Hard Properties with (Very) Short PCPPs and Their Applications

Omri Ben-Eliezer
Tel Aviv University, Israel
omrib@mail.tau.ac.il

Eldar Fischer
Technion – Israel Institute of Technology, Haifa, Israel
eldar@cs.technion.ac.il

Amit Levi
University of Waterloo, Canada
amit.levi@uwaterloo.ca

Ron D. Rothblum
Technion – Israel Institute of Technology, Haifa, Israel
rothblum@cs.technion.ac.il

Abstract

We show that there exist properties that are maximally hard for testing, while still admitting PCPPs with a proof size very close to linear. Specifically, for every fixed ℓ, we construct a property $\mathcal{P}^{(\ell)} \subseteq \{0,1\}^n$ satisfying the following: Any testing algorithm for $\mathcal{P}^{(\ell)}$ requires $\Omega(n)$ many queries, and yet $\mathcal{P}^{(\ell)}$ has a constant query PCPP whose proof size is $O(n \cdot \log^{(\ell)} n)$, where $\log^{(\ell)}$ denotes the ℓ times iterated log function (e.g., $\log^{(2)} n = \log \log n$). The best previously known upper bound on the PCPP proof size for a maximally hard to test property was $O(n \cdot \text{polylog } n)$.

As an immediate application, we obtain stronger separations between the standard testing model and both the tolerant testing model and the erasure-resilient testing model: for every fixed ℓ, we construct a property that has a constant-query tester, but requires $\Omega(n/\log^{(\ell)}(n))$ queries for every tolerant or erasure-resilient tester.

2012 ACM Subject Classification Theory of computation → Interactive proof systems

Keywords and phrases PCPP, Property testing, Tolerant testing, Erasure resilient testing, Randomized encoding, Coding theory

Digital Object Identifier 10.4230/LIPIcs.ITCS.2020.9

Related Version `https://arxiv.org/abs/1909.03255`

Funding *Amit Levi*: Supported by the David R. Cheriton Graduate Scholarship. Part of this work was done while the author was visiting the Technion.
Ron D. Rothblum: Supported in part by the Israeli Science Foundation (Grant No. 1262/18), a Milgrom family grant and the Technion Hiroshi Fujiwara cyber security research center and the Israel cyber directorate.

1 Introduction

Probabilistically checkable proofs (PCPs) are one of the landmark achievements in theoretical computer science. Loosely speaking, PCPs are proofs that can be verified by reading only a very small (i.e., constant) number of bits. Beyond the construction of highly efficient proof systems, PCPs have myriad applications, most notably within the field of hardness of approximation.

A closely related variant of PCPs, called *probabilistically checkable proofs of proximity* (PCPPs), was introduced independently by Ben-Sasson *et al.* [5] and Dinur and Reingold [13]. In the PCPP setting, a verifier is given oracle access to both an input x and a proof π. It should make a few (e.g., constant) number of queries to both oracles to ascertain whether $x \in \mathcal{L}$. Since the verifier can only read a few of the input bits, we only require that it rejects inputs that are *far* (in Hamming distance) from $\mathcal{L}$, no matter what proof π is provided. PCPPs are highly instrumental in the construction of standard PCPs. Indeed, using modern terminology, both the original algebraic construction of PCPs [1] (see also [5]) as well as Dinur's [12] combinatorial proof utilize PCPPs.

By combining the seminal works of Ben-Sasson and Sudan [7] and Dinur [12], one can obtain PCPs and PCPPs with only poly-logarithmic (multiplicative) overhead. More specifically, the usual benchmark for PCPPs is with respect to the `CircuitEval` problem, in which the verifier is given explicit access to a circuit C and oracle access to both an input x and a proof π, and needs to verify that x is close to the set $\{x' : C(x') = 1\}$. The works of [7, 12] yield a PCPP whose length is quasilinear in the size $|C|$ of the circuit C.[1]

Given the important connections both to constructions of efficient proof-systems, and to hardness of approximation, a central question in the area is whether this result can be improved: Do PCPPs with only a *constant* overhead exist? In a recent work, Ben Sasson *et al.* [6] construct PCPs with constant overhead, albeit with very large query complexity (as well as a non-uniform verification procedure).[2] To verify that $C(x) = 1$ the verifier needs to make $|C|^{\delta}$ queries, where $\delta > 0$ can be any fixed constant.

Given the lack of success (despite the significant interest) in constructing constant-query PCPPs with constant overhead, it may be the case that there exist languages that do not have such efficient PCPPs. A natural class of candidate languages for which such PCPPs may not exist are languages for which it is *maximally* hard to test whether $x \in \mathcal{L}$ or is far from such, *without* a PCPP proof. In other words, languages (or rather properties) that do not admit sub-linear query testers. Thus, we investigate the following question:

> *Supposing that $\mathcal{L}$ requires $\Omega(n)$ queries for every (property) tester, must any constant-query* PCPP *for $\mathcal{L}$ have proof length $n \cdot (\log n)^{\Omega(1)}$?*

1.1 Our Results

Our first main result answers the above question negatively, by constructing a property that is maximally hard for testing, while admitting a very short PCPP. For the exact theorem statement, we let $\log^{(\ell)}$ denote the ℓ times iterated log function. That is, $\log^{(\ell)}(n) = \log(\log^{(\ell-1)}(n))$ for $\ell \geq 1$ and $\log^{(0)} n = n$.

▶ **Theorem 1** (informal restatement of Theorem 30). *For every constant integer $\ell \in \mathbb{N}$, there exists a property $\mathcal{P} \subseteq \{0,1\}^n$ such that any testing algorithm for $\mathcal{P}$ requires $\Omega(n)$ many queries, while $\mathcal{P}$ admits a (constant query[3]) PCPP system with proof length $O(n \cdot \log^{(\ell)}(n))$.*

---

[1] Note that a PCPP for `CircuitEval` can be easily used to construct a PCP for `CircuitSAT` with similar overhead (see [5, Proposition 2.4]).

[2] Although it is not stated in [6], we believe that their techniques can also yield PCPPs with similar parameters.

[3] For detection radius (or proximity parameter) $\varepsilon > 0$ and constant soundness, the particular query complexity of the PCPP system is bounded by $(2^{\ell}/\varepsilon)^{O(\ell)}$.

We remark that all such maximally hard properties cannot have constant-query PCPP proof-systems with a *sub-linear* length proof string (see Proposition 13), leaving only a small gap of $\log^{(\ell)}(n)$ on the proof length in Theorem 1.

Beyond demonstrating that PCPPs with extremely short proofs exist for some hard properties, we use Theorem 1 to derive several applications. We proceed to describe these applications next.

Tolerant Testing

Recall that property testing (very much like PCPPs) deals with solving approximate decision problems. A tester for a property $\mathcal{P}$ is an algorithm that given a sublinear number of queries to its input x, should accept (with high probability) if $x \in \mathcal{P}$ and reject if x is *far* from $\mathcal{P}$ (where, unlike PCPPs, the tester is not provided with any proof).

The standard setting of property testing is arguably fragile, since the testing algorithm is only guaranteed to accept all functions that exactly satisfy the property. In various settings and applications, accepting only inputs that exactly have a certain property is too restrictive, and it is more beneficial to distinguish between inputs that are close to having the property, and those that are far from it. To address this question, Parnas, Ron and Rubinfeld [26] introduced a natural generalization of property testing, in which the algorithm is required to accept functions that are close to the property. Namely, for parameters $0 \leq \varepsilon_0 < \varepsilon_1 \leq 1$, an $(\varepsilon_0, \varepsilon_1)$-*tolerant testing algorithm* is given an oracle access to the input, and is required to determine (with high probability) whether a given input is ε_0-close to the property or whether it is ε_1-far from it. As observed in [26], any standard testing algorithm whose queries are uniformly (but not necessarily independently) distributed, is inherently tolerant to some extent. Nevertheless, for many problems, strengthening the tolerance requires applying advanced methods and devising new algorithms (see e.g., [16, 23, 10, 8, 9]).

It is natural to ask whether tolerant testing is strictly harder than standard testing. This question was explicitly studied by Fischer and Fortnow [15], who used PCPPs with polynomial size proofs to show that there exists a property $\mathcal{P} \subseteq \{0,1\}^n$ that admits a tester with constant query complexity, but such that every tolerant tester for $\mathcal{P}$ has query complexity $\Omega(n^c)$ for some $0 < c < 1$. Using modern quasilinear PCPPs [7, 12] in combination with the techniques of [15] it is possible to construct a property demonstrating a better separation, of constant query complexity for standard testing versus $\Omega(n/\text{polylog}\, n)$ for tolerant testing.

Using Theorem 1 we can obtain an improved separation between testing and tolerant testing:

▶ **Theorem 2** (informal restatement of Theorem 46). *For any constant integer $\ell \in \mathbb{N}$, there exist a property of boolean strings $\mathcal{P} \subseteq \{0,1\}^n$ and a constant $\varepsilon_1 \in (0,1)$ such that $\mathcal{P}$ is ε-testable for any $\varepsilon > 0$ with a number of queries[4] independent of n, but for any $\varepsilon_0 \in (0, \varepsilon_1)$, every $(\varepsilon_0, \varepsilon_1)$-tolerant tester for $\mathcal{P}$ requires $\Omega(n/\text{polylog}^{(\ell)} n)$ many queries.*

Erasure-Resilient Testing

Another variant of the property testing model is the *erasure-resilient testing* model. This model was defined by Dixit et. al. [14] to address cases where data cannot be accessed at some domain points due to privacy concerns, or when some of the values were adversarially erased. More precisely, an α-erasure-resilient ε-tester gets as input parameters $\alpha, \varepsilon \in (0,1)$,

[4] The query complexity of the intolerant ε-tester has the same asymptotic bound as in Theorem 1.

as well as oracle access to a function f, such that at most an α fraction of its values have been erased. The tester has to accept with high probability if there is a way to assign values to the erased points of f such that the resulting function satisfies the desired property. The tester has to reject with high probability if for every assignment of values to the erased points, the resulting function is still ε-far from the desired property.

Similarly to the tolerant testing scenario, PCPPs were also used in [14] to show that there exists a property of boolean strings of length n that has a tester with query complexity independent of n, but for any constant $\alpha > 0$, every α-erasure-resilient tester is required to query $\Omega(n^c)$ many bits for some $c > 0$, thereby establishing a separation between the models. Later, in [27] PCPP constructions were used to provide a separation between the erasure-resilient testing model and the tolerant testing model.

Similarly to the tolerant testing case, we use Theorem 1 to prove a stronger separation between the erasure-resilient testing model and the standard testing model.

▶ **Theorem 3** (informal restatement of Theorem 47). *For any constant integer $\ell \in \mathbb{N}$, there exist a property of boolean strings $\mathcal{P} \subseteq \{0,1\}^n$ and a constant $\varepsilon_1 \in (0,1)$ such that $\mathcal{P}$ is ε-testable for any $\varepsilon > 0$ with number of queries[5] independent of n, but for any $\alpha = \Omega(1/\log^{(\ell)} n)$ and $\varepsilon \in (0, \varepsilon_1)$ such that $\varepsilon < 1 - \alpha$, any α-erasure-resilient ε-tester is required to query $\Omega(n/\text{polylog}^{(\ell)} n)$ many bits.*

Secret Sharing applications

As an additional application of our techniques we also obtain a new type of *secret sharing scheme*. Recall that in a secret sharing scheme, a secret value $b \in \{0,1\}$ is shared between n parties in such a way that only an authorized subset of the users can recover the secret. We construct a secret sharing scheme in which no subset of $o(n)$ parties can recover the secret and yet it is possible for each one of the parties to recover the secret, if given access to a PCPP-like proof, with the guarantee that no matter what proof-string is given, most parties will either recover b or reject.

We obtain such a secret sharing scheme through a notion called *Probabilistically Checkable Unveiling of a Shared Secret (*PCUSS*)*, which will be central in our work. This notion is loosely described in Subsection 1.2 and formally defined in Section 4.

1.2 Techniques

Central to our construction are (univariate) polynomials over a finite field $\mathbb{F}$. A basic fact is that a random polynomial $p : \mathbb{F} \to \mathbb{F}$ of degree (say) $|\mathbb{F}|/2$, evaluated at any set of at most $|\mathbb{F}|/2$ points, looks exactly the same as a totally random function $f : \mathbb{F} \to \mathbb{F}$. This is despite the fact that a random function is very far (in Hamming distance) from the set of low degree polynomials. Indeed, this is the basic fact utilized by Shamir's secret sharing scheme [29].

Thus, the property of being a degree-$|\mathbb{F}|/2$ univariate polynomial is a hard problem to decide for any tester, in the sense that such a tester must make $\Omega(|\mathbb{F}|)$ queries to the truth table of the function in order to decide. Given that, it seems natural to start with this property in order to prove Theorem 1. Here we run into two difficulties. First, the property of being a low degree polynomial is defined over a large alphabet, whereas we seek a property over boolean strings. Second, the best known PCPPs for this property have quasi-linear length [7], which falls short of our goal.

[5] Again, the query complexity of the (non erasure resilient) ε-tester has the same asymptotic bound as in Theorem 1.

To cope with these difficulties, our approach is to use composition, or more accurately, an iterated construction. The main technical contribution of this paper lies in the mechanism enabling this iteration. More specifically, rather than having the property contain the explicit truth table of the low degree polynomial p, we would like to use a more redundant representation for encoding each value $p(\alpha)$. This encoding should have several properties:

- It must be the case that one needs to read (almost) the entire encoding to be able to decode $p(\alpha)$. This feature of the encoding, which we view as a secret-sharing type of property, lets us obtain a hard to test property over boolean strings.
- The encoding need not be efficient, and in fact it will be made long enough to eventually subsume the typical length of a PCPP proof-string for the low degree property, when calculated with respect to an *unencoded* input string.
- Last but not least, we need the value to be decodable using very few queries, when given access to an auxiliary PCP-like proof string. This would allow us to "propagate" the PCPP verification of the property across iterations.

In more detail, we would like to devise a (randomized) encoding of strings in $\{0,1\}^k$ by strings in $\{0,1\}^m$. The third requirement listed above can be interpreted as saying that given oracle access to $v \in \{0,1\}^m$ and explicit access to a value $w \in \{0,1\}^k$, it will be possible verify that v indeed encodes w using a PCPP-like scheme, i.e. by providing a proof that can be verified with a constant number of queries. We refer to this property as a *probabilistically checkable unveiling (*PCU*)*[6]. Note that in our setting a single value w may (and usually will) have more than one valid encoding.

Going back to the first requirement of the encoding, we demand that without a proof, one must query at least $\Theta(m)$ bits of v to obtain *any* information about the encoded w, or even discern that v is indeed a valid encoding of some value. Given this combination of requirements, we refer to the verification procedure as a *Probabilistically Checkable Unveiling of a Shared Secret (*PCUSS*)*.

