

Every set in $\mathcal{P}$ is strongly testable under a suitable encoding*

Irit Dinur[†]Oded Goldreich[‡]Tom Gur[§]

July 23, 2026

Abstract

We show that every set in $\mathcal{P}$ is strongly testable under a suitable encoding. By “strongly testable” we mean having a (proximity oblivious) tester that makes a constant number of queries and rejects with probability that is proportional to the distance of the tested object from the property. By a “suitable encoding” we mean one that is distance-preserving and length-regular as well as polynomial-time computable and invertible. This result stands in contrast to the known fact that some sets in $\mathcal{P}$ are extremely hard to test, providing another demonstration of the crucial role of representation in the context of property testing.

The testing result is proved by showing that any set in $\mathcal{P}$ has a *strong canonical PCP*, where canonical means that (for YES-instances) there exists a single proof that is accepted with probability 1 by the system, whereas all other potential proofs are rejected with probability proportional to their distance from this proof. In fact, we show that $\mathcal{UP}$ equals the class of sets having strong canonical PCPs (of logarithmic randomness). Actually, for the testing result, we use a PCP-of-Proximity version of the foregoing notion and an analogous positive result (i.e., strong canonical PCPPs of logarithmic randomness for any set in $\mathcal{UP}$).

*A preliminary version of this work appeared in the proceedings of ITCS 2019.

[†]Supported by ERC-CoG grant 772839. Weizmann Institute of Science, ISRAEL. irit.dinur@weizmann.ac.il

[‡]Supported by the Israel Science Foundation (grant No. 671/13). Weizmann Institute of Science, ISRAEL. oded.goldreich@weizmann.ac.il

[§]Supported by EPSRC award EP/D063191/1. University of Cambridge, UK. tom.gur@cl.cam.ac.uk

Contents

1	Introduction	1
1.1	Our main result: a POT for an encoding of any set in $\mathcal{P}$	1
1.2	The way to our main result: strong canonical PCPPs	4
1.3	Afterthoughts: the role of the encoding	5
1.4	Organization	5
2	Definitions of strong canonical PCPs and PCPPs	6
2.1	Preliminaries: The PCP model	6
2.2	Strong canonical PCPs	6
2.3	Adaptation to the model of PCP of Proximity	7
3	On the existence of strong canonical PCPs and PCPPs	8
4	The testing result	13
5	Directions for further research	15
	References	17
A	The gap amplification reduction is parsimonious	20
A.1	Degree reduction	20
A.2	Expanderization	20
A.3	Powering	20
A.4	Alphabet reduction	21
A.5	Gap amplification for PCPs of Proximity	23

1 Introduction

In the last couple of decades, the area of property testing has attracted much attention (see, e.g., a recent textbook [13]). Loosely speaking, property testing typically refers to super-fast probabilistic algorithms for deciding whether a given object has a predetermined property or is far from any object having this property. Such algorithms, called testers, obtain local views of the object by performing queries; that is, the tested object is modeled as a function and the testers get oracle access to this function (and thus may be expected to work in time that is sub-linear in the size of the object).

It is well known that property testing is very sensitive to the representation of the tested objects, far more so than standard studies in complexity theory (cf. [13, Sec. 1.2.5]). For example, while the adjacency matrix and the incident-list representations are equivalent as far as complexity classes such as $\mathcal{P}$ are concerned, in the case of property testing there is a significant difference between the *adjacency matrix model* (a.k.a. the *dense graph model* [14]) and the *incidence graph model* (a.k.a. the *bounded-degree graph model* [18]).

In this paper we provide another demonstration of the crucial role of representation in the context of property testing. Specifically, in contrast to the known fact that some sets in $\mathcal{P}$ are extremely hard to test (see, e.g., [16, Theorem 7]),¹ we show that, *under a suitable distance-preserving encoding, all sets in $\mathcal{P}$ are extremely easy to test*, where by “extremely easy to test” we mean having a Proximity Oblivious Tester (POT). Furthermore, the encoding is polynomial-time computable and invertible.

1.1 Our main result: a POT for an encoding of any set in $\mathcal{P}$

The standard definition of a property tester refers to randomized oracle machines that are given two parameters as explicit inputs along with oracle access to some string (or function). The two parameters are the *size parameter*, representing the size of the tested object, and a *proximity parameter*, denoted ϵ , which determines which objects are considered *far from the property*² (according to a fixed metric, typically the relative Hamming distance). Specifically, on input parameters n and ϵ , the test is required to distinguish (with constant probability) n -bit long strings that have the property from n -bit long strings that are ϵ -far from the property, where $x \in \{0, 1\}^n$ is ϵ -far from S if for every $x' \in S \cap \{0, 1\}^n$ it holds that $\delta(x, x') \stackrel{\text{def}}{=} |\{i \in [n] : x_i \neq x'_i\}|/n$ is greater than ϵ . (Otherwise, we say that x is ϵ -close to S .)

The query complexity of testers is stated as a function of the two explicit parameters, n and ϵ . Two extreme cases are the case of query complexity n , which can be obtained for any property, and the case that the query complexity depends only on the proximity parameter, which is sometimes considered the yardstick for “easy testability” (see, e.g., [1, 2]). Typically (see [9]), the query complexity is $\Omega(1/\epsilon)$, and so testers of such complexity are extremely efficient. An even more restricted case refers to one in which the tester operates by repeating some constant-query check for $O(1/\epsilon)$ times, where the celebrated linearity tester of Blum, Luby, and Rubinfeld [8] is an archetypical case. The effect of a single repetition of the constant-query check is captured by the notion of a *proximity oblivious tester* [19].

¹This is essentially due to [14, Prop 4.1.1].

²As usual in the area, we associate the notion of having a property with the notion of being in the (set of objects that have the) property.

A Proximity Oblivious Tester (POT) does not obtain a proximity parameter as input, but rather the probability gap with which it distinguishes inputs that have the property from ones that lack the property is allowed to be a function of the distance of the tested input from the property (defined in Eq. (1)). Further restricting ourselves to the case of one-sided error testers, we require that the POT always accepts inputs that have the property and rejects objects that lack the property with probability that increases with the distance of the object from the property.³ For the sake of clarity, we recall that the distance of x from S , denoted $\delta_S(x)$, is defined as follows

$$\delta_S(x) \stackrel{\text{def}}{=} \min_{x' \in \{0,1\}^{|x|} \cap S} \{\delta(x, x')\}, \quad (1)$$

where $\delta_S(x) = 1$ if $\{0,1\}^{|x|} \cap S = \emptyset$.

Definition 1.1 (Proximity Oblivious Testers):⁴ *Let $\varrho: (0, 1] \rightarrow (0, 1]$ be monotonically non-decreasing.⁵ A proximity oblivious tester (POT) with detection probability function ϱ for S is a probabilistic oracle machine, denoted T , that makes a constant number of queries and satisfies the following two conditions.*

1. T always accepts inputs in S : For every $n \in \mathbb{N}$ and every $w \in S \cap \{0,1\}^n$, it holds that $\Pr[T^w(n)=1] = 1$.
2. T rejects inputs that are not in S with probability that increases as a function of their distance from S : For every $n \in \mathbb{N}$ and every $w \in \{0,1\}^n \setminus S$, it holds that $\Pr[T^w(n)=0] \geq \varrho(\delta_S(w))$.

The case that ϱ is linear is of special interest; in this case the rejection probability is proportional to the distance of the input from the set S . (In general, a POT with any ϱ yields a standard (one-sided error) tester: when given proximity parameter $\epsilon > 0$, the tester invokes the POT for $O(1/\varrho(\epsilon))$ times (and accepts iff all invocations accepted).)

Our main result asserts the existence of a POT for some distance-preserving encoding of any set in $\mathcal{P}$. Starting with some natural representation of a set $S \subseteq \{0,1\}^*$, we consider a representation obtained by applying a distance-preserving encoding $E: \{0,1\}^* \rightarrow \{0,1\}^*$, where E is a **distance-preserving encoding** of S if for every w it holds that $\delta_{E(S)}(E(w)) = \Omega(\delta_S(w))$, where $E(S) \stackrel{\text{def}}{=} \{E(x) : x \in S\}$. Furthermore, we consider the natural case in which this encoding is polynomial-time computable and invertible. For example, we may consider an encoding such as $E(x_1 \cdots x_n) = x_1 x_1 \cdots x_n x_n$ or encodings that map graphs in the adjacency matrix representation to the incidence list representation.⁶

³A two-sided error version was also studied (see [20]), but the one-sided error version that we consider here is much better known.

⁴Unlike in [19], which considered POTs of arbitrary query complexity, here we mandate that a POT has constant query complexity. This choice is justified by the fact that our result establishes the existence of such POTs. Ditto regarding our choice to consider one-sided error only.

⁵The postulate that ϱ is monotonically non-decreasing means that *any input that is ϵ -far from S is rejected with probability at least $\varrho(\epsilon)$* ; that is, if $\delta_S(f) \geq \epsilon$ (and not only if $\delta_S(f) = \epsilon$), then f is rejected with probability at least $\varrho(\epsilon)$. This postulate is natural (and it can be enforced in general by redefining $\varrho(\epsilon) \leftarrow \inf_{\delta \geq \epsilon} \{\varrho(\delta)\}$).

⁶Note that an encoding that maps graphs in the incidence list representation to the adjacency matrix representation is not distance-preserving. This is the reason that testers for graphs in the adjacency matrix representation (a.k.a dense graph model) do not yield testers in the bounded-degree graph model.

