma) \cdot \mathrm{polylog}(k, n, a, \ell)$ factors, Note that we rely on the assumption that $\log |\mathbb{F}| = \tilde{O}(\sigma)$ to simplify the expressions.

- $\ell_{\mathsf{Dist}} = O(\ell)$,
- $a_{\mathsf{Dist}} = T_{\mathsf{cksum}} \cdot a \cdot \log |\mathbb{F}| = \tilde{O}(da)$.
- $b_{\mathsf{Dist}} = \max(b, T_{\mathsf{cksum}} \log |\mathbb{F}|) = \tilde{O}(\max(b, \sigma^2 d)) = \tilde{O}(da)$, assuming $b \le a$.
- $\mathsf{Ptime}_{\mathsf{Dist}} = \tilde{O}(g\ell\mathsf{Ptime} + T_{\mathsf{cksum}}(g\ell)(a\ell)\log |\mathbb{F}| + |\langle \mathcal{S} \rangle| + |\langle \Phi \rangle| + k \cdot n) = \tilde{O}(g\ell\mathsf{Ptime} + dga\ell^2 + |\langle \mathcal{S} \rangle| + |\langle \Phi \rangle| + k \cdot n)$.
- $\mathsf{Vtime}_{\mathsf{Dist}} = \tilde{O}(T_{\mathsf{cksum}} \cdot a \cdot \log |\mathbb{F}| \cdot \ell + |\langle \mathcal{S} \rangle| + |\langle \Phi \rangle| + k \cdot n) = \tilde{O}(da\ell + |\langle \mathcal{S} \rangle| + |\langle \Phi \rangle| + k \cdot n)$.

The runtimes are less immediate, so we explain them below.

- For the prover runtime, note that is has to run $g = |\mathcal{S}|$ instances of the UIP $(\mathsf{P}^{\mathsf{tc}}, \mathsf{V}^{\mathsf{tc}})$ in parallel, which is in turn $g\ell$ instances of the base (P, V), so that takes $O(g\ell\mathsf{Ptime})$. It also computes the checksums (which can be done in $\tilde{O}(T_{\mathsf{cksum}}(g\ell)(a\ell)\log |\mathbb{F}|)$ time using FFT), The term $|\langle \mathcal{S} \rangle| + |\langle \Phi \rangle| + k \cdot n$ comes from reading the input.
- The verifier does not perform any check in this sub-protocol, but it has to store the $T_{\mathsf{cksum}} \cdot a$ points it receives every round, which requires $\tilde{O}(T_{\mathsf{cksum}} \cdot a \cdot \log |\mathbb{F}| \ell)$ time. In the end, the time to compute $\langle \mathcal{S}_{\mathsf{Dist}} \rangle$ is $O(|\langle \mathcal{S}_{\mathsf{Dist}} \rangle|) = O(|\langle S \rangle| + b\ell)$, and the time to compute $\langle \Phi_{\mathsf{Dist}} \rangle$ is $O(|\langle \Phi_{\mathsf{Dist}} \rangle|) = \tilde{O}(|\langle \Phi \rangle| + |\langle \mathcal{S} \rangle| + da\ell + k \cdot n)$, since both of these consist of strings the verifier already has access to.

Circuit G_{Dist} for generating $\mathcal{S}_{\mathsf{Dist}}$. Let $\boldsymbol{q}'$ be the fresh random coin generated in Protocol 2. Let $p := \lceil \log k \rceil + b \cdot \ell$ be the bit length of a pair $(i, \boldsymbol{q}) \in \{0,1\}^{\lceil \log k \rceil} \times \{0,1\}^{b \cdot \ell}$. We can represent $\mathcal{S}_{\mathsf{Dist}}$ as a subset of $\{0,1\}^p$.

Claim 2. *The set $\mathcal{S}_{Dist}$ has a description $\langle \mathcal{S}_{Dist} \rangle = (\langle \mathcal{S} \rangle, \boldsymbol{q}')$, and $|\langle \mathcal{S}_{Dist} \rangle| = |\langle \mathcal{S} \rangle| + b\ell + O(1)$.*

Proof. Assume $\mathcal{S} = \{\mathcal{S}_1, \ldots, \mathcal{S}_g\}$, where $\mathcal{S}_s = (i^s, \boldsymbol{q}^s)$ for every $s \in [g]$, then by the definition of the hybrid coins in Protocol 2,

$$\mathcal{S}_{\mathsf{Dist}} = \mathcal{S}^{\mathsf{Hyb}} = \left\{ (i^1, (\boldsymbol{q}^1)_1^{\mathsf{Hyb}}), \ldots, (i^1, (\boldsymbol{q}^1)_\ell^{\mathsf{Hyb}}), \ldots, (i^g, (\boldsymbol{q}^g)_1^{\mathsf{Hyb}}), \ldots, (i^g, (\boldsymbol{q}^g)_\ell^{\mathsf{Hyb}}) \right\},$$

where $(\boldsymbol{q}^s)_r^{\mathsf{Hyb}} := (\boldsymbol{q}^s_{\le r}, \boldsymbol{q}'_{>r})$ for every $s \in [g], r \in [\ell]$.

Let $M = |\mathcal{S}_{\mathsf{Dist}}| = g\ell$, and $\langle \mathcal{S}_{\mathsf{Dist}} \rangle = (\langle \mathcal{S} \rangle, \boldsymbol{q}')$. We define a Turing machine $\mathcal{M}$ that takes in $(M, 1^{|\langle \mathcal{S}_{\mathsf{Dist}} \rangle|}, 1^p)$ as input, and outputs the circuit $G_{\mathsf{Dist}} : [M] \times \{0,1\}^{|\langle \mathcal{S}_{\mathsf{Dist}} \rangle|} \to \{0,1\}^p$. Since $\langle \mathcal{S} \rangle$ is a succinct description of $\mathcal{S}$, there exists a Turing machine $\mathcal{M}'$ that takes in $(g, 1^{|\langle \mathcal{S} \rangle|}, 1^p)$ as input, and outputs the circuit $G : [g] \times \{0,1\}^{|\langle \mathcal{S} \rangle|} \to \{0,1\}^p$, which, given $(s, \langle \mathcal{S} \rangle)$ as input, outputs $\mathcal{S}_s = (i^s, \boldsymbol{q}^s)$.

$\mathcal{M}$ first runs $\mathcal{M}'$ to get G, then moves onto defining G_{Dist}, which, given $(s', \langle \mathcal{S}_{\mathsf{Dist}} \rangle) \in [M] \times \{0,1\}^{|\langle \mathcal{S}_{\mathsf{Dist}} \rangle|}$ as input,

1. First, computes (g, r) such that $s' = s \cdot g + r$ where $s \in [g], r \in [\ell]$, e.g. by long division.
2. Then, evaluates $G(s, \langle \mathcal{S} \rangle)$ to get $(i^s, \boldsymbol{q}^s)$.

3. Finally, computes the corresponding hybrid coin $(\boldsymbol{q}^s)_r^{\mathsf{Hyb}}$ using $\boldsymbol{q}^s$ and $\boldsymbol{q}'$ (which is just copying the first r bits of $\boldsymbol{q}^s$ and appending the remaining bits of $\boldsymbol{q}'$), and outputs $(i^s, (\boldsymbol{q}^s)_r^{\mathsf{Hyb}})$.

Indeed,

- $\mathsf{size}(G_{\mathsf{Dist}}) = \mathsf{size}(G) + b\ell + \tilde{O}(1)$.
- $\mathsf{depth}(G_{\mathsf{Dist}}) = \mathsf{depth}(G) + \tilde{O}(1)$.

The Turing machine $\mathcal{M}$ uses $O(p) = O(\log k + b\ell)$ space because it only needs to simulate $\mathcal{M}'$ (which uses $O(p)$ space) and maintain pointers into the circuit G_{Dist} (which uses $O(p)+O(\log k+\log b+\log \ell)$ space). □

Circuit C_{Dist} for Φ_{Dist}. By construction, $\Phi_{\mathsf{Dist}}(\boldsymbol{a})$ checks four conditions, whose circuit complexities are:

1. Checking Φ on $\boldsymbol{a}[\mathcal{S}, :]$, requires $\mathsf{size}(\Phi)$ gates and $\mathsf{depth}(\Phi)$ depth.
2. For all $(i, \boldsymbol{q}) \in \mathcal{S}^{\mathsf{Hyb}}$, $\mathsf{V}(x_i, \boldsymbol{q}, \boldsymbol{a}'[(i, \boldsymbol{q}), :]) = 1$, and that for every $j \in [\ell]$, the $(j \cdot a)$-th prefix of $\boldsymbol{a}[(i, \boldsymbol{q}_j^{\mathsf{Hyb}}), :]$ and $\boldsymbol{a}[(i, \boldsymbol{q}), :]$ are identical.

   This requires $O(M \cdot S + ML) = \tilde{O}(g\ell S + dga\ell^2)$ gates and $D + O(\log(ML)) = D + \tilde{O}(1)$ depth (by a binary tree of AND gates).
3. $\mathsf{cksum}_{\boldsymbol{U}}(\boldsymbol{a}) = (\boldsymbol{S}_1, \ldots, \boldsymbol{S}_\ell)$.

   This requires re-computing the checksums for every column of $\boldsymbol{a} \in \{0,1\}^{M \times L}$. To compute one checksum, T_{cksum} LDE evaluation are needed. One LDE evaluation requires $\tilde{O}(M) = \tilde{O}(g\ell)$ gates, and $\tilde{O}(1)$ depth. Therefore, we need $\tilde{O}(T_{\mathsf{cksum}} ML) = \tilde{O}(dga\ell^2)$ gates and $\tilde{O}(1)$ depth to compute the checksums. Finally, we implement the consistency checks by equality tests. This requires an additional $\tilde{O}(T_{\mathsf{cksum}} L) = \tilde{O}(da\ell)$ gates and $\tilde{O}(1)$ depth.
4. $\boldsymbol{a} \in \{0,1\}^{M \times L}$ (i.e. all elements strictly lies in $\mathbb{GF}(2)$.)

   To check $\boldsymbol{a} \in \mathbb{GF}(2)^{M \times L}$, we just need to test if every element is in $\mathbb{GF}(2)$, and compute an AND over them. This requires $\tilde{O}(ML) = \tilde{O}(g\ell)$ gates and $\tilde{O}(1)$ depth, because the test can be implemented by evaluating the polynomial $X^2 + X$ over $\mathbb{F}$ and checking if the result is 0.

In order to perform the second check, C_{Dist} also needs to first enumerate the pairs $\mathcal{S}^{\mathsf{Hyb}}$ explicitly using $\langle \mathcal{S}^{\mathsf{Hyb}} \rangle$. This incurs an additional overhead of $\mathsf{depth}(G_{\mathsf{Dist}}) + \tilde{O}(1) = \mathsf{depth}(G) + \tilde{O}(1)$ and $g\ell \cdot \mathsf{size}(G_{\mathsf{Dist}}) = \tilde{O}(g\ell \cdot \mathsf{size}(G) + gb\ell^2)$ (see Claim 2).

Therefore, we have the following overall bounds.

- $\mathsf{size}(C_{\mathsf{Dist}}) = \tilde{O}(g\ell \cdot \mathsf{size}(G) + \mathsf{size}(C) + ga\ell S + dga\ell^2)$.
- $\begin{aligned}\mathsf{depth}(C_{\mathsf{Dist}}) &= \max(\mathsf{depth}(C), \mathsf{depth}(G) + D + \tilde{O}(1), \tilde{O}(1)) + \tilde{O}(1) \\ &= \max(\mathsf{depth}(C), \mathsf{depth}(G) + D) + \tilde{O}(1).\end{aligned}$

Similarly to Claim 2, the circuit Φ_{Dist} also has a succinct description, given by

$$\langle \Phi_{\mathsf{Dist}} \rangle = (\langle \mathcal{S}_{\mathsf{Dist}} \rangle, \langle \Phi \rangle, \boldsymbol{U}, (\boldsymbol{S}_1, \ldots, \boldsymbol{S}_\ell), \boldsymbol{x}),$$

and it is indeed log-space uniform, because its components (the four checks, as well as the circuit $C(\langle \mathcal{S}^{\mathsf{Hyb}} \rangle)$ that generates $\mathcal{S}^{\mathsf{Hyb}}$) are, and the additional binary trees of AND as well as the equality checks are simple to describe. The bit length $|\langle \Phi_{\mathsf{Dist}} \rangle| = \tilde{O}(|\langle \Phi \rangle| + |\langle \mathcal{S} \rangle| + da\ell + k \cdot n)$.

4.4 Proof of Lemma 4: Instance Reduction for Δ_c-Distance.

4.4.1 Additional Ingredients

Road map of our Construction. Recall that the intermediate claim is $\Phi_{\mathsf{Dist}}(\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}) = 1$. We also have the *distance guarantee* from Lemma 3: If $\Phi_{\mathsf{Dist}}(\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}) = 0$ then every matrix $\boldsymbol{a} \in \mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}})$ is rejected by Φ_{Dist}. Our goal is to output some reduced claim $(\mathcal{S}', \Phi')$ such that $|\mathcal{S}'| \leq |\mathcal{S}_{\mathsf{Dist}}|/d$.

The RR protocol provides a similar instance reduction, but only with respect to the Hamming distance.

Theorem 7 (The original RR Instance Reduction Protocol, informally rephrased; [RR20])**.** *Suppose $M \in \mathbb{N}$. There exists a protocol $(\mathsf{P}_{RR}, \mathsf{V}_{RR})$, where both the prover and verifier gets the description $\langle \Phi \rangle$ of a log-space uniform circuit $\Phi : \{0,1\}^M \to \{0,1\}$, and the prover gets the auxiliary input $\boldsymbol{w} \in \{0,1\}^M$, such that if both parties gets the parameter $d = d(n) \in \mathbb{N}$, then the protocol outputs a succinct description $\langle \mathcal{Q} \rangle$ of a set $\mathcal{Q} \subset [M]$ and a new circuit $\Phi' : \{0,1\}^{|\mathcal{Q}|} \to \{0,1\}$, such that $|\mathcal{Q}| \leq \frac{|M|}{d}$. The protocol is sound in the following sense: if $\Phi(\boldsymbol{w}) = 0$ for every $\boldsymbol{w}$ that is (Hamming) d-close to $\boldsymbol{w}$, then $\Phi'(\boldsymbol{w}|_{\mathcal{Q}}) = 0$. It has $\tilde{O}(1)$ rounds and communication complexity $\tilde{O}(d)$. The prover runtime is $\mathrm{poly}(M)$ and the verifier runtime is $\tilde{O}(d)$.*

As explained in Section 2.2, $\mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}})$ is a strict subset of the Hamming ball. The soundness of Theorem 7 applies only when *every Hamming-close string* is rejected, which is not always true in our case because our notion of distance is different.

The RR Instance Reduction Template Theorem 7 is originally proven by the following two steps:

1. Use the GKR protocol (Theorem 3) to transform the input claim $\Phi(\boldsymbol{w}) = 1$ into the claim $\boldsymbol{w} \in \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$. The parameters are chosen to ensure that *distance is preserved*: if every $\boldsymbol{w}'$ that is d-close to $\boldsymbol{w}$ is rejected by Φ, then $\boldsymbol{w}$ is d-far from $\mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$.
2. Apply a special-purpose protocol for reducing PVAL instances, on the input $\boldsymbol{w} \in \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$. This outputs a subset of coordinates $\mathcal{Q} \subset [M]$ that specifies the positions of the coordinates of $\boldsymbol{w}$ to read, and $|\mathcal{Q}| \leq |M|/d$, along with a predicate Φ', such that $\Phi'(\boldsymbol{w}|_{\mathcal{Q}}) = 1$ only if $\boldsymbol{w}$ is Hamming-d-close to $\mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$.

It turns out that the first step can be modified to work for Δ_c-distance quite straightforwardly (Lemma 2). However, for the second step, we have to make lower level modifications in order to make it work for our notion of distance. This discussion appears in Section 6.

In the rest of this section, we formally recall the guarantees of the two sub-protocols (Lemmas 2 and 6), corresponding to the two steps mentioned above, and use them to prove Lemma 4.

The GKR Protocol is Distance-Preserving Let $L := a\ell$. To preserve the distance, we choose the parameter $T = \tilde{\Theta}(dL)$ (see Lemma 2). In the GKR protocol, both parties take as input the description $\langle \Phi_{\mathsf{Dist}} \rangle$, and the prover additionally has access to $\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}$. After running the GKR protocol with parameter T, the verifier obtains $\boldsymbol{j} \in (\mathbb{F}^m)^T, \boldsymbol{v} \in \mathbb{F}^T$ as output, with the guarantee that $\mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$ is Δ_c-d-close to $\Phi_{\mathsf{Dist}}(\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}})$ iff $\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}$ is Δ_c-d-close to $\Phi_{\mathsf{Dist}}(\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}})$.[19]

Recall that $D_{\mathsf{Dist}} := \mathsf{depth}(\Phi_{\mathsf{Dist}})$, $S_{\mathsf{Dist}} := \mathsf{size}(\Phi_{\mathsf{Dist}})$. The complexity of the protocol is as follows, where $\tilde{O}$ hides $\mathrm{polylog}(\log |\mathbb{F}|, M, L)$ factors:

[19] Recall that a matrix $\boldsymbol{a}$ is Δ_c-d-close to $\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}$ iff $\boldsymbol{a} \in \mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}})$ (Definition 1).

- $\epsilon_{\mathsf{GKR}} = \frac{O(D_{\mathsf{Dist}} \log S_{\mathsf{Dist}})}{|\mathbb{F}|} + \left(O(\frac{D_{\mathsf{Dist}} \log S_{\mathsf{Dist}}}{|\mathbb{F}|})\right)^T \left(\binom{M}{d}|\mathbb{F}|^d\right)^L < 2^{-\sigma+2}$, so long as $O(\frac{D_{\mathsf{Dist}} \log S_{\mathsf{Dist}}}{|\mathbb{F}|}) < 2^{-\sigma-3}$.
- $\ell_{\mathsf{GKR}} = O(D_{\mathsf{Dist}} \log S_{\mathsf{Dist}})$.
- $a_{\mathsf{GKR}} = b_{\mathsf{GKR}} = \tilde{O}(T \log |\mathbb{F}|) = \tilde{O}(da\ell)$.
- $\mathsf{Ptime}_{\mathsf{GKR}} = \tilde{O}(T \cdot \mathrm{poly}(S_{\mathsf{Dist}}) \cdot \mathrm{polylog}|\mathbb{F}|) = \tilde{O}(da\ell) \cdot \mathrm{poly}(S_{\mathsf{Dist}})$.
- $\mathsf{Vtime}_{\mathsf{GKR}} = \tilde{O}(T \cdot D_{\mathsf{Dist}} \log S_{\mathsf{Dist}} \cdot \log |\mathbb{F}| + T \cdot |\langle \Phi_{\mathsf{Dist}} \rangle| \log |\mathbb{F}|) = \tilde{O}(da\ell \cdot (D_{\mathsf{Dist}} \log S_{\mathsf{Dist}} + |\langle \Phi_{\mathsf{Dist}} \rangle|))$.

And the verifier's verdict circuit satisfies:

- $\mathsf{size}(\mathsf{V}_{\mathsf{GKR}}) = \tilde{O}(T \cdot D_{\mathsf{Dist}} \log S_{\mathsf{Dist}} + T \cdot |\langle \Phi_{\mathsf{Dist}} \rangle| \log |\mathbb{F}|) = \tilde{O}(da\ell \cdot (D_{\mathsf{Dist}} \log S_{\mathsf{Dist}} + |\langle \Phi_{\mathsf{Dist}} \rangle|))$.
- $\mathsf{depth}(\mathsf{V}_{\mathsf{GKR}}) = \tilde{O}(1)$.

The Instance Reduction Protocol for PVAL with Δ_c-distance $(\mathsf{P}_{\Delta_c \mathsf{RR}}, \mathsf{V}_{\Delta_c \mathsf{RR}})$ is a special-purpose protocol for reducing PVAL instances with Δ_c distances. Both parties in the protocol take as input the parameters $\boldsymbol{j} \in (\mathbb{F}^m)^T, \boldsymbol{v} \in \mathbb{F}^T$ that specify $\mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$, and the prover additionally has the input $\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}$. The protocol outputs $\langle \mathcal{S}' \rangle, \langle \Phi' \rangle$ such that $|\mathcal{S}'| \leq |\mathcal{S}_{\mathsf{Dist}}|/d$, with the guarantee that $(\Phi', \mathcal{S}') \in \mathcal{L}'_{\boldsymbol{x}}$ iff $\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}$ is Δ_c-d-close to $\mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$.

The following is a restatement of Theorem 9. Its proof appears in Section 6.

Lemma 6 (The Instance Reduction Protocol for PVAL with Δ_c-distance (Theorem 9 in Section 6))**.** *Suppose σ, d, m, $\log L \in \mathbb{N}$, $M = 2^m$, and $\mathbb{F}$ is a field. Let $\tilde{O}$ be omitting $\mathrm{polylog}(|\mathbb{F}|, M, L)$ factors. Suppose $T \geq 8dLm = \tilde{O}(dL)$. Let $\boldsymbol{j} = (\boldsymbol{j}_1, \ldots, \boldsymbol{j}_T) \in (\mathbb{F}^{m+\log L})^T$ and $\boldsymbol{v} = (v_1, \ldots, v_T) \in \mathbb{F}^T$.*

There exists a constant C, and a public-coin unambiguous IPP $(\mathsf{P}_{\Delta_c RR}, \mathsf{V}_{\Delta_c RR})$ for $\mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$, where both the prover and verifier gets the input $\boldsymbol{j}, \boldsymbol{v}$, while the prover additionally gets the input $\boldsymbol{a} \in \mathbb{F}^{M \times L}$, such that if both parties additionally gets parameters $\sigma, d \in \mathbb{N}$ and $\mathbb{F}$, that satisfy the following preconditions:

- $d \geq 16m\sigma$,
- $|\mathbb{F}| \geq 32 \cdot 2^\sigma (T \log M \log L)^C$,

then the protocol outputs a succinct description $\langle \mathcal{Q} \rangle$ of descriptions $\langle \mathcal{Q} \rangle$ of a set of rows $\mathcal{Q} \subsetneq [M]$, and $\langle \Phi' \rangle$ of a predicate Φ',

- ***Prescribed Completeness:*** *If $\mathsf{V}_{\Delta_c RR}$ interacts with $\mathsf{P}_{\Delta_c RR}$, then $\Phi'(\boldsymbol{a}[\mathcal{Q}, :]) = 1$ iff $\boldsymbol{a} \in \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$.*
- $2^{-\sigma-2}$-***Unambiguity:*** *Suppose $\Delta_c(\mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{0})) \geq 4d$, then for any (unbounded) cheating prover strategy P^* that deviates from $\mathsf{P}_{\Delta_c RR}$ first in round $r^* \in [\ell_{\Delta_c RR}]$, with probability at least $1 - 2^{-\sigma-2}$ over the verifier's remaining coins, either $\mathsf{V}_{\Delta_c RR}$ rejects, or $\Phi'(\boldsymbol{a}[\mathcal{Q}, :]) = 0$.*
- ***Reduced Query:*** *The subset of rows $\mathcal{Q} \subsetneq [M]$ has size $|\mathcal{Q}| \leq \lceil 8\sigma \cdot \frac{M}{d} \rceil$.*
- *Regardless of the prover's strategy, the distribution of $\mathcal{Q}$ only depends on the verifier's random coins.*

The complexity of the protocol is as follows.