Low degree polynomials can be used to obtain a PCUSS based on Shamir's secret sharing scheme. More specifically, to encode a k bit string w, we take a random polynomial whose values on a subset $H \subseteq \mathbb{F}$ are exactly equal to the bits of w. However, we provide the values of this polynomial only over the sub domain $\mathbb{F} \setminus H$. Then, the encoded value is represented by the (interpolated) values of g over H, which admit a PCU scheme. On the other hand, the "large independence" feature of polynomials makes the encoded value indiscernible without a a supplied proof string, unless too many of the values of g over $\mathbb{F} \setminus H$ are read, thus allowing for a PCUSS.

This construction can now be improved via iteration. Rather than explicitly providing the values of the polynomial, they will be provided by a PCUSS scheme. Note that the PCUSS scheme that we now need is for strings of a (roughly) exponentially smaller size. The high level idea is to iterate this construction ℓ times to obtain the ℓ iterated log function in our theorems.

At the end of the recursion, i.e., for the smallest blocks at the bottom, we utilize a linear-code featuring both high distance and high dual distance, for a polynomial size PCUSS of the encoded value. This is the only "non-constructive" part in our construction, but since the relevant block size will eventually be less than $\log\log(n)$, the constructed property will still be uniform with polynomial calculation time (the exponential time in $\text{poly}(\log\log(n))$, needed to construct the linear-code matrix, becomes negligible).

[6] In fact, we will use a stronger variant where the access to w is also restricted.

Our PCUSS in particular provides a property that is hard to test (due to its shared secret feature), and yet has a near-linear PCPP through its unveiling, thereby establishing Theorem 1. We utilize this property for separation results in a similar manner to [15] and [14], by considering a weighted version of a "PCPP with proof" property, where the proof part holds only a small portion of the total weight. The PCPP proof part enables a constant query test, whereas if the PCPP proof is deleted, efficient testing is no longer possible.

1.3 Related work

Short PCPPs

For properties which can be verified using a circuit of size n, [5] gave PCPP constructions with proof length $n \cdot \exp(\text{poly}(\log\log n))$ and with a query complexity of $\text{poly}(\log\log n)$, as well as slightly longer proofs with constant query complexity. Later, Ben-Sasson and Sudan [7] gave constructions with quasilinear size proofs, but with slightly higher query complexity. The state of the art construction is due to Dinur [12] who, building on [7], showed a PCPP construction with proof length that is quasilinear in the circuit size and with constant query complexity. In a recent work Ben Sasson *et al.* [3] constructed an *interactive* version of PCPPs [4, 28] of strictly linear length and constant query complexity.

Tolerant Testing

The tolerant testing framework has received significant attention in the past decade. Property testing of dense graphs, initiated by [19], is inherently tolerant by the canonical tests of Goldreich and Trevisan [21]. Later, Fischer and Newman [16] (see also [2]) showed that every testable (dense) graph property admits a tolerant testing algorithm for *every* $0 < \varepsilon_0 < \varepsilon_1 < 1$, which implies that $O(1)$ query complexity testability is equivalent to distance approximation in the dense graph model. Some properties of boolean functions were also studied recently in the tolerant testing setting. In particular, the properties of being a *k-junta* (i.e. a function that depends on k variables) and being *unate* (i.e., a function where each direction is either monotone increasing or monotone decreasing) [9, 24, 11].

Erasure-resilient Testing

For the erasure resilient model, in addition to the separation between that model and the standard testing model, [14] designed efficient erasure-resilient testers for important properties, such as monotonicity and convexity. Shortly after, in [27] a separation between the erasure-resilient testing model and the tolerant testing model was established. The last separation requires an additional construction (outside PCPPs), which remains an obstacle to obtaining better than polynomial separations.

2 Preliminaries

We start with some notation and central definitions. For a set A, we let 2^A denote the power-set of A. For two strings $x, y \in \{0,1\}^*$ we use $x \sqcup y$ to denote string concatenation.

For an integer k, a field $\mathbb{F} = \text{GF}(2^k)$ and $\alpha \in \mathbb{F}$, we let $\langle\langle \alpha \rangle\rangle \in \{0,1\}^k$ denote the binary representation of α in some canonical way.

For two sets of strings A and B we use $A \sqcup B$ to denote the set $\{a \sqcup b \mid a \in A,\ b \in B\}$. For a collection of sets $\{A(d)\}_{d \in D}$ we use $\bigsqcup_{d \in D} A(d)$ to denote the set of all possible concatenations $\bigsqcup_{d \in D} a_d$, where $a_d \in A(d)$ for every $d \in D$.

Throughout this paper we use boldface letters to denote random variables, and assume a fixed canonical ordering over the elements in all the sets we define. For a set D, we write $\boldsymbol{v} \sim D$ to denote a random variable resulting from a uniformly random choice of an element $v \in D$.

2.1 Error correcting codes and polynomials over finite fields

The relative Hamming distance of two strings $x, y \in \Sigma^n$ is defined as $\mathrm{dist}(x,y) = \frac{1}{n} \cdot |\{i \in [n] \mid x_i \neq y_i\}|$. For a string $x \in \Sigma^n$ and a non-empty set $S \subseteq \Sigma^n$, we define $\mathrm{dist}(x,S) = \min_{y \in S} \mathrm{dist}(x,y)$. The following plays a central role in many complexity-related works, including ours.

▶ **Definition 4.** *A* code *is an injective function $C : \Sigma^k \to \Sigma^n$. If Σ is a finite field and C is a linear function (over Σ), then we say that C is a* linear code. *The* rate *of C is defined as k/n, whereas the* minimum relative distance *is defined as the minimum over all distinct $x, y \in \Sigma^k$ of* $\mathrm{dist}(C(x), C(y))$.

*An ε-*distance code *is a code whose minimum relative distance is at least ε. When for a fixed $\varepsilon > 0$ we have a family of ε-distance codes (for different values of k), we refer to its members as* error correcting codes.

In this work we use the fact that efficient codes with constant rate and constant relative distance exist. Moreover, there exist such codes in which membership can be decided by a quasi-linear size Boolean circuit.

▶ **Theorem 5** (see e.g., [30]). *There exists a linear code $\mathsf{Spiel} : \{0,1\}^k \to \{0,1\}^{100k}$ with constant relative distance, for which membership can be decided by a $k \cdot \mathrm{polylog}\, k$ size Boolean circuit.*

Actually, the rate of the code in [30] is significantly better, but since we do not try to optimize constants, we use the constant 100 solely for readability. In addition, the code described in [30] is *linear time decodeable*, but we do not make use of this feature throughout this work.

We slightly abuse notation, and for a finite field $\mathbb{F}$ of size 2^k, view the encoding given in Theorem 5 as $\mathsf{Spiel} : \mathbb{F} \to \{0,1\}^{100k}$, by associating $\{0,1\}^k$ with $\mathbb{F}$ in the natural way. Note that for $f : \mathbb{F} \to \mathbb{F}$, it holds that $\langle\!\langle f(\beta) \rangle\!\rangle \in \{0,1\}^k$ for every $\beta \in \mathbb{F}$, and therefore $\mathsf{Spiel}(f(\beta)) \in \{0,1\}^{100k}$. We slightly abuse notation, and for a function $f : \mathbb{F} \to \mathbb{F}$ we write $\mathsf{Spiel}(f)$ to denote the length $100k \cdot 2^k$ bit string $\bigsqcup_{\beta \in \mathbb{F}} \mathsf{Spiel}(f(\beta))$ (where we use the canonical ordering over $\mathbb{F}$).

▶ **Definition 6.** *Let $\mathcal{C}_{\mathbb{F}}$ denote the set of polynomials $g : \mathbb{F} \to \mathbb{F}$ such that $\deg(g) \leq \frac{|\mathbb{F}|}{2}$.*

The following lemma of [22], providing a fast univariate interpolation, will be an important tool in this work.

▶ **Lemma 7** ([22]). *Given a set of pairs $\{(x_1, y_1), \ldots, (x_r, y_r)\}$ with all x_i distinct, we can output the coefficients of $p(x) \in \mathbb{F}[X]$ of degree at most $r-1$ satisfying $p(x_i) = y_i$ for all $i \in [r]$, in $O(r \cdot \log^3(r))$ additions and multiplications in $\mathbb{F}$.*

The next lemma states that a randomly chosen function $\boldsymbol{\lambda} : \mathbb{F} \to \mathbb{F}$ is far from any low degree polynomial with very high probability.

▶ **Lemma 8.** *With probability at least $1 - o(1)$, a uniformly random function $\boldsymbol{\lambda} : \mathbb{F} \to \mathbb{F}$ is $1/3$-far from $\mathcal{C}_{\mathbb{F}}$.*

Proof. Consider the size of a ball of relative radius $1/3$ around some function $\lambda : \mathbb{F} \to \mathbb{F}$ in the space of functions from $\mathbb{F}$ to itself. The number of points (i.e., functions from $\mathbb{F} \to \mathbb{F}$) contained in this ball is at most

$$\binom{|\mathbb{F}|}{|\mathbb{F}|/3} \cdot |\mathbb{F}|^{|\mathbb{F}|/3} \leq (3e|\mathbb{F}|)^{|\mathbb{F}|/3}.$$

By the fact that the size of $\mathcal{C}_{\mathbb{F}}$ is $|\mathbb{F}|^{|\mathbb{F}|/2+1}$, the size of the set of points that are at relative distance at most $1/3$ from any point in $\mathcal{C}_{\mathbb{F}}$ is at most

$$|\mathbb{F}|^{|\mathbb{F}|/2+1} \cdot (3e|\mathbb{F}|)^{|\mathbb{F}|/3} = o(|\mathbb{F}|^{|\mathbb{F}|}).$$

The lemma follows by observing that there are $|\mathbb{F}|^{|\mathbb{F}|}$ functions from $\mathbb{F}$ to itself. ◀

2.1.1 Dual distance of linear codes

We focus here specifically on a linear code $C : \mathbb{F}^k \to \mathbb{F}^n$, and consider the linear subspace of its image, $V_C = \{C(x) : x \in \mathbb{F}^k\} \subseteq \mathbb{F}^n$. We define the *distance* of a linear space as $\text{dist}(V) = \min_{v \in V \setminus \{0^n\}} \text{dist}(v, 0^n)$, and note that in the case of V being the image V_C of a code C, this is identical to $\text{dist}(C)$. For a linear code, it helps to investigate also *dual distances*.

▶ **Definition 9.** *Given two vectors $u, v \in \mathbb{F}^n$, we define their* scalar product *as $u \cdot v = \sum_{i \in [n]} u_i v_i$, where multiplication and addition are calculated in the field $\mathbb{F}$. Given a linear space $V \subseteq \mathbb{F}^n$, its* dual space *is the linear space $V^\perp = \{u : \forall v \in V, u \cdot v = 0\}$. In other words, it is the space of vectors who are orthogonal to all members of V. The* dual distance *of the space V is simply defined as $\text{dist}(V^\perp)$.*

For a code C, we define its *dual distance*, $\text{dist}^\perp(C)$, as the dual distance of its image V_C. We call C an *η-dual-distance* code if $\text{dist}^\perp(C) \geq \eta$. The following well-known lemma is essential to us, as it will relate to the "secret-sharing" property that we define later.

▶ **Lemma 10** (See e.g., [25, Chapter 1, Theorem 10])**.** *Suppose that $C : \mathbb{F}^k \to \mathbb{F}^n$ is a linear η-dual distance code, let $Q \subset [n]$ be any set of size less than $\eta \cdot n$, and consider the following random process for picking a function $\boldsymbol{u} : Q \to \mathbb{F}$: Let $\boldsymbol{w} \in \mathbb{F}^k$ be drawn uniformly at random, and set $\boldsymbol{u}$ be the restriction of $C(\boldsymbol{w})$ to the set Q. Then, the distribution of $\boldsymbol{u}$ is identical to the uniform distribution over the set of all functions from Q to $\mathbb{F}$.*

2.2 Probabilistically checkable proofs of proximity (PCPP)

As described briefly in the introduction, a PCPP verifier for a property $\mathcal{P}$ is given access to an input x and a proof π, as well as a *detection radius* $\varepsilon > 0$ and *soundness parameter* $\delta > 0$. The verifier should make a constant number of queries (depending only on ε, δ) to the input x and the proof π, and satisfy the following. If $x \in \mathcal{P}$, then there exists π for which the verifier should always accept x. If $\text{dist}(x, \mathcal{P}) > \varepsilon$, the verifier should reject x with probability at least δ, regardless of the contents of π. More formally, we define the following.

▶ **Definition 11** (PCPP)**.** *For $n \in \mathbb{N}$, let $\mathcal{P} \subset \{0,1\}^n$ be a property of n-bit Boolean strings, and let $t \in \mathbb{N}$. We say that $\mathcal{P}$ has a $q(\varepsilon, \delta)$-*query, length-t Probabilistically Checkable Proof of Proximity (PCPP) system *if the following holds: There exists a verification algorithm V that takes as input $\varepsilon, \delta > 0$ and $n \in \mathbb{N}$, makes a total of $q(\varepsilon, \delta)$ queries on strings $w \in \{0,1\}^n$ and $\pi \in \{0,1\}^t$, and satisfies the following:*

1. *(Completeness) If $w \in \mathcal{P}$, then there exists a proof $\pi = \boldsymbol{Proof}_{\mathcal{P}}(w) \in \{0,1\}^t$ such that for every $\varepsilon, \delta > 0$, the verifier V accepts with probability 1.*
2. *(Soundness) If $\mathrm{dist}(w, \mathcal{P}) > \varepsilon$, then for every alleged proof $\pi \in \{0,1\}^t$, the verifier V rejects with probability greater than δ.*

Note that soundness is easy to amplify: Given a PCPP as above with parameters ε, δ, t and query complexity $q(\varepsilon, \delta)$, one can increase the soundness parameter to $1 - \tau$ (for any $\tau > 0$) by simply running $\Theta(\log(1/\tau)/\delta)$ independent instances of the verification algorithm V, and rejecting if at least one of them rejected; the query complexity then becomes $\Theta\left(q(\varepsilon, \delta) \cdot \log(1/\tau)/\delta\right)$, while the parameters ε and t remain unchanged.

The following lemma, establishing the existence of a quasilinear PCPP for any property $\mathcal{P}$ that is verifiable in quasilinear time, will be an important tool throughout this work.