Theorem 1.2 (a POT for a suitable encoding of any set in $\mathcal{P}$): *For any $S \in \mathcal{P}$ there exists distance-preserving encoding E of S such that the set $S' \stackrel{\text{def}}{=} \{E(x) : x \in S\}$ has a proximity oblivious tester of linear detection probability. Furthermore, the tester runs in polynomial-time, the encoding is length-regular (i.e., $|E(x)| = |E(1^{|x|})|$ for every x), and is polynomial-time computable and invertible. In fact, we present two versions of such encodings.*

1. (E is strictly distance preserving): *For every $x \neq x'$ of the same length, $\delta(E(x), E(x')) = \Theta(\delta(x, x'))$; that is, not only that $\delta(E(x), E(x')) = \Omega(\delta(x, x'))$ holds for every x, x' , but $\delta(E(x), E(x')) = O(\delta(x, x'))$ holds as well.*
2. (E is distance creating): *For every $x \neq x'$ of the same length, $\delta(E(x), E(x')) = \Omega(1)$.*

Actually, the tester runs in polylogarithmic time and has logarithmic randomness complexity.

Recall that POTs were defined as having constant query complexity. We note that at least some subset of added conditions (i.e., the tester running in polynomial-time, E being length-regular and polynomial-time computable) is essential for the meaningfulness of the result (see below). We also note that Theorem 1.2 cannot be significantly extended, since the existence of a polynomial-time tester (of arbitrary query complexity) for $\{E(x) : x \in S\}$ such that E is polynomial-time computable implies that $S \in \mathcal{BPP}$ (and $S \in \mathcal{P}$ follows if the tester has logarithmic randomness complexity).⁷

On the importance of the distance-preservation feature. Note that any property can be made almost-trivial to test under an encoding that is not distance-preserving. In particular, for any set S and an arbitrary monotone 1-t-1 function $\ell : \mathbb{N} \rightarrow \mathbb{N}$, consider the encoding E that maps any x to $x1^{\ell(|x|)-|x|}$. Then, every $y = x1^{\ell(|x|)-|x|}$ is $|x|/|\ell(|x|)|$ -close to $E(S) \stackrel{\text{def}}{=} \{E(x) : x \in S\}$. Consequently, for large ℓ (e.g., $\ell(n) = n^2$), testing $E(S)$ amounts to testing $E(\{0, 1\}^*)$, which totally misses the point of our main result. Specifically, on input $w \in \{0, 1\}^{\ell(n)}$, when given proximity parameter $\epsilon > 0$, if $\epsilon > \sqrt{4/\ell(n)}$, the tester just tests $E(\{0, 1\}^n)$ with proximity parameter $\epsilon/2$, and otherwise it just reads the entire input (which in this case means it makes at most $4/\epsilon^2$ queries). A POT for $E(S)$ can be easily constructed using a weak PCP (as defined at the end of Section 2.1).

On the importance of the length-regularity feature. Dropping the length-preservation requirement allows to easily distinguish the encoding of inputs in S from the encoding of inputs not in S by mapping the former (resp., latter) to odd-length (resp., even-length) strings. This is problematic because the tester gets the length of the input as an explicit input. The problem becomes acute if we encode $x \in S$ by a $(2|x| + 1)$ -bit long binary representation of the index of x in $S \cap \{0, 1\}^n$ and if the tester knows a good approximation of $s(n) \stackrel{\text{def}}{=} |S \cap \{0, 1\}^n|$. (In this case the tester accepts $w \in \{0, 1\}^{2n+1}$ if w represents a number that seems at most $s(n)$.)⁸

⁷The decision procedure maps x to $E(x)$ and invokes the tester with proximity parameter $1/2|E(x)|$. In case E has relative distance δ_E , invoking the tester with proximity parameter $\delta_E/2$ will do.

⁸This refers to a standard tester, which gets a proximity parameter $\epsilon > 0$. On input $w = w_1 \cdots w_{2n+1}$, knowing $s(n)$, it can test that $w_1 = \cdots = w_{2n - \log_2 s(n)} = 0$ and check that $w_{2n - \log_2 s(n) + 1} \cdots w_{2n - \log_2 s(n) / \epsilon}$ encode the top $\log_2(1/\epsilon)$ bits in the binary representation of a number that is at most $s(n)$.

1.2 The way to our main result: strong canonical PCPPs

For the distance-creating version of Theorem 1.2, we use a good error correcting code, denoted C , which has relative constant distance and efficient encoding and decoding algorithms, whereas for the strictly distance-preserving version we let C be the identity function (i.e., $C(x) = x$). In both cases, we encode an input x by a pair of the form $(C(x)^{t(|x|)}, \Pi(x))$, where $\Pi(x)$ is a PCP proof that $C(x) \in \{C(z) : z \in S\}$, and $t(|x|) = |\Pi(x)|$. Hence, in the first case, the encoding has relative distance that is approximately equal to the relative distance of C , whereas in the second case, for every $x, x' \in \{0, 1\}^n$, the relative distance between their encodings almost equals the relative distance between x and x' themselves.

The foregoing strategy, which can be traced back to [21], works only when the PCP system is of a certain type, as discussed next. First, we need a PCP of Proximity (a.k.a. assignment testers), rather than a PCP. The difference is that a PCP of Proximity does not get an explicit (main) input, but rather oracle access to both the main input and the alleged proof. Indeed, PCPs of Proximity can be viewed as a “property testing” variant of PCPs (or “PCP-aided variants” of property testers). Second, valid assertions in these PCPs of Proximity must have *unique* valid proofs, otherwise the mapping $x \mapsto \Pi(x)$ is not even well-defined. Furthermore, the PCP of Proximity should reject (with constant probability) not only inputs (that encode) strings far from S , but also proof-parts that are far from the corresponding (unique) valid proof. Last, to get a POT rather than a tester that works only for constant values of the proximity parameter (i.e., constant $\epsilon > 0$), also inputs and alleged proofs that are close to being valid should be rejected with probability that is related to their distance from a valid object. A PCP of Proximity that satisfies all of these conditions is called *strongly canonical*.⁹

The foregoing aspects were dealt with in [21], but only for the special case of $S = \{0, 1\}^*$, where the issue was to test that the input-part is a valid codeword (with respect to code C). Using a linear code C , this was reduced to the special case in which the set (for which the PCP of Proximity is designed) is a linear space, but even this case was not handled in full generality in [21]. Subsequent work [22, 15, 23] culminated in providing strongly canonical PCPs of Proximity for any linear space, but left open the problem of providing strongly canonical PCPs of Proximity for any set in $\mathcal{P}$, let alone $\mathcal{UP}$.

Recall that the class $\mathcal{UP}$ is defined as the subset of $\mathcal{NP}$ in which each YES-instance has a unique valid proof. In this paper, we show that *every set in $\mathcal{UP}$ has a strong canonical PCP of Proximity*. Furthermore, we provide a polynomial-time transformation of NP-witnesses (with respect to the original NP-witness relation of the set) to valid proofs (for the resulting PCP of Proximity).

We seize the opportunity to study the simpler case of strong canonical PCPs. Loosely speaking, a **strong canonical PCP** for a set S is a PCP system in which each $x \in S$ has a unique valid proof $\Pi(x)$ that is accepted with probability 1, whereas each other alleged proof is rejected with probability that is related to its distance from $\Pi(x)$. We show that *every set in $\mathcal{UP}$ has a strong canonical PCP of logarithmic randomness, and only sets in $\mathcal{UP}$ have such a PCP* (see Theorem 3.1).

Our constructions are obtained in two steps. First, we show that sets in the relevant class have PCP systems in which each string in the set has a unique valid proof (that is accepted with probability 1). Specifically, we show that the only proofs that are accepted with probability 1 by these PCP systems are the images of the standard transformation of NP-witnesses to PCP-oracles. Next, we observe that these PCP systems can be made strongly canonical by a suitable padding of

⁹See Definition 2.2, which requires that the rejection probability of the oracle pair (x, π) be related to the maximum, over all $x' \in \{0, 1\}^{|x|} \cap S$, of $\delta(x, x')$ and $\delta(\pi, \Pi(x'))$.

the proofs. Specifically, the padding is determined such that the ratio of the length of the original proof over the length of the padded proof equals the lower bound on the rejection probability of invalid proofs (under the original PCP). Indeed, this simple observation reduces the construction of strong canonical PCPs to the construction of PCPs that have unique valid proofs.

Focusing on the construction of PCPs that have unique valid proofs (for sets in $\mathcal{UP}$), we note that the original PCP construction of Arora *et al.* [4, 3] will not do. Still, it is possible that this construction can be modified and augmented so that it has unique valid proofs (or even becomes a strong canonical PCP). Such an augmentation was indeed performed by Goldreich and Sudan [21], alas only for the special case of linear spaces, and the route taken there was quite tedious. Hence, we preferred to work with the gap amplification construction of Dinur [10], which is more transparent. Starting with a trivial weak-PCP that has unique valid proofs, we observe that the gap amplification operation is a parsimonious reduction, and so we are done.

Subsequent work. The padding of canonical PCPs that yields strong canonical PCP has an unfortunate side effect of yielding a highly un-“smooth” verifier; that is, the transformation inherently yields a verifier that queries certain locations in the proof with much larger probability than other locations. In a subsequent work, Paradise [24] showed a construction of $O(1)$ -query strong canonical PCPs of polynomial length for any set in $\mathcal{NP}$, which satisfy the smoothness condition; that is, each location in the proof is queried with equal probability. We remark that the smoothness of the strong canonical PCPs is not only of natural appeal, but is necessary for some applications to hardness of approximation (see discussion in [24]).