- $\ell_{\Delta_c RR} = \tilde{O}(1)$.
- $a_{\Delta_c RR} = b_{\Delta_c RR} = \tilde{O}(L + \mathrm{poly}(d))$.
- $\mathsf{Ptime}_{\Delta_c RR} = \mathrm{poly}(ML, T \cdot \log|\mathbb{F}|)$.
- $\mathsf{Vtime}_{\Delta_c RR} = \tilde{O}(L + \mathrm{poly}(d))$.

The bit-lengths are $|\langle \mathcal{Q} \rangle| = \tilde{O}(\mathrm{poly}(d))$ *and* $|\langle \Phi' \rangle| = \tilde{O}(L + \mathrm{poly}(d))$. *Let* $G_{\mathcal{Q}}(i, \langle \mathcal{Q} \rangle)$ *be the circuit that returns the* i*-th element in* $\mathcal{Q}$*, and* $C'(\boldsymbol{a}, \langle \Phi' \rangle)$ *be the circuit that computes* $\Phi'(\boldsymbol{a})$*, then they satisfy the following.*

- $\mathsf{size}(G_{\mathcal{Q}}) = \tilde{O}(\mathrm{poly}(d))$.
- $\mathsf{depth}(G_{\mathcal{Q}}) = \tilde{O}(1)$.
- $\mathsf{size}(C') = \tilde{O}(|\mathcal{Q}| \cdot L)$.
- $\mathsf{depth}(C') = \tilde{O}(1)$.

The verifier's verdict circuit (which outputs 0 iff it rejects amidst the protocol) satisfies

- $\mathsf{size}(\mathsf{V}_{\Delta_c RR}) = \tilde{O}(L + \mathrm{poly}(d))$.
- $\mathsf{depth}(\mathsf{V}_{\Delta_c RR}) = \tilde{O}(1)$.

4.4.2 Construction of the Reduce Protocol

We give the construction in Protocol 3.

4.4.3 Analysis of the Reduce Protocol

We first verify that the preconditions for $(\mathsf{P}_{\Delta_c \mathsf{RR}}, \mathsf{V}_{\Delta_c \mathsf{RR}})$ are satisfied. Indeed, $d > 16m\sigma$, and $|\mathbb{F}| \geq 32 \cdot 2^{\sigma}(T \log M \log L)^C$ are assumptions of Lemma 4. By Lemma 6, $|\mathcal{Q}| \leq \lceil 8\sigma \cdot \frac{M}{d} \rceil = O(\sigma \frac{M}{d})$.

Prescribed Completeness. By the prescribed completeness of $(\mathsf{P}_{\mathsf{GKR}}, \mathsf{V}_{\mathsf{GKR}})$, $\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}} \in \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$ iff $\Phi_{\mathsf{Dist}}(\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}) = 1$ (which is equivalent to $(\mathcal{S}_{\mathsf{Dist}}, \Phi_{\mathsf{Dist}}) \in \mathcal{L}'_{\boldsymbol{x}}$ by the construction of Φ_{Dist} in Lemma 3), and then by the prescribed completeness of $(\mathsf{P}_{\Delta_c \mathsf{RR}}, \mathsf{V}_{\Delta_c \mathsf{RR}})$, $\Phi'(\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}[\mathcal{Q}, :]) = 1$ iff $\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}} \in \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$. Finally, $\mathcal{S}' = \mathcal{S}_{\mathsf{Dist}}[\mathcal{Q}, :]$, so $(\mathcal{S}', \Phi') \in \mathcal{L}'_{\boldsymbol{x}}$ iff $(\mathcal{S}_{\mathsf{Dist}}, \Phi_{\mathsf{Dist}}) \in \mathcal{L}'_{\boldsymbol{x}}$.

Unambiguity. With the same proof as in Proposition 1, with all but $2^{-\sigma-1}$ probability, the following claim holds, so the unambiguity of $(\mathsf{P}_{\Delta_c \mathsf{RR}}, \mathsf{V}_{\Delta_c \mathsf{RR}})$ applies.

Claim 3. *With probability at least* $1 - 2^{-\sigma-1}$, $\Delta(\mathit{PVAL}(\boldsymbol{j}, \boldsymbol{0})) \geq 4d$.

Proof. $(\mathsf{P}_{\mathsf{GKR}}, \mathsf{V}_{\mathsf{GKR}})$ always outputs T truly random points in $(\mathbb{K}^{m+\log L})^T$. The number of points with Δ_c distance at most $4d$ from the origin is at most $(\binom{M}{4d} \cdot |\mathbb{F}|^{4d})^L$. Fixing such a point $x \in \{0,1\}^{M \times L}$, since every $\boldsymbol{j}$ is uniformly random, by Lemma 1, it is a root of the LDE of x with

Protocol 3 ($\mathsf{P}_{\mathsf{Reduce}}, \mathsf{V}_{\mathsf{Reduce}}$) Instance Reduction for Δ_c-Distance.

Input Parameters: $\sigma, d \in \mathbb{N}$, and a field $\mathbb{F}$ with $|\mathbb{F}| \geq 32 \cdot 2^{\sigma}(da\ell m)^C$ (where C is some constant).
Input: $\boldsymbol{x}, \langle \mathcal{S}_{\mathsf{Dist}} \rangle, \langle \Phi_{\mathsf{Dist}} \rangle$.
Other Parameters: $M, L \in \mathbb{N}$, $T = 8dLm$.
Ingredients:

- *The GKR protocol* ($\mathsf{P}_{\mathsf{GKR}}, \mathsf{V}_{\mathsf{GKR}}$) (Theorem 3) for checking $\Phi(\boldsymbol{a}) = 1$.

  The prover and verifier gets the description $\langle \Phi \rangle$, while the prover additionally gets the input $\boldsymbol{a}$. Both parties additionally takes in the parameter $T \in \mathbb{N}$, and in the end outputs $\boldsymbol{j} \in (\mathbb{F}^{m+\log L})^T, v \in \mathbb{F}^T$.
- *The Instance Reduction Protocol for PVAL* ($\mathsf{P}_{\Delta_c\mathsf{RR}}, \mathsf{V}_{\Delta_c\mathsf{RR}}$) (Lemma 6) for checking that $\boldsymbol{a}$ is Δ_c-d-close to $\mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$.

  Both the prover and verifier gets the input $\boldsymbol{j}, \boldsymbol{v}$, and the prover additionally gets the input $\boldsymbol{a}$. The protocol outputs descriptions $\langle \mathcal{Q} \rangle$ of a set of rows $\mathcal{Q} \subsetneq [M]$, and $\langle \Phi' \rangle$ of a predicate Φ'.

Output: $\langle \mathcal{S}' \rangle, \langle \Phi' \rangle$, new descriptions for the claim $(\mathcal{S}', \Phi') \in \mathcal{L}'$.

1: $\mathsf{P}_{\mathsf{Reduce}}$ compute the prescribed $\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}$.
2: Both parties run ($\mathsf{P}_{\mathsf{GKR}}, \mathsf{V}_{\mathsf{GKR}}$) (Theorem 3) with $T = 8dLm$ on the claim

$$\Phi_{\mathsf{Dist}}(\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}}) = 1.$$

The verifier obtains $\boldsymbol{j}, \boldsymbol{v}$.
3: Both parties run the Instance Reduction Protocol for PVAL, ($\mathsf{P}_{\Delta_c\mathsf{RR}}, \mathsf{V}_{\Delta_c\mathsf{RR}}$) (Lemma 6), on the claim

$$\boldsymbol{a}^{\mathcal{S}_{\mathsf{Dist}}} \in \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v}),$$

obtaining descriptions $\langle \mathcal{Q} \rangle$ of a set of rows $\mathcal{Q} \subsetneq [M]$, and $\langle \Phi' \rangle$ of a predicate Φ'.
4: Let $\langle \mathcal{S}' \rangle = (\langle \mathcal{S}_{\mathsf{Dist}} \rangle, \langle \mathcal{Q} \rangle)$. ▷ *Implicitly define $\mathcal{S}' := \mathcal{S}_{Dist}[\mathcal{Q}, :]$ and thus $|\mathcal{S}'| = |\mathcal{Q}| \leq |\mathcal{S}_{Dist}|/d$.*
5: **return** $\langle \mathcal{S}' \rangle, \langle \Phi' \rangle$.

probability at most $\frac{m}{|\mathbb{F}|}$. Therefore, the probability $x \in \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$ is at most $\left(\frac{m}{|\mathbb{F}|}\right)^T$. Taking a union bound over all such points, and using that $T \geq 8dLm$,

$$\Pr[\Delta_c(\mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})) \geq 4d] \leq \left(\binom{M}{4d} \cdot |\mathbb{F}|^{4d}\right)^L \cdot \left(\frac{m}{|\mathbb{F}|}\right)^T \leq 2^{-\sigma-1}. \qquad \square$$

Let P^* be a cheating prover that deviates from $\mathsf{P}_{\mathsf{Reduce}}$, then it must be deviating in either of the two protocols. By the unambiguity of $(\mathsf{P}_{\mathsf{GKR}}, \mathsf{V}_{\mathsf{GKR}})$ and $(\mathsf{P}_{\Delta_c\mathsf{RR}}, \mathsf{V}_{\Delta_c\mathsf{RR}})$ and the last claim, the probability that the prover produces $\langle \mathcal{S}' \rangle, \langle \Phi' \rangle$ such that $\Phi'(\mathcal{S}') = 1$ is at most $\epsilon_{\mathsf{GKR}} + \epsilon_{\Delta_c\mathsf{RR}} + 2^{-\sigma-1} \leq 2^{-\sigma}$.

Complexities. The GKR protocol has $\ell_{\mathsf{GKR}} = O(D_{\mathsf{Dist}} \log S_{\mathsf{Dist}})$ rounds. We can reduce the message length per-round (a_{GKR}) by increasing the number of rounds to $\ell'_{\mathsf{GKR}} = \tilde{O}(\ell \cdot D_{\mathsf{Dist}} \log S_{\mathsf{Dist}})$, which changes its complexity to $a'_{\mathsf{GKR}} = b'_{\mathsf{GKR}} = \tilde{O}(da)$. Similarly, for $(\mathsf{P}_{\Delta_c\mathsf{RR}}, \mathsf{V}_{\Delta_c\mathsf{RR}})$, we increase its number of rounds to $\ell'_{\Delta_c\mathsf{RR}} = \tilde{O}(\ell)$, and its message lengths reduce to $a'_{\Delta_c\mathsf{RR}} = b'_{\Delta_c\mathsf{RR}} = \tilde{O}(da + \mathrm{poly}(d))$. This is a simplifying assumption to make per-round messages have comparable lengths as in $(\mathsf{P}_{\mathsf{Dist}}, \mathsf{V}_{\mathsf{Dist}})$.

- $\ell_{\mathsf{Reduce}} = \ell'_{\mathsf{GKR}} + \ell'_{\Delta_c\mathsf{RR}} = \tilde{O}(\ell \cdot D_{\mathsf{Dist}} \log S_{\mathsf{Dist}})$.
- $a_{\mathsf{Reduce}} = b_{\mathsf{Reduce}} = \max(a'_{\mathsf{GKR}}, a'_{\Delta_c\mathsf{RR}}) = \tilde{O}(da + \mathrm{poly}(d))$.
- $\begin{aligned}\mathsf{Ptime}_{\mathsf{Reduce}} &= \mathsf{Ptime}_{\mathsf{GKR}} + \mathsf{Ptime}_{\Delta_c\mathsf{RR}} \\ &= \tilde{O}(dL) \cdot \mathrm{poly}(S_{\mathsf{Dist}}) + \mathrm{poly}(ML, T \cdot \log|\mathbb{F}|) \\ &= \mathrm{poly}(\log|\mathbb{F}|, M, L, d, S_{\mathsf{Dist}}).\end{aligned}$
- $\begin{aligned}\mathsf{Vtime}_{\mathsf{Reduce}} &= \mathsf{Vtime}_{\mathsf{GKR}} + \mathsf{Vtime}_{\Delta_c\mathsf{RR}} + \tilde{O}(k \cdot n + |\langle \mathcal{S}_{\mathsf{Dist}} \rangle| + |\langle \Phi_{\mathsf{Dist}} \rangle|) \\ &\leq \tilde{O}(da\ell \cdot (D_{\mathsf{Dist}} \log S_{\mathsf{Dist}} + |\langle \Phi_{\mathsf{Dist}} \rangle|) + \mathrm{poly}(d) + |\langle \mathcal{S}_{\mathsf{Dist}} \rangle| + k \cdot n).\end{aligned}$ Note that the third term in the first summation is due to the overhead of reading the input $(\mathcal{S}_{\mathsf{Dist}}, \boldsymbol{x})$.

Circuit G' for generating $\mathcal{S}'$. The sub-protocol $(\mathsf{P}_{\Delta_c\mathsf{RR}}, \mathsf{V}_{\Delta_c\mathsf{RR}})$ outputs the description $\langle \mathcal{Q} \rangle$. Our new description $\langle \mathcal{S}' \rangle = (\langle \mathcal{S}_{\mathsf{Dist}} \rangle, \langle \mathcal{Q} \rangle)$. We represent every pair $(i, \boldsymbol{q})$ as elements in $\{0,1\}^p$, where $p = \lceil \log k \rceil + b \cdot \ell$. Let $g' = |\mathcal{S}'|$ be the number of pairs to output. The Turing machine $\mathcal{M}$ takes in $(g', 1^{|\langle \mathcal{S}' \rangle|}, 1^p)$, and outputs the circuit $G' : [g'] \times \{0,1\}^{|\langle \mathcal{S}' \rangle|} \to \{0,1\}^p$. $G'(\langle \mathcal{S}' \rangle)$ enumerates $\mathcal{S}'$ as follows: for every $s \in [g']$,

- Compute the s-th element in $\mathcal{Q} \in [M]$ (using $G_{\mathcal{Q}}(s, \langle \mathcal{Q} \rangle)$), and let it be $\mathcal{Q}_s \in [M]$.
- Output the $\mathcal{Q}_s$-th pair in $\mathcal{S}_{\mathsf{Dist}}$ (using $G_{\mathsf{Dist}}(\mathcal{Q}_s, \langle \mathcal{S}_{\mathsf{Dist}} \rangle)$).

The circuit satisfies:

- $\mathsf{size}(G') = \mathsf{size}(G_{\mathcal{Q}}) + \mathsf{size}(G_{\mathsf{Dist}}) = \mathsf{size}(G_{\mathsf{Dist}}) + \tilde{O}(\mathrm{poly}(d))$.
- $\mathsf{depth}(G') = \mathsf{depth}(G_{\mathsf{Dist}}) + \tilde{O}(1)$.

The bit length satisfies $|\langle \mathcal{S}' \rangle| = \tilde{O}(\mathrm{poly}(d) + |\langle \mathcal{S}_{\mathsf{Dist}} \rangle|)$. G' is p-space uniform because $G_{\mathcal{Q}}$ is p-space uniform and $\mathcal{M}$ only needs to maintain additional pointers into the circuit G'.

Circuit C' for checking Φ'. Note that the sub-protocol $(\mathsf{P}_{\Delta_c\mathsf{RR}}, \mathsf{V}_{\Delta_c\mathsf{RR}})$ already outputs the description $\langle\Phi'\rangle$. By Lemma 6, $|\langle\Phi'\rangle| = \tilde{O}(da\ell + \mathrm{poly}(d))$, and the complexity of $C'(\boldsymbol{a}, \langle\Phi'\rangle) = \Phi'(\boldsymbol{a})$ is the following:

- $\mathsf{size}(C') = \tilde{O}(\frac{M}{d} \cdot a\ell)$,
- $\mathsf{depth}(C') = \tilde{O}(1)$.

Verdict Circuit $\mathsf{V}_{\mathsf{Reduce}}$. The verdict circuit $\mathsf{V}_{\mathsf{Reduce}}$ does not reject iff neither $\mathsf{V}_{\mathsf{GKR}}$ nor $\mathsf{V}_{\Delta_c\mathsf{RR}}$ rejects. Therefore, the circuit that implements this check would satisfy:

- $$\begin{aligned}\mathsf{size}(\mathsf{V}_{\mathsf{Reduce}}) &= \tilde{O}(\mathsf{size}(\mathsf{V}_{\mathsf{GKR}}) + \mathsf{size}(\mathsf{V}_{\Delta_c\mathsf{RR}}) + |\langle\mathcal{S}_{\mathsf{Dist}}\rangle| + |\langle\Phi_{\mathsf{Dist}}\rangle| + k \cdot n) \\ &= \tilde{O}(da\ell(D_{\mathsf{Dist}} \log S_{\mathsf{Dist}} + |\langle\Phi_{\mathsf{Dist}}\rangle|) + \mathrm{poly}(d) + |\langle\mathcal{S}_{\mathsf{Dist}}\rangle| + k \cdot n).\end{aligned}$$
- $$\begin{aligned}\mathsf{depth}(\mathsf{V}_{\mathsf{Reduce}}) &= \max(\mathsf{depth}(\mathsf{V}_{\mathsf{GKR}}), \mathsf{depth}(\mathsf{V}_{\Delta_c\mathsf{RR}})) \\ &= \tilde{O}(1).\end{aligned}$$

4.5 Analysis of Other Sub-protocols

We present of the deferred sub-protocols and analyze their properties in this section.

4.5.1 Proof of Proposition 2: the Random Continuation Protocol

The protocol $(\mathsf{P}^{\mathsf{tc}}, \mathsf{V}^{\mathsf{tc}})$ is formally given in Protocol 4.

Prescribed Completeness. This is immediate from the prescribed completeness of (P, V).

Unambiguity. We consider the following two cases.

- If the input $\boldsymbol{a} = \boldsymbol{a}^{x,\boldsymbol{q}}$, let $r^* \in [\ell]$ be the first round when P^* deviates from P^{tc}. Then P^* sends a non-prescribed message $\tilde{\boldsymbol{a}}_{r^*}^{x,\boldsymbol{q}_j^{\mathsf{Hyb}}} \neq \boldsymbol{a}_{r^*}^{x,\boldsymbol{q}_j^{\mathsf{Hyb}}}$ for some $j \in [\ell]$. With probability at most ϵ over the remaining interaction, $\mathsf{V}(x_i, \boldsymbol{q}_j^{\mathsf{Hyb}}, \tilde{\boldsymbol{a}}^{x,\boldsymbol{q}_j^{\mathsf{Hyb}}}) = 1$ by the unambiguity of (P, V), so P^* would be rejected by V^{tc} with probability at least $1 - \epsilon$.
- If the input $\boldsymbol{a} \neq \boldsymbol{a}^{x,\boldsymbol{q}}$, let $j^* \in [\ell]$ be the first round for which $\boldsymbol{a}_{j^*} \neq \boldsymbol{a}_{j^*}^{x,\boldsymbol{q}}$. Since V^{tc} checks that $\boldsymbol{a}$ and $\boldsymbol{a}^{x,\boldsymbol{q}_{j^*}^{\mathsf{Hyb}}}$ share a prefix of length $j^* \cdot a$, V^{tc} would reject at once if it is not the case that $\boldsymbol{a}_{j^*} = \boldsymbol{a}_r^{x,\boldsymbol{q}_{j^*}^{\mathsf{Hyb}}}$ for every $1 \leq r \leq j^*$. Therefore, we assume $\boldsymbol{a}_{j^*} = \boldsymbol{a}_r^{x,\boldsymbol{q}_{j^*}^{\mathsf{Hyb}}}$ for every $1 \leq r \leq j^*$. This means that $\boldsymbol{a}^{x,\boldsymbol{q}_{j^*}^{\mathsf{Hyb}}}$ is the transcript of an interaction where P^* deviates firstly in round j^* of the base protocol. By the unambiguity of (P, V), with probability at most ϵ over the remaining interaction starting from the j^*-th round, $\boldsymbol{a}^{x,\boldsymbol{q}_{j^*}^{\mathsf{Hyb}}}$ would be rejected, which means P^* would be rejected by V^{tc} with probability at least $1 - \epsilon$.

Complexities. The number of rounds is ℓ. The prover message is $a^{\mathsf{tc}} = \ell \cdot a$, and the verifier message is $b^{\mathsf{tc}} = b$. The prover time is $\mathsf{Ptime}^{\mathsf{tc}} = \ell \cdot \mathsf{Ptime}$, and the verifier time is $\mathsf{Vtime}^{\mathsf{tc}} = \mathsf{Vtime} + \ell \cdot \mathsf{size}(\mathsf{V})$.

Transcript Check Structures. By inspection, indeed the j-th message of P^{tc} is

$$\boldsymbol{m}_r = \left((\boldsymbol{a}^{x,\boldsymbol{q}_j^{\mathsf{Hyb}}})_{r\in[\ell]} \right)^\top,$$

and the verifier V^{tc} simply checks all $\boldsymbol{a}^{x,\boldsymbol{q}_j^{\mathsf{Hyb}}}$ are accepted, and that $\boldsymbol{a}^{x,\boldsymbol{q}}$ and $\boldsymbol{a}^{x,\boldsymbol{q}_j^{\mathsf{Hyb}}}$ share a prefix of length $j \cdot a$.