▶ **Lemma 12** (Corollary 8.4 in [12], see also [20]). *Let $\mathcal{P}$ be a property of Boolean strings which is verifiable by a size t Boolean circuit. Then, there exists a length-t' PCPP system $\mathcal{P}$ with parameters $\varepsilon, \delta > 0$, that makes at most $q(\varepsilon, \delta)$ queries, where $t' = t \cdot \mathrm{polylog}\, t$.*

Specifically, $q(\varepsilon, \delta) = O(\varepsilon^{-1})$ suffices for any $\delta < 0.99$.

As described briefly in the introduction, maximally hard properties cannot have a constant query PCPP proof systems with a sublinear length proof string.

▶ **Proposition 13.** *Let $\mathcal{P} \subseteq \{0,1\}^n$ and $\varepsilon > 0$ be such that any ε-tester for $\mathcal{P}$ has to make $\Omega(n)$ many queries. Then, any constant query PCPP system for $\mathcal{P}$ (where e.g. $\delta = 1/3$) must have proof length of size $\Omega(n)$.*

Proof. Suppose that there exists a PCPP for $\mathcal{P}$ with $O(1)$ queries and proof length $t = o(n)$. Since the PCPP verifier has constant query complexity, we may assume that it is non adaptive and uses $q = O(1)$ queries. By an amplification argument as above, we can construct an amplified verifier that makes $O(q \cdot t) = o(n)$ queries, with soundness parameter $1 - 2^{-t}/3$. By the fact that the verifier is non-adaptive, it has the same query distribution regardless of the proof string. Therefore, we can run 2^t amplified verifiers in parallel while reusing queries, one verifier for each of the 2^t possible proof strings. If any of the 2^t amplified verifiers accept, we accept the input. If the input belongs to $\mathcal{P}$, one of the above 2^t verifiers will accept (the one that used the correct proof). If the input was ε-far from $\mathcal{P}$, then by a union bound, the probability that there was any accepting amplified verifier is at most $1/3$. This yields an $o(n)$ tester for $\mathcal{P}$, which contradicts our assumption. ◀

2.3 Testing, tolerant testing and erasure-resilient testing

In this subsection we define notions related to the property testing framework. We also formally define a few variants of the original testing model that will be addressed in this work. A *property* $\mathcal{P}$ of n-bit boolean strings is a subset of all those strings, and we say that a string x *has* the property $\mathcal{P}$ if $x \in \mathcal{P}$.

Given $\varepsilon \geq 0$ and a property $\mathcal{P}$, we say that a string $x \in \{0,1\}^n$ is *ε-far* from $\mathcal{P}$ if $\mathrm{dist}(x, \mathcal{P}) > \varepsilon$, and otherwise it is *$\varepsilon$-close* to $\mathcal{P}$. We next define the notion of a *tolerant* tester of which standard (i.e. intolerant) testers are a special case.

▶ **Definition 14** (Intolerant and tolerant testing). *Given $0 \leq \varepsilon_0 < \varepsilon_1 \leq 1$, a q-query $(\varepsilon_0, \varepsilon_1)$-*testing *algorithm T for a property $\mathcal{P} \subseteq \{0,1\}^n$ is a probabilistic algorithm (possibly adaptive) making q queries to an input $x \in \{0,1\}^n$ that outputs a binary verdict satisfying the following two conditions.*

1. *If* $\text{dist}(x, \mathcal{P}) \leq \varepsilon_0$*, then* T *accepts* x *with probability at least* $2/3$.
2. *If* $\text{dist}(x, \mathcal{P}) > \varepsilon_1$*, then* T *rejects* x *with probability at least* $2/3$.

When $\varepsilon_0 = 0$*, we say that* T *is an* ε_1-testing algorithm *for* $\mathcal{P}$*, and otherwise we say that* T *is an* $(\varepsilon_0, \varepsilon_1)$-tolerant testing algorithm *for* $\mathcal{P}$.

Next, we define the erasure-resilient testing model. We start with some terminology. A string $x \in \{0,1,\perp\}^n$ is α-erased if x_i is equal to $\perp$ on at most αn coordinates. A string $x' \in \{0,1\}^n$ that differs from x only on coordinates $i \in [n]$ for which $x_i = \perp$ is called a *completion* of x. The (pseudo-)distance $\text{dist}(x, \mathcal{P})$ of an α-erased string x from a property $\mathcal{P}$ is the minimum, over every completion x' of x, of the relative Hamming distance of x' from $\mathcal{P}$. Note that for a string with no erasures, this is simply the Hamming distance of x from $\mathcal{P}$. As before, x is ε-far from $\mathcal{P}$ if $\text{dist}(x, \mathcal{P}) > \varepsilon$, and ε-close otherwise.

▶ **Definition 15** (Erasure-resilient tester)**.** *Let* $\alpha \in [0,1)$ *and* $\varepsilon \in (0,1)$ *be parameters satisfying* $\alpha + \varepsilon < 1$*. A* q*-query* α*-erasure-resilient* ε*-tester* T *for* $\mathcal{P}$ *is a probabilistic algorithm making* q *queries to an* α*-erased string* $x \in \{0,1,\perp\}^n$*, that outputs a binary verdict satisfying the following two conditions.*

1. *If* $\text{dist}(x, \mathcal{P}) = 0$ *(i.e., if there exists a completion* x' *of* x*, such that* $x' \in \mathcal{P}$*), then* T *accepts* x *with probability at least* $2/3$.
2. *If* $\text{dist}(x, \mathcal{P}) > \varepsilon$ *(i.e., if every completion of* x' *of* x *is* ε*-far from* $\mathcal{P}$*), then* T *rejects* x *with probability at least* $2/3$.

The next lemma will be useful to prove that some properties are hard to test. The lemma states that if we have two distributions whose restrictions to any set of queries of size at most q are identical, then no (possibly adaptive) algorithm making at most q queries can distinguish between them.

▶ **Definition 16** (Restriction)**.** *Given a distribution* $\mathcal{D}$ *over functions* $f : D \to \{0,1\}$ *and a subset* $Q \subseteq D$*, we define the* restriction $\mathcal{D}|_Q$ of $\mathcal{D}$ to Q *to be the distribution over functions* $g : Q \to \{0,1\}$*, that results from choosing a function* $f : D \to \{0,1\}$ *according to* $\mathcal{D}$*, and setting* g *to be* $f|_Q$*, the restriction of* f *to* Q.

▶ **Lemma 17** ([17], special case)**.** *Let* $\mathcal{D}_1$ *and* $\mathcal{D}_2$ *be two distributions of functions over some domain* D*. Suppose that for any set* $Q \subset D$ *of size at most* q*, the restricted distributions* $\mathcal{D}_1|_Q$ *and* $\mathcal{D}_2|_Q$ *are identically distributed. Then, any (possibly adaptive) algorithm making at most* q *queries cannot distinguish* $\mathcal{D}_1$ *from* $\mathcal{D}_2$ *with any positive probability.*

3 Code Ensembles

It will be necessary for us to think of a generalized definition of an encoding, in which each encoded value has multiple legal encodings.

▶ **Definition 18** (Code ensemble)**.** *A* code ensemble *is a function* $\mathcal{E} : \Sigma^k \to 2^{\Sigma^m}$*. Namely, every* $x \in \Sigma^k$ *has a set of its valid encodings from* Σ^m*. We define the distance of the code ensemble as*

$$\min_{x \neq x' \in \{0,1\}^k} \min_{(v,u) \in \mathcal{E}(x) \times \mathcal{E}(x')} \text{dist}(v, u).$$

It is useful to think of a code ensemble $\mathcal{E} : \Sigma^k \to 2^{\Sigma^m}$ as a *randomized mapping*, that given $x \in \Sigma^k$, outputs a uniformly random element from the set of encodings $\mathcal{E}(x)$. Using the above we can define a *shared secret* property. In particular, we use a strong information

theoretic definition of a shared secret, in which $o(m)$ bits do not give *any information at all* about the encoded value. Later on, we construct code ensembles with a shared secret property.

▶ **Definition 19** (Shared Secret). *For $m, k \in \mathbb{N}$ and a constant $\zeta > 0$, we say that a code ensemble $\mathcal{C} : \{0,1\}^k \to 2^{(\{0,1\}^m)}$ has a* ζ-shared secret *property if it satisfies the following. For any $Q \subseteq [m]$ of size $|Q| \leq \zeta m$, any $w, w' \in \{0,1\}^k$ such that $w \neq w'$, and any $t \in \{0,1\}^{|Q|}$ it holds that*

$$\underset{\boldsymbol{v} \sim \mathcal{C}(w)}{\boldsymbol{Pr}}[\boldsymbol{v}|_Q = t] = \underset{\boldsymbol{v}' \sim \mathcal{C}(w')}{\boldsymbol{Pr}}[\boldsymbol{v}'|_Q = t].$$

Namely, for any $w \neq w'$ and any $Q \subseteq [m]$ of size at most ζm, the distribution obtained by choosing a uniformly random member of $\mathcal{C}(w)$ and considering its restriction to Q, is identical to the distribution obtained by choosing a uniformly random member of $\mathcal{C}(w')$ and considering its restriction to Q.

3.1 A construction of a hard code ensemble

We describe a construction of a code ensemble for which a linear number of queries is necessary to verify membership or to decode the encoded value. This code will be our *base* code in the iterative construction. The existence of such a code ensemble is proved probabilistically, relying on the following simple lemma.

▶ **Lemma 20.** *Fix constant $\alpha, \beta > 0$ where $\beta \log(e/\beta) < \alpha$. Let $s, t \in \mathbb{N}$ so that $s \leq (1-\alpha)t$. Then, with probability $1 - o(1)$, a sequence of s uniformly random vectors $\{v_1, \ldots, v_s\}$ from $\{0,1\}^t$ is linearly independent, and corresponds to a β-distance linear code.*

Proof. The proof follows from a straightforward counting argument. If we draw s uniformly random vectors $v_1, \ldots, v_s \in \{0,1\}^t$, then each non-trivial linear combination of them is in itself a uniformly random vector from $\{0,1\}^t$, and hence has weight less than β with probability at most

$$2^{-t} \cdot \binom{t}{\beta t} \leq 2^{-t} \left(\frac{et}{\beta t}\right)^{\beta t} = 2^{-t} \cdot 2^{\beta \log(e/\beta) t} = 2^{(\gamma - 1)t},$$

where we set $\gamma = \beta \log(e/\beta) < \alpha$.

By a union bound over all $2^s \leq 2^{(1-\alpha)t}$ possible combinations, the probability that there exists a linear combination with weight less than β is at most $2^{(\gamma-\alpha)t} = o(1)$. If this is not the case, then $v_1, \ldots, v_s$ are linearly independent, and moreover, $\{v_1, \ldots, v_s\}$ corresponds to a β-distance linear code (where we use the fact that the distance of a linear code is equal to the minimal Hamming weight of a non-zero codeword). ◀

Our construction makes use of a sequence of vectors that correspond to a high-distance and high-dual distance code, as described below.

▶ **Definition 21** (Hard code ensemble $\mathcal{H}_k$). *Let $k \in \mathbb{N}$ and let $\{v_1, \ldots, v_{3k}\}$ be a sequence of vectors in $\{0,1\}^{4k}$ such that* $\mathrm{Span}\{v_1, \ldots, v_{3k}\}$ *is a* $1/30$*-distance code, and that* $\mathrm{Span}\{v_{k+1}, \ldots, v_{3k}\}$ *is a* $1/10$*-dual distance code. Let*

$$A = \begin{bmatrix} | & & | \\ v_1 & \cdots & v_{3k} \\ | & & | \end{bmatrix}.$$

We define the code ensemble $\mathcal{H}_k : \{0,1\}^k \to 2^{\{0,1\}^{4k}}$ *as*

$$\mathcal{H}_k(w) = \{Au \ : \ u \in \{0,1\}^{3k} \text{ where } u|_{\{1,\ldots,k\}} = w\},$$

where all operations are over GF(2)*.*

The next lemma states that a collection of random vectors $\{v_1, \ldots, v_{3k}\}$ in $\{0,1\}^{4k}$ satisfies the basic requirements of a code ensemble $\mathcal{H}_k$ with high probability (that is, with probability tending to one as $k \to \infty$), and hence such a code ensemble exists.

▶ **Lemma 22.** *A set* $\{v_1, \ldots, v_{3k}\}$ *of random vectors in* $\{0,1\}^{4k}$ *satisfies with high probability the following two conditions:* $\mathrm{Span}\{v_1, \ldots, v_{3k}\}$ *is a* 1/30*-distance code, and furthermore,* $\mathrm{Span}\{v_{k+1}, \ldots, v_{3k}\}$ *is a* 1/10*-dual distance code. In particular, for all* k *large enough the code ensemble* $\mathcal{H}_k$ *exists.*

Proof. We apply Lemma 20 multiple times. First, picking $t = 4k$, $s = 3k$, $\alpha = 1/4$, and $\beta = 1/30$, we conclude that $v_1, \ldots, v_{3k}$ with high probability correspond to a 1/30-distance code.

To show that with high probability the code spanned by the last $2k$ vectors has high dual distance, we compare the following two processes, whose output is a linear subspace of $(\mathrm{GF}(2))^{4k}$, that we view as a code: (i) Choose $2k$ vectors and return their span. (ii) Choose $4k - 2k = 2k$ vectors and return the dual of their span. Conditioning on the chosen $2k$ vectors being linearly independent, the output distributions of these two processes are identical. Indeed, by a symmetry argument it is not hard to see that under the conditioning, the linear subspace generated by Process (i) is uniformly distributed among all rank-$2k$ subspaces V of $(\mathrm{GF}(2))^{4k}$. Now, since we can uniquely couple each such V with its dual $V^\perp$ (also a rank-$2k$ subspace) and since $V = (V^\perp)^\perp$, this means that the output distribution of Process (ii) is uniform as well.

However, it follows again from Lemma 20 (with $t = 4k$, $s = 2k$, $\alpha = 1/2$, and any $\beta > 0$ satisfying the conditions of the lemma) that the chosen $2k$ vectors are independent with high probability. This means that (without the conditioning) the output distributions of Process (i) and Process (ii) are $o(1)$-close in variation distance. Applying Lemma 20 with $t = 4k$, $s = 2k$, $\alpha = 1/2$, and $\beta = 1/10$ we get that the distance of the code generated by Process (i) is at least $\beta = 1/10$ with high probability. However, the latter distance equals by definition to the dual distance of the code generated by Process (ii). By the closeness of the distributions, we conclude that the dual distance of Process (i) is also at least 1/10 with high probability. ◀

We next state a simple but important observation regarding membership verification.

▶ **Observation 23.** *Once a matrix* A *with the desired properties is constructed (which may take* $\exp(k^2)$ *time if we use brute force), given* $w \in \{0,1\}^k$*, the membership of* v *in* $\mathcal{H}_k(w)$ *can be verified in* $\mathrm{poly}(k)$ *time (by solving a system of linear equations over* GF(2)*).*

4 PCUs and PCUSSs

Next, we define the notion of Probabilistically Checkable Unveiling (PCU). This notion is similar to PCPP, but here instead of requiring our input to satisfy a given property, we require our input to encode a value $w \in \{0,1\}^k$ (typically using a large distance code ensemble). We then require that given the encoded value w, it will be possible to prove in a PCPP-like fashion that the input is indeed a valid encoding of w.