1.3 Afterthoughts: the role of the encoding

As stressed at the end of Section 1.1 the distance-preserving condition prevents a trivialization of the encoded property (i.e., of $E(S)$). It ensures that if x is far from S , then $E(x)$ is far from $E(S)$, and so testing x for membership in S does not become trivial when moving to the encoded form (i.e., $E(x)$). The distance preservation version of Theorem 1.2 makes testing potentially harder by causing every $x \notin S$ to be mapped to $E(x)$ that is far from $E(S)$. In contrast, the strictly distance-preserving version prevents this effect.

In general, there is a tension between constraining the encoding and the ease of testing the resulting property. As noted upfront, there are properties in $\mathcal{P}$ that are extremely hard to test, hence an encoding that satisfies Theorem 1.2 cannot be trivial (e.g., we cannot use $E(x) = x$). On the other hand, allowing any encoding yields meaningless results (as indicated above). We also note that dropping any computational restriction allows using the Long-Code, under which every property has a proximity-oblivious tester [5].

1.4 Organization

In Section 2 we recall the definitions of strong canonical PCPs and PCPPs, starting with the basic PCP model. In Section 3 we characterize the classes of sets having strong canonical PCPs of certain types (see Theorems 3.1 and 3.2), and obtain analogous PCPP systems (see Theorem 3.3). The latter PCPP systems will be used in Section 4 towards establishing Theorem 1.2. We conclude by spelling out some directions for further research (see Section 5).

2 Definitions of strong canonical PCPs and PCPPs

Throughout this work, we follow the standard convention by which the verifiers and testers are probabilistic polynomial-time oracle machines. We stress that in the context of PCPP and property testing the main input, with respect to which complexities are measured, is accessed via queries (and only its length is given as an explicit input).

In this section, we recall the definitions of strong canonical PCPs and PCPPs. Essentially, we follow the definitional approach presented in [21, Sec. 5.3] (while correcting an error in one of the actual definitions [21, Def. 5.7]).

2.1 Preliminaries: The PCP model

We start by recalling the basic definition of Probabilistically Checkable Proofs (PCPs): These are randomized verification procedures that are given an explicit input and oracle access to an alleged proof π , and are aimed to verify the membership of the (main) input in a predetermined set by making few (random) queries to the proof (see, e.g., [12, Sec. 9.3]). Specifically, for a predetermined set $S \subseteq \{0, 1\}^*$, on input x and oracle access to an alleged proof π , a PCP verifier V reads x , makes a constant number of random queries to the proof π , and satisfies the following conditions.

Completeness: If $x \in S$, then there exists a valid proof π such that V always accepts x when given oracle access to π ; that is, $\Pr[V^\pi(x)=1] = 1$.

Soundness: If $x \notin S$, then for every string π , with probability at least $1/2$ the verifier V rejects x when given oracle access to π ; that is, $\Pr[V^\pi(x)=0] \geq 1/2$.¹⁰

Indeed, a string π that makes V always accept x (i.e., that satisfies $\Pr[V^\pi(x)=1] = 1$) is called a **valid proof for x** ; the soundness condition implies that valid proofs exist only for members of S , and the completeness condition asserts that each member of S has a valid proof. We stress that it is not necessarily the case that the valid proofs are unique; that is, the same $x \in S$ may have several valid proofs (with respect to a fixed verifier).

Weak-PCPs. We shall also refer to the notion of a *weak-PCP*, which is defined as above with the crucial exception that its soundness condition is extremely weak. Specifically, this weak soundness condition only requires that for every $x \notin S$ and π , with positive probability, the verifier rejects x when given oracle access to π (i.e., $\Pr[V^\pi(x)=0] > 0$). Indeed, an oracle machine that on input a 3CNF and oracle access to a truth assignment to its variables checks the values assigned to the variables of a uniformly selected clause constitutes such a trivial weak-PCP. (Recall that Dinur's construction [10], which we shall use, gradually transforms such a weak-PCP into a full-fledged PCP.)

2.2 Strong canonical PCPs

We focus on the special case of PCP verifiers, for a set S , with respect to which each $x \in S$ has a *unique* valid proof, and call such verifiers **canonical**. Furthermore, we are interested in the case that invalid proofs are not merely rejected with positive probability, but are rather rejected

¹⁰Actually, the constant $1/2$ can be replaced by any other constant in $(0, 1)$.

with probability that is related to their distance from the (unique) valid proof. We shall call such verifiers **strongly canonical**, and quantify their strength by a function ϱ that relates their rejection probability to the latter distance. Details follow.

We denote the empty string by λ . For two strings $w, w' \in \{0, 1\}^m$, we let $\delta(w, w')$ denote the relative Hamming distance between w and w' ; that is, $\delta(w, w') = |\{i \in [m] : w_i \neq w'_i\}|/m$. For the sake of convenience, we define $\delta(w, w') = 1$ if w and w' have different lengths (e.g., the distance between a non-empty string and the empty string is 1).

Definition 2.1 (strong canonical PCPs): *For a set $S \subseteq \{0, 1\}^*$, a monotonically non-decreasing function $\varrho: [0, 1] \rightarrow [0, 1]$ such that $\varrho(\alpha) = 0$ if and only if $\alpha = 0$, and an oracle machine V , we say that V is a ϱ -strong canonical PCP for S if V makes a constant number of queries to the oracle and there exist functions $\ell: \mathbb{N} \rightarrow \mathbb{N}$ and $\Pi: \{0, 1\}^* \rightarrow \{0, 1\}^*$ such that the following conditions hold.*

- **Canonical Completeness:** *For every $x \in S$, it holds that $\Pi(x) \in \{0, 1\}^{\ell(|x|)}$, and the verifier always accepts x when given oracle access to $\Pi(x)$; that is, $\Pr[V^{\Pi(x)}(x) = 1] = 1$.*
- **Strong Canonical Soundness:** *For every $x \in \{0, 1\}^*$ and $\pi \in \{0, 1\}^*$, the verifier rejects x when given access to the oracle π with probability at least $\varrho(\delta(\pi, \Pi(x)))$, where $\Pi(x) \stackrel{\text{def}}{=} \lambda$ if $x \notin S$ (and in this case $\delta(\pi, \Pi(x)) = 1$); that is, $\Pr[V^\pi(x) = 0] \geq \varrho(\delta(\pi, \Pi(x)))$.*

The function ϱ is called V 's detection probability function, and ℓ is called its proof complexity. We say that V is a strong canonical PCP for S if, for some ϱ as above, V is a ϱ -strong canonical PCP for S .

Indeed, the foregoing conditions assert that $\Pi(x)$ is the unique valid proof for $x \in S$, and that the verifier is strongly canonical with strength ϱ . Note that strong canonical soundness implies (standard) soundness by the convention that $\varrho(\delta(\pi, \Pi(x))) = \varrho(1) = \Omega(1)$ for $x \notin S$. More generally, recall that $\delta(\pi, \Pi(x)) = 1$ if $|\pi| \neq |\Pi(x)|$. The case that ϱ is linear is of special interest; in this case invalid proofs are rejected with probability that is proportional to their distance from the valid proof.

2.3 Adaptation to the model of PCP of Proximity

Probabilistically checkable proofs of proximity (PCPs of Proximity, abbreviated PCPPs and a.k.a. assignment testers) are proof systems in which the verifier has oracle access to both its main input and an alleged proof, and is required to decide whether the main input is in some predetermined set or is far from any string that is in this set (cf. [6, 11]). We call such a PCPP system **strong** if it rejects every NO-instance with probability that is related to the distance of the instance from the predetermined set. For simplicity, when we say a PCPP system, we mean a strong one.

Analogously to the case of PCPs, we consider **strong canonical PCPs of Proximity**¹¹ (henceforth scPCPs of Proximity), which are PCPs of Proximity in which every statement has a unique valid proof such that a statement–proof pair is rejected with probability that is related to its distance from a true statement and its corresponding unique valid proof. The actual definition builds on Definition 2.1, while adapting it to the proofs of proximity model.

¹¹Alternatively, we use the term *strongly canonical*.

Definition 2.2 (strong canonical PCPs of Proximity): For a set S , a function ϱ as in Definition 2.1, and an oracle machine V that accesses two oracles, we say that V is a ϱ -strong canonical PCP of Proximity for S if V makes a constant number of queries to each of its oracles and there exist functions $\ell: \mathbb{N} \rightarrow \mathbb{N}$ and $\Pi: \{0, 1\}^* \rightarrow \{0, 1\}^*$ such that the following conditions hold.

- **Canonical Completeness:** For every $x \in S$, it holds that $\Pi(x) \in \{0, 1\}^{\ell(|x|)}$ and the verifier always accepts the pair of oracles $(x, \Pi(x))$; that is, $\Pr[V^{x, \Pi(x)}(1^{|x|}) = 1] = 1$.
- **Strong Canonical Soundness:** For every $x \in \{0, 1\}^*$ and $\pi \in \{0, 1\}^*$, the verifier rejects the pair of oracles (x, π) with probability at least $\varrho(\delta_\Pi(x, \pi))$, where¹²

$$\delta_\Pi(x, \pi) \stackrel{\text{def}}{=} \min_{x' \in \{0, 1\}^{|x|}} \{ \max(\delta(x, x'), \delta(\pi, \Pi(x'))) \}; \quad (2)$$

that is, $\Pr[V^{x, \pi}(1^{|x|}) = 0] \geq \varrho(\delta_\Pi(x, \pi))$.

The function ϱ is called V 's detection probability function, and ℓ is called its proof complexity. We say that V is a strong canonical PCPP for S if, for some ϱ as above, V is a ϱ -strong canonical PCPP for S .