Protocol 4 $(\mathsf{P}^{\mathsf{tc}}, \mathsf{V}^{\mathsf{tc}})$ Random Continuation.

Input: $x \in \{0,1\}^n$, $\boldsymbol{q} \in (\{0,1\}^b)^\ell$, and a transcript $\boldsymbol{a} \in \{0,1\}^{a\times\ell}$ purported to be $\boldsymbol{a}^{x,\boldsymbol{q}}$.
Output: the verifier decides to accept or reject the claim that $\boldsymbol{a}$ is indeed $\boldsymbol{a}^{x,\boldsymbol{q}}$, and that $x \in \mathcal{L}$.

1: V^{tc} samples $\boldsymbol{q}' = (q'_1, \ldots, q'_\ell) \in (\{0,1\}^b)^\ell$.
2: **for** $r = 1, \ldots, \ell$ **do**
3: $\quad$ V^{tc} sends q'_r to P^{tc}.
4: $\quad$ **for** $j = 1, \ldots, \ell$ **do**
5: $\quad\quad$ P^{tc} let $\boldsymbol{a}_r^{x,\boldsymbol{q}_j^{\mathsf{Hyb}}} \leftarrow \mathsf{P}(x, (\boldsymbol{q}_j^{\mathsf{Hyb}})_{\le r})$, where $\boldsymbol{q}_j^{\mathsf{Hyb}} = (q_1, \ldots, q_j, q'_{j+1}, \ldots, q'_\ell)$.
6: $\quad$ P^{tc} sends $\left\{\boldsymbol{a}_r^{x,\boldsymbol{q}_j^{\mathsf{Hyb}}}\right\}_{j\in[r]}$ to V^{tc}.
7: V^{tc} let $\boldsymbol{a}^{x,\boldsymbol{q}_j^{\mathsf{Hyb}}} \leftarrow (\boldsymbol{a}_1^{x,\boldsymbol{q}_j^{\mathsf{Hyb}}}, \ldots, \boldsymbol{a}_\ell^{x,\boldsymbol{q}_j^{\mathsf{Hyb}}})$ for every $j \in [\ell]$.
8: V^{tc} accepts iff $\bigwedge_{j\in[\ell]} \left[\mathsf{V}(x, \boldsymbol{q}_j^{\mathsf{Hyb}}, \boldsymbol{a}^{x,\boldsymbol{q}_j^{\mathsf{Hyb}}}) = 1\right]$ and that for every $j \in [\ell]$, $\boldsymbol{a}$ and $\boldsymbol{a}^{x,\boldsymbol{q}_j^{\mathsf{Hyb}}}$ share a prefix of length $j \cdot a$.

4.5.2 Proof of Lemma 5: the Explicit UIP for the associated language

We construct $(\mathsf{P}_{\mathcal{L}'_{\boldsymbol{x}}}, \mathsf{V}_{\mathcal{L}'_{\boldsymbol{x}}})$, an explicit UIP for the associated language $\mathcal{L}'_{\boldsymbol{x}}$, which takes as input $\boldsymbol{x}$ and $\langle\mathcal{S}\rangle, \langle\Phi\rangle$ and verifies the claim $(\mathcal{S}, \Phi) \in \mathcal{L}'_{\boldsymbol{x}}$. It consists of the following two steps:

1. $\mathsf{P}_{\mathcal{L}'_{\boldsymbol{x}}}$ sends the prescribed transcript matrix $\boldsymbol{a}^{\mathcal{S}} \in \{0,1\}^{|\mathcal{S}|\times(a\ell)}$ to $\mathsf{V}_{\mathcal{L}'_{\boldsymbol{x}}}$.
2. $\mathsf{V}_{\mathcal{L}'_{\boldsymbol{x}}}$ enumerates $G(s, \langle\mathcal{S}\rangle)$ for every $s \in [|\mathcal{S}|]$.
3. Both parties execute $(\mathsf{P}^{\mathsf{tc}}, \mathsf{V}^{\mathsf{tc}})$ explicitly on the input $((i, \boldsymbol{q}), \boldsymbol{a}^{\mathcal{S}}[(i, \boldsymbol{q}), :])$, for every $(i, \boldsymbol{q}) \in \mathcal{S}$ in parallel.

   By Remark 10, the messages sent by $\mathsf{P}_{\mathcal{L}'_{\boldsymbol{x}}}$ constitute the transcript matrix $\boldsymbol{a} = \boldsymbol{a}^{\mathcal{S}^{\mathsf{Hyb}}}$, where $\mathcal{S}^{\mathsf{Hyb}}$ is the set of hybrid pairs in constructed between $\mathcal{S}$ and $\boldsymbol{q}' \in \{0,1\}^{b\ell}$, the random coin sampled by $\mathsf{V}_{\mathcal{L}'_{\boldsymbol{x}}}$.
4. $\mathsf{V}_{\mathcal{L}'_{\boldsymbol{x}}}$ accepts iff every execution of $(\mathsf{P}^{\mathsf{tc}}, \mathsf{V}^{\mathsf{tc}})$ in $\boldsymbol{a}$ is accepted, $G(\boldsymbol{a}^{\mathcal{S}}, \langle\Phi\rangle) = 1$ and $\boldsymbol{a}[\mathcal{S}, :] = \boldsymbol{a}^{\mathcal{S}}$.

Let $\mathcal{I} \coloneqq \{(i, \boldsymbol{q}) : (i, \boldsymbol{q}) \in \mathcal{S}\}$. Recall that $(\mathcal{S}, \Phi) \in \mathcal{L}'_{\boldsymbol{x}}$ iff $(\Phi(\boldsymbol{a}^{\mathcal{S}}) = 1) \wedge (\boldsymbol{x}|_{\mathcal{I}} \in \mathcal{L}^{\otimes|\mathcal{S}|})$.

Prescribed Completeness. $\mathsf{P}_{\mathcal{L}'_{\boldsymbol{x}}}$ sends the verifier the prescribed $\boldsymbol{a}^{\mathcal{S}}$ in the first step, and executes $\mathsf{P}_{\mathcal{L}'_{\boldsymbol{x}}}$ honestly for every $(i, \boldsymbol{q}) \in \mathcal{S}$ in parallel, so $\boldsymbol{a} = \boldsymbol{a}^{\mathcal{S}^{\mathsf{Hyb}}}$ (and thus $\boldsymbol{a}[\mathcal{S}, :] = \boldsymbol{a}^{\mathcal{S}}$). Since $\boldsymbol{a}^{\mathcal{S}}$ is the prescribed transcript matrix, all executions of $(\mathsf{P}^{\mathsf{tc}}, \mathsf{V}^{\mathsf{tc}})$ in $\boldsymbol{a}$ are accepted iff $\boldsymbol{x}|_{\mathcal{I}} \in \mathcal{L}^{\otimes|\mathcal{S}|}$. Therefore, the verifier's check passes exactly when $(\mathcal{S}, \Phi) \in \mathcal{L}'_{\boldsymbol{x}}$.

Unambiguity. Suppose that P^* deviates from $\mathsf{P}_{\mathcal{L}'_{\boldsymbol{x}}}$ in the first step, i.e. it sends some $\boldsymbol{a}^* \neq \boldsymbol{a}^{\mathcal{S}}$ to $\mathsf{V}_{\mathcal{L}'_{\boldsymbol{x}}}$. This means for some row-index $(i, \boldsymbol{q}) \in \mathcal{S}$, $\boldsymbol{a}^*[(i, \boldsymbol{q})] \neq \boldsymbol{a}^{i,\boldsymbol{q}}$. If P^* does not deviate in the rest of the protocol, then by prescribed completeness of $(\mathsf{P}^{\mathsf{tc}}, \mathsf{V}^{\mathsf{tc}})$ (Proposition 2), $\mathsf{V}_{\mathcal{L}'_{\boldsymbol{x}}}$ rejects on the input $((i, \boldsymbol{q}), \boldsymbol{a}^*[(i, \boldsymbol{q})])$, and thus $\mathsf{V}_{\mathcal{L}'_{\boldsymbol{x}}}$ rejects. Otherwise, P^* deviates in one of the ℓ rounds while executing $(\mathsf{P}^{\mathsf{tc}}, \mathsf{V}^{\mathsf{tc}})$ in parallel. By the ϵ-unambiguity of $(\mathsf{P}^{\mathsf{tc}}, \mathsf{V}^{\mathsf{tc}})$ (Proposition 2), that execution of $(\mathsf{P}^{\mathsf{tc}}, \mathsf{V}^{\mathsf{tc}})$ is accepted with probability at most ϵ, so $\mathsf{V}_{\mathcal{L}'_{\boldsymbol{x}}}$ accepts with probability at most ϵ.

Complexities. Let $g = |\mathcal{S}|$. The number of rounds is $\ell_{\mathcal{L}'_{\boldsymbol{x}}} = \ell + 1$, The prover message (per round) is of length $a_{\mathcal{L}'_{\boldsymbol{x}}} = g\ell \cdot a$. This can be seen by noting that the message in the first step is of length $g \cdot a\ell$ and the rest are of length $g \cdot a\ell$ per round as well (since g instances of the protocol in Proposition 2 are in parallel). The verifier message (per round) is of length $b_{\mathcal{L}'_{\boldsymbol{x}}} = b$. The prover runtime is $\mathsf{Ptime}_{\mathcal{L}'_{\boldsymbol{x}}} = O(g\ell \cdot \mathsf{Ptime})$, because it simply runs g instances P^{tc} in parallel (each of which has runtime $\ell \cdot \mathsf{Ptime}$), and the verifier runtime is $\mathsf{Vtime}_{\mathcal{L}'_{\boldsymbol{x}}} = O(g\ell \cdot \mathsf{Vtime} + g\mathsf{size}(G) + \mathsf{size}(C))$. This is because it first computes $G(s, \langle \mathcal{S} \rangle)$ for every $s \in [g]$, then runs g instances of V^{tc} in parallel (each of which has runtime $\ell \cdot \mathsf{Vtime}$), and finally evaluates $C(\boldsymbol{a}^{\mathcal{S}}, \langle \Phi \rangle)$ (which takes time $\mathsf{size}(C)$).

The verifier's decision circuit $\mathsf{V}_{\mathcal{L}'_{\boldsymbol{x}}}$ has the following complexities:

- $\mathsf{size}(\mathsf{V}_{\mathcal{L}'_{\boldsymbol{x}}}) = \tilde{O}(g \cdot \ell \cdot \mathsf{size}(\mathsf{V}) + g\mathsf{size}(G) + \mathsf{size}(C))$.
- $\mathsf{depth}(\mathsf{V}_{\mathcal{L}'_{\boldsymbol{x}}}) = \max(\mathsf{depth}(G) + D, \mathsf{depth}(C)) + \tilde{O}(1)$.

5 Doubly-Efficient Interactive Proofs

5.1 A New UIP for Verifying Space-Bounded Computations

We demonstrate the power of Batch-UIP by constructing a doubly-efficient interactive proof for delegating space-bounded computations. Denote by $\mathsf{DTISP}(T, S)$ the class of languages decidable by a Turing machine in time $T(n)$ and space $S(n)$.

Theorem 8 (Doubly-efficient Interactive Proof for $\mathsf{DTISP}(T, S)$)**.** *For any $T(n) \leq n^{O\left(\left(\frac{\log n}{\log\log n}\right)^{1/2}\right)}$ and $S(n) = \mathrm{poly}(n)$, any $\sigma = \sigma(n) = \mathrm{polylog}(n)$, any $\delta = \delta(n) \in (0, 1)$, and any language $L \in \mathsf{DTISP}(T, S)$, there exists a UIP with the following properties:*

- *The soundness error is $2^{-\sigma} \cdot (1/\delta) \cdot (\log n)^{O(1/\delta)}$.*
- *The number of rounds is $(\log n)^{O(1/\delta)}$.*
- *The prover and verifier message length per-round is at most $T^{\delta} \cdot S(n) \cdot (\log n)^{O(1/\delta^2)}$.*
- *The prover runtime is $T(n) \cdot \mathrm{poly}(n) \cdot (\log n)^{O(1/\delta^2)}$.*
- *The verifier runtime is $T^{O(\delta)} \cdot S(n)^2 \cdot \mathrm{poly}(n) \cdot (\log n)^{O(1/\delta^2)}$.*

Setting $\delta = \sqrt{\frac{\log\log n}{\log n}}$ (so that $(\log n)^{O(1/\delta^2)} = \mathrm{poly}(n)$), and $\sigma = \Theta(\sqrt{\log n \cdot \log\log n})$, we have the following corollary.

Corollary 2 (Formal statement of Corollary 1)**.** *Let language $L \in \mathsf{DTISP}(T, S)$ where $T(n) = n^{O\left(\left(\frac{\log n}{\log\log n}\right)^{1/2}\right)}$ and $S(n) = \mathrm{poly}(n)$, there exists a UIP with the following properties:*

- *The soundness error is* $o(1)$.
- *The number of rounds is* $o(n)$.
- *The prover and verifier message length per-round is* $S(n) \cdot \mathrm{poly}(n)$.
- *The prover runtime is* $T(n) \cdot \mathrm{poly}(n)$.
- *The verifier runtime is* $S(n)^2 \cdot \mathrm{poly}(n)$.

5.1.1 Augmenting Interactive Proofs of Transition of Turing Machines

Our approach is similar to that in [RRR16]. Fix $\mathcal{M}$ to be some Turing machine with $T(n)$ time and $S(n)$ space. For every $t \leq T(n)$, let the language $\mathcal{L}_t$ consist of all tuples $(x, w_1, w_2) \in \{0,1\}^n \times \{0,1\}^{O(S(n))} \times \{0,1\}^{O(S(n))}$ such that $\mathcal{M}$ transitions from configuration w_1 to configuration w_2 in exactly t steps when the input is x.

The crucial observation is captured in the following *Augmentation Lemma* (Lemma 7): let $k = k(n) \in \mathrm{poly}(n)$ be some batch size parameter. The language $\mathcal{L}_{k \cdot t}$ can be expressed as a batch of $\mathcal{L}_t$. Therefore, an efficient batching protocol such as Theorem 6 allows the verifier to verify $(k \cdot t)$-step transitions by batch-verifying t-step transitions, at the cost of a small increase in complexity. By repeated application of Lemma 7, the verifier gradually increase the transition length t that it can verify, until it reaches $\mathcal{L}_{T(n)}$.

Lemma 7 (Augmentation Lemma)**.** *Let* $\mathcal{M}$ *be a Turing machine with* $T(n)$ *time and* $S(n) \geq n$ *space. Let* $t = t(n)$ *be some time bound, and let* $k = k(n) \in \mathrm{poly}(n)$ *be some batch size parameter. Assume that* $\mathcal{L}_t$ *has an* ϵ*-sound* $(\ell, a, b, \mathsf{Ptime}, \mathsf{Vtime})$ *UIP* (P, V)*. For every* $\sigma \in \mathbb{N}$*, there exists a protocol* ϵ_A*-unambiguous* $(\ell', a', b', \mathsf{Ptime}', \mathsf{Vtime}')$ *UIP* $(\mathsf{P}', \mathsf{V}')$ *for* $\mathcal{L}_{k \cdot t}$*. With* $\tilde{O}$ *hiding* $\mathrm{poly}(\sigma) \cdot \mathrm{polylog}(k, S(n), a, \ell)$ *factors, it has the following properties:*

- $\epsilon' = 2 \log k (\epsilon + 2^{-\sigma})$.
- $\ell' = \tilde{O}(\ell \cdot \mathsf{depth}(\mathsf{V}) \log \mathsf{size}(\mathsf{V}))$.
- $a' = b' = \tilde{O}(a\ell + \mathrm{poly}(\ell)) + O(k \cdot S(n))$.
- $\mathsf{Ptime}' = \tilde{O}(k \cdot \ell \cdot \mathsf{Ptime} + \mathrm{poly}(k, a, \ell, \mathsf{size}(\mathsf{V})))$.
- $\mathsf{Vtime}' = \tilde{O}(\ell \cdot \mathsf{Vtime} + a\ell^2(kS(n) + \mathsf{depth}(\mathsf{V}) \log \mathsf{size}(\mathsf{V})) + a^2 \cdot \mathrm{poly}(\ell))$.

Furthermore,

- $\mathsf{size}(\mathsf{V}') = \tilde{O}(\ell \cdot \mathsf{size}(\mathsf{V}) + a\ell^2(kS(n) + \mathsf{depth}(\mathsf{V}) \log \mathsf{size}(\mathsf{V})) + a^2 \cdot \mathrm{poly}(\ell))$.
- $\mathsf{depth}(\mathsf{V}') = \mathsf{depth}(\mathsf{V}) + \tilde{O}(1)$.

Proof Sketch. The protocol $(\mathsf{P}', \mathsf{V}')$ on input $(x, w_0, w_{k \cdot t})$ is constructed as follows.

1. P' runs $\mathcal{M}$ on input x and starting configuration w_1 for $k \cdot t$ steps, obtaining all intermediate configurations $w_1, w_2, \ldots, w_{k \cdot t}$. *Note that if* P' *already has access to* $w_1, w_2, \ldots, w_{k \cdot t}$*, then it can skip this step.*

2. P' sends $w_t, w_{2t}, \ldots, w_{kt}$ to $\mathsf{V}_{k\cdot t}$.

3. Both parties run the protocol $(\mathsf{P}_{\mathsf{Batch\text{-}UIP}}, \mathsf{V}_{\mathsf{Batch\text{-}UIP}})$ to batch (P, V), with security parameter σ, that batch-verifies $\mathcal{L}_t^{\otimes k}$, on input $\{(x, w_t, w_{2t})\}_{t=0}^{k}$.

4. The verifier V' accepts iff the batched protocol accepts.

Prescribed Completeness. This is immediate.

ϵ_A-Unambiguity. A deviating prover can either deviate by sending some incorrect configuration w_t, which leads to at least one incorrect t-step transition and hence a false statement in $\mathcal{L}_t$, or by deviating in the batched protocol. In either case, the unambiguity of $(\mathsf{P}_{\mathsf{Batch\text{-}UIP}}, \mathsf{V}_{\mathsf{Batch\text{-}UIP}})$ ensures that the verifier accepts with probability at most ϵ_A.

Complexities. The complexities remain the same as in Theorem 6, with the input (n in Theorem 6) being replaced by $O(S)$, except that $a_A = b_A \leq \tilde{O}(a\ell + \mathrm{poly}(\ell)) + O(k \cdot S)$, where the $O(k \cdot S)$ term comes from the prover sending the configurations $w_t, w_{2t}, \ldots, w_{kt}$. □

5.1.2 Our Construction of the Doubly-efficient UIP for $\mathsf{DTISP}(T, S)$

For the base case, we have the following (trivial) proof system for verifying one transition. On input (x, w_0, w_1), the prover and verifier does not communicate, and the verifier simply simulate $\mathcal{M}$ on x with w_0 by one step and verify it actually transitions to w_1, which requires $O(S(n))$ time.

Proposition 4 (A UIP for verifying one transition)**.** *There exists a $(\ell_0, a_0, b_0, \mathsf{Ptime}_0, \mathsf{Vtime}_0)$ UIP $(\mathsf{P}_0, \mathsf{V}_0)$ for $\mathcal{L}_1$ with the following properties:*

- *The unambiguity error is $\epsilon_0 = 0$.*
- *The number of rounds is $\ell_0 = 0$.*
- *The prover message length is $a_0 = 0$.*
- *The verifier message length is $b_0 = 0$.*
- *The prover does not do anything, so its runtime is 0.*
- *The verifier runs in time $O(S(n))$.*

Furthermore, the verifier's decision circuit V_0 has the following properties:

- *The circuit size is $\mathsf{size}(\mathsf{V}_0) = O(S(n))$.*
- *The circuit depth is $\mathsf{depth}(\mathsf{V}_0) = O(1)$.*

Using Lemma 7, we can show the following claim by induction.

Proposition 5. *Let $k = k(n) \in \mathrm{poly}(n), \gamma = \gamma(n) \in O\left(\sqrt{\frac{\log n}{\log\log n}}\right), \sigma = \sigma(n) \in \mathrm{polylog}(n)$ be parameters. There exist universal constants C_1, C_2, C_3 such that for each $i \leq \gamma$, there exists a $(\ell_i, a_i, b_i, \mathsf{Ptime}_i, \mathsf{Vtime}_i)$ UIP $(\mathsf{P}_i, \mathsf{V}_i)$ for $\mathcal{L}_{k^i}$, and for large n, the following holds:*

1. $\max(k, S(n), a_i, \ell_i, \mathsf{size}(\mathsf{V}_i)) = O(n^c)$, *where*

$$c = \limsup_{n\to\infty} \lceil 8C_1C_3 + 2\log_n(kS(n)) + 1 \rceil$$

is a constant independent of n *and* i *(since* $k, S(n) \in \mathrm{poly}(n)$*).*

2. *Regarding the complexities of* $(\mathsf{P}_i, \mathsf{V}_i)$*:*

   - *The unambiguity error is* $\epsilon_i \le i \cdot (2\log k)^i 2^{-\sigma}$.
   - *The number of rounds is* $\ell_i \le \log^{3C_1 i} n$.
   - *The prover and verifier message length per round is* $a_i = b_i \le kS(n) \cdot \log^{4C_1C_3i^2} n$.
   - *The prover runtime is* $\mathsf{Ptime}_i = k^i \cdot n^{C_2C_3} \log^{3C_1i^2} n$.
   - *The verifier runs in time* $\mathsf{Vtime}_i \le (kS(n))^2 \log^{8C_1C_3i^2} n$.

   Furthermore, the verifier's decision circuit V_i *has the following properties:*

   - *The circuit size is* $\mathsf{size}(\mathsf{V}_i) \le (kS(n))^2 \log^{8C_1C_3i^2} n$.
   - *The circuit depth is* $\mathsf{depth}(\mathsf{V}_i) \le i \cdot \log^{C_1} n$.