▶ **Definition 24** (PCU). *Fix $m, t, k \in \mathbb{N}$, and let $\mathcal{C} : \{0,1\}^k \to 2^{\{0,1\}^m}$ be a code ensemble. We say that $\mathcal{C}$ has a $q(\varepsilon,\delta)$-query, length-t* PCU *if the following holds. There exists a verification algorithm V that takes as inputs $\varepsilon, \delta > 0$, $m \in \mathbb{N}$, and $w \in \{0,1\}^k$, makes at most $q(\varepsilon,\delta)$ queries to the strings $v \in \{0,1\}^m$ and $\pi \in \{0,1\}^t$, and satisfies the following:*

1. *If $v \in \mathcal{C}(w)$, then there exists a proof $\pi = \boldsymbol{Proof}_{\mathcal{C}}(v) \in \{0,1\}^t$ such that for every $\varepsilon, \delta > 0$, the verifier V accepts with probability 1.*
2. *If $\mathrm{dist}(v, \mathcal{C}(w)) > \varepsilon$, then for every alleged proof $\pi \in \{0,1\}^t$, the verifier V rejects v with probability greater than δ.*

In order to facilitate the proof of the main theorem, we utilize a more stringent variant of the above PCU definition. Instead of supplying $w \in \{0,1\}^k$ to the algorithm, we supply oracle access to a a string $\tau \in \{0,1\}^{100k}$ that is supposed to represent $\mathsf{Spiel}(w)$, along with the proof π, and the algorithm only makes $q(\varepsilon,\delta)$ queries to the proof string π, the original encoding v *and* the string τ. For cases where $v \in \mathcal{C}(w)$, we use $\mathbf{Value}(v)$ to denote $\mathsf{Spiel}(w)$.

▶ **Definition 25** (Spiel-PCU). *Fix $m, t, k \in \mathbb{N}$, and let $\mathcal{C} : \{0,1\}^k \to 2^{\{0,1\}^m}$ be a code ensemble. We say that $\mathcal{C}$ has a $q(\varepsilon,\delta)$-query, length-t* Spiel-PCU *if the following holds. There exists a verification algorithm V that takes as inputs $\varepsilon, \delta > 0$, $m \in \mathbb{N}$, makes at most $q(\varepsilon,\delta)$ queries to the strings $v \in \{0,1\}^m$, $\tau \in \{0,1\}^{100k}$ and $\pi \in \{0,1\}^t$, and satisfies the following:*

1. *If there exists $w \in \{0,1\}^k$ for which $v \in \mathcal{C}(w)$ and $\tau = \boldsymbol{Value}(v) = \mathsf{Spiel}(w)$, then there exists a proof $\pi = \boldsymbol{Proof}_{\mathcal{C}}(v) \in \{0,1\}^t$ such that for every $\varepsilon, \delta > 0$, the verifier V accepts with probability 1.*
2. *If for every $w \in \{0,1\}^k$ either $\mathrm{dist}(\tau, \mathsf{Spiel}(w)) > \varepsilon$ or $\mathrm{dist}(v, \mathcal{C}(w)) > \varepsilon$, then for every alleged proof $\pi \in \{0,1\}^t$, the verifier V rejects v with probability greater than δ.*

Note that a code ensemble admitting a Spiel-PCU automatically admits a PCU. Indeed, given the string w, an oracle for $\mathsf{Spiel}(w)$ can be simulated.

The following lemma states the existence of Spiel-PCU for efficiently computable code ensembles, and will be used throughout this work. The proof follows from Lemma 12 together with a simple concatenation argument.

▶ **Lemma 26.** *Let $k, m, t \in \mathbb{N}$ be such that $t \geq m$, and let $\mathcal{C} : \{0,1\}^k \to 2^{\{0,1\}^m}$ be a code ensemble. If given $w \in \{0,1\}^k$ and $v \in \{0,1\}^m$, it is possible to verify membership of v in $\mathcal{C}(w)$ using a circuit of size t, then there is a $q(\varepsilon,\delta)$-query, length-t'* Spiel-PCU *for $\mathcal{C}$ where $t' = t \cdot \mathrm{polylog}\, t$.*

Proof. Assume without loss of generality that $m \geq |\mathsf{Spiel}(0^k)|$. Let $\xi = \left\lfloor \frac{m}{|\mathsf{Spiel}(0^k)|} \right\rfloor$ (note that $\xi \geq 1$), and define

$$\mathcal{C}_{eq} \stackrel{\text{def}}{=} \left\{ v \sqcup (\mathsf{Spiel}(w))^{\xi} \;\middle|\; \exists w \in \{0,1\}^k \text{ for which } v \in \mathcal{C}(w) \right\},$$

where $(\mathsf{Spiel}(w))^{\xi}$ denotes the ξ-times concatenation of $\mathsf{Spiel}(w)$.

For any string u it is possible to check, using a quasilinear size circuit (see [30]), that the substring that corresponds to the domain of $(\mathsf{Spiel}(w))^{\xi}$ is a ξ-times repetition of $\mathsf{Spiel}(w)$ for some w. After doing so, we decode w using a quasilinear size circuit (as in [30]), and then, by the premise of the lemma, we can verify membership in $\mathcal{C}(w)$ using a circuit of size t. Therefore, membership in $\mathcal{C}_{eq}$ can be decided using a $O(t)$ size boolean circuit, and therefore by Lemma 12 admits a PCPP system whose proof length is quasilinear in t.

Given an input v to Spiel-PCU, let $v' = v \sqcup (\mathsf{Spiel}(w))^{\xi}$ and use the PCPP system for $\mathcal{C}_{eq}$, with detection radius $\varepsilon/3$ and soundness parameter δ, where each query to v' is emulated by a corresponding query to v or $\mathsf{Spiel}(w)$. Note that if $v \in \mathcal{C}(w)$, then $v' \in \mathcal{C}_{eq}$, so the PCPP system for $\mathcal{C}_{eq}$ will accept with probability 1.

Next, suppose that $\mathrm{dist}(v, \mathcal{C}(w)) > \varepsilon$, and observe that this implies that v' is at least $\varepsilon/3$-far from $\mathcal{C}_{eq}$. Thus, by the soundness property of the PCPP for $\mathcal{C}_{eq}$, the verifier rejects with probability at least δ, regardless of the contents of the alleged proof π it is supplied with. ◀

Next we define Probabilistically Checkable Unveiling of a Shared Secret (PCUSS).

▶ **Definition 27.** *For $m, k, t \in \mathbb{N}$, we say that a function $\mathcal{C}\colon \{0,1\}^k \to 2^{(\{0,1\}^n)}$ has a $q(\varepsilon,\delta)$-query, length-t* PCUSS*, if $\mathcal{C}$ has a shared secret property, as well as $\mathcal{C}$ has a $q(\varepsilon,\delta)$-query, length-t* PCU*. Similarly, when $\mathcal{C}$ has a shared secret property (for constant ζ), as well as $\mathcal{C}$ has a $q(\varepsilon,\delta)$-query, length-t* Spiel-PCU*, we say that $\mathcal{C}$ has a $q(\varepsilon,\delta)$-query, length-t* Spiel-PCUSS*.*

Note that $\mathcal{C}$ admitting a Spiel-PCUSS directly implies that it admits a PCUSS with similar parameters.

The following lemma establishes the existence of a Spiel-PCUSS for $\mathcal{H}_k$, where $\mathcal{H}_k$ is the code ensemble from Definition 21.

▶ **Lemma 28.** *For any $k \in \mathbb{N}$, $\mathcal{H}_k$ has a $q(\varepsilon,\delta)$-query, length-t'* Spiel-PCUSS *where $t' = \mathrm{poly}(k)$.*

Proof. By Observation 23, given w, membership in $\mathcal{H}_k(w)$ can be checked in $\mathrm{poly}(k)$ time, which means that there exists a polynomial size circuit that decides membership in $\mathcal{H}_k(w)$. Combining the above with Lemma 26 implies a $q(\varepsilon,\delta)$-query, length-t' Spiel-PCU where $t' = \mathrm{poly}(k)$. By Lemma 10, the large dual distance property of $\mathcal{H}_k$ implies its shared secret property for some constant ζ, which concludes the proof of the lemma. ◀

5 PCUSS construction

In this section we give a construction of code ensembles that admit a PCUSS. First we show that our code ensemble has a PCU with a short proof. Specifically,

▶ **Lemma 29.** *For any fixed $\ell \in \mathbb{N}$ and any $k \in \mathbb{N}$, there exists $n_0(\ell, k)$ and a code ensemble $\mathcal{E}^{(\ell)} : \{0,1\}^k \to 2^{(\{0,1\}^n)}$, such that for all $n > n_0(\ell, k)$, the code ensemble $\mathcal{E}^{(\ell)}$ has a $q(\varepsilon,\delta)$-query length-t* PCU*, for $t = O(n \cdot \mathrm{polylog}^{(\ell)} n)$.*

Later, we prove that our code ensemble has a shared secret property, which implies that it has a PCUSS (which implies Theorem 1, as we shall show).

▶ **Theorem 30.** *For any fixed $\ell \in \mathbb{N}$ and any $k \in \mathbb{N}$, there exists $n_0(\ell, k)$ and a code ensemble $\mathcal{E}^{(\ell)} : \{0,1\}^k \to 2^{(\{0,1\}^n)}$, such that for all $n > n_0(\ell, k)$, the code ensemble $\mathcal{E}^{(\ell)}$ has a $q(\varepsilon,\delta)$-query length-t* PCUSS*, for $t = O(n \cdot \mathrm{polylog}^{(\ell)} n)$.*

Specifically, by the discussion before Lemma 37, for any fixed soundness parameter $0 < \delta < 1$ it suffices to take

$$q(\varepsilon,\delta) \leq \left(2^{\ell}/\varepsilon\right)^{O(\ell)},$$

and for the high soundness regime where $\delta = 1 - \tau$ (and $\tau > 0$ is small), it suffices to have

$$q(\varepsilon,\delta) \leq \left(2^{\ell}/\varepsilon\right)^{O(\ell)} \log(1/\tau).$$

5.1 The iterated construction

Our iterative construction uses polynomials over a binary finite field $\mathrm{GF}(2^t)$. In our proof we will need to be able to implement arithmetic operations over this field efficiently (i.e., in $\mathrm{poly}(t)$ time). This can be easily done given a suitable representation of the field: namely, a degree t irreducible polynomial over $\mathrm{GF}(2)$. It is unclear in general whether such a polynomial can be found in $\mathrm{poly}(t)$ time. Fortunately though, for $t = 2 \cdot 3^r$ where $r \in \mathbb{N}$, it is known that the polynomial $x^t + x^{t/2} + 1$ is irreducible over $\mathrm{GF}(2)$ (see, e.g., [18, Appendix G]). We will therefore restrict our attention to fields of this form. At first glance this seems to give us a property that is defined only on a sparse set of input lengths. However, towards the end of this section, we briefly describe how to bypass this restriction.

We next formally define our iterated construction, starting with the "level-0" construction as a base case. The constants c, d in the definition will be explicitly given in the proof of Lemma 36. Additionally, for any $\ell \in \mathbb{N}$, we shall pick a large enough constant c_ℓ that satisfies several requirements for the "level-ℓ" iteration of the construction.

▶ **Definition 31** (Iterated coding ensemble). *For $k \in \mathbb{N}$ and $w \in \{0,1\}^k$, we define the* base code ensemble of w *(i.e., level-ℓ code ensemble of w for $\ell = 0$) as*

$$\mathcal{E}_k^{(0)}(w) = \mathcal{H}_k(w).$$

Let $c, d \in \mathbb{N}$ be large enough global constants, fix $\ell > 0$, let c_ℓ be large enough, and let $\mathbb{F}$ be a finite field for which $|\mathbb{F}| \geq \max\{c_\ell, c \cdot k\}$.

We define the level-ℓ code ensemble of $w \in \{0,1\}^k$ over $\mathbb{F}$ *as follows. Let $r \in \mathbb{N}$ be the smallest integer such that $(\log|\mathbb{F}|)^d \leq 2^{2 \cdot 3^r}$, set $\mathbb{F}' = \mathrm{GF}\left(2^{2 \cdot 3^r}\right)$ and $k' = \log|\mathbb{F}|$. Note that these satisfy the recursive requirements of a level-$(\ell-1)$ code ensemble provided that c_ℓ is large enough (specifically we require $(\log|\mathbb{F}|)^{d-1} > c$, so that $|\mathbb{F}'| \geq ck'$). Finally, let $H \subseteq \mathbb{F}$ be such that $|H| = k$, and define*

$$\mathcal{E}_{\mathbb{F},k}^{(\ell)}(w) = \bigcup_{g \in \mathcal{C}_\mathbb{F}:\ g|_H = w} \bigsqcup_{\beta \in \mathbb{F} \setminus H} \mathcal{E}_{\mathbb{F}',k'}^{(\ell-1)}(\langle\!\langle g(\beta) \rangle\!\rangle).$$

(Note that for $\ell = 1$ we just use $\mathcal{E}_{\mathbb{F},k}^{(1)}(w) = \bigcup_{g \in \mathcal{C}_\mathbb{F}:\ g|_H = w} \bigsqcup_{\beta \in \mathbb{F} \setminus H} \mathcal{E}_{k'}^{(0)}(\langle\!\langle g(\beta) \rangle\!\rangle)$).

That is, $v \in \mathcal{E}_{\mathbb{F},k}^{(\ell)}(w)$ if there exists a polynomial $g \in \mathcal{C}_\mathbb{F}$ such that $v = \bigsqcup_{\beta \in \mathbb{F} \setminus H} v_\beta$, where $v_\beta \in \mathcal{E}_{\mathbb{F}',k'}^{(\ell-1)}(\langle\!\langle g(\beta) \rangle\!\rangle)$ for every $\beta \in \mathbb{F} \setminus H$ and $g|_H = w$ (where we identify the 0 and 1 elements of $\mathbb{F}$ with 0 and 1 bits respectively). When the context is clear, we sometimes omit the subscripts.

Our choice of the constants c, d, c_ℓ needs to satisfy the following conditions. The constant c is chosen such that H will not be an overly large portion of $\mathbb{F}$ (this requirement is used in Lemma 42). The constant d is needed to subsume the length of PCPP proof string which is part of the construction (this requirement is used in Lemma 36). Finally, the constant c_ℓ needs to be large enough to enable iteration (as explained in Definition 31 itself).