We stress that the rejection probability depends on the distance of the oracle-pair (x, π) from a valid pair consisting of $x' \in S \cap \{0, 1\}^{|x|}$ and the corresponding valid proof $\Pi(x')$, where the distance between pairs is defined as the maximum of the distance between the corresponding elements.¹³ This represents the fact that we wish to reject with probability that not only depends on the distance of the input x to a string $x' \in S$, but also depends on the distance of the alleged proof π to the corresponding valid proof $\Pi(x')$. Indeed, proximity oblivious testers (POTs) can be viewed as strong canonical PCPs of Proximity with proof complexity zero.

3 On the existence of strong canonical PCPs and PCPPs

Our first result is a characterization of the class of sets having strong canonical PCPs of *logarithmic randomness*. It turns out that this class equals $\mathcal{UP}$. Recall that the class $\mathcal{UP}$ is defined as the subset of $\mathcal{NP}$ in which each YES-instance has a unique valid proof; that is, $S \in \mathcal{UP}$ if there exists a polynomially-bounded relation R that is recognizable in polynomial-time such that for every $x \in S$ there exists a unique $w \in R(x) = \{y : (x, y) \in R\}$ whereas $R(x) = \emptyset$ if $x \notin S$.

Theorem 3.1 ($\mathcal{UP}$ and strong canonical PCPs): *The set S has a strong canonical PCP of logarithmic randomness if and only if $S \in \mathcal{UP}$. Furthermore, the resulting PCP is ϱ -strong for $\varrho(\alpha) = \alpha/4$ and there exists a polynomial-time transformation of NP-witnesses for $S \in \mathcal{UP}$ to valid proofs for the resulting PCP.*

Proof: The necessary condition is quite straightforward: Let V be a strong canonical PCP of logarithmic randomness for S , and assume for simplicity that its proof complexity is polynomial. Now, define $R = \{(x, \pi) : \Pr[V^\pi(x) = 1] = 1\}$, and observe that membership in R can be decided in

¹²Recall that $\Pi(x') \stackrel{\text{def}}{=} \lambda$ if $x' \notin S$, and in this case $\delta(\pi, \Pi(x')) = 1$.

¹³That is, we effectively define $\delta(\langle x, y \rangle, \langle x', y' \rangle)$ as $\max(\delta(x, x'), \delta(y, y'))$. Taking the sum of the latter distances (or their average) would have been as good, since $\frac{\alpha + \beta}{2} \leq \max(\alpha, \beta) \leq \alpha + \beta$.

polynomial-time by trying all possible random choices of V . Hence, $S = \{x : R(x) \neq \emptyset\}$ is in $\mathcal{NP}$, and the hypothesis that the valid proofs (with respect to V) are unique implies that $S \in \mathcal{UP}$. In the general case (i.e., when the proof length may be super-polynomial), one may consider the “effective proofs” (i.e., the values of π at locations that are read by V on some random choices). That is, in this case, we define $R = \{(x, (I(x), \pi_{I(x)})) : \Pr[V^\pi(x) = 1] = 1\}$, where $I(x)$ is the set of locations that are in the “effective proof” (i.e., locations that $V(x)$ probes with positive probability).

Note that the foregoing argument holds also for very weak PCP systems, provided that they have logarithmic randomness complexity and unique valid proofs. That is, we only used the hypothesis that for every $x \in S$, there exists a unique π such that $p_x(\pi) \stackrel{\text{def}}{=} \Pr[V^\pi(x) = 1] = 1$, and capitalized on the fact that it is feasible to compute $p_x(\pi)$ exactly.

Turning to the opposite direction, we show that each $S \in \mathcal{UP}$ has a strong canonical PCP of logarithmic randomness by presenting such a PCP for **USAT** and recalling that each set in $\mathcal{UP}$ is reducible to **USAT** via a parsimonious reduction (see, e.g., [12, Ex 2.29]). Recall that **USAT** is the promise problem in which YES-instances are 3CNF formulas with a *unique* satisfying assignment and NO-instances are formulas with no satisfying assignments. (Actually, we need to define PCPs for promise problems and state, as well as prove, Proposition 3.1.1 in this more general setting, but we avoid doing so while commenting that the extension is straightforward.)¹⁴

The key observation is that it suffices to show a PCP for S in which each $x \in S$ has a unique valid proof. This is the case because such PCPs can be transformed into strong canonical ones, as stated next.

Proposition 3.1.1 (deriving strong canonical PCPs from PCPs with unique valid proofs): *Let V be a PCP system of logarithmic randomness complexity for S , and suppose that for every $x \in S$ there exists a unique π such that $\Pr[V^\pi(x) = 1] = 1$. Then, there exists strong canonical PCP of logarithmic randomness for S and a polynomial-time transformation of valid proofs with respect to V to valid proofs for the resulting PCP. Furthermore, the resulting PCP is ϱ -strong for $\varrho(\alpha) = \alpha/4$ and its proof complexity is $2^r \cdot \ell$, where r and ℓ are the randomness and proof complexity of V .*

Proof: Again, we may assume, without loss of generality, that V has polynomial proof complexity, since we can efficiently determine all relevant locations (i.e., those queried under any choice of randomness) without making any queries.

Letting Π be the function mapping instances to their canonical proofs, as in Definition 2.1, we define $\Pi'(x) = 1^{(2^{r(|x|)} - 1) \cdot \ell(|x|)} \Pi(x) \in \{0, 1\}^{2^{r(|x|)} \cdot \ell(|x|)}$ if $\Pi(x) \neq \lambda$ and $\Pi'(x) = \lambda$ otherwise. Note that, for every $x \in S$ and $\pi \in \{0, 1\}^{\ell(|x|)}$, it holds that

$$\delta(1^{(2^{r(|x|)} - 1) \cdot \ell(|x|)} \pi, \Pi'(x)) \leq 2^{-r(|x|)},$$

which means that invalid proofs for $x \in S$ are extremely close to valid proofs, and so it suffices to reject them with tiny probability. This suggests the following verifier, which on input $x \in \{0, 1\}^n$ and access to oracle $\pi' \in \{0, 1\}^{2^{r(n)} \cdot \ell(n)}$, selects uniformly at random one of the following two tests and performs it.

1. The verifier selects at random $i \in [(2^{r(n)} - 1) \cdot \ell(n)]$, and accepts if and only if the i^{th} bit of π' equals 1.

¹⁴Specifically, the canonical soundness condition has to be satisfied only for inputs that satisfy the promise, whereas the canonical completeness condition is stated for the YES-instances only.

2. The verifier invokes V on input x , while providing it with oracle access to the $\ell(n)$ -bit long suffix of π' , and outputs the verdict of V .

Turning to the analysis, we first note that if $x \notin S$, then (by virtue of V) the resulting verifier rejects x with probability at least $\frac{1}{2} \cdot \frac{1}{2}$, regardless of the identity of the oracle π' . Hence, from this point on we assume $x \in S$, and let $\Pi(x)$ denote the unique valid proof with respect to V .

Now, let $\pi' \equiv (w, \pi) \in \{0, 1\}^{2^{r(n)} \cdot \ell(n)}$ such that $|\pi| = \ell(n)$. Observe that, on input x and access to π' , the new verifier rejects with probability that is lower-bounded by (half) the fraction of 0's in w , since with probability $1/2$ this verifier tests whether the $(2^{r(n)} - 1) \cdot \ell(n)$ -bit long prefix equals the all-1 string. Next, recall that, for any $\pi \in \{0, 1\}^{\ell(n)}$, it holds that $\pi'' = 1^{(2^{r(n)} - 1) \cdot \ell(n)} \pi$ is $2^{-r(|x|)}$ -close to $\Pi'(x) = 1^{(2^{r(n)} - 1) \cdot \ell(n)} \Pi(x)$, since $\delta(\pi, \Pi(x)) \leq 1$. Hence, rejecting π'' with probability at least $\frac{1}{2} \cdot 2^{-r(n)}$ suffices when $\pi \neq \Pi(x)$. It follows that a generic $\pi' \equiv (w, \pi) \neq \Pi'(x) \neq \lambda$ is rejected with probability

$$\begin{aligned} \frac{1}{2} \cdot \delta(w, 1^{(2^{r(n)} - 1) \cdot \ell(n)}) + \frac{1}{2} \cdot 2^{-r(n)} &\geq \frac{1}{2} \cdot \delta(w, 1^{(2^{r(n)} - 1) \cdot \ell(n)}) + \frac{1}{2} \cdot 2^{-r(n)} \cdot \delta(\pi, \Pi(x)) \\ &\geq \frac{1}{2} \cdot \delta(\pi', \pi'') + \frac{1}{2} \cdot \delta(\pi'', \Pi'(x)), \end{aligned}$$

where the second inequality uses $\delta(w, 1^{(2^{r(n)} - 1) \cdot \ell(n)}) \geq \delta(w\pi, 1^{(2^{r(n)} - 1) \cdot \ell(n)} \pi) = \delta(\pi', \pi'')$ and

$$\begin{aligned} \delta(\pi, \Pi(x)) &= 2^{r(|x|)} \cdot \delta(1^{(2^{r(n)} - 1) \cdot \ell(n)} \pi, 1^{(2^{r(n)} - 1) \cdot \ell(n)} \Pi(x)) \\ &= 2^{r(|x|)} \cdot \delta(\pi'', \Pi'(x)). \end{aligned}$$

Using $\delta(\pi', \pi'') + \delta(\pi'', \Pi'(x)) \geq \delta(\pi', \Pi'(x))$, it follows that, on input x and access to π' , the new verifier rejects with probability at least $\delta(\pi', \Pi'(x))/2$, which means that it constitutes a strong canonical PCP for S . Indeed, the PCP is ϱ -strong for $\varrho(\alpha) = \alpha/4$.¹⁵ ■

In light of Proposition 3.1.1, it suffices to show a PCP of logarithmic randomness and unique valid proofs for **USAT**. This PCP is constructed by merely following the construction of Dinur [10], while noting that her gap amplification transformation is a parsimonious reduction.