To simplify our calculation, we use the following loose bounds on the constants involved.

Remark 11 (Determining the constants C_1, C_2, C_3.). The hidden constants in Lemma 7 determine the constants C_1, C_2, C_3, as follows.

- Consider the upper bound of all $\mathrm{poly}(\sigma)\cdot\mathrm{polylog}(k, S(n), a_i, \ell_i)$ factors hidden in $\tilde{O}$ of Lemma 7. Assume that indeed $k, S(n), a_i, \ell_i$ are each upper-bounded by n^c for large n, and that $\sigma = \mathrm{polylog}(n)$, then for some constant c_1 and large n, the following holds,

  $$\mathrm{poly}(\sigma)\mathrm{polylog}(k, S(n), a_i, \ell_i) \le (c\log n)^{c_1} \le (\lceil 8C_1C_3 + 2\log_n(kS(n)) + 1 \rceil \log n)^{c_1}.$$

  For large n,

  $$\log n \ge \lceil 8C_1C_3 + 2\log_n(kS(n)) + 1 \rceil,$$

  so the above is upper-bounded by $\log^{2c_1} n$, irrespectively of the value of c (or C_1). Therefore, if we take $C_1 \coloneqq 2c_1 + 1$, then we can replace all occurrence of $\mathrm{poly}(\sigma) \cdot \mathrm{polylog}(k, S(n), a_i, \ell_i)$ with $\log^{C_1} n$.

- There exists a constant c_2 such that the term $\mathrm{poly}(k, a_i, \ell_i, \mathsf{size}(\mathsf{V}_i)) < (ka_i\ell_i\mathsf{size}(\mathsf{V}_i))^{c_2}$ in the Ptime recurrence relation. If $\max(k, S(n), a_i, \ell_i, \mathsf{size}(\mathsf{V}_i)) \le n^c$ for large n, then $(ka_i\ell_i\mathsf{size}(\mathsf{V}_i))^{c_2} \le n^{4cc_2}$ for large n.

  Therefore, we choose $C_2 \coloneqq 4cc_2$.

- C_3 is simply defined to be the power of ℓ in the terms $\mathrm{poly}(\ell)$ appearing in the recurrence relations.

Proof of Proposition 5. Let $C_1 \coloneqq 2c_1 + 1$, $C_2 \coloneqq 4cc_2$ be the constants as defined in Remark 11. (Recall that $c = \limsup_{n\to\infty} \lceil 8C_1C_3 + 2\log_n(kS(n)) + 1 \rceil$.)

We prove the following two claims simultaneously by induction on i.

1. $\max(k, S(n), a_i, b_i, \mathsf{size}(\mathsf{V}_i)) \le n^c$ for large n.
2. The bounds about each of ϵ_i, ℓ_i, a_i, b_i, Ptime_i, Vtime_i, $\mathsf{size}(\mathsf{V}_i)$, $\mathsf{depth}(\mathsf{V}_i)$ hold.

The base case, $i = 0$, corresponds to a transition of one step and holds trivially as in Proposition 4.

Inductive Step. Assume that we have a UIP $(\mathsf{P}_i, \mathsf{V}_i)$ for $\mathcal{L}_{k^i}$ with the stated complexity bounds. Note that an important consequence of $i \le \gamma \le \sqrt{\frac{\log n}{\log\log n}}$ is that $i^2 \log\log n < \log n$.

By our second inductive hypothesis, the following hold for large n.

1. $\mathsf{depth}(\mathsf{V}_i) \le i \cdot \log^{C_1} n$.
2. From $\mathsf{size}(\mathsf{V}_i) \le O((kS(n))^2 \log^{8C_1C_3i^2} n)$ and $\log(kS(n)) = \log_n(kS(n)) \log n$ we deduce

$$\begin{aligned}\log \mathsf{size}(\mathsf{V}_i) &\le O(1) + 2\log(kS(n)) + 8C_1C_3i^2 \log\log n \\ &= O(1) + (2\log_n(kS(n)) + 8C_1C_3i^2)\log n \\ &< (2\log_n(kS(n)) + 8C_1C_3i^2 + 1)\log n < \log^2 n.\end{aligned}$$

Therefore, for large n, $\mathsf{depth}(\mathsf{V}_i) \log \mathsf{size}(\mathsf{V}_i) < i \cdot \log^{C_1} n \log^2 n < \log^{C_1+3} n$.

Complexities Applying Lemma 7 with batch size k and security parameter σ, we obtain a UIP $(\mathsf{P}_{i+1}, \mathsf{V}_{i+1})$ for $\mathcal{L}_{k^{i+1}}$. As in Remark 11, the first inductive hypothesis implies that all $\tilde{O}(1)$ terms hidden can be upper bounded by $\log^{C_1} n$ when n is large. We work out the recurrence relations to obtain the following.

- $\mathsf{depth}(\mathsf{V}_{i+1}) = \mathsf{depth}(\mathsf{V}_i) + \tilde{O}(1) \le \mathsf{depth}(\mathsf{V}_i) + \log^{C_1} n \le (i+1) \cdot \log^{C_1} n$.
- $\ell_{i+1} = \ell_i \cdot (\mathsf{depth}(\mathsf{V}_i) \log \mathsf{size}(\mathsf{V}_i) \log^{C_1} n) \le \log^{3C_1 i + (2C_1+3)} n < \log^{3C_1(i+1)} n$, (w.l.o.g. $C_1 > 3$).
- $$\begin{aligned}a_{i+1} = b_{i+1} &\le a_i \cdot \ell_i \cdot \log^{C_1} n + \ell_i^{C_3} \log^{C_1} n + O(k \cdot S(n)) \\ &= k \cdot S(n) \cdot (\log^{4C_1C_3i^2 + C_1 i + C_1} n + \log^{3C_1C_3 i + C_1} n + O(1)) \\ &< k \cdot S(n) \cdot \log^{4C_1C_3(i+1)^2} n. \text{ (w.l.o.g. } C_3 \ge 1).\end{aligned}$$
- $$\begin{aligned}\mathsf{Vtime}_{i+1} &\le \mathsf{Vtime}_i \cdot (\ell_i \cdot \log^{C_1} n) + \left(a_i \ell_i^2 (kS(n) + \mathsf{depth}(\mathsf{V}_i) \log \mathsf{size}(\mathsf{V}_i)) + a_i^2 \ell_i^{C_3}\right) \log^{C_1} n \\ &\le (kS(n))^2 \log^{8C_1C_3i^2 + 3C_1 i + C_1} n \\ &\quad + kS(n)(kS(n) + \log^{C_1+3} n) \log^{4C_1C_3i^2 + 6C_1 i} n \cdot \log^{C_1} n \\ &\quad + (kS(n))^2 \cdot \log^{8C_1C_3i^2} n \cdot \log^{C_1C_3 i + C_1} n \\ &< (kS(n))^2 \cdot \log^{8C_1C_3(i+1)^2} n.\end{aligned}$$
- Similarly, $\mathsf{size}(\mathsf{V}_{i+1}) < (kS(n))^2 \cdot \log^{8C_1C_3(i+1)^2} n$.
- $$\begin{aligned}\mathsf{Ptime}_{i+1} &\le \mathsf{Ptime}_i \cdot (k \cdot \ell_i \cdot \log^{C_1} n) + \mathrm{poly}(k, a_i, \ell_i, \mathsf{size}(\mathsf{V}_i)) \\ &\le \mathsf{Ptime}_i \cdot k \log^{3C_1 i^2 + C_1} n \log^{C_1} n + n^{C_2C_3} \\ &< k^{i+1} n^{C_2C_3} \log^{3C_1(i+1)^2 + C_1} n.\end{aligned}$$

Recall that $c = \lceil \limsup_{n\to\infty} 8C_1C_3 + 2\log_n(kS(n)) + 1 \rceil$. The condition in the induction follows from that $\max(k, S(n), a_{i+1}, b_{i+1}, \mathsf{size}(\mathsf{V}_{i+1})) = (kS(n))^2 \cdot \log^{8C_1C_3(i+1)^2} n \le n^c$ for large n, since $(i+1)^2 \le \gamma^2 \le \frac{\log n}{\log\log n}$ and $(\log n)^{\frac{\log n}{\log\log n}} = n$.

Finally, the unambiguity error satisfies $\epsilon_{i+1} = 2\log k(\epsilon_i + 2^{-\sigma}) \le 2\log k(i(2\log k)^i 2^{-\sigma} + 2^{-\sigma}) < (i+1)(2\log k)^{i+1} 2^{-\sigma}$. □

Let $\delta(n) \in (0,1)$ be a parameter. Theorem 8 now follows from Proposition 5 by setting $k = T^{\delta}$ and $\gamma = \frac{1}{\delta}$. In this case, $\mathcal{L}_{k^\gamma} = \mathcal{L}$ and it has a UIP with the following complexities:

- $\epsilon_\gamma \leq \gamma(2\log n)^\gamma 2^{-\sigma} = 2^{-\sigma} \cdot (1/\delta) \cdot (\log n)^{O(1/\delta)}$.
- $\ell_\gamma \leq \log^{3C_1\gamma} n = (\log n)^{O(1/\delta)}$.
- $a_\gamma = b_\gamma \leq k \cdot S(n) \log^{4C_1C_3\gamma^2} n = T^\delta \cdot S(n) \cdot (\log n)^{O(1/\delta^2)}$.
- $\mathsf{Ptime}_\gamma \leq k^\gamma n^{C_2C_3} \log^{3C_1\gamma^2} n = T^\delta \cdot \mathrm{poly}(n) \cdot (\log n)^{O(1/\delta^2)}$.
- $\mathsf{Vtime}_\gamma = (kS(n))^2 \log^{8C_1C_3\gamma^2} n = T^{2\delta} \cdot S(n)^2 \cdot (\log n)^{O(1/\delta^2)}$.

6 IPP for PVAL with Column Distance

The work of [RR20] constructed an IPP (same as Definition 11 without the unambiguous soundness property) for the PVAL problem with respect to the Hamming distance. We use their ideas to construct a UIPP for a slightly different formulation of the PVAL problem which has a different distance measure. The same ideas also result in an instance reduction protocol for PVAL. In particular, we use the column distance Δ_c (Definition 1), as opposed to the Hamming distance.

We impose the additional constraint on our IPP to be a UIP, as it is used as a building block in our batch UIP protocol.[20] Thus, we also prove the unambiguous soundness of our protocol. We now present our main result for UIPPs.

Theorem 9 (UIPP for PVAL with column distance; Restatement of Lemma 6)**.** *Suppose σ, d, m, $\log L \in \mathbb{N}$, $M = 2^m$, and $\mathbb{F}$ is a field. Let $\tilde{O}$ be omitting $\mathrm{polylog}(|\mathbb{F}|, M, L)$ factors. Suppose $T \geq 8dLm = \tilde{O}(dL)$. Let $\boldsymbol{j} = (\boldsymbol{j}_1, \ldots, \boldsymbol{j}_T) \in (\mathbb{F}^{m+\log L})^T$ and $\boldsymbol{v} = (v_1, \ldots, v_T) \in \mathbb{F}^T$.*

There exists a constant C, and a public-coin unambiguous IPP $(\mathsf{P}_{\Delta_c RR}, \mathsf{V}_{\Delta_c RR})$ for $\mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$, where both the prover and verifier gets the input $\boldsymbol{j}, \boldsymbol{v}$, while the prover additionally gets the input $\boldsymbol{a} \in \mathbb{F}^{M \times L}$, such that if both parties additionally gets parameters $\sigma, d \in \mathbb{N}$ and $\mathbb{F}$, that satisfy the following preconditions:

- $d \geq 16m\sigma$,
- $|\mathbb{F}| \geq 32 \cdot 2^\sigma (T \log M \log L)^C$,

then the protocol outputs a succinct description $\langle \mathcal{Q} \rangle$ of descriptions $\langle \mathcal{Q} \rangle$ of a set of rows $\mathcal{Q} \subsetneq [M]$, and $\langle \Phi' \rangle$ of a predicate Φ',

- ***Prescribed Completeness:*** *If $\mathsf{V}_{\Delta_c RR}$ interacts with $\mathsf{P}_{\Delta_c RR}$, then $\Phi'(\boldsymbol{a}[\mathcal{Q}, :]) = 1$ iff $\boldsymbol{a} \in \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$.*
- $2^{-\sigma-2}$-***Unambiguity:*** *Suppose $\Delta_c(\mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{0})) \geq 4d$, then for any (unbounded) cheating prover strategy P^* that deviates from $\mathsf{P}_{\Delta_c RR}$ first in round $r^* \in [\ell_{\Delta_c RR}]$, with probability at least $1 - 2^{-\sigma-2}$ over the verifier's remaining coins, either $\mathsf{V}_{\Delta_c RR}$ rejects, or $\Phi'(\boldsymbol{a}[\mathcal{Q}, :]) = 0$.*

[20] We note that [RR20] did not require their IPP to be unambiguous. Their protocol, despite being slightly unambiguous, does not have an exponentially small unambiguity error. Given that parallel repetition does not reduce unambiguity error, we have to modify the protocol.

- ***Reduced Query:*** *The subset of rows $\mathcal{Q} \subsetneq [M]$ has size $|\mathcal{Q}| \leq \lceil 8\sigma \cdot \frac{M}{d} \rceil$.*

- *Regardless of the prover's strategy, the distribution of $\mathcal{Q}$ only depends on the verifier's random coins.*

The complexity of the protocol is as follows.

- $\ell_{\Delta_c RR} = \tilde{O}(1)$.
- $a_{\Delta_c RR} = b_{\Delta_c RR} = \tilde{O}(L + \text{poly}(d))$.
- $\mathsf{Ptime}_{\Delta_c RR} = \text{poly}(ML, T \cdot \log|\mathbb{F}|)$.
- $\mathsf{Vtime}_{\Delta_c RR} = \tilde{O}(L + \text{poly}(d))$.

The bit-lengths are $|\langle \mathcal{Q} \rangle| = \tilde{O}(\text{poly}(d))$ and $|\langle \Phi' \rangle| = \tilde{O}(L + \text{poly}(d))$. Let $G_{\mathcal{Q}}(i, \langle \mathcal{Q} \rangle)$ be the circuit that returns the i-th element in $\mathcal{Q}$, and $C'(\boldsymbol{a}, \langle \Phi' \rangle)$ be the circuit that computes $\Phi'(\boldsymbol{a})$, then they satisfy the following.

- $\mathsf{size}(G_{\mathcal{Q}}) = \tilde{O}(\text{poly}(d))$.
- $\mathsf{depth}(G_{\mathcal{Q}}) = \tilde{O}(1)$.
- $\mathsf{size}(C') = \tilde{O}(|\mathcal{Q}| \cdot L)$.
- $\mathsf{depth}(C') = \tilde{O}(1)$.

The verifier's verdict circuit (which outputs 0 iff it rejects amidst the protocol) satisfies

- $\mathsf{size}(\mathsf{V}_{\Delta_c RR}) = \tilde{O}(L + \text{poly}(d))$.
- $\mathsf{depth}(\mathsf{V}_{\Delta_c RR}) = \tilde{O}(1)$.

6.1 Overview of the RR IPP for PVAL.

The setup for the RR IPP is as follows:

- The verifier has oracle access to a function $\boldsymbol{a} : \{0,1\}^m \mapsto \mathbb{F}$ (note the notation $\boldsymbol{a}$ often refers to the truth table of the function which is a string in $\mathbb{F}^{2^m}$), where $\mathbb{F}$ is a field of characteristic 2. The verifier also has elements $\boldsymbol{j} \in (\mathbb{F}^m)^T$ and $\boldsymbol{v} \in \mathbb{F}^T$.
- $\hat{\boldsymbol{a}} : \mathbb{F}^m \mapsto \mathbb{F}$ is the multilinear extension of $\boldsymbol{a}$.
- The prover and verifier want to run a UIP protocol to verify whether $\hat{\boldsymbol{a}}(\boldsymbol{j}) = \boldsymbol{v}$ or if $\hat{\boldsymbol{a}}$ is δ far from $\mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$ by making only $\tilde{O}\left(\frac{1}{\delta}\right)$ oracle queries to the function $\boldsymbol{a}$.

Since the verifier only cares about the case when the distance is at least δ, it could hope to query $\boldsymbol{a}$ on just $O(\frac{1}{\delta})$ uniformly random points to get a false claim. Unfortunately, the verifier does not begin with a claim about any subset of $\frac{1}{\delta}$ coordinates, but only global constraints imposed by $\boldsymbol{j}$ and $\boldsymbol{v}$.

The original idea for getting around this difficulty was given in [RVW13]. Their idea was to break $\boldsymbol{a}$ into two parts: $\boldsymbol{a}_0, \boldsymbol{a}_1$, each corresponding to a function in $\{0,1\}^{m-1} \to \mathbb{F}$ such that $\boldsymbol{a}_b(x) = \boldsymbol{a}(b, x)$ for $b \in \{0,1\}$, and thus

$$\boldsymbol{a}(x_1, \vec{x}) = (1 - x_1)\boldsymbol{a}_0(\vec{x}) + x_1\boldsymbol{a}_1(\vec{x}).$$

This, together with the fact that $\hat{\boldsymbol{a}}$ is the multilinear extension of $\boldsymbol{a}$, implies that

$$\hat{\boldsymbol{a}}(x_1, \vec{x}) = (1 - x_1)\hat{\boldsymbol{a}}_0(\vec{x}) + x_1\hat{\boldsymbol{a}}_1(\vec{x})$$

Exploiting this observation, one can ask the prover to reduce the PVAL claims about $\hat{\boldsymbol{a}}$ to $\hat{\boldsymbol{a}}_0$ and $\hat{\boldsymbol{a}}_1$. Assuming $\boldsymbol{a}$ was far from the PVAL set, regardless of the prover's answers, we would expect a random linear combination $\boldsymbol{a}_0 + c\boldsymbol{a}_1$ to be far from the resulting PVAL claim. Let's try to understand how far from the claim we can expect $\boldsymbol{a}_0 + c\boldsymbol{a}_1$ to be.

We know that the prover needs to change $\boldsymbol{a}$ in $\delta \cdot 2^m$ points to satisfy the PVAL instance. Thus, the prover could answer such that it would need to change the answers in both $\boldsymbol{a}_0$ and $\boldsymbol{a}_1$ in $\delta \cdot 2^{m-1}$ positions each. Therefore, the prover may need to change $\boldsymbol{a}_0 + c\boldsymbol{a}_1$ in only $\delta \cdot 2^{m-1}$ points to satisfy the PVAL instance if the error positions are aligned. Thus, even if the error positions are aligned, we could hope that $\boldsymbol{a}_0 + c\boldsymbol{a}_1$ remains $(\delta 2^{m-1})/2^{m-1} = \delta$-far from satisfying the resulting PVAL claim.

[RVW13] proved that with high probability, $\boldsymbol{a}_0 + c\boldsymbol{a}_1$ is at least $\frac{\delta}{2}$-far from the resulting PVAL since they did not have any assumption that the base PVAL sets themselves have any distance. In contrast, [BKS18] got improved upon [RVW13] by additionally assuming that every two elements in the PVAL set are also far apart. This allowed them to associate a unique point in the set PVAL to which $\boldsymbol{a}_0 + c\boldsymbol{a}_1$ is close.

Now we have the guarantee that the reduced instance has δ-distance from the PVAL set. However, observe that if we want to query $\boldsymbol{a}_0 + c\boldsymbol{a}_1$ on one point then we need to query $\boldsymbol{a}$ on two points. Even though the "instance size" is halved, the "effective instance size", measured by the number of needed queries to $\boldsymbol{a}$, remains the same. Specifically, if we have to query $\boldsymbol{a}_0 + c\boldsymbol{a}_1$ on q points, then we need to query $\boldsymbol{a}$ on $2q$ points. To make progress, we would in fact like to *double* the fractional distance, i.e. maintain the absolute number of error positions in the new claim. If this is achieved, the number of verifier queries to $\boldsymbol{a}_0 + c\boldsymbol{a}_1$ can be halved while retaining the desired soundness. Through this, the number of queries to $\boldsymbol{a}$ remains the same, and the PVAL constraint is indeed only over the folded instance, which is half the original size.

The above construction is indeed quite lossy since 1 error each in $\boldsymbol{a}_0$ and $\boldsymbol{a}_1$ can add up to just count as 1 error in $\boldsymbol{a}_0 + c\boldsymbol{a}_1$ when they align in the truth tables. Thus, [RR20] had the idea of instead reducing not to $\boldsymbol{a}_0 + c\boldsymbol{a}_1$ but to $\boldsymbol{a}_0 + c(\boldsymbol{a}_1 \circ \pi)$ where π is a sufficiently random permutation. This step distributes the errors, and $\boldsymbol{a}_0 + c(\boldsymbol{a}_1 \circ \pi)$ is indeed about 2δ-far from the resulting instance.

In the next subsection, we show that this approach is compatible with not just the Hamming distance but also column distance. Essentially, this is the same as running L instances of RR in parallel with the same randomness and with a PVAL claim over the L instances together. Additionally, since we want to use this protocol as a sub-protocol of a UIP, we prove that the resulting protocol is unambiguous. This requires $\mathbb{F}$ to be a large enough field, which is without loss of generality since we can always just replace $\mathbb{F}$ with a sufficiently large extension.

6.2 Gap Amplification Lemmas for Column Distance

In the section above, we gave an overview of how [RR20] used ideas from [RVW13] and [BKS18]. These results are actively studied in the SNARKs literature [BSCI+23, AHIV23, ACFY25, Zei24, ACFY24] and are known as "divergence" and "correlated-agreement" theorems. We prove the Δ_c analogues of [RVW13] and [BKS18] here now.

We begin by giving an analogous version of the [RVW13] batch amplification lemma. The following lemma only shows that if $\boldsymbol{a}_0$ and $\boldsymbol{a}_1$ are δ-far from vector spaces V then $\boldsymbol{a}_0 + c\boldsymbol{a}_1$ is at least $\delta/2$-far from V with high probability.[21] This result might seem suboptimal, but it is important to note that we do not have any assumption on the distance of the space V itself.