Let $\ell \geq 0$ be some fixed iteration. The following simple observation follows by a simple inductive argument using the definition of the level-ℓ coding ensemble, and in particular that $|\mathbb{F}'| = \mathrm{polylog}\,|\mathbb{F}|$.

▶ **Observation 32.** *For $\ell > 0$, let $n = |\mathbb{F}|$ and $w \in \{0,1\}^k$. If $v \in \mathcal{E}^{(\ell)}(w)$, then $m_\mathbb{F}^{(\ell)} \stackrel{\text{def}}{=} |v| = n \cdot \mathrm{poly}(\log n) \cdot \mathrm{poly}(\log\log n) \cdots \mathrm{poly}(\log^{(\ell)} n)$, where $\log^{(\ell)} n$ is the* log *function iterated ℓ times.*

When the field $\mathbb{F}$ is clear from context, we shall usually write $m^{(\ell)}$ as a shorthand for $m^{(\ell)}_{\mathbb{F}}$. The following lemma, proved in the next subsection, establishes the existence of short length Spiel-PCUs for our code ensembles.

▶ **Lemma 33.** *For any $\ell \geq 0$, the code ensemble $\mathcal{E}^{(\ell)}_{\mathbb{F},k}$ admits a $q(\varepsilon,\delta)$-query, length-t* Spiel-PCU *for $t = O(m^{(\ell)} \cdot \mathrm{polylog}^{(\ell)} m^{(\ell)})$.*

5.2 Proof of Lemma 33

We start by defining the PCU proof string for a given $v \in \mathcal{E}^{(\ell)}_{\mathbb{F},k}(w)$ for some $w \in \{0,1\}^k$.

▶ **Definition 34** (The PCU Proof String)**.** *For $\ell = 0$, let $v \in \mathcal{E}^{(0)}_k(w)$ and $\boldsymbol{Value}^{(0)}(v) = \mathsf{Spiel}(w)$. We define the proof string for v, $\boldsymbol{Proof}^{(0)}(v)$, as the one guaranteed by Lemma 28 (note that the length of $\boldsymbol{Proof}^{(0)}(v)$ is $\mathrm{poly}(k)$).*

For $\ell > 0$, let $g \in \mathcal{C}_{\mathbb{F}}$ and $w \in \{0,1\}^k$ be such that $v \in \bigsqcup_{\beta\in\mathbb{F}\setminus H} \mathcal{E}^{(\ell-1)}(\langle\!\langle g(\beta)\rangle\!\rangle)$, $\boldsymbol{Value}^{(\ell)}(v) = \mathsf{Spiel}(w)$ and $g|_H = w$. In addition, set $S_v \stackrel{\text{def}}{=} \bigsqcup_{\beta\in\mathbb{F}\setminus H} \boldsymbol{Value}^{(\ell-1)}(v_\beta) = \bigsqcup_{\beta\in\mathbb{F}\setminus H} \mathsf{Spiel}(g(\beta))$. The proof string for $v \in \mathcal{E}^{(\ell)}_{\mathbb{F},k}$ is defined as follows.

$$\boldsymbol{Proof}^{(\ell)}(v) = S_v \sqcup \bigsqcup_{\beta\in\mathbb{F}\setminus H} \boldsymbol{Proof}^{(\ell-1)}(v_\beta) \sqcup \boldsymbol{Proof}_{\mathcal{L}}(S_v)$$

where the code ensemble $\mathcal{L} : \{0,1\}^k \to 2^{\{0,1\}^{O(|\mathbb{F}|\cdot\log|\mathbb{F}|)}}$ is defined as follows. Given $w \in \{0,1\}^k$, $S \in \mathcal{L}(w)$ if and only if there exists a polynomial $g \in \mathcal{C}_{\mathbb{F}}$ such that the following conditions are satisfied.

1. $g|_H = w$.
2. $S = \bigsqcup_{\beta\in\mathbb{F}\setminus H} \mathsf{Spiel}(g(\beta))$.

The following lemma establishes the existence of a Spiel-PCU for $\mathcal{L}$.

▶ **Lemma 35.** *$\mathcal{L}$ has a $q(\varepsilon,\delta)$-query length-t* Spiel-PCU *for $t = O(|\mathbb{F}| \cdot \mathrm{polylog}\,|\mathbb{F}|)$.*

Proof. By Theorem 5, there exists a quasilinear size circuit that decodes $\mathsf{Spiel}(\alpha)$. Using such a circuit, we can decode $g(\beta)$ from S for every $\beta \in \mathbb{F}$. Then, using all the values $g(\beta)$ and w (where the i-th bit of w correspond to the value of the i-th element in H according to the ordering), we use Theorem 7 to interpolate the values and achieve a representation of a polynomial $g : \mathbb{F} \to \mathbb{F}$. If $g \in \mathcal{C}_{\mathbb{F}}$ we accept S and otherwise we reject. Since deciding if $S \in \mathcal{L}(w)$ has a quasilinear size circuit, by Lemma 26, there is a quasilinear length Spiel-PCU for $\mathcal{L}$. ◀

Having defined $\mathbf{Proof}^{(\ell)}$, we first provide an upper bound on the bit length of the prescribed proof string. For $\ell > 0$, let $z^{(\ell)}_{\mathbb{F},k}$ denote the bit length of the proof for membership in $\mathcal{E}^{(\ell)}$ as defined in Definition 34, where for $\ell = 0$ we replace the (nonexistent) field $\mathbb{F}$ with $|w|$.

The following lemma, establishing the proof string's length, relies on our choice of the constant d in Definition 31. In particular, d needs to be large enough to subsume the size of $\mathbf{Proof}_{\mathcal{L}}(\cdot)$

▶ **Lemma 36.** *For any $\ell \geq 0$, we have that $z^{(\ell)}_{\mathbb{F},k} = O\big(m^{(\ell)} \cdot \mathrm{polylog}^{(\ell)} m^{(\ell)}\big)$.*

Proof. The proof follows by induction on ℓ. The base case ($\ell = 0$) follows directly from the definition of $\mathcal{P}^{(0)}$ by our convention that $\log^{(0)}|w| = |w|$.

Consider $\ell > 0$, and note that since the size of S_v is $O(|\mathbb{F}| \log |\mathbb{F}|)$, the size of $\mathbf{Proof}_{\mathcal{L}}(S_v)$ is $O(|\mathbb{F}| \cdot \text{polylog } |\mathbb{F}|)$. By combining the above with the definition of the proof string we have

$$z^{(\ell)}_{\mathbb{F},k} \leq |\mathbb{F}| \cdot \text{polylog } |\mathbb{F}| + |\mathbb{F}| \cdot z^{(\ell-1)}_{\mathbb{F}',k'}.$$

Now, assume that $z^{(\ell-1)}_{\mathbb{F}',k'} = O(m^{(\ell-1)} \cdot \text{polylog}^{(\ell-1)}|\mathbb{F}'|)$. Note that since the global constant d was chosen so that $|\mathbb{F}| \cdot |\mathbb{F}'| \geq |\mathbf{Proof}_{\mathcal{L}}(S_v)|$, we have that $|\mathbb{F}| \cdot z^{(\ell-1)}_{\mathbb{F}',k'} \geq |\mathbb{F}| \cdot |\mathbb{F}'| \geq |\mathbf{Proof}_{\mathcal{L}}(S_v)|$. Therefore,

$$m^{(\ell)} = \Theta(|\mathbb{F}| \cdot m^{(\ell-1)}) = \Omega(|\mathbb{F}| \cdot |\mathbb{F}'|) = \Omega(|\mathbb{F}| \cdot \text{polylog } |\mathbb{F}|),$$

so that $|\mathbb{F}| \cdot \text{polylog } |\mathbb{F}| = O(m^{(\ell)})$, and

$$z^{(\ell)}_{\mathbb{F},k} = O(|\mathbb{F}| \cdot z^{(\ell-1)}_{\mathbb{F}'}).$$

In addition, by the fact that $m_\ell = \Theta(|\mathbb{F}| \cdot m^{(\ell-1)})$ and the induction hypothesis we obtain

$$|\mathbb{F}| \cdot z^{(\ell-1)}_{\mathbb{F}',k'} = O(|\mathbb{F}| \cdot m^{(\ell-1)} \cdot \text{polylog}^{(\ell-1)}|\mathbb{F}'|) = O(m^{(\ell)} \cdot \text{polylog}^{(\ell)}|\mathbb{F}|) = O(m^{(\ell)} \cdot \text{polylog}^{(\ell)} m_\ell).$$

So overall, we get that $z^{(\ell)}_{\mathbb{F},k} = O(m^{(\ell)} \cdot \text{polylog}^{(\ell)} m^{(\ell)})$ as required. ◀

Next, for an alleged proof $\pi = \mathbf{Proof}^{(\ell)}(v)$, we use the notation $\pi|_{\text{Dom}(X)}$ to denote the restriction of π to the bits that correspond to X in π as defined in Definition 34. For example, $\pi|_{\text{Dom}(\mathbf{Value}^{(\ell-1)}(v_\beta))}$ refers to the bits that represent $\mathbf{Value}^{(\ell-1)}(v_\beta)$.

We introduce the verifier procedure for $\mathcal{E}^{(\ell)}_{\mathbb{F},k}$ (see Figure 1), and prove its completeness and soundness. For technical considerations, the verifier procedure is only defined when the soundness parameter δ is small enough (as a function of ℓ); the soundness amplification argument from Subsection 2.2 can easily take care of the situation where δ is larger, by running sufficiently many independent instances of the verification step.

Before proceeding to the completeness and soundness proofs, let us analyze the query complexity. Denote by $Q_\ell(\varepsilon, \delta)$ the query complexity of the verifier in the above procedure for a given $\ell \geq 0$. It follows from the recursive description of `Verifier-Procedure` $_{\mathcal{E}^{(\ell)}}$ and the proof of Lemmas 26 and 35 that the query complexity satisfies the recurrence relation $Q_\ell(\varepsilon, \delta) \leq O(1/\varepsilon) \cdot Q_{\ell-1}(\varepsilon/O(1)), \delta \cdot O(1)) + q^*(\Theta(\varepsilon), \Theta(\delta))$, where $q^*(\varepsilon^*, \delta^*) = O((\varepsilon^*)^{-1})$ is the query complexity of Dinur's PCP [12] with detection radius ε^* and soundness parameter $\delta^* \leq 1/2$; and furthermore, that $Q_0(\varepsilon, \delta) \leq q^*(\Theta(\varepsilon), \Theta(\delta))$. Thus, we conclude by induction that, provided that $\delta \leq 2^{-\ell-1}$,

$$Q_\ell(\varepsilon, \delta) \leq \frac{C}{\varepsilon} \cdot \frac{C^2}{\varepsilon} \cdot \ldots \cdot \frac{C^\ell}{\varepsilon} \cdot q^*(\varepsilon/2^{O(\ell)}, \delta \cdot 2^\ell) = 2^{O(\ell^2)} \varepsilon^{-O(\ell)},$$

where $C > 0$ is a large enough absolute constant. To achieve any given soundness $\delta > 1/2$, we can amplify by repeating the verifier procedure with parameter $\delta' = 2^{-\ell-1}$ a total of $2^{O(\ell)} \cdot \log\left((1-\delta)^{-1}\right)$ times and rejecting if any of these instances rejected. The query complexity is bounded by

$$2^{O(\ell^2)} \varepsilon^{-O(\ell)} \cdot 2^{O(\ell)} \cdot \log\left((1-\delta)^{-1}\right) = (2^\ell/\varepsilon)^{O(\ell)} \cdot \log\left((1-\delta)^{-1}\right),$$

as desired. The next two lemmas establish the completeness and soundeness of the verifier procedure, respectively.

`Verifier-Procedure` $_{\mathcal{E}^{(\ell)}}$

Input: Parameters $\varepsilon, \delta \in (0,1)$ where $\delta \leq 2^{-\ell-1}$, an input $v \in \{0,1\}^{m^{(\ell)}}$, an alleged value $\tau \in \{0,1\}^{100k}$ of v, and an alleged proof $\pi \in \{0,1\}^{z^{(\ell)}_{\mathbb{F},k}}$ for v.

1. If $\ell = 0$, use the PCU for $\mathcal{E}^{(0)}$ with parameters ε and δ.
2. If $\ell > 0$:
   a. Use the PCU verifier for $\mathcal{L}$ with radius $\varepsilon/300$ and soundness δ, to verify the unveiling of $\pi|_{\mathrm{Dom}(S_v)}$, using τ as the value oracle and $\pi|_{\mathrm{Dom}(\mathbf{Proof}_{\mathcal{L}}(S_v))}$ as the proof oracle.
   b. For $6/\varepsilon$ many times:
      i. Pick $\boldsymbol{\beta} \in \mathbb{F} \setminus H$ uniformly at random.
      ii. Use the PCU verifier procedure for $\mathcal{E}^{(\ell-1)}$ with parameters $\varepsilon/3$ and 2δ, to verify the unveiling of $v_{\boldsymbol{\beta}}$, using $\pi|_{\mathrm{Dom}(\mathbf{Value}^{(\ell-1)}(v_\beta))}$ as the value oracle and $\pi|_{\mathrm{Dom}(\mathbf{Proof}^{(\ell-1)}(v_{\boldsymbol{\beta}}))}$ as the proof oracle.

If any of the stages rejected then **Reject**, and otherwise **Accept**.

Figure 1 Description of `Verifier-Procedure` $_{\mathcal{E}^{(\ell)}}$.

▶ **Lemma 37.** *If there exist $w \in \{0,1\}^k$ for which $v \in \mathcal{E}^{(\ell)}_{\mathbb{F},k}(w)$, then* `Verifier-Procedure` $_{\mathcal{E}^{(\ell)}}$ *accepts v with probability 1 when supplied with oracle access to the corresponding $\boldsymbol{Proof}^{(\ell)}(v)$ and $\tau = \boldsymbol{Value}^{(\ell)}(v) = \mathsf{Spiel}(w)$.*

Proof. The proof follows by induction on ℓ. The base case follows directly from Lemma 28. Hence, the verifier for $\mathcal{E}^{(0)}$ supplied with $\mathbf{Proof}^{(0)}(v)$ as the proof oracle and $\mathbf{Value}^{(\ell)}(v)$ as the value oracle, will accept v with probability 1.