Proposition 3.1.2 (PCPs with unique valid proofs for **USAT**): *There exists a PCP system of logarithmic randomness for **USAT** such that for every satisfiable formula there exists a unique valid proof with respect to this system. Furthermore, there exists a polynomial-time transformation of satisfying assignments for the input formula to valid proofs for the resulting PCP.*

Proof: Let ψ be an m -clause 3CNF formula over n variables, promised to have at most one satisfying assignment. Let V_0 be the trivial weak-PCP system with soundness $1/m$, in which the oracle is allegedly the unique satisfying assignment of ψ , and the verifier checks that this assignment satisfies a random clause of ψ . By construction, V_0 has unique valid proofs. Applying the gap amplification transformation of Dinur [10] to V_0 , we obtain a PCP system V of logarithmic randomness for **USAT**.

As stated above, the crucial point is showing that the aforementioned transformation is a parsimonious reduction. The argument is detailed in Appendix A; it consists of showing that gap amplification is a one-to-one transformation, and that the only valid proofs with respect to the resulting proof system are those in the range of the transformation. These facts are demonstrated

¹⁵The factor of $1/4$ is due to the case that $x \notin S$, which is rejected with probability at least $1/4$.

by closely inspecting each of the four steps in the gap amplification procedure: degree reduction, “expanderization”, powering, and alphabet reduction. We show that each of these steps satisfies the two aforementioned properties, where in the analysis of the alphabet reduction step we assume that it is performed by composition with a PCPP that has unique valid proofs. Such a PCPP is immediately implied by the Hadamard code (alternatively, by the long code); see details at the end of Appendix A.4. ■

Combining Proposition 3.1.1 and 3.1.2, the theorem follows. ■

A detour: Removing the upper-bound on the randomness complexity. A natural question that arises in light of Theorem 3.1 refers to the power of strong canonical PCPs of polynomial proof length (but with no bound on their randomness complexity). We observe that only sets in “unambiguous- $\mathcal{MA}$ ” (denoted $\mathcal{UM\!A}$, and defined next) have such *poly*-strong PCPs (i.e., ϱ -strong canonical PCPs of polynomial proof length for $\varrho(\alpha) = \text{poly}(\alpha)$). Recall that the class $\mathcal{MA}$ consists of all sets having a non-interactive probabilistic proof system; that is, $S \in \mathcal{MA}$ if there exists a polynomially-bounded relation R that is recognizable in $\text{co}\mathcal{RP}$ such that $S = \{x : \exists w (x, w) \in R\}$.¹⁶ We define $\mathcal{UM\!A}$ as the subset of $\mathcal{MA}$ in which the non-interactive proof system has unique valid proofs; that is, $S \in \mathcal{UM\!A}$ if there exists a polynomially-bounded relation R that is recognizable in $\text{co}\mathcal{RP}$ such that for every $x \in S$ there exists a unique $w \in R(x) = \{y : (x, y) \in R\}$, whereas $R(x) = \emptyset$ if $x \notin S$. (Note that in this case $|R(x)| \leq 1$ for every x .)

Theorem 3.2 ($\mathcal{UM\!A}$ and strong canonical PCPs):¹⁷ *The set S has a poly-strong canonical PCP of polynomial proof complexity only if $S \in \mathcal{UM\!A}$.*

Recall that by poly-strong we mean ϱ -strong with respect to $\varrho(\alpha) = \text{poly}(\alpha)$. We stress that, unlike in Theorem 3.1, here we do not know whether a ϱ -strong canonical PCP with arbitrary ϱ for S implies that $S \in \mathcal{UM\!A}$. The point is that invalid proofs of length ℓ are only guaranteed to be rejected with probability at least $\varrho(1/\ell)$, which may be negligible.

Proof: Following the outline of the proof of the relevant part of Theorem 3.1, we face the fact that we are no longer guaranteed that the randomness complexity is logarithmic. This implies that we can no longer afford trying all possible random choices of the verifier. Instead, using the hypothesis that the PCP is poly-strong, we infer that invalid proofs are rejected with noticeable probability (i.e., probability at least $\varrho(1/\ell) = \text{poly}(1/\ell)$). This fact allows for the rejection of invalid proofs by invoking the PCP verifier polynomially many times. Using the (explicit) hypothesis that the PCP has polynomial proof complexity, we obtain a verification procedure that places the set in $\mathcal{UM\!A}$ (rather than in $\mathcal{UP}$). Specifically, let V be a ϱ -strong canonical PCP of proof complexity $\ell = \text{poly}$ for S , and define $R = \{(x, \pi) : \Pr[V^\pi(x) = 1] = 1\}$. Then, $R \in \text{co}\mathcal{RP}$, by letting the decision procedure emulate $O(1/\varrho(1/\ell(|x|))) = \text{poly}(|x|)$ executions of $V^\pi(x)$, and accept if and only if all executions accepted. Hence, $S = \{x : R(x) \neq \emptyset\}$ is in $\mathcal{MA}$, and the hypothesis that the valid proofs (with respect to V) are unique implies that $S \in \mathcal{UM\!A}$. ■

¹⁶This perfect completeness version of $\mathcal{MA}$ equals the non-perfect one in which R is only required to be recognizable in $\mathcal{BPP}$ (see [12, Ex. 6.12 (2)]).

¹⁷Prior versions of this paper claimed that the necessary condition is also sufficient, but the claimed proof had a fundamental gap.

Strong canonical PCPs of Proximity. Next, we adapt the proof of the positive direction of Theorem 3.1 to the PCPP model.

Theorem 3.3 ($\mathcal{UP}$ and strong canonical PCPPs): *Every set in $\mathcal{UP}$ has a strong canonical PCP of Proximity of logarithmic randomness and linear detection probability function. Furthermore, there exists a polynomial-time transformation of NP-witnesses for membership in the set to valid proofs for the resulting PCP.*

Indeed, the positive direction of Theorem 3.1 follows from Theorem 3.3 by applying the latter to the set $S' = \{C(x) : x \in S\}$, where C is a good error correcting code. Note that the claimed PCP system (for S) emulates the input-oracle of the PCPP system (for S') by applying C to its own input x , and emulating the proof-oracle of the PCPP system by using its own proof-oracle. The canonical soundness of the PCP system (for S) follows from the canonical soundness of the PCPP system (for S'), since in the case that $x \notin S$ it holds that the relative distance of $C(x)$ from the set S' is a constant.

Proof: The construction and its analysis are analogous to those in the proof of Theorem 3.1, except that here we start with a trivial weak-PCPP for the set $S \in \mathcal{UP}$, use (parsimonious) gap amplification for PCPPs (see Appendix A.5), and apply a PCPP version of Proposition 3.1.1. Details follow.

Starting with the PCPP analogue of Proposition 3.1.2, we use a similar construction except that we apply it to a fixed 3CNF (which is generated based on the input length only). Recall that the construction consists of two steps: First, we construct a trivial weak-PCPP with unique proofs, and then we apply the gap amplification procedure to it (obtaining a PCPP with unique proofs).

Specifically, in the first step, we reduce the verification of the claim $x \in S$ to the satisfiability of a fixed 3CNF by an assignment that extends x , where the formula is derived by the standard Cook–Levin reduction of S to 3SAT. The fixed formula has main variables X (which are set by the assignment x) and auxiliary variables Y (which represent the NP-witness for x as well as intermediate gate-values in the corresponding computation), and the question is whether this formula is satisfiable by an assignment in which $X = x$. (Note that when $x \in S$ there is a unique assignment y to Y such that the assignment $(X, Y) = (x, y)$ satisfies the fixed formula.) The first step is completed by observing that the foregoing formula yields a trivial weak-PCPP (with small but noticeable soundness) that is given oracle access to the input x (i.e., x is the input-oracle) as well as to a proof that corresponds to an assignment to the auxiliary variables. This PCPP has unique valid proofs.

In the second step, we apply the gap amplification procedure, which treats the foregoing PCPP execution as a 2CSP instance such that some variables of the 2CSP are identified with the bits of the input-oracle and the other variables represent various auxiliary values. The set of variables that represents bits of the input-oracle will remain intact throughout the entire process of gap amplification (see Appendix A.5). Hence, when viewing the resulting 2CSP as a PCPP, the input-oracle of the resulting PCPP equals the input-oracle of the original (trivial) PCPP. Observing (as in the proof of Proposition 3.1.2) that the gap amplification process maintains the number of valid proofs for each input (see Appendix A.5), we obtain a PCPP with unique proofs for S .