Lemma 8 ([RVW13] gap amplification for Δ_c)**.** *Let $\mathbb{F}$ be a field of characteristic 2 and integers $m, \log L, T > 0$. Let $V \subset \mathbb{F}^{\{0,1\}^{m+\log L}}$ be a non-empty $\mathbb{F}$-linear vector space, and $\boldsymbol{a}_0, \boldsymbol{a}_1 : \{0,1\}^{m+\log L} \to \mathbb{F}$ be functions. Suppose*

$$d_i = \Delta_c(\boldsymbol{a}_i, V).$$

Then,

$$\Pr_{c \leftarrow \mathbb{F}^*}[\Delta_c(\boldsymbol{a}_0 + c\boldsymbol{a}_1, V) < \max(d_0, d_1)/2] \leq \frac{1}{|\mathbb{F}| - 1}.$$

Proof. We emulate the proof given in [RVW13]. For the sake of contradiction assume that

$$\Pr[\Delta_c(\boldsymbol{a}_0 + c\boldsymbol{a}_1, V) < d_0/2] > \frac{1}{|\mathbb{F}|}.$$

Call a c *bad* if $\Delta_c(\boldsymbol{a}_0 + c\boldsymbol{a}_1, V) < d_0/2$. Now, if the theorem is false, then there exist 2 bad values of c, let these be c' and c''.

We would like to use triangle inequality to say that if $g' = \boldsymbol{a}_0 + c'\boldsymbol{a}_1$, $g'' = \boldsymbol{a}_0 + c''\boldsymbol{a}_1$ are both less than $\frac{d_0}{2}$ far from V then their linear combinations are less than δ-far as well. In particular, for any $r \in \mathbb{F}$, we have that

$$\Delta_c(r \cdot g' + (1-r) \cdot g'', V) \leq \Delta_c(g, V) + \Delta_c(g', V) < d_0$$

Now, if $rc' + (1-r)c'' = 0$ then $r \cdot g' + (1-r) \cdot g'' = \boldsymbol{a}_0$ which would imply that $\Delta_c(\boldsymbol{a}_0, V) < d_0$ which is a contradiction. Thus, there is no $r \in \mathbb{F}$ such that $r(c' - c'') = -c''$. But, then we must have $c' = c''$ and there is at most one *bad* value of c as desired. □

6.2.1 Correlated-Agreement under Column Distance

[BKS18] improves the [RVW13] analysis by observing that if we have an additional assumption about distance on the base vector space V then we could get a better result. Their ideas also provide an improved gap amplification lemma which served as the point upon which [RR20] built and improved. We prove the version of [BKS18] in our Δ_c setting. We later use this directly when we prove the RR gap amplification lemma over Δ_c distance.

[BKS18] observed that if we could assume that if $\boldsymbol{a}_0$ is close to v_0 and $\boldsymbol{a}_1$ is close to v_1 where $v_0, v_1 \in V$ where V is a vector space then we could assume that the closest point to $\boldsymbol{a}_0 + c\boldsymbol{a}_1$ is

[21] Note that absolute distances are used in place of the relative forms.

$v_0 + cv_1$ as long as V has large distance. As soon as we have such an assumption, we can immediately improve the [RVW13] analysis.

Similarly to what is done in [RR20], instead of stating the main [BKS18] result on gap amplification directly, we will first prove a "correlated-agreement" lemma. Since the plan is to fold the functions as $\boldsymbol{a}_0 + c(\boldsymbol{a}_1 \circ \pi)$, we need to understand how the errors in the sum look more carefully.

[BKS18] observed that if V is some vector space with large distance and $\boldsymbol{a}_0, \boldsymbol{a}_1$ are different points in the ambient space of V, then if many points along the line $\boldsymbol{a}_0 + c\boldsymbol{a}_1$ are close to V then not only are both $\boldsymbol{a}_0$ and $\boldsymbol{a}_1$ close to V but the number of coordinates that both of them simultaneously agree with elements of V on is also large.

Lemma 9 (Correlated Agreement for Δ_c)**.** *Let $V \subseteq \mathbb{F}^{\{0,1\}^{m+\log L}}$ be a non-empty vector space over $\mathbb{F}$ and $\Delta_c(V) = \Upsilon$. For every $\varepsilon, \delta > 0$ and $\boldsymbol{a}_0, \boldsymbol{a}_1 \in \mathbb{F}^{\{0,1\}^{m+\log L}}$ such that*

- $\delta(1-\varepsilon) \leq \Upsilon/3$
- $|\{c \mid \Delta_c(\boldsymbol{a}_0 + c\boldsymbol{a}_1, V) < \delta(1-\epsilon)\}| > 1/\epsilon$

there exist $v_0, v_1 \in V$ such that for all $t \in \{0,1\}^{\log L}$,

$$|\{i \mid (\boldsymbol{a}_0[i,t] = v_0[i,t]) \wedge (\boldsymbol{a}_1[i,t] = v_1[i,t])\}| \geq (1-\delta)2^m$$

Proof. Let $S \subseteq \{c \mid \Delta_c(\boldsymbol{a}_0 + c\boldsymbol{a}_1, V) < \delta(1-\epsilon)\}$ such that $|S| = 1 + \lfloor 1/\varepsilon \rfloor$. Now, for each $c \in S$, let $v^{(c)} \in V$ be such that $\Delta_c(\boldsymbol{a}_0 + c\boldsymbol{a}_1, v^{(c)}) < \delta(1-\epsilon)$.

Observe that now if all these points $(c, v^{(c)})$ lied on a line then we can get the result. In particular, we can write $v^{(c)} = v_0 + cv_1$ for some fixed v_0, v_1 independent of c. And, we would get

$$\Delta_c(\boldsymbol{a}_0 + c\boldsymbol{a}_1, v_0 + cv_1) < \delta(1-\varepsilon)$$

Rearranging, we get for every $c \in S$,

$$\Delta_c(\boldsymbol{a}_0 - v_0, c(v_1 - \boldsymbol{a}_1)) < \delta(1-\epsilon)$$

Now, observe that for any $i \in \{0,1\}^m, t \in \{0,1\}^{\log L}$ if $\boldsymbol{a}_0[i,t] \neq v_0[i,t]$ then there is at most one value of c for which $\boldsymbol{a}_0[i,t] - v_0[i,t] = c(v_1[i,t] - \boldsymbol{a}_1[i,t])$ and similarly if $\boldsymbol{a}_1[i,t] \neq v_1[i,t]$.

Thus, fix any $t \in \{0,1\}^{\log L}$ and let

$$T_t = \{i \mid (\boldsymbol{a}_0[i,t] = v_0[i,t]) \wedge (\boldsymbol{a}_1[i,t] = v_1[i,t])\}$$

we get that there exists a $c \in S$ such that

$$\frac{|\{i \mid (\boldsymbol{a}_0[i,t] - v_0[i,t]) \neq c(v_1[i,t] - \boldsymbol{a}_{i,t})\}|}{2^m} \geq \left(1 - \frac{|T_t|}{2^m}\right)\left(1 - \frac{1}{|S|}\right)$$

However, since, for any T, we have that:

$$\Delta_c(\boldsymbol{a}_0 - v_0, c(v_1 - \boldsymbol{a}_1)) \geq \frac{|\{i \mid (\boldsymbol{a}_0[i,t] - v_0[i,t]) \neq c(v_1[i,t] - \boldsymbol{a}_1[i,t])\}|}{2^m}$$

Thus,

$$\left(1 - \frac{|T|}{2^m}\right)\left(1 - \frac{1}{|S|}\right) \leq \delta(1-\varepsilon) \implies 1 - \frac{|T|}{2^m} < \delta \implies 2^m(1-\delta) < |T|$$

The final result is since $|S| > \frac{1}{\varepsilon}$.

Thus, we just need to show that indeed the $(c, v^{(c)})$ lie on a line. This is feasible to show because we additionally assume that V has large distance. We will now use the same triangle inequality trick to show that if all $v^{(c)}$ are not on a line then there are two distinct points of V which are $< \Upsilon$ far from each other.

Claim. All the points $(c, v^{(c)})$ lie on a line.

Proof. Observe that the above claim follows trivially when $\epsilon > 1/2$ since then $|S| = 2$.

We are interested in the regime when $\varepsilon \leq 1/2$. Consider any $c_1, c_2, c_3 \in S$. Then, we can take a linear combination of $(\boldsymbol{a}_0 + c_1\boldsymbol{a}_1)$ and $(\boldsymbol{a}_0 + c_2\boldsymbol{a}_1)$ to get $(\boldsymbol{a}_0 + c_3\boldsymbol{a}_1)$. Observe that

$$\Delta_c\left(\frac{(c_2 - c_3)(\boldsymbol{a}_0 + c_1\boldsymbol{a}_1) + (c_3 - c_1)(\boldsymbol{a}_0 + c_2\boldsymbol{a}_1)}{c_2 - c_1}, \frac{(c_2 - c_3)v^{(c_1)} + (c_3 - c_1)v^{(c_2)}}{c_2 - c_1}\right) < 2\delta(1 - \epsilon)$$

Let $v' = \frac{(c_2-c_3)v^{(c_1)}+(c_3-c_1)v^{(c_2)}}{c_2-c_1}$. Thus, by the above we have

$$\Delta_c(v', \boldsymbol{a}_0 + c_3\boldsymbol{a}_1) < 2\delta(1 - \epsilon)$$

and $\Delta_c(v^{(c_3)}, \boldsymbol{a}_0 + c_3\boldsymbol{a}_1) < \delta(1 - \epsilon)$. Finally, by the triangle inequality, we have that:

$$\Delta_c(v', v^{(c_3)}) < 3\delta(1 - \varepsilon) \leq \Upsilon \implies v' = v^{(c_3)}.$$

Hence, $v' = v^{(c_3)}$ so $v^{(c_1)}, v^{(c_2)}$, and $v^{(c_3)}$ are indeed collinear as desired. □

□

6.2.2 An RR-Style Gap Amplification

Now, we are ready to prove the RR style gap amplification lemma for Δ_c.

[RR20]'s idea is to amplify distance using not just $\boldsymbol{a}_0 + c\boldsymbol{a}_1$ as in [RVW13] but instead to work with $\boldsymbol{a}_0 + c(\boldsymbol{a}_1 \circ \pi)$ where π is an efficient to compute permutation. If we assume that $\pi \sim \Pi$ is η-close to being $d-$wise independent then we can hope that the errors in $\boldsymbol{a}_0$ and $\boldsymbol{a}_1 \circ \pi$ do not align with very high probability. The issue here is that even though we have that $\Delta_c(\boldsymbol{a}_0 \| \boldsymbol{a}_1, \mathsf{PVAL}(j, v)) \geq \delta$, but that does not translate into any immediate claim about distance of $\boldsymbol{a}_1 \circ \pi$ with any immediate new PVAL claim.

Thus, we need to add a GKR based protocol first to create claims about $\mathsf{PVAL}(\boldsymbol{a}_1 \circ \tau)$ where $\tau = \pi \times I$[22]. The permutation π is sampled as $\pi \sim \Pi$ where Π is an η-almost d-wise independent distribution as defined in Theorem 4.

Definition 14. $\Phi_{\pi,\boldsymbol{j},v}$ is the circuit that takes in an input $\boldsymbol{a}$ and computes whether

$$\left(\widehat{\boldsymbol{a} \circ (\pi \times I)}\right)(\boldsymbol{j}) = v$$

Here, we assume that π has a succinct description.

Lemma 10 (Complexity of $\Phi_{\pi,\boldsymbol{j},\boldsymbol{v}}$)**.** *The circuit $\Phi_{\pi,\boldsymbol{j},\boldsymbol{v}}$ has size $\tilde{O}(T \cdot (2^{m+\log L}) \cdot \log |\mathbb{F}|)$ and depth* $\mathrm{poly}(m + \log L, \log(T \log |\mathbb{F}|))$.

[22] It implements π in each column of the matrix as in Figure 1c.

Proof. The permutation is only 1 layer of the circuit and size linear in the input. Beyond that, it is verifying T distinct multilinear evaluations which can be evaluated by simply taking the sum of the indicator variables. Thus, the result follows. □

Now, consider the following sub-protocol:

Protocol 5 $(\mathsf{P}_\Pi, \mathsf{V}_\Pi)$: GKR to get claims about $\boldsymbol{a} \circ \tau$.

Input Parameters: m, L, $T \in \mathbb{N}$ and $\mathbb{F}$ is a field.
Input: $\pi \in \Pi$, a permutation, $\boldsymbol{j} \in \left(\mathbb{F}^{m+\log L}\right)^T$ and $\boldsymbol{v} \in \mathbb{F}^T$.
Prover Auxiliary Input: $\boldsymbol{a} \in \{0,1\}^{m \times L} \to \mathbb{F}$.
Ingredients:

- *The GKR protocol* $(\mathsf{P}_{\mathsf{GKR}}, \mathsf{V}_{\mathsf{GKR}})$ (Theorem 3) for checking $\Phi(\boldsymbol{a}) = 1$.

  The prover and verifier gets the description $\langle \Phi \rangle$, while the prover additionally gets the input $\boldsymbol{a}$. Both parties additionally takes in the parameter $T \in \mathbb{N}$, and in the end outputs $\boldsymbol{j} \in (\mathbb{F}^{m+\log L})^T, v \in \mathbb{F}^T$.

Output: $\boldsymbol{j}' \in \left(\mathbb{F}^{m+\log L}\right)^T, v' \in \mathbb{F}^T$.

1: Let $\tau = \pi \times I$. Let $\langle \Phi_{\pi,\boldsymbol{j},\boldsymbol{v}} \rangle$ be the description of the circuit $\Phi_{\pi,\boldsymbol{j},\boldsymbol{v}}$ that takes in a function $\boldsymbol{a}$ and checks whether $\widehat{(\boldsymbol{a} \circ \tau)}(\boldsymbol{j}) = \boldsymbol{v}$.
2: Run the GKR protocol T times on $\langle \Phi_{\pi,\boldsymbol{j},\boldsymbol{v}} \rangle$ and prover auxiliary input $\boldsymbol{a}$ and output the resulting $\boldsymbol{j}', \boldsymbol{v}'$.

We can directly use this sub-protocol to create claims about $\boldsymbol{a}_1 \circ \tau$ and thus use [RR20]'s idea of folding the function $\boldsymbol{a}$ as $\boldsymbol{a}_0 + c(\boldsymbol{a}_1 \circ \tau)$.

Lemma 11 (RR gap amplification for Δ_c)**.** *Let $m, \log L, T, \sigma, d > 0$ be integers and $0 < 4d \leq 2^{m/50}$. Let $\mathbb{F}$ be a field of characteristic 2. For $\boldsymbol{j} \in \left(\mathbb{F}^{m+\log L}\right)^T$ suppose that has minimum column distance $\geq 4d$. Let $\boldsymbol{v}_0, \boldsymbol{v}_1 \in \mathbb{F}^T$ be vectors such that both $\mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v}_0)$ and $\mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v}_1)$ are non-empty. Let $\boldsymbol{a}_0, \boldsymbol{a}_1 : \{0,1\}^{m+\log L} \to \mathbb{F}$ be functions. Suppose*

$$d \leq \Delta_c\left(\begin{bmatrix}\boldsymbol{a}_0\\ \boldsymbol{a}_1\end{bmatrix}, \begin{bmatrix}\mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v}_0)\\ \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v}_1)\end{bmatrix}\right)$$

Consider a uniformly random $\pi \in \Pi_{m,4d,2^{-\sigma}}$ where Π is the set of permutations on $\{0,1\}^m$ defined in Section 3.4. Let $\tau = \pi \times I$ and $\boldsymbol{j}' \in \left(\mathbb{F}^{m+\log L}\right)^T, \boldsymbol{v}' \in \mathbb{F}^T$ such that $\boldsymbol{j}', \boldsymbol{v}'$ is outputted by the Protocol 5 protocol on input $\boldsymbol{a}_1 \circ \tau, \pi^{-1}, \boldsymbol{j}, v_1$ and the same additional parameters i.e. $m, \log L, T$.

Now, for $c \leftarrow \mathbb{F}$ define:

$$g = \boldsymbol{a}_0 + c(\boldsymbol{a}_1 \circ \tau).$$

Let $S_\pi \subseteq \mathbb{F}^{2t}$ be the set of pairs of vectors (u_0, u_1) such that the sets $\mathsf{PVAL}((\boldsymbol{j}, \boldsymbol{j}'), (\boldsymbol{v}_0, u_0))$ and $\mathsf{PVAL}((\boldsymbol{j}, \boldsymbol{j}', (u_1, \boldsymbol{v}'))$ are not empty. For $(u_0, u_1) \in S_\pi$, define

$$\delta_{\pi,j',v',(u_0,u_1),c} = \Delta_c(g, \mathsf{PVAL}((\boldsymbol{j}, \boldsymbol{j}'), (v_0, u_0) + c(u_1, v'))$$

Then for every $\gamma < (2/d, 1)$ and $\epsilon \in (0,1)$, taking

$$\delta' = d(1-\gamma)(1-\varepsilon),$$

we have that

$$\Pr_{\pi \leftarrow \Pi, Protocol\ 5}\left[\exists(\boldsymbol{u_0}, \boldsymbol{u_1}) \in S_\pi \ s.t. \ \Pr_{c_0,c_1 \leftarrow \mathbb{F}}\left[\delta_{\pi,(u_0,u_1),c,j'} < \delta'\right] > \frac{1}{\varepsilon|\mathbb{F}|} + \frac{1}{|\mathbb{F}|}\right] < 2^{-\sigma} + \exp(-\gamma d/6) + \mathsf{err}_{GKR}$$

where $\mathsf{err}_{\mathsf{GKR}}$ *refers to the failure probability of Protocol 5 as described in Lemma 2. Thus, we have that*

$$\mathsf{err}_{\mathsf{GKR}} \leq \left(\epsilon_{\mathsf{GKR}}(D, \log S, |\mathbb{F}|) + (\epsilon_{\mathsf{GKR}}(D, \log S, |\mathbb{F}|))^T \cdot \left(\binom{M}{4d}|\mathbb{F}|^{4d}\right)^L\right)$$

where D *and* S *are the depth and size of* $\Phi_{\pi,\boldsymbol{j},v}$ *respectively.*

Proof. For the sake of contradiction, assume that the lemma is false.

Observe that for any $\pi \in \Pi$, with probability $\geq 1 - \mathsf{err}_{\mathsf{GKR}}$, by Lemma 2 and Theorem 3, there exists a $g \in \mathsf{PVAL}(\boldsymbol{j}', v') \cap B_{4d,\mathbb{F}}(\boldsymbol{a}_1 \circ \tau)$ iff the prover was honest with implicit input g and $\left(g \circ \tau^{-1}\right) \in \mathsf{PVAL}(\boldsymbol{j}, v_1)$.

Thus, we condition on this event happening and that accounts for the $\mathsf{err}_{\mathsf{GKR}}$ term in the lemma by union bound.

Let $\bar{\delta} = \frac{\delta'}{1-\epsilon} < d < \frac{4d}{3}$. Now, with probability $> p = 2^{-\sigma} + \exp(-\gamma \cdot d/6)$ over the choice of π, there exists $(u_0, u_1) \in S_\pi$ and $c_0 \neq 0$, such that

$$\Pr_{c_1 \leftarrow \mathbb{F}}[\delta_{\pi,(u_0,u_1),c,\boldsymbol{j}'} < \bar{\delta}(1-\epsilon)] > \frac{1}{\epsilon|\mathbb{F}|}$$

Since $\bar{\delta} < 4d/3$, with probability $> p$, over the choice of $\pi \in \Pi$, let the choice of $(u_0, u_1) \in S_\pi$ and $c \neq 0$ be as above. Thus, let $V = \mathsf{PVAL}((\boldsymbol{j}, \boldsymbol{j}'), 0)$ and observe that $\Delta_c(V) \geq 4d$. Let

$$h_0 \in \mathsf{PVAL}((\boldsymbol{j}, \boldsymbol{j}'), (v_0, u_0)), \qquad h_1 \in \mathsf{PVAL}((\boldsymbol{j}, \boldsymbol{j}'), (u_1, v')),$$

then

$$\Pr_{c \leftarrow \mathbb{F}}[\Delta_c(V + h_0 + c(V + h_1), \boldsymbol{a}_0 + c(\boldsymbol{a}_1 \circ \tau)) < \bar{\delta}(1-\epsilon)] > \frac{1}{\epsilon|\mathbb{F}|},$$

and thus

$$\Pr_{c \leftarrow \mathbb{F}}[\Delta_c(V, (\boldsymbol{a}_0 - h_0) + c(\boldsymbol{a}_1 \circ \tau - h_1)) < \bar{\delta}(1-\epsilon)] > \frac{1}{\epsilon|\mathbb{F}|}.$$

Letting $u^* = \boldsymbol{a}_0 - h_0$ and $u = \boldsymbol{a}_1 \circ \tau - h_1$ in Lemma 9, we get that there exist $y_0, y_1 \in V$ such that for all $r \in \{0,1\}^{\log L}$

$$|\{i \in \{0,1\}^m \mid (\boldsymbol{a}_0(i,r) - h_0(i,r) = y_0(i,r)) \wedge (\boldsymbol{a}_1 \circ \tau(i,r) - h_1(i,r) = y_1(i,r))\}| \geq 2^m - \bar{\delta},$$

or equivalently for all $r \in \{0,1\}^{\log L}$,

$$|\{i \in \{0,1\}^m \mid (\boldsymbol{a}_0(i,r) = (y_0 + h_0)(i,r)) \wedge (\boldsymbol{a}_1 \circ \tau(i,r) = (y_1 + h_1)(i,r))\}| \geq 2^m - \bar{\delta}$$

Now, observe that since $(y_0 + h_0) \in \mathsf{PVAL}(\boldsymbol{j}, v_0)$, $(y_1 + h_1) \in \mathsf{PVAL}(\boldsymbol{j}', v')$, and $\Delta_c(\mathsf{PVAL}((\boldsymbol{j}, \boldsymbol{j}'), 0)) > 3\bar{\delta}$, we have that there is a unique element in $z_0 \in \mathsf{PVAL}(\boldsymbol{j}, v_0)$ such that $\Delta_c(\boldsymbol{a}_0, z_0) \leq \bar{\delta}$ and similarly, there is a unique element $z_1 \in \mathsf{PVAL}(\boldsymbol{j}', v')$ such that $\Delta_c(\boldsymbol{a}_1 \circ \tau, z_1) \leq \bar{\delta}$. Now, observe that since $z_1 \in \mathsf{PVAL}(\boldsymbol{j}', v')$ and $\Delta_c(\boldsymbol{a}_1 \circ \tau, z_1) \leq 4d$, we must have that the prover was honest with implicit input z_1 in mind and $z_1 \circ \tau^{-1} \in \mathsf{PVAL}(\boldsymbol{j}, v_1)$. Observe that $\Delta_c(\mathsf{PVAL}(\boldsymbol{j}, v_1)) \geq 4d$. Thus, $z_1' = z_1 \circ \tau^{-1}$ is the unique element such that $z_1' \in \mathsf{PVAL}(\boldsymbol{j}, v_1)$ and $\Delta_c(\boldsymbol{a}_1, z_1') < \bar{\delta}$.