Assume that `Verifier-Procedure` $_{\mathcal{E}^{(\ell-1)}}$ accepts with probability 1 any valid encoding v' when supplied with the corresponding oracles for $\mathbf{Value}^{(\ell-1)}(v')$ and $\mathbf{Proof}^{(\ell-1)}(v')$. Let $v \in \mathcal{E}^{(\ell)}$ and write $v = \bigsqcup_{\beta \in \mathbb{F} \setminus H} v_\beta$, where there exist $w \in \{0,1\}^k$ and $g \in \mathcal{C}_{\mathbb{F}}$ such that for all $\beta \in \mathbb{F} \setminus H$, $v_\beta \in \mathcal{E}^{(\ell-1)}(g(\beta))$, where $g|_H = w$ and $\tau = \mathbf{Value}^{(\ell)}(v) = \mathsf{Spiel}(w)$. Then, by the definition of the language $\mathcal{L}$ and the first two components of $\mathbf{Proof}^{(\ell)}(v)$, Step (2a) of `Verifier-Procedure` $_{\mathcal{E}^{(\ell)}}$ will always accept. In addition, for every $\beta \in \mathbb{F} \setminus H$, we have that $v_\beta \in \mathcal{E}^{(\ell-1)}$, and therefore by the induction hypothesis, Step (2b) of `Verifier-Procedure` $_{\mathcal{E}^{(\ell)}}$ will accept the corresponding unveiling for any picked $\beta \in \mathbb{F} \setminus H$. ◀

▶ **Lemma 38.** *If for every $w \in \{0,1\}^k$ either $\mathrm{dist}(\tau, \mathsf{Spiel}(w)) > \varepsilon$ or $\mathrm{dist}(v, \mathcal{E}^{(\ell)}(w)) > \varepsilon$ (or both), then with probability greater than δ,* `Verifier-Procedure` $_{\mathcal{E}^{(\ell)}}$ *will reject v regardless of the contents of the supplied proof string.*

Proof. Let $\tau \in \{0,1\}^{100k}$ be an alleged value for v, and $\pi \in \{0,1\}^{z^{(\ell)}_{\mathbb{F},k}}$ be an alleged proof string for v. We proceed by induction on ℓ. For $\ell = 0$ we use the PCU verifier for $\mathcal{E}^{(0)}$ with error ε and soundness δ to check that v is a member of the code ensemble $\mathcal{E}^{(0)}$ and τ is its value. If the PCU verifier for $\mathcal{E}^{(0)}$ rejects with probability at most δ, then there exist $w \in \{0,1\}^k$ such that $\mathrm{dist}(v, \mathcal{E}^{(0)}(w)) \leq \varepsilon$ and $\mathrm{dist}(\tau, \mathsf{Spiel}(w)) \leq \varepsilon$, and the base case is complete.

Next assume that the lemma holds for $\ell - 1$. If the PCU verifier for $\mathcal{L}$ in Step (2a) rejects with probability at most δ, then there exist a function $g \in \mathcal{C}_{\mathbb{F}}$ and $w \in \{0,1\}^k$ for which $g|_H = w$ so that

$$\mathrm{dist}(\pi|_{\mathrm{Dom}(S_v)}, \mathsf{Spiel}(g|_{\mathbb{F}\setminus H})) \leq \varepsilon/300 \qquad \text{and} \qquad \mathrm{dist}(\tau, \mathsf{Spiel}(w)) \leq \varepsilon/300.$$

In particular, the leftmost inequality means that for at most $\frac{\varepsilon}{3}|\mathbb{F}\setminus H|$ of the elements $\beta \in \mathbb{F}\setminus H$, it holds that

$$\mathrm{dist}(\pi|_{\mathrm{Dom}(\mathbf{Value}^{(\ell-1)}(v_\beta))}, \mathsf{Spiel}(g(\beta)) > 1/100.$$

We refer to elements $\beta \in \mathbb{F} \setminus H$ satisfying the above inequality as *bad* elements, and to the rest as *good* elements. Let G denote the set of good elements.

Next, we show that if the loop that uses the PCU verifier for $\mathcal{E}^{(\ell-1)}$ in Step (2b) rejects with probability at most δ, then for at most an $\varepsilon/3$ fraction of the good $\beta \in \mathbb{F} \setminus H$, it holds that

$$\mathrm{dist}\left(v_\beta, \mathcal{E}^{(\ell-1)}(\langle\!\langle g(\beta)\rangle\!\rangle)\right) > \varepsilon/3.$$

Assume that there are more than $\frac{\varepsilon}{3} \cdot |G|$ good elements such that $\mathrm{dist}\left(v_\beta, \mathcal{E}^{(\ell-1)}(\langle\!\langle g(\beta)\rangle\!\rangle)\right) > \varepsilon/3$. Then, by our induction hypothesis, each of them will be rejected by the PCU verifier for $\mathcal{E}^{(\ell-1)}$ with probability more than 2δ. In addition, with probability at least $1/2$ we sample at least one such good β, and then during this iteration the verifier in Step (2b(ii)) rejects with conditional probability more than 2δ, and hence the verifier will reject with overall probability more than δ. Summing everything up, when the input is rejected with probability at most δ,

$$\mathrm{dist}\left(v, \bigsqcup_{\beta\in\mathbb{F}\setminus H} \mathcal{E}^{(\ell-1)}(\langle\!\langle g(\beta)\rangle\!\rangle)\right) \leq \varepsilon/3 + (1-\varepsilon/3)\cdot\varepsilon/3 + (1-\varepsilon/3)^2\cdot\varepsilon/3 \leq \varepsilon,$$

where the three summands are respectively the contribution to the distance of the bad elements, the good elements with v_β being far from any level $\ell - 1$ encoding of $\langle\!\langle g(\beta)\rangle\!\rangle$, and all the other elements. ◀

The proof of Lemma 33 follows directly by combining Lemma 36, Lemma 37 and Lemma 38.

The following corollary follows directly from Lemma 33 and the definition of Spiel-PCU (Definition 25), and implies Lemma 29.

▶ **Corollary 39.** *Let $\mathbb{F}$ be a finite field and $k \in \mathbb{N}$ which satisfy the requirements in Definition 31. Then, for every $\ell \geq 0$ the coding ensemble $\mathcal{E}^{(\ell)}_{\mathbb{F},k} : \{0,1\}^k \to 2^{\left(\{0,1\}^{m^{(\ell)}}\right)}$ has a $q(\varepsilon,\delta)$-query, length-t Spiel-PCU for $t = O(m^{(\ell)}\mathrm{polylog}^{(\ell)}m^{(\ell)})$.*

5.3 The Lower Bound

We turn to prove the linear query lower bound for the testability of our property. We start by defining distributions over strings of length $m^{(\ell)}$.

Distribution $\mathcal{D}^{(\ell)}_{\mathsf{yes}}(w)$: Given $w \in \{0,1\}^k$, we define the distribution $\mathcal{D}^{(\ell)}_{\mathsf{yes}}(w)$ to be the uniform distribution over elements in $\mathcal{E}^{(\ell)}(w)$.

Distribution $\mathcal{D}^{(\ell)}_{\mathsf{no}}$: An element v from $\mathcal{D}^{(\ell)}_{\mathrm{no}}$ is drawn by the following process. For $\ell = 0$, $\boldsymbol{v}$ is a uniformly random string in $\{0,1\}^{4k}$. For $\ell > 0$, we pick a uniformly random function $\boldsymbol{\lambda} : \mathbb{F} \setminus H \to \mathbb{F}$, and let v be a uniformly random element of $\bigsqcup_{\beta \in \mathbb{F} \setminus H} \mathcal{E}^{(\ell-1)}(\langle\!\langle \boldsymbol{\lambda}(\beta) \rangle\!\rangle)$

▶ **Lemma 40.** *For any $\ell \geq 0$, every $w \in \{0,1\}^k$ and $q = o(m^{(\ell)}/10^{\ell})$, any algorithm making at most q queries cannot distinguish (with constant probability) between $\boldsymbol{v} \sim \mathcal{D}^{(\ell)}_{yes}(w)$ and $\boldsymbol{u}$ which is drawn according to any of the following distributions:*
1. *$\mathcal{D}^{(\ell)}_{yes}(w')$ for any $w' \neq w$.*
2. *$\mathcal{D}^{(\ell)}_{no}$.*

Note that Item (1) in the above follows immediately from Item (2). Additionally, the first item implies the shared secret property of the code ensemble $\mathcal{E}^{(\ell)}$. Furthermore, we remark that that above lemma implies a more stringent version of PCUSS. In addition to the shared secret property, Item (2) implies that the ensemble $\mathcal{E}^{(\ell)}$ is indistinguishable from strings that are mostly far from any encoding (i.e., drawn from $\mathcal{D}^{(\ell)}_{\mathrm{no}}$).

The proof of Lemma 40 follows by induction over ℓ. Before we continue, we introduce some useful lemmas that will be used in the proof.

▶ **Lemma 41.** *For any $\ell \geq 0$ and $w, w' \in \{0,1\}^k$ for which $w \neq w'$ it holds that*

$$\min_{(v,v') \in \mathcal{E}^{(\ell)}(w) \times \mathcal{E}^{(\ell)}(w')} \mathrm{dist}(v, v') = \Theta\left(1/4^{\ell+1}\right).$$

Proof. The proof follows by induction over ℓ. The base case for $\ell = 0$ follows directly by the fact that the code from Definition 21 has high distance, and in particular $\mathrm{dist}(\mathcal{E}^{(0)}(w), \mathcal{E}^{(0)}(w')) > 1/10$. Assume that the lemma holds for $\ell - 1$. Namely, for $w, w' \in \{0,1\}^{k'}$ for which $w \neq w'$ it holds that

$$\min_{(v,v') \in \mathcal{E}^{(\ell-1)}(w) \times \mathcal{E}^{(\ell-1)}(w')} \mathrm{dist}(v, v') = \Theta\left(1/4^{\ell}\right).$$

Let $\tilde{w}, \tilde{w}' \in \{0,1\}^k$ be such that $\tilde{w}' \neq \tilde{w}$. Then we can write $(\tilde{v}, \tilde{v}') \in \mathcal{E}^{(\ell)}(\tilde{w}) \times \mathcal{E}^{(\ell)}(\tilde{w}')$ as

$$\tilde{v} = \bigsqcup_{\beta \in \mathbb{F} \setminus H} \mathcal{E}^{(\ell-1)}(\langle\!\langle g(\beta) \rangle\!\rangle) \qquad \text{and} \qquad \tilde{v}' = \bigsqcup_{\beta \in \mathbb{F} \setminus H} \mathcal{E}^{(\ell-1)}(\langle\!\langle g'(\beta) \rangle\!\rangle),$$

for some $g, g' \in \mathcal{C}_{\mathbb{F}}$ such that $g|_H = \tilde{w}$ and $g'|_H = \tilde{w}'$. By the fact that g and g' are degree $|\mathbb{F}|/2$ polynomials (which are not identical), we have that g and g' disagree on at least $|\mathbb{F} \setminus H|/4$ of the elements $\beta \in \mathbb{F} \setminus H$. By applying the induction hypothesis on the minimum distance between $\mathcal{E}^{(\ell)}(\langle\!\langle g(\beta) \rangle\!\rangle)$ and $\mathcal{E}^{(\ell)}(\langle\!\langle g'(\beta) \rangle\!\rangle)$, for all β such that $g(\beta) \neq g'(\beta)$, we have that

$$\min_{(\tilde{v},\tilde{v}') \in \mathcal{E}^{(\ell)}(\tilde{w}) \times \mathcal{E}^{(\ell)}(\tilde{w}')} \mathrm{dist}(\tilde{v}, \tilde{v}') > \frac{1}{4} \cdot \Theta\left(\frac{1}{4^{\ell}}\right) = \Theta\left(1/4^{\ell+1}\right). \qquad \blacktriangleleft$$

▶ **Lemma 42.** *For any $\ell \geq 0$, with probability at least $1 - o(1)$, a string v drawn from $\mathcal{D}^{(\ell)}_{no}$ satisfies $\mathrm{dist}(v, \mathcal{E}^{(\ell)}(w)) = \Theta\left(1/4^{\ell+1}\right)$ for all $w \in \{0,1\}^k$.*

Proof. The proof follows by induction over ℓ. For $\ell = 0$, fix some $w \in \{0,1\}^k$. Consider the size of a ball of relative radius $1/40$ around some $v \in \mathcal{E}^{(0)}(w)$ in the space of all strings $\{0,1\}^{4k}$. The number of strings contained in this ball is at most

$$\binom{4k}{k/10} \leq (40e)^{k/10} = 2^{k/10 \cdot \log(40e)}.$$

Thus, the size of the set of strings which are at relative distance $1/40$ from any legal encoding of some word $w \in \{0,1\}^k$ is at most

$$2^{3k} \cdot 2^{k/10 \cdot \log(40e)} = o(2^{4k}).$$

This implies that with probability at least $1 - o(1)$, a random string from $\{0,1\}^{4k}$ is $1/40$-far from $\mathcal{E}^{(0)}(w)$ for any $w \in \{0,1\}^k$.