Next, we turn to establish a PCPP analogue of Proposition 3.1.1. This version asserts a transformation of PCPPs with unique proofs to strong canonical PCPPs, and its proof is obtained by a straightforward adaptation of the original (PCP) version.¹⁸ Using the notation of Proposi-

¹⁸Alternatively, the current version can be derived as a special case of Proposition 5.3.

tion 3.1.1, the crux of the analysis is that the pair of oracles (x, π') , where $x \in \{0, 1\}^n$ (such that $S \cap \{0, 1\}^n \neq \emptyset$)¹⁹ and $\pi' \equiv (w, \pi) \in \{0, 1\}^{(2^{r(n)}-1) \cdot \ell(n) + \ell(n)}$, is rejected with probability at least

$$\min_{x' \in S \cap \{0, 1\}^n} \{ \max(\Omega(\delta(x, x')), 0.5 \cdot \delta(w, 1^{(2^{r(n)}-1) \cdot \ell(n)}) + 0.5 \cdot 2^{-r(|x|)} \cdot \delta(\pi, \Pi(x')))) \}, \quad (3)$$

where the foregoing lower bound of $\Omega(\delta(x, x'))$ follows from the soundness of the original PCPP system outlined above. As shown in the proof of Proposition 3.1.1, it holds that $\delta(w, 1^{(2^{r(n)}-1) \cdot \ell(n)}) + 2^{-r(|x|)} \cdot \delta(\pi, \Pi(x')) \geq \delta(\pi', \Pi'(x'))$, which implies that Eq. (3) is lower-bounded by $\Omega(\delta_\Pi(x, \pi'))$. ■

4 The testing result

With strong canonical PCPs of Proximity (as provided by Theorem 3.3) at our disposal, it is quite straightforward to obtain a proximity oblivious tester for a suitable encoding of any set in $\mathcal{P}$. Such an encoding incorporates copies of the target object as well as a corresponding PCPP-oracle that attests its membership in the set. As stated in the introduction, to be meaningful, this encoding should be distance-preserving and better be “length regular” (i.e., equal length strings have an equal encoding length).

Theorem 1.2 (restated): *For every $S \in \mathcal{P}$, there exists distance-preserving encoding E of S such that the set $S' \stackrel{\text{def}}{=} \{E(x) : x \in S\}$ has a proximity oblivious tester of linear detection probability. Furthermore, the tester runs in polylogarithmic-time, $|E(x)| = |E(1^{|x|})|$ for every x , and E is polynomial-time computable and invertible. In fact, we present two versions of such encodings.*

1. (E is strictly distance preserving): $\delta(E(x), E(x')) = \Theta(\delta(x, x'))$, for every $x \neq x'$.
2. (E is distance creating): $\delta(E(x), E(x')) = \Omega(1)$, for every $x \neq x'$.

Moreover, the tester has logarithmic randomness complexity.

Recall that proximity oblivious testers (POTs) were defined as having constant query complexity, and that their detection probability function represents a lower bound on their rejection probability as a function of the distance of the tested object from the property.

Proof: For the distance-creating version of Theorem 1.2, we use a good error correcting code, denoted C ; that is, let $C: \{0, 1\}^* \rightarrow \{0, 1\}^*$ be an arbitrary (length-regular) code of constant relative distance such that the mapping $x \mapsto C(x)$ can be computed and inverted in polynomial time. In particular, efficient inversion follows easily if the code is systematic code (i.e., x is a prefix of $C(x)$). For the strictly distance-preserving version we let C be the identity function (i.e., $C(x) = x$). In both cases, we define $C(S) \stackrel{\text{def}}{=} \{C(x) : x \in S\}$, and note that $C(S)$ is in $\mathcal{P}$.

We use a strong canonical PCPP for the set $C(S) \in \mathcal{P}$, as guaranteed by Theorem 3.3, and let $\Pi(C(x))$ denote the (unique) valid proof for $C(x) \in C(S)$. The basic idea is to combine $C(x)$ and $\Pi(C(x))$ into a single codeword that will be accessed by the POT as an oracle. The POT will emulate the PCPP system, on input-oracle $C(x)$ and proof-oracle $\Pi(C(x))$, in order to verify that $C(x)$ is indeed in $C(S)$. The distance-preserving feature easily follows if the input (i.e., $C(x)$) is

¹⁹If $S \cap \{0, 1\}^n = \emptyset$, then (x, π') is rejected with probability at least $\Omega(1)$, and the claim follows (since in this case $\delta_\Pi(x, \pi') = 1$).

much longer than the proof (i.e., $\Pi(x)$), alas the proof-oracle in PCPP is typically longer than the input-oracle. This problem is solved by repeating the $C(x)$ sufficiently many times.

Recalling that $|\Pi(C(x))| = \ell(|C(x)|)$ for some polynomial ℓ , we proceed to present the claimed algorithms.

The encoding function E : On input $x \in \{0, 1\}^*$, we let $n = |C(x)|$ and $t = \ell(n)$. Then, $E(x) = C(x)^t \pi$ such that $\pi = \Pi(C(x))$ if $x \in S$, and $\pi = 1^{\ell(n)}$ otherwise.

The encoding can be computed in polynomial time, since the canonical proof $\Pi(C(x))$ can be computed in polynomial-time because $C(S) \in \mathcal{P}$. (Formally, the reader may think of S as being in $\mathcal{UP}$ by virtue of the witness relation $R = \{(x, x) : x \in S\}$, and recall that given an NP-witness one can efficiently obtain the corresponding proof-oracle.)

Note that for every $x, x' \in \{0, 1\}^k$ such that $|C(1^k)| = n$, it holds that

$$\begin{aligned} \delta(E(x), E(x')) &= \frac{t \cdot n \cdot \delta(C(x), C(x'))}{tn + \ell(n)} + \frac{\ell(n) \cdot \delta(\Pi(x), \Pi(x'))}{tn + \ell(n)} \\ &= \frac{n}{n+1} \cdot \delta(C(x), C(x')) + \frac{\delta(\Pi(x), \Pi(x'))}{n+1} \end{aligned}$$

whereas $x \neq x'$ implies that $\delta(C(x), C(x')) \geq 1/n$, which is larger than $\frac{\delta(\Pi(x), \Pi(x'))}{n+1}$. Depending on our choice of C , this establishes either the distance-creating feature or the strictly distance-preserving feature.

The decoding function D : On input $y \in \{0, 1\}^*$, the algorithm outputs x if $y = C(x)^{\ell(n)} \Pi(C(x))$, and outputs a special failure symbol otherwise. Specifically, the algorithm first determines n such that $|y| = (n+1) \cdot \ell(n)$, then tries to recover x from the n -bit prefix of y , and outputs x if and only if $E(x) = y$.

The proximity-oblivious tester T : On input $y \in \{0, 1\}^{(n+1) \cdot \ell(n)}$, the tester parses y into $(w_1, \dots, w_t, \pi)$ such that $|w_1| = \dots = |w_t| = n$ and $|\pi| = \ell(n)$, where $t = \ell(n)$. First, T tests that the w_i 's are all identical to w_1 , by selecting a random $i \in [t]$ and comparing the value of a random location in w_i to the value of this location in w_1 . Next, T invokes the (strong canonical) PCPP verifier, providing it with access to the input-oracle w_1 and the proof-oracle π .

We first note that $D(E(x)) = x$ for every x . Next, we show that T is a POT for the set $S' = \{E(x) : x \in S\}$. On the one hand, for every $y \in S'$, it holds that $y = E(x) = C(x)^t \Pi(C(x))$ for some $x \in S$, and the tester T accepts y with probability 1 (by virtue of the perfect completeness of the PCPP verifier). On the other hand, turning to the case of $y \notin S'$ and letting $y \equiv (w_1, \dots, w_t, \pi) \in \{0, 1\}^{t \cdot n + \ell(n)}$, we consider three cases (where below, by “far” we mean being at relative distance $\Omega(\delta_{S'}(y))$).

1. If $z \stackrel{\text{def}}{=} (w_1, \dots, w_t)$ is far from w_1^t , then y is rejected with proportional probability by the first check of T .
2. If (z is close to w_1^t but) w_1 is far from $C(S)$, then y is rejected with proportional probability by the (strong canonical) PCPP verifier (which is invoked with input-oracle w_1).
3. If w_1 is close to some $C(x) \in C(S)$ (e.g., $w_1 = C(x)$) but π is far from the canonical proof $\Pi(C(x))$, then y is rejected with proportional probability by the (strong canonical) PCPP verifier (which is invoked with input-oracle w_1 and the proof-oracle π).

We stress that one of the three cases must hold, since otherwise we derive a contradiction to the definition of $\delta_{S'}(y)$. (That is, if z is δ_1 -close to w_1^t , and w_1 is δ_2 -close to $C(x)$ and π is δ_3 -close to $\Pi(C(x))$, then y is $(\delta_1 + \delta_2 + \delta_3)$ -close to S' .) Hence, T is a POT (of linear detection probability) for S' . ■

5 Directions for further research

The natural question is whether a result like Theorem 1.2 can be proved when using an encoding function of smaller stretch, where the stretch of $E: \{0,1\}^* \rightarrow \{0,1\}^*$ is the function that maps n to $|E(1^n)|$. Specifically, which sets S satisfy the conclusion of Theorem 1.2 with respect to an encoding of almost linear stretch?

Recalling that our proof of Theorem 1.2 is pivoted at the existence of strong canonical PCPs of Proximity, it is natural to ask which sets have a strong canonical PCP of Proximity of almost-linear proof complexity. We believe that **USAT** has such a PCP of Proximity, and suggest establishing this conjecture as an open problem.

Problem 5.1 (almost-linear length strong canonical PCPPs): *Show that **USAT** has a strong canonical PCP of Proximity of almost-linear proof complexity. Furthermore, show that valid proofs for this PCPP can be constructed in polynomial-time when given the input formula and a satisfying assignment to it.*

Recall that **3SAT** has a PCP of Proximity of almost-linear proof complexity: We refer to the PCPP system of Dinur [10], which builds upon the work of Ben-Sasson and Sudan [7]. A possible route towards resolving Problem 5.1 is to show that this PCPP is a strong canonical PCPP, or can be transformed into such a PCPP at moderate cost (i.e., increasing the proof complexity by a polylogarithmic factor). We actually believe that such a transformation is needed, since we believe that the PCPP of [10] is almost there (i.e., it satisfies a relaxed form of being strongly canonical (detailed below)), and that the extra step can be made by a generalization (of a PCPP version) of Proposition 3.1.1.