Plugging, this back in, we get that with probability $\geq p$ over the choice of $\pi \in \Pi$, we have that for all $r \in \{0,1\}^{\log L}$

$$|\{i \in \{0,1\}^m \mid (\boldsymbol{a}_0(i,r) = z_0(i,r)) \wedge (\boldsymbol{a}_1 \circ \tau(i,r) = z_1' \circ \tau(i,r))\}| > 2^m - \bar{\delta}$$

Since $\Delta_c(\boldsymbol{a}_0\|\boldsymbol{a}_1, z_0\|z_1') \geq d$, for each $r \in \{0,1\}^{\log L}$, let

$$\delta_{r,0} = \Delta_c(\boldsymbol{a}_0(\cdot, r), z_0(\cdot, r)) \qquad \delta_{r,1} = \Delta_c(\boldsymbol{a}_1(\cdot, r), z_1'(\cdot, r)).$$

By our assumption, we know that $\max_{r\in\{0,1\}^m}(\delta_{r,0} + \delta_{r,1}) = \Delta_c(\boldsymbol{a}_0\|\boldsymbol{a}_1, z_0\|z_1') \geq d$.

Now, we fix our focus to a fixed $r \in \{0,1\}^{\log L}$ such that $\delta_{r,0} + \delta_{r,1} \geq d$. Now, for this r, with probability $\geq p$ over the choice of π, we have that

$$|\{i \in \{0,1\}^m \mid (\boldsymbol{a}_0(i,r) = z_0) \wedge \big(\boldsymbol{a}_1 \circ \tau(i,r) = z_1' \circ \tau(i,r)\big)\}| \geq 2^m - \bar{\delta}$$

Let

$$E_0 = \{i \mid (\boldsymbol{a}_0(i,r) \neq z_0(i,r))\} \qquad E_1 = \{i \mid \big(\boldsymbol{a}_1(i,r) \neq z_1'(i,r)\big)\}.$$

We know $|E_0|+|E_1| \geq d$. Thus, we can take subsets $A_0 \subseteq E_0$ and $E_1 \subseteq A_1$ such that $|A_0|+|A_1| \geq d$ and $|A_0|, |A_1| \leq d$. Therefore, probability $\geq p$ over the choice of π, we have that $|A_0 \cap \pi A_1| \geq d - (1-\gamma)d \geq \gamma d$.

Since $\pi \sim \Pi_{m,4d,2^{-\sigma}}$ and $p = 2^{-\sigma} + \exp(-\gamma d/6)$, this would contradict Theorem 5. Thus, we have a contradiction. □

6.3 Efficient IPP for PVAL with Column Distance

Now, we give an analogous efficient Interactive proof of proximity (IPP) for PVAL to [RR20] in the column distance setting. The key differences here from their work is that we add a soundness parameter as we want the protocol to be unambiguous, and our distance notion is different. Besides these differences which mostly just affect our choices of parameters, the protocol is essentially the same as RR.

For the applications that we use the protocol, we do not require it to be an IPP. We mention the IPP here for completeness and historical reasons.

Before presenting the protocol, we go over the ideas in the protocol first:

- We begin with a function $\boldsymbol{a} : \{0,1\}^{m+\log L} \to \mathbb{F}$ and a $\mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$ set. The verifier would like to check whether $\boldsymbol{a} \in \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$ or if $\Delta_c(\boldsymbol{a}, \mathsf{PVAL}(\boldsymbol{j}, v)) > d$. The verifier would also like the final check to be efficient and succinctly described so that it could be further delegated.

- **Breaking into Multilinear Functions:** The verifier asks the prover to break $\boldsymbol{a}$ into multilinear functions $\boldsymbol{a}_0, \boldsymbol{a}_1 : \{0,1\}^{m+\log L-1} \mapsto \mathbb{F}$ such that $\hat{\boldsymbol{a}}(\chi, x) = (1-\chi)\hat{\boldsymbol{a}}_0(x) + \chi\hat{\boldsymbol{a}}_1(x)$ for $\chi \in \mathbb{F}$. This is possible due to the multilinearity of $\hat{\boldsymbol{a}}$. The verifier then asks the prover to break $j_i = (\chi_i, j_i')$ and send $\boldsymbol{a}_0(j_i')$ and$\boldsymbol{a}_1(j_i')$. Let the resulting PVAL instances be $\mathsf{PVAL}(\boldsymbol{j}', \boldsymbol{v}_0)$ and $\mathsf{PVAL}(\boldsymbol{j}', \boldsymbol{v}_1)$.

  The verifier can now use the linearity to check if these claims are consistent.

- **Permuting Errors:** Now, the prover uses Protocol 5 to convert the claim $\boldsymbol{a}_1 \in \mathsf{PVAL}(\boldsymbol{j}', \boldsymbol{v}_1)$ into a new claim that $\boldsymbol{a}_1 \circ \tau \in \mathsf{PVAL}(\boldsymbol{j}', \boldsymbol{v}')$.

- **Filling in Values and emptiness checks:** The verifier would want to create now a new PVAL claim about $\boldsymbol{a}_0 + c(\boldsymbol{a}_1 \circ \tau)$ but does not know $\hat{\boldsymbol{a}}_0(\boldsymbol{j}')$ or $\widehat{\boldsymbol{a}_1 \circ \tau}(\boldsymbol{j}')$. Thus, the verifier simply asks the prover to fill in these values. This is all compatible with the Lemma 11 gap amplification that we have proved above.

- **Curve fitting:** Assuming all this goes well, the verifier has reduced to a claim about a function $\boldsymbol{a}' : \{0,1\}^{m+\log L-1} \mapsto \mathbb{F}$ but the resulting PVAL instance now has $2T$ evaluations instead of T. Thus, the verifier can ask the prover to construct a low degree curve through j' and j' and ask the prover for values of $\boldsymbol{a}_0$ and $\boldsymbol{a}_1 \circ \tau$ on all the points on the curve.
- **Reducing number of evaluation points:** Now, the verifier just picks T uniformly random points on the curve. Since j' was uniformly random, these T points also end up being uniformly random elements of $\mathbb{F}^{m+\log L-1}$ and the distance guarantee continues to hold with high probability.

The protocol (Protocol 10) is essentially just the above with some parameter balancing to get the appropriate soundness error. We need to do all the above over larger fields than [RR20] did since we also prove unambiguity of the procedure and thus cannot use repetition to amplify the unambiguous soundness error. In subsequent subsections - we describe each of these components of the full protocol. We then combine all of these parts to get Protocol 10 and the following theorem:

Theorem 10 (IPP for PVAL with column distance)**.** *Let $m, \log L, T, \sigma, d \in \mathbb{N}$. Let $\mathbb{F}$ be a constructible field ensembles such that $\mathbb{F}$ has characteristic 2. If*

- $|\mathbb{F}| \geq \Omega(2^\sigma \cdot T^2(m + \log L)^2)$,
- $T > 8d \cdot 2^{\log L} \cdot m$,
- $d \geq 16 \cdot m \cdot \sigma$

Then for any $\boldsymbol{j} \in (\mathbb{F}^{m+\log L})^T, v \in \mathbb{F}^T$, the matrix $\mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$ has a public coin unambiguous IPP as long as $\Delta_c(\mathsf{PVAL}(\boldsymbol{j}, 0)) > 4d$:

- *Soundness error:* $m \cdot 2^{4-\sigma}$
- *Prover to Verifier communication complexity:* $T \log |\mathbb{F}| \cdot \mathrm{poly}(m, \log L) + L \cdot \log |\mathbb{F}| \cdot \mathrm{poly}(d)$.
- *Verifier to Prover communication complexity:* $T \log |\mathbb{F}| \cdot \mathrm{poly}(m, \log L)$
- *Round Complexity:* $\mathrm{poly}(m, \log L, \log(T \log |\mathbb{F}|))$
- *Prover runtime:* $\mathrm{poly}(2^{m+\log L}, T \log |\mathbb{F}|)$
- *Verifier runtime:* $T \log |\mathbb{F}| \cdot \mathrm{poly}(m, \log L, \log(T \log |\mathbb{F}|), \log |\mathbb{F}|) + \tilde{O}(L \cdot \log |\mathbb{F}| \cdot \mathrm{poly}(d))$
- *Query complexity :* $|\mathcal{Q}| = \left\lceil \frac{4\sigma \cdot 2^m}{d} \right\rceil \cdot L$.

The final verifier verdict is given by a succinctly described predicate $\Phi : \mathbb{F}^{\left\lceil \frac{4\sigma \cdot 2^m}{d} \right\rceil \cdot 2^{\log L}} \mapsto \{0,1\}$ along with a succinctly described set $Q \subseteq [2^m]$ such that $|Q| = \left\lceil \frac{4\sigma \cdot 2^m}{d} \right\rceil$, and the verdict is $\Phi(f|_Q \times \{0,1\}^{\log L})$ with $|\langle Q \rangle| = \tilde{O}(m^2 d\sigma \cdot + m \log |\mathbb{F}| + \sigma \cdot \mathrm{poly}(d))$ and $|\langle \Phi \rangle| = |\langle Q \rangle| + \sigma \cdot \mathrm{poly}(d) \cdot L \cdot \log |\mathbb{F}|$.

Protocol 6 Breaking $\boldsymbol{a}$ into two multilinear functions

Input Parameters: m, L, $T \in \mathbb{N}$ and $\mathbb{F}$ is a field.

Input: $\boldsymbol{j} \in (\mathbb{F}^{m+\log L})^T$ and $\boldsymbol{v} \in \mathbb{F}^T$.

Prover Auxiliary Input: $\boldsymbol{a} : \{0,1\}^{m+\log L} \to \mathbb{F}$.

Output: $\{j_i', \zeta_i^{(0)}, \zeta_i^{(1)}\}_{i\in[T]}$.

1: P lets $\boldsymbol{j}_i = (\chi_i, \boldsymbol{j}_i') \in \mathbb{F} \times \mathbb{F}^{m-1+\log L}$.

2: P sends $\left\{\zeta_i^{(b)} = \hat{\boldsymbol{a}}(b, \boldsymbol{j}_i')\right\}_{i\in[T],b\in\{0,1\}}$ to V.

3: **if** $\bigwedge_{i\in[T]} \left[v_i = (1-\chi_i)\zeta_i^{(0)} + \chi_i\zeta_i^{(1)}\right]$ **then**

4: V outputs $\{\boldsymbol{j}_i', \zeta_i^{(b)}\}_{b\in\{0,1\},i\in[T]}$.

5: **else**

6: V halts and rejects.

6.3.1 Breaking into Multilinear Functions and Consistency Checks

Lemma 12. *Let $T, d, m, \log L \in \mathbb{N}$ and $\mathbb{F}$, a finite field of characteristic 2. $\boldsymbol{j} \in (\mathbb{F}^{m+\log L})^T$ and $\boldsymbol{v} \in \mathbb{F}^T$. If, $\Delta_c(\mathit{PVAL}(j, 0)) \geq 4d$, then Protocol 6 has the following properties:*

- ***Unambiguous Distance Preservation:***
  - *If $\Delta_c(\boldsymbol{a}, \mathit{PVAL}(\boldsymbol{j}, \boldsymbol{v})) \leq d$ and the prover answers according to the prover's prescribed strategy for the unique $\boldsymbol{a}^* \in \mathit{PVAL}(\boldsymbol{j}, \boldsymbol{v}) \cap \mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a})$ then $\boldsymbol{a}^*$ remains the unique element in $\mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a}) \cap (\mathit{PVAL}(\boldsymbol{j}', \boldsymbol{\zeta}^{(0)}) \| \mathit{PVAL}(\boldsymbol{j}', \zeta^{(1)}))$.*
  - *If $\Delta_c(\boldsymbol{a}, \mathit{PVAL}(\boldsymbol{j}, \boldsymbol{v})) > d$ or the prover is not honest with respect to the unique $\boldsymbol{a}^* \in \mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a}) \cap \mathit{PVAL}(\boldsymbol{j}, \boldsymbol{v})$ then, $\Delta_c(\boldsymbol{a}, \mathit{PVAL}(\boldsymbol{j}', \zeta^{(0)}) \| \mathit{PVAL}(\boldsymbol{j}', \zeta^{(1)}).) > d$.*
- ***Round Complexity:*** 1
- ***Communication Complexity:*** $T \cdot \log \mathbb{F} \cdot (m + \log L + 1)$.
- ***Verifier Time Complexity:*** $O(T\text{polylog}(\mathbb{F}))$
- ***Prover Runtime:*** $\text{poly}(2^{m+\log L}, T, \log \mathbb{F})$.
- *In all cases:* $\Delta_c(\mathit{PVAL}(\boldsymbol{j}', \boldsymbol{0})) \geq 4d$.

Proof. The round, communication and time complexity follow directly. The final property is since $\mathsf{PVAL}(\boldsymbol{j}', \boldsymbol{0}) \| \boldsymbol{0} \subseteq \mathsf{PVAL}(\boldsymbol{j}, 0)$ and the first has distance $\Delta_c(\mathsf{PVAL}(\boldsymbol{j}', \boldsymbol{0}))$.

Now, if the verifier doesn't reject and $\boldsymbol{a}_0 \in \mathsf{PVAL}(\boldsymbol{j}', \zeta^{(0)})$ and $\boldsymbol{a}_1 \in \mathsf{PVAL}(\boldsymbol{j}', \zeta^{(1)})$ then, we have that for all $i \in [T]$:

$$\widehat{\boldsymbol{a}_0 \| \boldsymbol{a}_1}(\boldsymbol{j}_i) = \widehat{\boldsymbol{a}_0 \| \boldsymbol{a}_1}(\chi_i, \boldsymbol{j}_i') = (1-\chi_i)\widehat{\boldsymbol{a}_0}(\boldsymbol{j}_i') + \chi_i \widehat{\boldsymbol{a}_1}(\boldsymbol{j}_i') = (1-\chi_i)\zeta_i^{(0)} + \chi_i \cdot \zeta_i^{(1)} = \boldsymbol{v}_i$$

by multilinearity. We get that $\boldsymbol{a}_0 \| \boldsymbol{a}_1 \in \mathsf{PVAL}(\mathbf{j}, \mathbf{v})$. This clearly implies the honest prover case and the $\Delta_c(\boldsymbol{a}, \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})) > d$ cases.

If the prover is dishonest with respect to $\boldsymbol{a}^*$, then we know that $\boldsymbol{a}^* \notin \mathsf{PVAL}(\boldsymbol{j}', \zeta^{(0)}) \| \mathsf{PVAL}(\boldsymbol{j}', \zeta^{(1)})$. Thus, $\Delta_c(\boldsymbol{a}^*, \mathsf{PVAL}(\boldsymbol{j}', \zeta^{(0)}) \| \mathsf{PVAL}(\boldsymbol{j}', \zeta^{(1)})) \geq 4d \implies \Delta_c(\boldsymbol{a}, \mathsf{PVAL}(\boldsymbol{j}', \zeta^{(0)}) \| \mathsf{PVAL}(\boldsymbol{j}', \zeta^{(1)})) > d$. □

6.3.2 Permuting Errors and Curve Fitting

Protocol 7 Permuting Errors and Curve Fitting

Input parameters: m, L, T, $\sigma \in \mathbb{N}$ and $\mathbb{F}$ is a field.
Input: $\boldsymbol{j} \in (\mathbb{F}^{m+\log L})^T$ and $\boldsymbol{v}_0, \boldsymbol{v}_1 \in \mathbb{F}^T$.
Prover Auxiliary Input: $\boldsymbol{a}_0, \boldsymbol{a}_1 : \{0,1\}^{m+\log L} \to \mathbb{F}$.
Other parameters: $\boldsymbol{\Lambda} = (\Lambda_1, \Lambda_2, \ldots, \Lambda_{2T})$ are distinct fixed canonical points in $\mathbb{F}$.
Output: A permutation $\tau : \{0,1\}^{m+\log L} \to \{0,1\}^{m+\log L}$, a curve $\mathcal{C} : \mathbb{F} \to \mathbb{F}^{m+\log L}$, $\boldsymbol{j}' \in (\mathbb{F}^{m+\log L})^T$, $\boldsymbol{v}' \in \mathbb{F}^T$, and $g^{(0)}$ and $g^{(1)} : \mathbb{F} \to \mathbb{F}$ which are univariate polynomials of degree at most $2T(m + \log L)$.

1: V samples $\pi \leftarrow \Pi_{m,4d,2^{-\sigma}}$ and sends it to the prover. They both let $\tau = \pi \times I$.
2: Both parties run Protocol 5 on $\boldsymbol{a}_1 \circ \tau, \pi^{-1}, \boldsymbol{j}, \boldsymbol{v}_1$, obtaining $\boldsymbol{j}', \boldsymbol{v}'$.
3: Let $\mathcal{C} : \mathbb{F} \to \mathbb{F}^{m+\log L}$ be the canonical low degree curve passing through $\boldsymbol{j}$ and $\boldsymbol{j}'$ i.e. the unique degree $\leq 2T-1$ curve such that $\mathcal{C}(\boldsymbol{\Lambda}) = (\boldsymbol{j}, \boldsymbol{j}')$.
4: P computes the univariate polynomials of degree $\leq (2T-1)(m + \log L)$, denoted by $g^{(0)}$ and $g^{(1)}$, that satisfy

$$g^{(0)}(x) = \widehat{\boldsymbol{a}_0}(\mathcal{C}(x)) \qquad g^{(1)}(x) = \widehat{\boldsymbol{a}_1 \circ \tau}(\mathcal{C}(x)),$$

and sends them to V.
5: V accepts and outputs $\pi, \mathcal{C}, \boldsymbol{j}', \boldsymbol{v}', g^{(0)}, g^{(1)}$ if

$$g^{(0)}(\Lambda_1, \Lambda_2, \ldots, \Lambda_T) = \boldsymbol{v}_0 \qquad \wedge \qquad g^{(1)}(\Lambda_{T+1}, \ldots, \Lambda_{2T}) = \boldsymbol{v}'.$$

Lemma 13. *Let $m, \log L, T, \sigma, r, d > 0$ be integers with $r < d/4$, $4d \leq 2^{m/50}$. Let $\mathbb{F}$ be a constructible field ensemble of characteristic 2. We also have the inputs $\boldsymbol{j}, \boldsymbol{v_0}, \boldsymbol{v_1}$ where $\boldsymbol{j} \in (\mathbb{F}^{m+\log L})^T$ and $\boldsymbol{v}_0, \boldsymbol{v}_1 \in \mathbb{F}^T$, with prover having access to functions $\boldsymbol{a}_0, \boldsymbol{a}_1 : \{0,1\}^{m+\log L} \to \mathbb{F}$. Additionally, if $\Delta_c(\mathsf{PVAL}(\boldsymbol{j}, 0)) \geq 4d$ then we have that the protocol either breaks and rejects or outputs univariate polynomials $g_0, g_1 : \mathbb{F} \to \mathbb{F}$ of degree at most $2T(m + \log L)$, a permutation $\pi : \{0,1\}^m \to \{0,1\}^m$, and a uniformly random $\boldsymbol{j}' \in (\mathbb{F}^{m+\log L})^T$. Then Protocol 7 has the following properties:*

- ***Unambiguous Distance Preservation:*** *Let $\boldsymbol{a} = \boldsymbol{a}_0 \| \boldsymbol{a}_1$. Now, with probability, $\geq 1 - (2^{-\sigma} + \mathsf{err}_{GKR} + \exp(-d/12r))$, the following properties hold:*
  - *If $\Delta_c(\boldsymbol{a}, \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v}_0) \| \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v}_1)) \leq d$ and the prover answers according to the prover's prescribed strategy for the unique $\boldsymbol{a}_0^* \| \boldsymbol{a}_1^* = \boldsymbol{a}^* \in (\mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v}_0) \| \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v}_1)) \cap \mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a})$ then:*
    - *For any $c \in \mathbb{F}$: $\boldsymbol{a}_0^* + c\boldsymbol{a}_1^* \circ \tau$ is the unique element in $\mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a}_0 + c\boldsymbol{a}_1 \circ \tau) \cap (\mathsf{PVAL}(\mathcal{C}(\mathbb{F}), g_0 + cg_1(\mathbb{F})))$.*
    - *$\Pr_c \left[\Delta_c(\boldsymbol{a}_0 + c\boldsymbol{a}_1 \circ \tau, \boldsymbol{a}_0^* + c\boldsymbol{a}_1^* \circ \tau) < \Delta_c(\boldsymbol{a}, \boldsymbol{a}^*)(1 - \frac{1}{r}) \right] < \frac{2r+1}{\mathbb{F}}$.*
  - *If $\Delta_c(\boldsymbol{a}, \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v}_0) \| \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v}_1)) > d$ or the prover is not honest with respect to the unique $\boldsymbol{a}^* \in \Delta_c(\boldsymbol{a}, \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v}_0) \| \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v}_1))$ then either:*
    - *$\mathsf{PVAL}((\boldsymbol{j}, \boldsymbol{j}'), g_0(\Lambda))$ is empty; or*
    - *$\mathsf{PVAL}((\boldsymbol{j}, \boldsymbol{j}'), g_1(\Lambda))$ is empty; or*

$$* \ \Pr_c\left[\Delta_c(\boldsymbol{a}_0 + c\boldsymbol{a}_1 \circ \tau, \mathit{PVAL}(\mathcal{C}(\mathbb{F}), g_0 + cg_1(\mathbb{F})) < d(1 - \tfrac{1}{r})\right] < \tfrac{2r+1}{\mathbb{F}}.$$

- ***Round Complexity:*** $O(D \log S)$
- ***Message length per round:*** $\tilde{O}(T(m + \log L) \log F + dm\sigma)$.
- ***Verifier Runtime:*** $\tilde{O}(D \log S \cdot \text{polylog}(\mathbb{F}) + dm\sigma + T(m + \log L))$.
- ***Prover Runtime:*** $\text{poly}(S, \log \mathbb{F}, T, 2^{m+\log L})$.