For any $\ell > 0$, consider v' sampled according to $\mathcal{D}^{(\ell)}_{\text{no}}$. Then, v' can be written as

$$v' = \bigsqcup_{\beta \in \mathbb{F} \setminus H} \mathcal{E}^{(\ell-1)}(\langle\langle \boldsymbol{\lambda}(\beta) \rangle\rangle),$$

where $\boldsymbol{\lambda} : \mathbb{F} \setminus H \to \mathbb{F}$ is a uniformly random function. On the other hand, each member $\tilde{v}$ of $\mathcal{P}^{(\ell)}$ can be written as

$$\tilde{v} = \bigsqcup_{\beta \in \mathbb{F} \setminus H} \mathcal{E}^{(\ell-1)}(\langle\langle g(\beta) \rangle\rangle),$$

for some $g \in \mathcal{C}_{\mathbb{F}}$ such that $g|_H = w$ for some $w \in \{0,1\}^k$. Note that by Lemma 41, whenever $\boldsymbol{\lambda}(\beta) \neq g(\beta)$, we have that the minimum distance between any $\tilde{v} \in \mathcal{E}^{(\ell-1)}(\langle\langle g(\beta) \rangle\rangle)$ and $v' \in \mathcal{E}^{(\ell-1)}(\langle\langle \boldsymbol{\lambda}(\beta) \rangle\rangle)$ is at least $\Theta(1/4^\ell)$. In addition, by Lemma 8, we have that that with probability at least $1 - o(1)$, a uniformly random function $\boldsymbol{\lambda} : \mathbb{F} \to \mathbb{F}$ is $1/3$-far from any $g \in \mathcal{C}_{\mathbb{F}}$. By the restrictions on k in Definition 31, which implies that $|H| \leq |F|/c$, we can ensure (by the choice of c) that with probability at least $1 - o(1)$, that a uniformly random $\boldsymbol{\lambda} : \mathbb{F} \setminus H \to \mathbb{F}$ is at least $1/4$-far from the restriction $g|_{\mathbb{F} \setminus H}$. This implies that for at least $|\mathbb{F} \setminus H|/4$ of the elements $\beta \in \mathbb{F} \setminus H$, we have that $\boldsymbol{\lambda}(\beta) \neq g(\beta)$. Therefore, we have that $\text{dist}(v', \mathcal{E}^{(\ell)}(w)) = \frac{1}{4} \cdot \Theta\left(\frac{1}{4^\ell}\right) = \Theta\left(1/4^{\ell+1}\right)$ for all $w \in \{0,1\}^k$, and the proof is complete. ◀

▶ **Lemma 43.** *Fix any $\ell > 0$, and suppose that for any $w' \in \{0,1\}^{k'}$, and for any set Q' of at most $q^{(\ell-1)}_{\mathbb{F}',k'}$ queries (where $\mathbb{F}'$ and k' are picked according to the recursive definition of the level ℓ-encoding, and for $q^{(0)}$ we substitute k' for the nonexistent $\mathbb{F}'$) the restricted distributions $\mathcal{D}^{(\ell-1)}_{yes}(w')|_{Q'}$ and $\mathcal{D}^{(\ell-1)}_{no}|_{Q'}$ are identical. Then, for any $w \in \{0,1\}^k$, and any set Q of at most $\frac{|\mathbb{F} \setminus H|}{10} \cdot q^{(\ell-1)}_{\mathbb{F}',k'}$ queries, the restricted distributions $\mathcal{D}^{(\ell)}_{yes}(w)|_Q$ and $\mathcal{D}^{(\ell)}_{no}|_Q$ are identical.*

Proof. Let $Q \subset [m^{(\ell)}]$ be the set of queries, and fix a canonical ordering over the elements in $\mathbb{F} \setminus H$. Let $\boldsymbol{v}$ be an element drawn according to distribution $\mathcal{D}^{(\ell)}_{\text{yes}}(w)$, and let $\boldsymbol{v}'$ be an element drawn according to distribution $\mathcal{D}^{(\ell)}_{\text{no}}$. The sampling process from $\mathcal{D}^{(\ell)}_{\text{yes}}(w)$ can be thought of as first drawing a uniformly random function $\boldsymbol{g} \in \mathcal{C}_{\mathbb{F}}$ such that $\boldsymbol{g}|_H = w$, and for every $\beta \in \mathbb{F} \setminus H$, letting $\boldsymbol{v}_\beta$ be a uniformly random element in $\mathcal{E}^{(\ell-1)}(\langle\langle \boldsymbol{g}(\beta) \rangle\rangle)$.

For each $\beta \in \mathbb{F} \setminus H$ we set $Q_\beta = Q \cap \text{Dom}(v_\beta)$, and define the set of *big clusters*

$$I = \left\{ \beta \in \mathbb{F} \setminus H \ : \ |Q_\beta| \geq q^{(\ell-1)}_{\mathbb{F}',k'} \right\}.$$

Note that since $|Q| \leq |\mathbb{F} \setminus H| \cdot q^{(\ell-1)}_{\mathbb{F}',k'}/10$, we have that $|I| \leq |\mathbb{F} \setminus H|/10$.

By the fact that $\boldsymbol{g}$ is a uniformly random polynomial of degree $|\mathbb{F}|/2 > |I|$, we have that $\boldsymbol{g}|_I$ is distributed exactly as $\boldsymbol{\lambda}|_I$ (both are a sequence of $|I|$ independent uniformly random values), which implies that $\boldsymbol{v}|_{\bigcup_{j \in I} Q_j}$ is distributed exactly as $\boldsymbol{v}'|_{\bigcup_{j \in I} Q_j}$.

Next, let $\mathbb{F} \setminus (I \cup H) = \{i_1, \ldots, i_{|\mathbb{F} \setminus (I \cup H)|}\}$ be a subset ordered according to the canonical ordering over $\mathbb{F}$. We proceed by showing that $\boldsymbol{v}|_{\bigcup_{j \in I \cup \{i_1, \ldots, i_t\}} Q_j}$ is distributed identically to $\boldsymbol{v}'|_{\bigcup_{j \in I \cup \{i_1, \ldots, i_t\}} Q_j}$ by induction over t.

The base case ($t = 0$) corresponds to the restriction over $\bigcup_{j \in I} Q_j$, which was already proven above. For the induction step, let $T = \{i_1, \ldots, i_{t-1}\} \subseteq \mathbb{F} \setminus (I \cup H)$ be an ordered subset that agrees with the canonical ordering on $\mathbb{F}$, and let $i_t \in \mathbb{F} \setminus (H \cup T \cup I)$ be the successor of i_{t-1} according to the ordering. We now prove that for each $x \in \{0,1\}^{m^{(\ell)}}$ for which $\boldsymbol{v}|_{\bigcup_{j \in I \cup T} Q_j}$ has a positive probability of being equal to $x|_{\bigcup_{j \in I \cup T} Q_j}$, conditioned on the above event taking place (and its respective event for v'), $\boldsymbol{v}|_{Q_{i_t}}$ is distributed exactly as $\boldsymbol{v'}|_{Q_{i_t}}$.

Observe that conditioned on the above event, $\boldsymbol{v}|_{Q_{i_t}}$ is distributed exactly as a uniformly random element in $\mathcal{E}^{(\ell-1)}(\boldsymbol{\rho})$ for some $\boldsymbol{\rho} \in \{0,1\}^{k'}$ (which follows some arbitrary distribution, possibly depending on $x|_{\bigcup_{j \in I \cup T} Q_j}$), while $\boldsymbol{v'}|_{Q_{i_t}}$ is distributed exactly as a uniformly random element in $\mathcal{E}^{(\ell-1)}(\boldsymbol{y})$ for a uniformly random $\boldsymbol{y} \in \{0,1\}^{k'}$. By the fact that $|Q_{i_t}| \leq q^{(\ell-1)}_{\mathbb{F}',k'}/10$, we can apply the induction hypothesis and conclude that $\boldsymbol{v}|_{Q_{i_t}}$ is distributed exactly as $\boldsymbol{v'}|_{Q_{i_t}}$, because by our hypothesis both are distributed identically to the corresponding restriction of $\mathcal{D}^{(\ell-1)}_{\text{no}}$, regardless of the values picked for $\boldsymbol{\rho}$ and $\boldsymbol{y}$. This completes the induction step for t. The lemma follows by setting $t = |\mathbb{F} \setminus H \cup I|$. ◀

▶ **Lemma 44.** *For any $\ell \geq 0$, $w \in \{0,1\}^k$ and any set of queries $Q \subset [m^{(\ell)}]$ such that $|Q| = O\left(\frac{m^{(\ell)}}{10^\ell}\right)$, the restricted distributions $\mathcal{D}^{(\ell)}_{yes}(w)|_Q$ and $\mathcal{D}^{(\ell)}_{no}|_Q$ are identically distributed.*

Proof. By induction on ℓ. For $\ell = 0$ and any $w \in \{0,1\}^k$, by the fact that our base encoding $\mathcal{E}^{(0)}(w)$ is a high dual distance code, we can select (say) $q^{(0)} = k/c$ (for some constant $c > 0$), making the assertion of the lemma trivial.

Assume that for any $w' \in \{0,1\}^{k'}$, and any set of queries Q' of size up to $O(m^{(\ell-1)}/10^{\ell-1})$ the conditional distributions $\mathcal{D}^{(\ell-1)}_{\text{yes}}(w')|_{Q'}$ and $\mathcal{D}^{(\ell-1)}_{\text{no}}|_{Q'}$ are identically distributed. Then, by Lemma 43, we have that for any $w \in \{0,1\}^k$ and any set of queries Q of size at most

$$O\left(\frac{|\mathbb{F} \setminus H|}{10^\ell} \cdot m^{(\ell-1)}\right),$$

the restricted distributions $\mathcal{D}^{(\ell)}_{\text{yes}}(w)|_Q$ and $\mathcal{D}^{(\ell)}_{\text{no}}|_Q$ are identically distributed. Note that by definition of the level ℓ-encoding, $m^{(\ell)} = |\mathbb{F} \setminus H| \cdot m^{(\ell-1)}$, which implies the conclusion of the lemma. ◀

Proof of Lemma 40. Lemma 40 follows directly by combining Lemma 17, and Lemma 44. ◀

Combining Lemma 40 with the definition of Spiel-PCU (Definition 25) establishes that we have constructed a Spiel-PCUSS, which implies Theorem 30.

▶ **Corollary 45.** *Let $\mathbb{F}$ be a finite field and $k \in \mathbb{N}$ which satisfy the requirements in Definition 31. Then, for every $\ell \geq 0$, the coding ensemble $\mathcal{E}^{(\ell)}_{\mathbb{F},k} : \{0,1\}^k \to 2^{\left(\{0,1\}^{m^{(\ell)}}\right)}$ has $q(\varepsilon,\delta)$-query length-t Spiel-PCUSS for $t = O(m^{(\ell)}\text{polylog}^{(\ell)}m^{(\ell)})$.*

5.4 Handling arbitrary input lengths

As mentioned in the beginning of this section, our construction of code ensembles relies on the fact that operations over a finite field $\text{GF}(2^t)$ can be computed efficiently. In order to do so we need to have an irreducible polynomial of degree t over $\text{GF}(2)$, so that we have a representation $\text{GF}(2^t)$. Given such a polynomial, operations over the field can be

implemented in polylogarithmic time in the size of the field. By [18] (Appendix G), we know that for $t = 2 \cdot 3^r$ where $r \in \mathbb{N}$, we do have such a representation. However, the setting of t restricts the sizes of the fields that we can work with, which will limit our input size length.

We show here how to extend our construction to a set of sizes that is "log-dense". For a global constant c', our set of possible input sizes includes a member of $[m', c'm']$ for every m'. Moving from this set to the set of all possible input sizes now becomes a matter of straightforward padding.

For any $n \in \mathbb{N}$, let r be the smallest integer such that $n < 2^{2 \cdot 3^r}$ and let $\mathbb{F} = \mathrm{GF}(2^{2 \cdot 3^r})$. We make our change only at the level-ℓ construction. First, we use $4d$ instead of d in the calculation of the size of $\mathbb{F}'$. Then, instead of using $\mathbb{F} \setminus H$ as the domain for our input, we use $E \setminus H$, for any arbitrary set $E \subseteq \mathbb{F}$ of size $n \geq \max\{4k, |\mathbb{F}|^{1/4}, c_\ell\}$ that contains H. Then, for the level-ℓ, instead of considering polynomials of degree $|\mathbb{F}|/2$, we consider polynomials of degree $|E|/2$. The rest of the construction follows the same lines as the one defined above. This way, all of our operations can be implemented in polylogarithmic time in $|E|$.

6 Separations of testing models

In this section we use Theorem 30 to prove a separation between the standard testing model, and both the tolerant and the erasure resilient testing models. Specifically, we prove the following.

▶ **Theorem 46** (Restatement of Theorem 2). *For every constant $\ell \in \mathbb{N}$, there exist a property $\mathcal{Q}^{(\ell)}$ and $\varepsilon_1 = \varepsilon_1(\ell) \in (0,1)$ such that the following hold.*

1. *For every $\varepsilon \in (0,1)$, the property $\mathcal{Q}^{(\ell)}$ can be ε-tested using a number of queries depending only on ε (and ℓ).*
2. *For every $\varepsilon_0 \in (0, \varepsilon_1)$, any $(\varepsilon_0, \varepsilon_1)$-tolerant tester for $\mathcal{Q}^{(\ell)}$ needs to make $\Omega(N/10^\ell \cdot \mathrm{polylog}^{(\ell)} N)$ many queries on inputs of length N.*

▶ **Theorem 47** (Restatement of Theorem 3). *For every constant $\ell \in \mathbb{N}$, there exist a property $\mathcal{Q}^{(\ell)}$ and $\varepsilon_1 = \varepsilon_1(\ell) \in (0,1)$ such that the following hold.*

1. *For every $\varepsilon \in (0,1)$, the property $\mathcal{Q}^{(\ell)}$ can be ε-tested using a number of queries depending only on ε (and ℓ).*
2. *For every $\varepsilon \in (0, \varepsilon_1)$ and any $\alpha = \Omega(1/\log^{(\ell)} N)$ satisfying $\varepsilon + \alpha < 1$, any α-erasure resilient ε-tester for $\mathcal{Q}^{(\ell)}$ needs to make $\Omega(N/10^\ell \cdot \mathrm{polylog}^{(\ell)} N)$ many queries on inputs of length N.*

In order to prove the separation we use the code ensemble $\mathcal{E}^{(\ell)}_{\mathbb{F},k}$ where k is set to 0. Namely, we consider $\mathcal{E}_{\mathbb{F},0}(\emptyset)$. Note that in this case, the code ensemble becomes a property (i.e. a subset of the set of all possible strings).

Next, we define the property that exhibits the separation between the standard testing model and both the tolerant testing model and the erasure resilient model. We prove Theorem 46 and mention the small difference between the proof of Theorem 46 and the proof of Theorem 47.

▶ **Definition 48.** *Fix a finite field $\mathbb{F}$ and a constant integer $\ell \in \mathbb{N}$ and let $\varepsilon(\ell) = \Theta(1/4^\ell)$. Let $n \stackrel{\text{def}}{=} m^{(\ell)}_{\mathbb{F}}$, $z^{(\ell)}_{\mathbb{F},0} \leq n \cdot \mathrm{polylog}^{(\ell)} n$ denote the length of the proof for the* PCUSS *from Theorem 30, and let $N = (\log^{(\ell)} n + 1) \cdot z^{(\ell)}_{\mathbb{F},0}$. Let $\mathcal{Q}^{(\ell)} \subseteq \{0,1\}^N$ be defined as follows. A string $x \in \{0,1\}^N$ satisfies $\mathcal{Q}^{(\ell)}$ if the following hold.*

1. *The first $z^{(\ell)}_{\mathbb{F},0} \cdot \log^{(\ell)} n$ bits of x consist of $s = \frac{z^{(\ell)}_{\mathbb{F},0} \cdot \log^{(\ell)} n}{n}$ copies of $y \in \mathcal{E}^{(\ell)}_{\mathbb{F},0}$.*
2. *The remaining $z^{(\ell)}_{\mathbb{F},0}$ bits of x consist of a proof string $\pi \in \{0,1\}^{z^{(\ell)}_{\mathbb{F},0}}$, for which the* `Verifier-Procedure` $_{\mathcal{E}^{(\ell)}_{\mathbb{F},0}}$ *in Figure 1 accepts y given oracle access to y and π.*

We first show that $\mathcal{Q}^{(\ell)}$ can be tested using a constant number of queries in the standard testing model.

`Testing Algorithm for` $\mathcal{Q}^{(\ell)}$

Input: Parameter $\varepsilon \in (0,1)$, an oracle access to $x \in \{0,1\}^N$.