In order to detail this idea, we need a refinement of Definition 2.2. In this refinement, the rejection probability is not lower-bounded by a function of the maximum of the distances $\delta(x, x')$ and $\delta(\pi, \Pi(x))$, but is rather the maximum of two (potentially) different functions applied to the two distances. Maybe more importantly, we allow these functions to depend also on the input length.

Definition 5.2 (Definition 2.2, refined): *Let $\varrho_I, \varrho_P : \mathbb{N} \times [0, 1] \rightarrow [0, 1]$ be monotonically non-decreasing in their second argument such that $\varrho_I(n, \alpha) = 0$ (resp., $\varrho_P(n, \alpha) = 0$) if and only if $\alpha = 0$. For a set $S \subseteq \{0, 1\}^*$ and an oracle machine V that accesses two oracles, we say that V is a (ϱ_I, ϱ_P) -strong canonical PCP of Proximity for S if V makes a constant number of queries to each of its oracles and there exist functions $\ell : \mathbb{N} \rightarrow \mathbb{N}$ and $\Pi : \{0, 1\}^* \rightarrow \{0, 1\}^*$ such that the following conditions hold.*

- **Canonical Completeness:** *As in Definition 2.2.*²⁰

²⁰That is, for every $x \in S$, it holds that $\Pi(x) \in \{0, 1\}^{\ell(|x|)}$ and the verifier always accepts the pair of oracles $(x, \Pi(x))$; i.e., $\Pr[V^{x, \Pi(x)}(1^{|x|}) = 1] = 1$.

- **Strong Canonical Soundness:** For every $x \in \{0, 1\}^*$ and $\pi \in \{0, 1\}^{\ell(|x|)}$, the verifier rejects the pair of oracles (x, π) with probability at least

$$\min_{x' \in \{0, 1\}^{|x|}} \left\{ \max(\varrho_I(|x|, \delta(x, x')), \varrho_P(|x|, \delta(\pi, \Pi(x')))) \right\}. \quad (4)$$

We say that V is a **strong canonical PCP of Proximity** for S if both ϱ_I and ϱ_P are oblivious of the length parameter (i.e., if $\varrho_I(n, \alpha) = \varrho_I'(\alpha)$ for some $\varrho_I' : [0, 1] \rightarrow [0, 1]$, and ditto for ϱ_P), and say that V is a **semi-strong canonical PCP of Proximity** for S if ϱ_I is oblivious of the length parameter. As in Definition 2.2, ℓ is called the **proof complexity** of V .

Indeed, Definition 2.2 is obtained as a special case by letting $\varrho_I(n, \alpha) = \varrho_P(n, \alpha) = \varrho(\alpha)$ for $\varrho : [0, 1] \rightarrow [0, 1]$. We conjecture that the PCPP system of Dinur [10], which builds on the work of Ben-Sasson and Sudan [7], yields a semi-strong canonical PCP of Proximity of logarithmic randomness and almost linear proof complexity for **USAT**, and that the corresponding function ϱ_P has the form $\varrho_P(n, \alpha) = \alpha / \text{poly}(\log n)$. If this is indeed the case, then using the following transformation, which generalizes Proposition 3.1.1, yields a strong canonical PCP of Proximity of almost-linear proof complexity for **USAT**, which in turn resolves Problem 5.1.

Proposition 5.3 (deriving strong canonical PCPPs from semi-strong ones): *Let V be a (ϱ_I, ϱ_P) -strong canonical PCPP system of logarithmic randomness complexity and proof complexity ℓ for S , and suppose that $\varrho_I(n, \alpha) = \varrho(\alpha) \leq \alpha$ and $\varrho_P(n, \alpha) = \varrho'_P(n) \cdot \alpha$ for some $\varrho'_P : \mathbb{N} \rightarrow (0, 1]$. Then, there exists a $0.5 \cdot \varrho$ -strong canonical PCP of logarithmic randomness and proof complexity ℓ / ϱ'_P for S . Furthermore, there exists a polynomial-time transformation of valid proofs with respect to V to valid proofs for the resulting PCPP.*

Note that the PCPP analogue of Proposition 3.1.1 follows as a special case by observing that any canonical PCPP system (i.e., one that has unique valid proofs) of randomness complexity r is an (ϱ_I, ϱ_P) -strong canonical PCPP, where ϱ_I is oblivious of the length parameter and $\varrho_P(n, \alpha) = 2^{-r(n)}$.

Proof: We observe that the proof of Proposition 3.1.1 can be adapted and generalized as follows. Again, we may assume, without loss of generality, that V has polynomial proof complexity, and let $t(n) = 1 / \varrho'_P(n)$. Letting Π be as in Definition 2.2, we define $\Pi'(x) = 1^{(t(n)-1) \cdot \ell(|x|)} \Pi(x)$ if $\Pi(x) \neq \lambda$, and $\Pi'(x) = \lambda$ otherwise. Analogously to the proof of Proposition 3.1.1, we consider the following verifier, which given oracle access to an input $x \in \{0, 1\}^n$ and an alleged proof $\pi' \in \{0, 1\}^{t(n) \cdot \ell(n)}$, selects uniformly at random one of the following two tests and performs it.

1. The verifier selects at random $i \in [(t(n) - 1) \cdot \ell(n)]$, and accepts if and only if the i^{th} bit of π' equals 1.
2. The verifier invokes V on input x and the $\ell(n)$ -bit long suffix of π' . That is, V 's queries to the input-oracle are answered by the input-oracle of the new verifier, whereas V 's queries to the proof-oracle are answered by accessing the suffix of the proof-oracle of the new verifier (i.e., query $i \in [\ell(n)]$ is answered by querying the location $(t(n) - 1) \cdot \ell(n) + i$ in π').

Turning to the analysis, we first note that if $S \cap \{0, 1\}^n = \emptyset$, then (x, π') is rejected with probability at least $\delta(1) = \Omega(1)$, and the claim follows. Hence, we may assume that $S \cap \{0, 1\}^n \neq \emptyset$. Letting

$\pi' \equiv (w, \pi) \in \{0, 1\}^{(t(n)-1) \cdot \ell(n) + \ell(n)}$, we infer that the pair of oracles $(x, \pi') \in \{0, 1\}^n \times \{0, 1\}^{t(n) \cdot \ell(n)}$ is rejected with probability at least

$$\min_{x' \in S \cap \{0, 1\}^n} \{ \max(\varrho(\delta(x, x')), 0.5 \cdot \delta(w, 1^{(t(n)-1) \cdot \ell(n)}) + 0.5 \cdot \varrho'_P(n) \cdot \delta(\pi, \Pi(x'))) \}.$$

Observing that

$$\begin{aligned} & \delta(w, 1^{(t(n)-1) \cdot \ell(n)}) + \varrho'_P(n) \cdot \delta(\pi, \Pi(x')) \\ & \geq \delta(w\pi, 1^{t(n)-1 \cdot \ell(n)}\pi) + \varrho'_P(n) \cdot t(n) \cdot \delta(1^{t(n)-1 \cdot \ell(n)}\pi, 1^{t(n)-1 \cdot \ell(n)}\Pi(x')) \\ & = \delta(\pi', 1^{t(n)-1 \cdot \ell(n)}\pi) + \delta(1^{t(n)-1 \cdot \ell(n)}\pi, \Pi'(x')) \\ & \geq \delta(\pi', \Pi'(x')), \end{aligned}$$

it follows that (x, π') is rejected with probability at least

$$\min_{x' \in S \cap \{0, 1\}^n} \{ \max(\varrho(\delta(x, x')), 0.5 \cdot \delta(\pi', \Pi'(x'))) \}.$$

Using $\varrho(\alpha) \leq \alpha$, it follows that the new verifier is a $0.5 \cdot \varrho$ -strong canonical PCPP of proof complexity ℓ / ϱ'_P . ■

Acknowledgments

The third author wishes to thank Igor Shinkar for helpful discussions regarding the parsimony of the gap amplification reduction. We are grateful to the reviewers for their useful comments.

References

- [1] N. Alon, E. Fischer, M. Krivelevich and M. Szegedy. Efficient Testing of Large Graphs. *Combinatorica*, Vol. 20, pages 451–476, 2000.
- [2] N. Alon, E. Fischer, I. Newman, and A. Shapira. A Combinatorial Characterization of the Testable Graph Properties: It's All About Regularity. In *38th ACM Symposium on the Theory of Computing*, pages 251–260, 2006.
- [3] S. Arora, C. Lund, R. Motwani, M. Sudan and M. Szegedy. Proof Verification and Intractability of Approximation Problems. *Journal of the ACM*, Vol. 45, pages 501–555, 1998. Preliminary version in *33rd FOCS*, 1992.
- [4] S. Arora and S. Safra. Probabilistic Checkable Proofs: A New Characterization of NP. *Journal of the ACM*, Vol. 45, pages 70–122, 1998. Preliminary version in *33rd FOCS*, 1992.
- [5] M. Bellare, O. Goldreich and M. Sudan. Free Bits, PCPs and Non-Approximability – Towards Tight Results. *SIAM Journal on Computing*, Vol. 27, No. 3, pages 804–915, 1998. Extended abstract in *36th FOCS*, 1995.
- [6] E. Ben-Sasson, O. Goldreich, P. Harsha, M. Sudan, and S. Vadhan. Robust PCPs of Proximity, Shorter PCPs, and Applications to Coding. *SIAM Journal on Computing*, Vol. 36 (4), pages 889–974, 2006. Extended abstract in *36th STOC*, 2004.