Here D and S are the depth and size respectively of $\Phi_{\pi,\boldsymbol{j},\boldsymbol{v}}$ as defined in Definition 14.

Proof. We again don't focus on round, communication, and time complexities as they follow directly. The complexities of π, g_0, g_1 and Protocol 5 directly imply the results. Focusing on the unambiguous distance preservation property, the result is essentially a restatement of Lemma 11 with $\epsilon = \gamma = \frac{1}{2r}$. In particular:

- If $\Delta_c(\boldsymbol{a}, \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v}_0) \| \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v}_1)) > d$, then the result follows by *Lemma* 11 with $\epsilon = \gamma = \frac{1}{2r}$.
- If $\Delta_c(\boldsymbol{a}, \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v}_0) \| \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v}_1)) \leq d$ and the prover is honest then:
  - The first condition follows since $\Delta_c(\mathsf{PVAL}(\mathcal{C}(\mathbb{F}), g_0 + cg_1(\mathbb{F}))) \geq \Delta_c(\mathsf{PVAL}(\boldsymbol{j}, 0)) \geq 4d$, and we know that by the honesty definition that $\boldsymbol{a}_0^* + c\boldsymbol{a}_1^* \circ \tau \in \mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a}_0 + c\boldsymbol{a}_1 \circ \tau)$. Thus, it must also be unique.
  - The second part follows from *Lemma* 11.
- If the prover is dishonest first in:
  - The GKR i.e. Protocol 5 then with probability $\geq 1 - \mathsf{err}_{\mathsf{GKR}}$, we have that $\Delta_c(\boldsymbol{a}_1 \circ \pi, \mathsf{PVAL}(\boldsymbol{j}', v')) \geq 4d$. Now, by Lemma 8, the result follows.
  - In returning g: If the checks pass, but the prover lied in the value of g, then we know that either $\Delta_c(\boldsymbol{a}_0^*, \mathsf{PVAL}(\mathcal{C}(\mathbb{F}), g_0(\mathbb{F}))) \geq 4d$ or $\Delta_c(\boldsymbol{a}_1^* \circ \tau, \mathsf{PVAL}(\mathcal{C}(\mathbb{F}), g_1(\mathbb{F}))) \geq 4d$. Thus, since $\boldsymbol{a}_0^*$ and $\boldsymbol{a}_1^* \circ \tau$ are within d distance of $\boldsymbol{a}_0$ and $\boldsymbol{a}_1 \circ \tau$ by the GKR correctness and our assumption respectively. Thus, we can again simply conclude by the triangle inequality and Lemma 8.

□

6.3.3 Efficient UIP for PVAL emptiness

Since we have used non-emptiness conditions on the PVAL instances defined in Lemma 11, we need to be able to check the same. The following is a standard interactive proof for checking whether a given PVAL instance is empty. This is based on the observation that checking emptiness of a PVAL instance is simply a rank computation, and thus we can use Theorem 3 on such a circuit. We will make use of this lemma freely now.

Lemma 14 (Lemma 5.10 in [RR20])**.** *Let $T, m, \sigma \in \mathbb{N}$ and $\mathbb{F}$, a finite field of characteristic 2. There is a public coin unambiguous interactive proof for the language*

$$\mathcal{L} = \left\{(\boldsymbol{j}, \boldsymbol{v}) \in \left((\mathbb{F}^m)^T, \mathbb{F}^T\right) \mid \mathit{PVAL}(\boldsymbol{j}, \boldsymbol{v}) \neq \varnothing\right\},$$

with perfect completeness and the following parameters:

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

Proof. Let matrix $M \in \mathbb{F}^{2^m \times T}$ where

$$M_{a,b} = \hat{x}^a(j_b)$$

where $x^a = \prod_{a_r=1} x_r$. Observe that each entry of the matrix can be computed using an arithmetic circuit over $\mathbb{F}$ of depth $O(\log m)$ and size m. Now, since each multiplication and addition over $\mathbb{F}$ as a boolean circuit is of depth $\log |\mathbb{F}|$ and size $\log^2 |\mathbb{F}|$. Thus, each coordinate of M can be computed in size $m \log |\mathbb{F}|^2$ and depth $O(\log |\mathbb{F}| \log m|)$.

The rank of a matrix can be computed in depth $O(\log^2(T \cdot 2^m)$ as a circuit over $\mathbb{F}$ and in size $\text{poly}(2^m T)$. Now, each operation over $|\mathbb{F}|$ multiplies another $\log^2 |\mathbb{F}|$ to size and another $\log |\mathbb{F}|$ to the depth. Now, the verifier can simply check using GKR whether $\mathsf{rank}(M) = \mathsf{rank}(M \| \boldsymbol{v})$. Observe that if these two ranks are equal then $\boldsymbol{v} \in \mathbb{F}^T$ is in the $\mathbb{F}$ linear span of the columns of M i.e. there is a polynomial whose extension applied to $\boldsymbol{j}$ has the image $\boldsymbol{v}$ and the affine space has dimension > 0 if the rank is not full. Similarly, we could have also just checked if $\dim \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v}) > 0$ instead of just checking if $|\mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})| > 0$. □

6.3.4 Reducing Number of Evaluation Points:

Lemma 15. *$m, \log L, T, \sigma, d > 0$ are integers. $\mathbb{F}$ is a constructible field of characteristic 2. We also have $\boldsymbol{j} \in (\mathbb{F}^{m+\log L})^T$ and uniformly random points $\boldsymbol{j}' \in (\mathbb{F}^{m+\log L})^T$. $\boldsymbol{\Lambda} = (\Lambda_1, \ldots, \Lambda_{2T})$ are canonical distinct fixed points in $\mathbb{F}$. Let $\boldsymbol{a} : \{0,1\}^{m+\log L} \to \mathbb{F}$ and let $g : \mathbb{F} \to \mathbb{F}$ be a univariate polynomial of degree at most $2T(m + \log L)$. Additionally, assume that $\Delta_c(\mathsf{PVAL}(\boldsymbol{j}, 0))$ is non-empty.*

Then, we have the following unambiguous distance preservation property:

- *If $\Delta_c(\boldsymbol{a}, \mathsf{PVAL}(\mathcal{C}(\mathbb{F}), g(\mathbb{F}))) \leq d$ then:*

$$\Pr_{\alpha=(\alpha_1,\ldots,\alpha_T)\in\mathbb{F}^T}[\mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a}) \cap \mathsf{PVAL}(\mathcal{C}(\alpha), g(\alpha)) = \mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a}) \cap \mathsf{PVAL}(\mathcal{C}(\mathbb{F}, g(\mathbb{F})))] \geq p$$

- *If $\Delta_c(\boldsymbol{a}, \mathsf{PVAL}(\mathcal{C}(\mathbb{F}), g(\mathbb{F}))) > d$ then:*

$$\Pr_{\alpha=(\alpha_1,\ldots,\alpha_T)\in\mathbb{F}^T}[\Delta_c(\mathsf{PVAL}(\mathcal{C}(\alpha), g(\alpha)), \boldsymbol{a}) > d] \geq p,$$

where $p = 1 - \left(\binom{M}{d}|\mathbb{F}|^d\right)^L \left(\frac{2T(m+\log L)}{\mathbb{F}}\right)^T$.

Additionally,

$$\Pr_{\alpha=(\alpha_1,\ldots,\alpha_T)\in\mathbb{F}^T}[\Delta_c(\mathsf{PVAL}(\mathcal{C}(\alpha), 0)) \geq 4d] \geq 1 - \left(\binom{M}{4d}|\mathbb{F}|^{4d}\right)^L \left(\frac{m+\log L}{\mathbb{F}}\right)^T$$

where $\mathcal{C} : \mathbb{F} \to \mathbb{F}^{m+\log L}$ is the canonical $2T-1$ degree curve passing through $\boldsymbol{j}, \boldsymbol{j}'$.

Proof. For the first part, observe that for if $\boldsymbol{a}' \in \mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a}) \backslash \mathsf{PVAL}(\mathcal{C}(\mathbb{F}), g(\mathbb{F}))$ then we have that $\widehat{\boldsymbol{a}'} \circ \mathcal{C} \neq g$. Thus, $\Pr_{\alpha \in \mathbb{F}^T}[\widehat{\boldsymbol{a}'} \circ \mathcal{C}(\alpha) = g(\alpha)] \leq \left(\frac{2T(m+\log L)}{\mathbb{F}}\right)^T$ by the Schwartz-Zippel Lemma(Lemma 1). Thus, we finish by a union bound over all elements of $\mathcal{B}_{d,\mathbb{F}}(\boldsymbol{a})$.

For the second part, observe that since $\boldsymbol{j}'$ is uniformly random, thus so is $\mathcal{C}(\alpha)$. Thus, for any element $\boldsymbol{a}'$ such that $0 < \Delta_c(0, \boldsymbol{a}') < 4d$, we have that $\Pr_{\alpha \in \mathbb{F}^T}[\widehat{\boldsymbol{a}'}(\mathcal{C}(\alpha)) = 0] \leq \left(\frac{m+\log L}{|\mathbb{F}|}\right)^T$ by Lemma 1. We can now finish by the union bound over all elements.

Remark. In the proof of the second part, we used the fact that j' was uniformly random. If we do not want to use this fact then we can observe that $\widehat{\boldsymbol{a}'} \circ \mathcal{C} \neq 0$ for any $\boldsymbol{a}'$ since $\Delta_c(\mathsf{PVAL}(j, 0)) \geq 8d$ and have a loss of a factor of T, but this will not matter much anyway. □

6.3.5 Putting together the Reduction Protocol:

Protocol 8 The Reduction Protocol

Input Parameters: m, L, T, $\sigma \in \mathbb{N}$ and a field $\mathbb{F}$.

Input: $\boldsymbol{j} \in (\mathbb{F}^{m+\log L})^T$, $\boldsymbol{v} \in \mathbb{F}^T$.

Prover Auxiliary Input: $\boldsymbol{a} : \{0,1\}^{m+\log L} \to \mathbb{F}$.

Output: A permutation $\tau \in \{0,1\}^{m-1+\log L} \to \{0,1\}^{m-1+\log L}$, a scalar $c \in \mathbb{F}$, and a smaller instance: $\boldsymbol{a}_{\text{new}} : \{0,1\}^{m-1+\log L} \to \mathbb{F}$, $\boldsymbol{j}_{\text{new}} \in (\mathbb{F}^{m-1+\log L})^T$ and $\boldsymbol{v}_{\text{new}} \in \mathbb{F}^T$.

1: Run Protocol 6 on input $\boldsymbol{j}, \boldsymbol{v}$ and $\boldsymbol{a}$ to get output $\boldsymbol{j}'', \zeta^{(0)}, \zeta^{(1)}$.
2: Break $\boldsymbol{a}$ into $\boldsymbol{a}_0 \| \boldsymbol{a}_1$.
3: Run Protocol 7 on input $\boldsymbol{j}'', \boldsymbol{v}_0 = \zeta^{(0)}, \boldsymbol{v}_1 = \zeta^{(1)}$, obtaining $\tau, \mathcal{C}, \boldsymbol{j}', \boldsymbol{v}', g_0, g_1$.
4: **PVAL Emptiness Check:** Both parties run the Protocol in Lemma 14 with error parameter σ to check the non-emptiness of $\mathsf{PVAL}((\boldsymbol{j}, \boldsymbol{j}'), g^{(0)}(\Lambda))$ and $\mathsf{PVAL}((\boldsymbol{j}, \boldsymbol{j}'), g^{(1)}(\Lambda))$.
5: V rejects if the check fails.
6: V samples $\alpha_1, \ldots, \alpha_T \leftarrow \mathbb{F}$ and $c \leftarrow \mathbb{F}$ and sends them to P.
7: V outputs $\tau, c, \boldsymbol{a}_{\text{new}} = \boldsymbol{a}_0 + c(\boldsymbol{a}_1 \circ \tau)$, $\boldsymbol{j}_{\text{new}} = (\mathcal{C}(\alpha_i))_{i=1}^T$ and $\boldsymbol{v}_{\text{new}} = ((g_0 + cg_1)(\alpha_i))_{i=1}^T$.

Lemma 16. *Let $m, \log L, T, \sigma, d, r, \in \mathbb{N}$ such that $r < d/4$ and $4d < 2^{m/50}$ and let $\mathbb{F}$ be a constructible field. Let $\boldsymbol{a} : \{0,1\}^{m+\log L} \to \mathbb{F}$ and $\boldsymbol{j} \in (\mathbb{F}^{m+\log L})^T, \boldsymbol{v} \in \mathbb{F}^T$ such that $\Delta_c(\mathsf{PVAL}(\boldsymbol{j}, 0)) \geq 4d$. Define the following quantity.*

$$p := 1 - \left(\frac{2r+1}{|\mathbb{F}|} + 2\left(\binom{M}{4d}|\mathbb{F}|^{4d}\right)^L \left(\frac{m+\log L}{\mathbb{F}}\right)^T\right)$$
$$- \left(\left(\binom{M}{d}|\mathbb{F}|^d\right)^L \left(\frac{2T(m+\log L)}{\mathbb{F}}\right)^T + 2^{2-\sigma} + \mathsf{err}_{GKR} + \exp(-d/12r)\right).$$

If we run Protocol 8 and get outputs $\boldsymbol{a}_{new}, \boldsymbol{j}_{new}, \boldsymbol{v}_{new}$ then we have the following properties:

- ***Completeness:*** *If $\boldsymbol{a} \in \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$ and the prover was honest then $\boldsymbol{a}_{new} \in \mathsf{PVAL}(\boldsymbol{j}_{new}, \boldsymbol{v}_{new})$.*
- ***Soundness:*** *If $\boldsymbol{a} \notin \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$ with respect to $\boldsymbol{a}$ then*

$$\Pr[\Delta_c(\boldsymbol{a}_{new}, \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})) \geq d(1 - \frac{1}{r})] \geq p.$$

- ***Unambiguity:*** *If* $\boldsymbol{a} \in \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$ *but the prover deviates from the honest strategy then*

$$\Pr[\Delta_c(\boldsymbol{a}_{new}, \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})) \geq d(1 - \frac{1}{r})] \geq p.$$

- *With probability at least* $1 - \left(\binom{M}{4d}|\mathbb{F}|^{4d}\right)^L \left(\frac{m+\log L}{\mathbb{F}}\right)^T$, *we have that* $\Delta_c(\mathsf{PVAL}(\boldsymbol{j}_{new}, 0)) \geq 4d$.

Additionally, the protocol has the following complexities:

- ***Round Complexity:*** $\tilde{O}(D \log S \cdot \mathrm{poly}(m + \log L, \log(T \log |\mathbb{F}|), \sigma))$
- ***Communication Complexity:*** $(T + d) \cdot \mathrm{poly}(m + \log L, \log(T \log |\mathbb{F}|), \sigma)$
- ***Verifier Runtime:*** $(T + d)D \log S \cdot \mathrm{poly}(m + \log L, \log(T \log |\mathbb{F}|), \sigma)$.
- ***Prover Runtime:*** $\mathrm{poly}(S, T, 2^{m+\log L}, \log |\mathbb{F}|, \sigma)$.
- ***Verifier's verification Circuit Size:*** $\tilde{O}(T)\mathrm{poly}(\log \mathbb{F}, m + \log L, \sigma)$.
- ***Verifier's verification circuit depth:*** $\mathrm{polylog}(T, m + \log L, \log |\mathbb{F}|, \sec)$.

If we further assume that

- $|\mathbb{F}| \geq \tilde{\Omega}(2^\sigma \cdot T^2(m + \log L + \log T)^2)$.
- $T > 8d \cdot 2^{\log L} \cdot m$
- $d \geq 16 \cdot r \cdot \sigma$
- $r = o(|\mathbb{F}| \cdot 2^{-\sigma})$,

then

$$\left(\frac{2r+1}{|\mathbb{F}|} + 3\left(\binom{M}{4d}|\mathbb{F}|^{4d}\right)^L \left(\frac{m+\log L}{\mathbb{F}}\right)^T\right)$$
$$+ \left(\left(\binom{M}{d}|\mathbb{F}|^{d}\right)^L \left(\frac{2T(m+\log L)}{\mathbb{F}}\right)^T + 2^{2-\sigma} + \mathsf{err}_{GKR} + \exp(-d/12r)\right) \leq 2^{3-\sigma}.$$

Proof. This statement is a combination of Lemmas 12 to 15. The only thing left to show is the final verifier's verification circuit and the setting of parameters.

The verifier's verification checks are:

- Checking Linear Combinations: Size: $T\mathrm{polylog}(\mathbb{F})$ and Depth $O(\log(T) + \log\log |\mathbb{F}|)$.
- Protocol 5: Depth D and size S.
- Checking $O(T)$ polynomial evaluations:
  Size $\tilde{O}(T \cdot \log |\mathbb{F}|(m + \log L))$ and depth $\mathrm{poly}(\log(T \log |\mathbb{F}|(m + \log L)))$ since we can run multipoint polynomial evaluation.
- Emptiness checks: Size $T \cdot \mathrm{poly}(m + \log L, \sigma, \log |\mathbb{F}|)$ and depth $\mathrm{polylog}(T, m + \log L, \log |\mathbb{F}|, \sigma)$.

Thus, putting all these together, we get the desired result.

We verify the parameters set:

- $\frac{2r+1}{|\mathbb{F}|} \leq 2^{-\sigma}$
- $\left(\binom{M}{4d} \cdot |\mathbb{F}|^d\right)\left(\frac{(m+\log L)}{|\mathbb{F}|}\right)^T \leq \left(\frac{(2m+2\log L)^m \cdot |\mathbb{F}|}{|\mathbb{F}|^m}\right)^{4dL} \leq \frac{1}{|\mathbb{F}|} \leq 2^{-\sigma}.$
- $\left(\binom{M}{d} \cdot |\mathbb{F}|^d\right)\left(\frac{T(m+\log L)}{|\mathbb{F}|}\right)^T \leq \left(\frac{T(m+2\log L)^{8m} \cdot |\mathbb{F}|}{|\mathbb{F}|^{8m}}\right)^{dL} \leq \frac{1}{|\mathbb{F}|} \leq 2^{-\sigma}$
- $\mathsf{err}_{\mathsf{GKR}} \leq \left(\epsilon_{\mathsf{GKR}}(D, \log S, |\mathbb{F}|) + (\epsilon_{\mathsf{GKR}}(D, \log S, |\mathbb{F}|))^T \cdot \left(\binom{M}{4d}|\mathbb{F}|^{4d}\right)^L\right)$
  $\leq O\left(\frac{D \log S}{|\mathbb{F}|}\right) + \left(\frac{(O(D \log S)^m \cdot |\mathbb{F}|}{|\mathbb{F}|^m}\right)^{4dL} \leq \frac{1}{|\mathbb{F}|} \leq 2^{-\sigma}$
- $\exp(-d/12r) \leq 2^{-\sigma}$.

Combining these we get the desired result. □

6.3.6 The Base Protocol

Protocol 9 Base protocol in RR.

Input Parameters: m, m_{new} (where $m \geq m_{\text{new}}$), L, σ, $d \in \mathbb{N}$ and $\mathbb{F}$ is a field.
Input: $\boldsymbol{j} \in \left(\mathbb{F}^{m_{\text{new}}+\log L}\right)^T$, $\boldsymbol{v} \in \mathbb{F}^T$.
Other Parameters: Let $M := 2^m$ and $M^{\text{new}} := 2^{m_{\text{new}}}$.
Prover Auxiliary Input: A function $\boldsymbol{a}_{\text{new}} \in \{0,1\}^{m_{\text{new}}} \times \{0,1\}^{\log L} \to \mathbb{F}$.
Verifier Auxiliary Input: A description $\langle H \rangle$ of a $\mathbb{F}$-linear function $H : (\mathbb{F}^L)^M \to (\mathbb{F}^L)^{M_{\text{new}}}$, such that its every output coordinate is a linear combination of exactly $\frac{M}{M_{\text{new}}}$ elements of $\mathbb{F}^L$. (We assume that the set of rows that fold into any given row of M_{new} are easy to compute given the description $\langle H \rangle$.)
Verifier Query Access: Query access to $\boldsymbol{a} : \{0,1\}^m \times \{0,1\}^{\log L} \to \mathbb{F}$.
Output: A succinct description $\langle Q \rangle$ of $Q \subset \{0,1\}^m$ of size $\left\lceil \frac{\sigma \cdot M}{d} \right\rceil$ and a succinct predicate $\langle \Phi \rangle : \mathbb{F}^{|Q| \times L} \to \{0,1\}$.

1: P explicitly sends all of $\boldsymbol{a}_{\text{new}}$ i.e. the entire string in $\mathbb{F}^{2^{m_{\text{new}}} \times L}$ to V.
2: V checks if all the extensions are correct i.e. $\boldsymbol{a}_{\text{new}} \in \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$ (and rejects otherwise).
3: V randomly selects Q' to be a uniform subset of $\left\lceil \frac{\sigma \cdot M}{d} \right\rceil$ of $[M_{\text{new}}]$.
4: V outputs Q as the set of rows that the rows of $H|_{Q'}$ depends on. Φ checks if $H(\boldsymbol{a}|_Q) = \boldsymbol{a}_{\text{new}}|_{Q'}$.