1. Set $s \stackrel{\text{def}}{=} \frac{z^{(\ell)}_{\mathbb{F},0} \cdot \log^{(\ell)} n}{n}$.
2. Repeat $4/\varepsilon$ times:
   a. Sample $\boldsymbol{j} \in [n]$ and $\boldsymbol{i} \in [s] \setminus \{1\}$ uniformly at random.
   b. If $x_{\boldsymbol{j}} \neq x_{(\boldsymbol{i}-1)\cdot n+\boldsymbol{j}}$, then **Reject**.
3. Let $v = (x_1, \ldots, x_n)$, $\pi = (x_{z^{(\ell)}_{\mathbb{F},0} \cdot \log^{(\ell)} n+1}, \ldots, x_{(\log^{(\ell)} n+1) z^{(\ell)}_{\mathbb{F},0}})$ and τ be the empty string.
4. Run the PCU verifier for $\mathcal{E}^{(\ell)}_{\mathbb{F},0}$ with parameters $\varepsilon/3$ and $\delta = 2/3$ on v, using π as the alleged proof for v, and τ as the alleged value for v.
5. If the PCU verifier rejects, then **Reject**; otherwise **Accept**.

Figure 2 Description of `Testing Algorithm for` $\mathcal{Q}^{(\ell)}$.

For Item 4 in Figure 2, recall that running the PCU verifier with parameter $\delta = 2/3$ actually involves running multiple instances of the verifier with smaller δ, as discussed in Subsection 5.2.

▶ **Lemma 49.** *The property $\mathcal{Q}^{(\ell)}$ has a tester with query complexity depending only on ε.*

Proof. We show that the algorithm described in Figure 2 is a testing algorithm for $\mathcal{Q}^{(\ell)}$. We assume that n is large enough so that $\log^{(\ell)} n > 6/\varepsilon$.

Assume that $x \in \mathcal{Q}^{(\ell)}$. Then, there exists a string $y \in \mathcal{E}^{(\ell)}_{\mathbb{F},0}$ with $x_1, \ldots, x_{z^{(\ell)}_{\mathbb{F},0} \log^{(\ell)} n} = (y)^s$ (where $(y)^s$ is the concatenation of s copies of y), and $x_{z^{(\ell)}_{\mathbb{F},0} \cdot \log^{(\ell)} n+1}, \ldots, x_{(\log^{(\ell)}+1) z^{(\ell)}_{\mathbb{F},0}} = \pi \in \{0,1\}^{z^{(\ell)}_{\mathbb{F},0}}$, where π is a proof that makes the PCU verifier for $\mathcal{E}^{(\ell)}_{\mathbb{F},0}$ accept when given oracle access to y and π. Therefore, the algorithm in Figure 2 accepts x.

Next, assume that x is ε-far from $\mathcal{Q}^{(\ell)}$, and let $y' = x_1, \ldots, x_n$. Note that if the string $x_1, \ldots, x_{z^{(\ell)}_{\mathbb{F},0} \cdot \log^{(\ell)} n}$ is $\varepsilon/2$-far from being $(z')^s$, then the loop in Step 2 rejects x with probability at least $2/3$, and we are done. If $x_1, \ldots, x_{z^{(\ell)}_{\mathbb{F},0} \cdot \log^{(\ell)} n}$ is $\varepsilon/2$-close to $(y')^s$, then y' must be $\varepsilon/3$-far from $\mathcal{E}^{(\ell)}_{\mathbb{F},0}$. To see this, assume toward a contradiction that y' is $\varepsilon/3$-close to $\mathcal{E}^{(\ell)}_{\mathbb{F},0}$. Then, by modifying at most $\frac{\varepsilon \cdot z^{(\ell)}_{\mathbb{F},0} \cdot \log^{(\ell)} n}{2}$ bits, we can make $x_1, \ldots, x_{z^{(\ell)}_{\mathbb{F},0} \cdot \log^{(\ell)} n}$ equal to $(y')^s$. Since, by our assumption y' is $\varepsilon/3$-close to $\mathcal{E}^{(\ell)}_{\mathbb{F},0}$, we can further modify the string $(y')^s$ to $(\tilde{y})^s$, where $\tilde{y} \in \mathcal{E}^{(\ell)}_{\mathbb{F},0}$, by changing at most $\frac{\varepsilon \cdot z^{(\ell)}_{\mathbb{F},0} \cdot \log^{(\ell)} n}{3}$ bits. Finally, by changing at most $z^{(\ell)}_{\mathbb{F},0}$ bits from π, we can get a proof string $\tilde{\pi}$ which will make the PCPP verifier accept $\tilde{y}$. By our assumption that $6/\varepsilon < \log^{(\ell)} n$, the total number of changes to the input string x is at most

$$\frac{\varepsilon \cdot z^{(\ell)}_{\mathbb{F},0} \cdot \log^{(\ell)} n}{2} + \frac{\varepsilon \cdot z^{(\ell)}_{\mathbb{F},0} \cdot \log^{(\ell)} n}{3} + z^{(\ell)}_{\mathbb{F},0} \leq \varepsilon \cdot (\log^{(\ell)} n + 1) \cdot z^{(\ell)}_{\mathbb{F},0} = \varepsilon N,$$

which is a contradiction to the fact that x is ε-far from $\mathcal{E}^{(\ell)}_{\mathbb{F},0}$.

Finally, having proved that y' is $\varepsilon/3$-far from $\mathcal{E}^{(\ell)}_{\mathbb{F},0}$, the PCU verifier for $\mathcal{E}^{(\ell)}_{\mathbb{F},0}$ (when called with parameters $\varepsilon/3$ and $\delta = 2/3$) rejects with probability at least $2/3$. ◀

▶ **Lemma 50.** *For every constant $\ell \in \mathbb{N}$, there exists $\varepsilon_1 \stackrel{\text{def}}{=} \Theta(1/4^\ell)$ such that for every $\varepsilon_0 < \varepsilon_1$, any $(\varepsilon_0, \varepsilon_1)$-tolerant tester for $\mathcal{Q}^{(\ell)}$ needs to make at least $\Omega\left(\frac{N}{10^\ell \cdot \text{polylog}^{(\ell)} N}\right)$ many queries.*

Proof. Fix some constant $\ell \in \mathbb{N}$. The proof follows by a reduction from $2\varepsilon_1$-testing of $\mathcal{E}^{(\ell)}_{\mathbb{F},0}$. Given oracle access to a string $y \in \{0,1\}^n$ which we would like to $2\varepsilon_1$-test for $\mathcal{E}^{(\ell)}_{\mathbb{F},0}$, we construct an input string $x \in \{0,1\}^N$ where $N = (\log^{(\ell)} n + 1) \cdot z^{(\ell)}_{\mathbb{F},0}$ as follows.

$$x \stackrel{\text{def}}{=} (y)^{\frac{z^{(\ell)}_{\mathbb{F},0} \cdot \log^{(\ell)} n}{n}} \sqcup (0)^{z^{(\ell)}_{\mathbb{F},0}}.$$

That is, we concatenate $z^{(\ell)}_{\mathbb{F},0} \cdot \log^{(\ell)} n / n$ copies of y, and set the last $z^{(\ell)}_{\mathbb{F},0}$ bits to 0. Note that a single query to the new input string x can be simulated using at most one query to the string y.

If $y \in \mathcal{E}^{(\ell)}_{\mathbb{F},0}$, then for large enough n we have that x is ε_0-close to $\mathcal{Q}^{(\ell)}$, since the last $z^{(\ell)}_{\mathbb{F},0}$ bits that are set to 0 are less than an ε_0-fraction of the input length.

On the other hand, if $\text{dist}(x, \mathcal{E}^{(\ell)}_{\mathbb{F},0}) > 2\varepsilon_1$, since each copy of y in x is $2\varepsilon_1$-far from $\mathcal{E}^{(\ell)}_{\mathbb{F},0}$, then x is $\frac{2\varepsilon_1 \cdot \log^{(\ell)} n}{\log^{(\ell)} n + 1}$-far from $\mathcal{Q}^{(\ell)}$ (note that $\frac{\log^{(\ell)} n}{\log^{(\ell)} n + 1} > 1/2$). Therefore, an $(\varepsilon_0, \varepsilon_1)$-tolerant tester for $\mathcal{Q}^{(\ell)}$ would imply an $2\varepsilon_1$-tester for $\mathcal{E}^{(\ell)}_{\mathbb{F},0}$ with the same query complexity. By Lemma 40, since for some $\varepsilon_1 = \Theta(1/4^\ell)$, every $2\varepsilon_1$-tester for $\mathcal{E}^{(\ell)}_{\mathbb{F},0}$ requires $\Omega(n/10^\ell)$ queries on inputs of length n, any $(\varepsilon_0, \varepsilon_1)$-tolerant tester for $\mathcal{Q}^{(\ell)}$ requires to make $\Omega\left(\frac{N}{10^\ell \cdot \text{polylog}^{(\ell)} N}\right)$ many queries. ◀

Proof of Theorem 46. The proof follows by combining Lemma 49 and Lemma 50. ◀

Proof of Theorem 47. The proof of Theorem 47 is almost identical to the proof of Theorem 46. The only difference is that we replace Lemma 50 with a counterpart for erasure resilient testing, where instead of setting the last $z^{(\ell)}_{\mathbb{F},0}$ bits of x to $(0)^{z^{(\ell)}_{\mathbb{F},0}}$, we use $(\perp)^{z^{(\ell)}_{\mathbb{F},0}}$, noting that the relative size of this part of the input is $1/(s+1) = \Theta(1/\log^{(\ell)}(N))$. ◀

References

1. Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof Verification and the Hardness of Approximation Problems. *J. ACM*, 45(3):501–555, 1998.
2. Omri Ben-Eliezer and Eldar Fischer. Earthmover Resilience and Testing in Ordered Structures. In *Proceedings of the 33rd Conference on Computational Complexity (CCC)*, pages 18:1–18:35, 2018.
3. Eli Ben-Sasson, Alessandro Chiesa, Ariel Gabizon, Michael Riabzev, and Nicholas Spooner. Interactive Oracle Proofs with Constant Rate and Query Complexity. In *Proceedings of the 44th International Colloquium on Automata, Languages and Programming (ICALP)*, pages 40:1–40:15, 2017.
4. Eli Ben-Sasson, Alessandro Chiesa, and Nicholas Spooner. Interactive Oracle Proofs. In *Theory of Cryptography - 14th International Conference TCC Proceedings, Part II*, pages 31–60, 2016.
5. Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil P. Vadhan. Robust PCPs of Proximity, Shorter PCPs, and Applications to Coding. *SIAM Journal on Computing*, 36(4):889–974, 2006.
6. Eli Ben-Sasson, Yohay Kaplan, Swastik Kopparty, Or Meir, and Henning Stichtenoth. Constant Rate PCPs for Circuit-SAT with Sublinear Query Complexity. *Journal of the ACM*, 63(4):32:1–32:57, 2016.

7 Eli Ben-Sasson and Madhu Sudan. Short PCPs with polylog query complexity. *SIAM Journal on Computing*, 38(2):551–607, 2008.

8 Piotr Berman, Meiram Murzabulatov, and Sofya Raskhodnikova. Tolerant Testers of Image Properties. In *Proceedings of the 43rd International Colloquium on Automata, Languages and Programming (ICALP)*, pages 90:1–90:14, 2016.

9 Eric Blais, Clément L. Canonne, Talya Eden, Amit Levi, and Dana Ron. Tolerant junta testing and the connection to submodular optimization and function isomorphism. In *Proceedings of the 29th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2113–2132, 2018.

10 Andrea Campagna, Alan Guo, and Ronitt Rubinfeld. Local reconstructors and tolerant testers for connectivity and diameter. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 411–424. Springer, 2013.

11 Anindya De, Elchanan Mossel, and Joe Neeman. Junta correlation is testable. In *Proceedings of the 60th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 1549–1563, 2019.

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

13 Irit Dinur and Omer Reingold. Assignment Testers: Towards a Combinatorial Proof of the PCP Theorem. *SIAM Journal on Computing*, 36(4):975–1024, 2006.

14 Kashyap Dixit, Sofya Raskhodnikova, Abhradeep Thakurta, and Nithin M. Varma. Erasure-Resilient Property Testing. *SIAM Journal on Computing*, 47(2):295–329, 2018.

15 Eldar Fischer and Lance Fortnow. Tolerant versus intolerant testing for Boolean properties. *Theory of Computing*, 2(9):173–183, 2006.

16 Eldar Fischer and Ilan Newman. Testing versus estimation of graph properties. *SIAM Journal on Computing*, 37(2):482–501, 2007.

17 Eldar Fischer, Ilan Newman, and Jiří Sgall. Functions that have read-twice constant width branching programs are not necessarily testable. *Random Structures & Algorithms*, 24(2):175–193, 2004.

18 Oded Goldreich. *Computational complexity - A conceptual perspective*. Cambridge University Press, 2008.

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

20 Oded Goldreich and Or Meir. A small gap in the gap amplification of assignment testers, 2007. In ECCC, 2007, TR05-46, Comment 3.

21 Oded Goldreich and Luca Trevisan. Three theorems regarding testing graph properties. *Random Structures & Algorithms*, 23(1):23–57, 2003.

22 Ellis Horowitz. A fast method for interpolation using preconditioning. *Information Processing Letters*, 1(4):157–163, 1972.

23 Swastik Kopparty and Shubhangi Saraf. Tolerant linearity testing and locally testable codes. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM)*, pages 601–614. Springer, 2009.

24 Amit Levi and Erik Waingarten. Lower Bounds for Tolerant Junta and Unateness Testing via Rejection Sampling of Graphs. In *Proceedings of the 10th Innovations in Theoretical Computer Science Conference (ITCS)*, pages 52:1–52:20, 2019.

25 Florence Jessie MacWilliams and Neil James Alexander Sloane. *The theory of error-correcting codes*, volume 16. Elsevier, 1977.

26 Michal Parnas, Dana Ron, and Ronitt Rubinfeld. Tolerant property testing and distance approximation. *Journal of Computer and System Sciences*, 72(6):1012–1042, 2006.

27 Sofya Raskhodnikova, Noga Ron-Zewi, and Nithin M. Varma. Erasures vs. Errors in Local Decoding and Property Testing. In *Proceedings of the 10th Innovations in Theoretical Computer Science Conference (ITCS)*, pages 63:1–63:21, 2019.

28 Omer Reingold, Guy N. Rothblum, and Ron D. Rothblum. Constant-round interactive proofs for delegating computation. In *Proceedings of the 48th ACM Symposium on the Theory of Computing (STOC)*, pages 49–62, 2016.

29 Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.
30 Daniel A. Spielman. Linear-time encodable and decodable error-correcting codes. *IEEE Transactions on Information Theory*, 42(6):1723–1731, 1996.