Claim 4. *If $\Delta_c(\mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v}), H(\boldsymbol{a})) \geq d$ then with probability $\geq 1 - 2^{-\sigma}$, we have that $\Phi(\boldsymbol{a}|Q) = 0$. Additionally, Protocol 9 has the following complexities:*

- *Round Complexity: $O(m_{new})$*

- *Communication Complexity:* $O(2^{m_{new}} \cdot L \cdot \log(\mathbb{F}) + T \cdot (m_{new} + \log L) \cdot \log(\mathbb{F}))$.
- *Verifier runtime:* $O(M_{new} \cdot L \cdot \text{poly}(\log L + m, \log \mathbb{F}) \cdot T)$
- *Prover runtime:* $M_{new} L \cdot \log |\mathbb{F}| \text{poly}(T, m_{new} + \log L)$

Proof. When the distance is at least d, the verification passes exactly when none of the rows in the sampled Q' are inconsistent with H. This probability is at most $(1 - \frac{d}{M_{\text{new}}})^{|Q'|} \leq 2^{-\sigma}$. □

6.3.7 The complete protocol:

Theorem 11 (IPP for PVAL with column distance)**.** *Let $m, \log L, T, \sigma, d \in \mathbb{N}$. Let $\mathbb{F}$ be a constructible field ensembles such that $\mathbb{F}$ has characteristic 2. If*

- $|\mathbb{F}| \geq \Omega(2^\sigma \cdot T^2 (m + \log L)^2)$,
- $T > 8d \cdot 2^{\log L} \cdot m$,
- $d \geq 16 \cdot m \cdot \sigma$

Then for any $\boldsymbol{j} \in (\mathbb{F}^{m+\log L})^T, v \in \mathbb{F}^T$, the matrix $\mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})$ has a public coin unambiguous IPP (Protocol 10) as long as $\Delta_c(\mathsf{PVAL}(\boldsymbol{j}, 0)) > 4d$:

- *Soundness error:* $m \cdot 2^{4-\sigma}$
- *Prover to Verifier communication complexity:* $T \log |\mathbb{F}| \cdot \text{poly}(m, \log L) + L \cdot \log |\mathbb{F}| \cdot \text{poly}(d)$.
- *Verifier to Prover communication complexity:* $T \log |\mathbb{F}| \cdot \text{poly}(m, \log L)$
- *Round Complexity:* $\text{poly}(m, \log L, \log(T \log |\mathbb{F}|))$
- *Verifier runtime:* $T \log |\mathbb{F}| \cdot \text{poly}(m, \log L, \log(T \log |\mathbb{F}|), \log |\mathbb{F}|) + \tilde{O}(L \cdot \log |\mathbb{F}| \cdot \text{poly}(d))$
- *Prover runtime:* $\text{poly}(2^{m+\log L}, T \log |\mathbb{F}|)$
- *Query complexity :* $|\mathcal{Q}| = \frac{4\sigma \cdot 2^{m+\log L}}{d}$.

The final verifier verdict is given by a succinctly described predicate $\Phi : \mathbb{F}^{\frac{4\sigma}{d} \cdot 2^{m+\log L}} \mapsto \{0,1\}$ along with a succinctly described set $Q \subseteq [2^m]$ such that $|Q| = \frac{4\sigma \cdot 2^m}{d}$, and the verdict is $\Phi(f|_{Q \times \{0,1\}^{\log L}})$ with $|\langle Q \rangle| = \tilde{O}(m^2 d\sigma \cdot + m \log |\mathbb{F}| + \sigma \cdot \text{poly}(d))$ and $|\langle \Phi \rangle| = |\langle Q \rangle| + \sigma \cdot \text{poly}(d) \cdot L \cdot \log |\mathbb{F}|$.

Proof. Observe that the total error is at most the sum of the errors in each round which we know is at most $2^{3-\sigma}$ for the first m rounds if $r = m$ in Lemma 16. This would result in that if $\Delta_c(\boldsymbol{a}, \mathsf{PVAL}(\boldsymbol{j}, v)) \geq d$ initially, then after i rounds, we have $\Delta_c(\boldsymbol{a}_{\text{prot}}, \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})) \geq d(1 - \frac{1}{m})^i$. Thus, with probability at least $m \cdot 2^{3-\sigma}$, we have that when Protocol 9 is called then indeed $\Delta_c(\boldsymbol{a}_{\text{prot}}, \mathsf{PVAL}(\boldsymbol{j}, \boldsymbol{v})) > d/4$ as required. Thus, the total error is at most $m \cdot 2^{4-\sigma}$ as required.

The total round, time, communication complexities are just the sum of m times the complexities of Protocol 8 and the complexity of Protocol 9, as desired. The final query complexity is as described in Protocol 9.

We discuss the succinct descriptions in the next subsection. □

Protocol 10 Efficient IPP for PVAL with column distance

Input Parameters: m, $L \in \mathbb{N}$, σ, $d \geq 16\sigma \cdot m$, $\mathbb{F}$ is a field with $|\mathbb{F}| \geq 32 \cdot 2^{\sigma}(T \log M \log L)^{C}$.
Input: $\boldsymbol{j} \in (\mathbb{F}^{m+\log L})^{T}, \boldsymbol{v} \in \mathbb{F}^{T}$.
Other Parameters: Let $r \coloneqq m$.
Prover Auxiliary Input: $\boldsymbol{a} : \{0,1\}^{m+\log L} \to \mathbb{F}$.
Verifier Query Access: Query access to $\boldsymbol{a} : \{0,1\}^{m+\log L} \to \mathbb{F}$.
Output: Succinct descriptions of $Q \subset [M]$ and $\Phi : \mathbb{F}^{|Q| \times L} \to \{0,1\}$.

V maintains a linear function $H : \left(\mathbb{F}^{\log L}\right)^{M} \to \left(\mathbb{F}^{\log L}\right)^{M_{\text{prot}}}$.
Set the variable parameters: $m_{\text{prot}} \coloneqq m$, $H_{\text{prot}} \coloneqq Id$ and $\boldsymbol{a}_{\text{prot}} \coloneqq \boldsymbol{a}$.
while True **do**
  if $2^{m_{\text{prot}}/50} \leq 4d$: **then**
    Both parties run Protocol 9 on input parameters m, m_{prot}, L, σ, $d/4$, $\mathbb{F}$, and input $\boldsymbol{j}$, $\boldsymbol{v}$, prover auxiliary input $\boldsymbol{a}_{\text{prot}}$, verifier auxiliary input $\langle H_{\text{prot}} \rangle$ and verifier query access to $\boldsymbol{a}$, obtaining $\langle Q \rangle$ and $\langle \Phi \rangle$.
    V outputs $\langle Q \rangle$ and $\langle \Phi \rangle$.
  else
    Both parties run Protocol 8 on input parameters m_{prot}, L, σ and T, input $\boldsymbol{j}, \boldsymbol{v}$ and prover auxiliary input $\boldsymbol{a}_{\text{prot}} = \boldsymbol{a}_{0,\text{prot}} \| \boldsymbol{a}_{1,\text{prot}}$, obtaining τ, c, $\boldsymbol{j}_{\text{new}}, \boldsymbol{v}_{\text{new}}$ and $\boldsymbol{a}_{\text{new}} = \boldsymbol{a}_0 + c\boldsymbol{a}_1 \circ \tau$.
    Update $\boldsymbol{j} = \boldsymbol{j}_{\text{new}}$, $\boldsymbol{v} = \boldsymbol{v}_{\text{new}}$, $m_{\text{prot}} = m_{\text{prot}} - 1$, and $\boldsymbol{a}_{\text{prot}} = \boldsymbol{a}_{\text{new}}$.
    Update $H_{\text{prot}} = G \circ H_{\text{prot}}$ where $G(x_0 \| x_1) = x_0 + cx_1 \circ \tau$. $\langle H_{\text{prot}} \rangle = \langle H_{\text{prot}} \rangle \| (\tau, c)$.

6.4 Succinct Descriptions

We verify that the output parameters (Q, Φ) have succinct descriptions. The size of $\langle Q \rangle$ is given by the randomness of the choice in *Protocol* 9 which is $\text{poly}(d) \cdot \sigma \cdot \log |\mathbb{F}|$ and the (c, π) pairs from the m rounds. Thus, giving us a total complexity of $\tilde{O}(m^2 d\sigma \cdot + m \log |\mathbb{F}| + \sigma \cdot \text{poly}(d))$ since the seed length of a single π is $\tilde{O}(m \cdot d \cdot \sigma)$.

Additionally, $|\langle \Phi \rangle|$ is just given by $|\langle Q \rangle| + |Q'| \cdot L \cdot \log |\mathbb{F}|$ since $\langle Q \rangle$ contains the entire details of $\langle H \rangle$. Thus, we only need $\boldsymbol{a}_{\text{prot}} | Q'$ which is $\log L \cdot |Q'| \cdot \log |\mathbb{F}| = \log L \log |\mathbb{F}| \sigma \cdot \text{poly}(d)$ bits.

Circuit G for generating Q Let $m' = m_{\text{prot}}$ when *Protocol* 9 is called.

The set Q has succinct description

$$\langle Q \rangle = (\langle Q \rangle', \{\pi : \pi \text{ is used in the folding steps of the protocol}\}),$$

where $Q' \subset \{0,1\}^{m'}$ are the $\frac{4\sigma \cdot 2^{m'}}{d}$ random rows selected in the end of the iteration and $\langle R \rangle \in \{0,1\}^{|Q'| \cdot m'}$ are the random coins used to generate Q'.

Given $(i, \langle Q \rangle)$, G needs to generate the i-th element of Q. The idea is to carry out the folding steps in reverse, which can be done as follows. First, it generates the full set Q' using $\langle R \rangle$, and let $Q_0 \coloneqq Q'$, which requires $\tilde{O}(|Q'|)$ gates and $\tilde{O}(1)$ depth. If $i \leq |Q|/2$, the i-th element of Q would lie in the first half before folding, so G prepends a 0 to every element in Q_0. Otherwise, G prepends a 1 to every element in Q_0. Since the indices in the second half were permuted by π, G also applies π^{-1} to every element in R_0 to reconstruct their locations in the original index space.

The circuit G repeats this process $(m - m')$ times, until it obtains $|Q'|$ indices that lies in $\{0,1\}^m$, and it outputs the $(i \bmod |Q'|)$-th index.

Circuit $C(\boldsymbol{a}|Q \times \{0,1\}^{\log L}, \langle\Phi\rangle)$ for checking $\Phi(\boldsymbol{a}|Q \times \{0,1\}^{\log L})$ Let $Q' \subset \{0,1\}^{m'}$ be the $\left\lceil \frac{4\sigma \cdot 2^{m'}}{d} \right\rceil$ random rows selected in the end of the iteration, and $\langle Q\rangle' \in \{0,1\}^{|Q'| \cdot m'}$ be the random coin used to generate it. The predicate Φ can be described by the pair

$$(\langle Q\rangle', \{(c,\pi) : (c,\pi) \text{ is used in the folding steps of the protocol}\}, \boldsymbol{\psi}),$$

where ψ is the set of values claimed by the prover on Q'.

The circuit first enumerates the entire Q using $\langle Q\rangle'$ and all the π's (as in the last paragraph), and then it applies the $\mathbb{F}$-linear function $\phi : \mathbb{F}^{|Q| \times 2^{\log L}} \to \mathbb{F}^{2^{\log L}}$ determined by all the (c,π) used in the folding steps of the protocol, on the $2^{\log L}$ columns of the truth table of f in parallel. This can be done by iteratively applying the corresponding π and figuring out which of c to multiply in every folding step for each position in Q, and requires $\tilde{O}(|Q| \cdot 2^{\log L} + \sigma + 2^{\log L} \log |\mathbb{F}|)$ gates and $\tilde{O}(1)$ depth.

6.5 Construction of the Verifier's Verification Circuit

We now wish to understand the size and depth of the verifier's verification circuit $\mathsf{V}_{\Delta_c \mathsf{RR}}$. $\mathsf{V}_{\Delta_c \mathsf{RR}}$ on reading the transcript of the interaction and the verifier decides whether to reject or to check Φ.

Claim 5. *If the transcript of the interaction is of length* L*, then* $\mathsf{V}_{\Delta_c RR}$*, the verification circuit, has the following properties:*

- *Size of* $\mathsf{V}_{\Delta_c RR}$ *is* $\tilde{O}(\mathrm{L})$ *where we ignore* $\mathrm{poly}(m + \log L)$ *factors.*
- *Depth of* $\mathsf{V}_{\Delta_c RR}$ *is* $\tilde{O}(\log \mathrm{L})$ *where we ignore* $\mathrm{poly}(m + \log L)$ *factors.*

Proof. Observe that the verifier conducts the following checks:

(1) $\tilde{O}(Tm)$ linearity checks to see if the verifier broke the instances into two claims correctly.

(2) $\tilde{O}(m)$ PVAL emptiness checks which are run via a GKR.

(3) $\tilde{O}(m)$ runs of Protocol 5 to apply the pairwise independent map and convert f_1 into $f_1 \circ \tau$.

(4) $\tilde{O}(m)$ recalculations of input points $\boldsymbol{j}$ for recursing via evaluating low degree curves.

(5) Checking if $\tilde{O}(T)$ low degree extensions are correct.

Now, step (1) is clearly linear in it's input. Step (2) and (3) can be done in nearly linear input size and logarithmic depth due to Theorem 3 and Lemma 2. We may also need to check $\tilde{O}(m)$ low degree extensions which can be done in nearly linear of their input length as well. Thus, we just need to show that interpolation and evaluation of univariate polynomials can be done in nearly linear number of field operations. We know that this is possible due to our assumptions about our fields and multipoint interpolation and evaluation results (Corollary 10.8 and 10.12 in [vzGG13]).

Thus, all the checks can be done in nearly linear size of the input transcript. The depth of the overall circuit is also clearly $\tilde{O}(\log \mathrm{L})$. □

Acknowledgement

The authors thank Ron Rothblum, Guy Rothblum, and the anonymous FOCS reviewers for their helpful comments and suggestions.

M.M.H. is partially supported by National Institute of Health (NIH) R01HG010959 (to B.B.). M.M.H., R.G. and Y.T.K. are supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR0011-25-C-0300 (to Y.T.K.). Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Defense Advanced Research Projects Agency (DARPA).

References

[ACFY24] Gal Arnon, Alessandro Chiesa, Giacomo Fenzi, and Eylon Yogev. STIR: Reed-solomon proximity testing with fewer queries. In Leonid Reyzin and Douglas Stebila, editors, *CRYPTO 2024, Part X*, volume 14929 of *LNCS*, pages 380–413. Springer, Cham, August 2024.

[ACFY25] Gal Arnon, Alessandro Chiesa, Giacomo Fenzi, and Eylon Yogev. WHIR: Reed-solomon proximity testing with super-fast verification. In Serge Fehr and Pierre-Alain Fouque, editors, *EUROCRYPT 2025, Part IV*, volume 15604 of *LNCS*, pages 214–243. Springer, Cham, May 2025.

[AHIV23] Scott Ames, Carmit Hazay, Yuval Ishai, and Muthuramakrishnan Venkitasubramaniam. Ligero: lightweight sublinear arguments without a trusted setup. *DCC*, 91(11):3379–3424, 2023.

[ALM+92] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and hardness of approximation problems. In *33rd FOCS*, pages 14–23. IEEE Computer Society Press, October 1992.

[AS92] Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs; A new characterization of NP. In *33rd FOCS*, pages 2–13. IEEE Computer Society Press, October 1992.

[BCS16] Eli Ben-Sasson, Alessandro Chiesa, and Nicholas Spooner. Interactive oracle proofs. In Martin Hirt and Adam D. Smith, editors, *TCC 2016-B, Part II*, volume 9986 of *LNCS*, pages 31–60. Springer, Berlin, Heidelberg, October / November 2016.

[BFL90] László Babai, Lance Fortnow, and Carsten Lund. Non-deterministic exponential time has two-prover interactive protocols. In *31st FOCS*, pages 16–25. IEEE Computer Society Press, October 1990.

[BFLS91] László Babai, Lance Fortnow, Leonid A. Levin, and Mario Szegedy. Checking computations in polylogarithmic time. In *23rd ACM STOC*, pages 21–31. ACM Press, May 1991.

[BGKW88] Michael Ben-Or, Shafi Goldwasser, Joe Kilian, and Avi Wigderson. Multi-prover interactive proofs: How to remove intractability assumptions. In *20th ACM STOC*, pages 113–131. ACM Press, May 1988.

[BH08] Alex Brodsky and Shlomo Hoory. Simple permutations mix even better. *Random Structures & Algorithms*, 32(3):274–289, 2008.

[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, *56th ACM STOC*, pages 435–443. ACM Press, June 2024.

[BKS18] Eli Ben-Sasson, Swastik Kopparty, and Shubhangi Saraf. Worst-case to average case reductions for the distance to a code. In *Proceedings of the 33rd Computational Complexity Conference*, CCC '18. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2018.

[BM88] László Babai and Shlomo Moran. Arthur-Merlin games: A randomized proof system, and a hierarchy of complexity classes. *J. Comput. Syst. Sci.*, 36(2):254–276, 1988.

[BSCI+23] Eli Ben-Sasson, Dan Carmon, Yuval Ishai, Swastik Kopparty, and Shubhangi Saraf. Proximity gaps for Reed-Solomon codes. *Journal of the ACM*, 70(5), 2023.

[Coo71] Stephen A Cook. The complexity of theorem-proving procedures. In *3rd ACM STOC*, pages 151–158, 1971.

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

[FGL+91] Uriel Feige, Shafi Goldwasser, László Lovász, Shmuel Safra, and Mario Szegedy. Approximating clique is almost NP-complete (preliminary version). In *32nd FOCS*, pages 2–12. IEEE Computer Society Press, October 1991.

[GHKO25] William Gay, William He, Nicholas Kocurek, and Ryan O'Donnell. Pseudorandomness properties of random reversible circuits. In Yael Tauman Kalai and Seny F. Kamara, editors, *CRYPTO 2025, Part I*, volume 16000 of *LNCS*, pages 651–678. Springer, Cham, August 2025.

[GHP25] Lucas Gretta, William He, and Angelos Pelecanos. More efficient approximate k-wise independent permutations from random reversible circuits via log-sobolev inequalities. In Yossi Azar and Debmalya Panigrahi:, editors, *36th SODA*, pages 5582–5598. ACM-SIAM, January 2025.

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

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

[Gow96] W. T. Gowers. An almost m-wise independent random permutation of the cube. *Combinatorics, Probability and Computing*, 5(2):119–130, 1996.

[GR15] Tom Gur and Ron D. Rothblum. Non-interactive proofs of proximity. In Tim Roughgarden, editor, *ITCS 2015*, pages 133–142. ACM, January 2015.

[GR17] Tom Gur and Ron D. Rothblum. A hierarchy theorem for interactive proofs of proximity. In Christos H. Papadimitriou, editor, *ITCS 2017*, volume 4266, pages 39:1–39:43, 67, January 2017. LIPIcs.

[GS86] Shafi Goldwasser and Michael Sipser. Private coins versus public coins in interactive proof systems. In *18th ACM STOC*, pages 59–68. ACM Press, May 1986.

[HMMR04] Shlomo Hoory, Avner Magen, Steven Myers, and Charles Rackoff. Simple permutations mix well. In Josep Díaz, Juhani Karhumäki, Arto Lepistö, and Donald Sannella, editors, *ICALP 2004*, volume 3142 of *LNCS*, pages 770–781. Springer, Berlin, Heidelberg, July 2004.

[JKKZ21] Ruta Jawale, Yael Tauman Kalai, Dakshita Khurana, and Rachel Yun Zhang. SNARGs for bounded depth computations and PPAD hardness from sub-exponential LWE. In Samir Khuller and Virginia Vassilevska Williams, editors, *53rd ACM STOC*, pages 708–721. ACM Press, June 2021.

[KNR05] Eyal Kaplan, Moni Naor, and Omer Reingold. Derandomized constructions of k-wise (almost) independent permutations. In Chandra Chekuri, Klaus Jansen, José D. P. Rolim, and Luca Trevisan, editors, *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques*, pages 354–365. Springer Berlin Heidelberg, 2005.

[KR08] Yael Tauman Kalai and Ran Raz. Interactive PCP. In Luca Aceto, Ivan Damgård, Leslie Ann Goldberg, Magnús M. Halldórsson, Anna Ingólfsdóttir, and Igor Walukiewicz, editors, *ICALP 2008, Part II*, volume 5126 of *LNCS*, pages 536–547. Springer, Berlin, Heidelberg, July 2008.

[Lev73] Leonid A Levin. Universal sequential search problems. *Problems of information transmission*, 9(3):265–266, 1973.

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

[RR20] Guy N. Rothblum and Ron D. Rothblum. Batch verification and proofs of proximity with polylog overhead. In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020, Part II*, volume 12551 of *LNCS*, pages 108–138. Springer, Cham, November 2020.

[RRR16] Omer Reingold, Guy N. Rothblum, and Ron D. Rothblum. Constant-round interactive proofs for delegating computation. In Daniel Wichs and Yishay Mansour, editors, *48th ACM STOC*, pages 49–62. ACM Press, June 2016.

[RRR18] Omer Reingold, Guy N. Rothblum, and Ron D. Rothblum. Efficient batch verification for UP. In *Proceedings of the 33rd Computational Complexity Conference*, CCC '18. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2018.

[RVW13] Guy N. Rothblum, Salil P. Vadhan, and Avi Wigderson. Interactive proofs of proximity: delegating computation in sublinear time. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pages 793–802. ACM Press, June 2013.

[Sch80] Jacob T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *Journal of the ACM*, 27(4):701–717, 1980.

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

[Vad00] Salil P. Vadhan. On transformation of interactive proofs that preserve the prover's complexity. In *32nd ACM STOC*, pages 200–207. ACM Press, May 2000.

[vzGG13] Joachim von zur Gathen and Juergen Gerhard. *Modern Computer Algebra*. Cambridge University Press, 3 edition, 2013.

[Zei24] Hadas Zeilberger. Khatam: Reducing the communication complexity of code-based SNARKs. Cryptology ePrint Archive, Report 2024/1843, 2024.

[Zip79] Richard Zippel. Probabilistic algorithms for sparse polynomials. In *EUROSAM '79: International Symposium on Symbolic and Algebraic Computation*, volume 72 of *Lecture Notes in Computer Science*, pages 216–226. Springer, 1979.

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il