- [7] E. Ben-Sasson and M. Sudan. Short PCPs with polylog query complexity. *SIAM Journal on Computing*, Vol. 38 (2), pages 551–607, 2008. Preliminary Version in *37th STOC*, 2005.
- [8] M. Blum, M. Luby and R. Rubinfeld. Self-Testing/Correcting with Applications to Numerical Problems. *Journal of Computer and System Science*, Vol. 47, No. 3, pages 549–595, 1993. Extended abstract *22nd STOC*, 1990.
- [9] N.H. Bshouty and O. Goldreich. On Properties that are Non-trivial to Test. In *Computational Complexity and Local Algorithms*, LNCS 1570, pages 383–387, 2025.
- [10] I. Dinur. The PCP Theorem by Gap Amplification. *Journal of the ACM*, Vol. 54 (3), Art. 12, 2007. Extended abstract in *38th STOC*, 2006.
- [11] I. Dinur and O. Reingold. Assignment-testers: Towards a combinatorial proof of the PCP-Theorem. Extended abstract in *45th FOCS*, 2004.
- [12] O. Goldreich. *Computational Complexity: A Conceptual Perspective*. Cambridge University Press, 2008.
- [13] O. Goldreich. *Introduction to Property Testing*. Cambridge University Press, 2017.
- [14] O. Goldreich, S. Goldwasser, and D. Ron. Property testing and its connection to learning and approximation. *Journal of the ACM*, pages 653–750, July 1998. Extended abstract in *37th FOCS*, 1996.
- [15] O. Goldreich, T. Gur, and I. Komargodski. Strong Locally Testable Codes with Relaxed Local Decoders. In *30th Conference on Computational Complexity*, pages 1–41, 2015.
- [16] O. Goldreich, M. Krivelevich, I. Newman, and E. Rozenberg. Hierarchy Theorems for Property Testing. *Computational Complexity*, Vol. 21(1), pages 129–192, 2012.
- [17] O. Goldreich and O. Meir. A small gap in the gap amplification of assignment testers. Comment 3 on ECCC Technical Report TR05-046, Oct. 2007.
- [18] O. Goldreich and D. Ron. Property testing in bounded degree graphs. *Algorithmica*, pages 302–343, 2002. Extended abstract in *29th STOC*, 1997.
- [19] O. Goldreich and D. Ron. On Proximity Oblivious Testing. *SIAM Journal on Computing*, Vol. 40, No. 2, pages 534–566, 2011. Extended abstract in *41st STOC*, 2009.
- [20] O. Goldreich and I. Shinkar. Two-Sided Error Proximity Oblivious Testing. *Random Structures and Algorithms*, Vol. 48 (2), pages 341–383, 2016.
- [21] O. Goldreich and M. Sudan. Locally testable codes and PCPs of almost-linear length. *Journal of the ACM*, Vol. 53 (4), pages 558–655, 2006. Extended abstract in *43rd FOCS*, 2002.
- [22] T. Gur and R. Rothblum. Non-interactive proofs of proximity. *ECCC*, TR13-078, 2013.
- [23] T. Gur, G. Ramnarayan, and R. Rothblum. Relaxed Locally Correctable Codes. In *9th ITCS*, pages 27:1–27:11, 2018.

- [24] O. Paradise. Smooth and Strong PCPs. ECCC Technical Report TR19-023, Feb. 2019.
- [25] R. Rubinfeld and M. Sudan. Robust characterization of polynomials with applications to program testing. *SIAM Journal on Computing*, Vol. 25(2), pages 252–271, 1996.
- [26] M. Viderman. Explicit strong LTCs with inverse poly-log rate and constant soundness. *ECCC*, TR15-020, 2015.

A The gap amplification reduction is parsimonious

Throughout this section we assume familiarity with [10]. Following [10] (see also [12, Sec. 9.3.2.3]), the PCP is presented as a 2-variable CSP over a constant non-Boolean alphabet Σ , which in turn is presented as a graph with constraints (i.e., predicates over Σ^2) associated with its edges. The gap amplification procedure in [10] consists of four steps that are repeated iteratively: (1) degree reduction, (2) “expanderization”, (3) powering, and (4) alphabet reduction. We recall these steps and argue that each one of them preserves the number of satisfying assignments that are accepted with probability 1. To this end, we show that each one of the foregoing steps is a one-to-one proof transformation such that there are no valid proofs for the resulting system except those in the range of the transformation. Finally, we also show how to extend this argument to the setting of PCPs of proximity.

A.1 Degree reduction

Let $G = \langle (V, E), \Sigma, \mathcal{C} \rangle$ be a constraint graph with vertices V , edges E , and constraints $\mathcal{C}$ over alphabet Σ . For $d = O(1)$, a d -regular constraint graph G_1 is obtained from G as follows. Denote by $\deg_G(v)$ the degree of $v \in V$ in G . Each vertex $v \in V$ is replaced by an $(d-1)$ -regular expander graph H_v with $\deg_G(v)$ vertices, where the edges correspond to equality constraints. Then, for every edge $(u, v) \in E$, the expander graphs H_u and H_v are connected by one edge that inherits its constraint from (u, v) . (The alphabet remains the same.)

Let φ be the natural transformation that maps a satisfying assignment $\alpha: V \rightarrow \Sigma$ of G into a satisfying assignment $\alpha_1: \cup_{v \in V} H_v \rightarrow \Sigma$ of G_1 by assigning values to each vertex of H_v in G_1 according to the assignment of v in G ; that is, for every $v \in V$ and $h_v \in H_v$, we have $\alpha_1(h_v) = \alpha(v)$.

Clearly, φ is one-to-one, since it is a systematic transformation (i.e., there exists $S \subseteq \cup_{v \in V} H_v$ such that $\varphi(\alpha)|_S = \alpha$). To see that every satisfying assignment of G_1 is in the range of φ , observe that (1) the connectivity and equality constraints of each expander H_v in G_1 imply that each satisfying assignment must assign all vertices of H_v the same value, denoted a_v , and (2) the constraints on the edges connecting each monochromatic H_u and H_v assert that the corresponding values of a_u and a_v must satisfy the constraint associated with the edge (u, v) in G .

A.2 Expanderization

Let $G_1 = \langle (V, E), \Sigma, \mathcal{C} \rangle$ be a $O(1)$ -regular constraint graph. The graph G_1 is transformed to an expander with self-loops G_2 as follows. The vertices and alphabet remain intact. A self-loop is added to each vertex, and in addition, the edges of an $O(1)$ -regular expander graph on V are added to G_2 . The constraints on the edges in E remain unchanged, and the constraints on the new edges are all trivial (i.e., are always satisfied). Since this transformation only adds edges that are always satisfied, it trivially preserves the number of satisfying assignments.

A.3 Powering

For $d = O(1)$, let $G_2 = \langle (V, E), \Sigma, \mathcal{C} \rangle$ be a d -regular expander constraint graph with self-loops. For a parameter $t = O(1)$, the amplified graph G_3 is obtained by raising G_2 to the power of t . Hence, the vertices remain intact, and the new edges correspond to walks on G_2 of length at most t . The alphabet is extended such that the assignment to each vertex v assigns values to all vertices that are within $t/2$ steps from v in G_2 , hereafter called the $t/2$ -neighborhood of v (in G_2). The

constraints are defined as follows: For each new edge, corresponding to a walk of length at most t on G_2 starting at vertex u and ending at v , the new constraint requires that all vertices that are in the $t/2$ -neighborhood of both u and v are assigned consistently (in u and v) and that these assignments satisfy the corresponding G_2 -constraints (of the edges in G_2).

Let φ be the natural transformation that maps a satisfying assignment $a_2: V \rightarrow \Sigma$ of G_2 to a satisfying assignment $a_3: V \rightarrow \Sigma^{d^{t/2}}$ of G_3 , by letting each vertex assign values to its $t/2$ -neighborhood according to a_2 (i.e., $a_3(v)|_u = a_2(u)$ for every v and every u in v 's $t/2$ -neighborhood). The transformation φ is one-to-one since it is a systematic map (we obtain a_2 by restricting $\varphi(a_2)$ such that each vertex only assigns a value to itself).

It remains to prove that every satisfying assignment of G_3 is in the range of φ . Letting $a_3: V \rightarrow \Sigma^{d^{t/2}}$ be a satisfying assignment of G_3 , set $a_2: V \rightarrow \Sigma$ to be the assignment in which each vertex is assigned a value according to its own assignment in a_3 (i.e., $a_2(v) = a_3(v)|_v$), and note that a_2 satisfies G_2 . We show that $a_3 = \varphi(a_2)$. Suppose otherwise (i.e., that there exists vertices u, v such that u is in v 's $t/2$ -neighborhood and $a_3(v)|_u \neq \varphi(a_2)(v)|_u = a_2(u)$). However, since $a_3(u)|_u = a_2(u)$, this violates the constraint corresponding to the edge (u, v) in G_3 , which mandates that the value assigned to u by u equals the value assigned to u by v (i.e., $a_3(u)|_u = a_3(v)|_u$), whereas we have $a_3(u)|_u = a_2(u)$ and $a_3(v)|_u \neq a_2(u)$.

A.4 Alphabet reduction

Let $G_3 = \langle (V, E), \Sigma, \mathcal{C} \rangle$ be a constraint graph, where $|\Sigma| = O(1)$. Let Σ_0 be an alphabet, where $|\Sigma_0|$ is a *universal constant*. The alphabet reduction procedure generates a constraint graph G_4 over alphabet Σ_0 by composing G_3 with a PCPP as follows.

Let $C: \Sigma \rightarrow \{0, 1\}^\ell$ be an arbitrary code with relative distance $\delta > 0$ and $\ell = O(1)$. We replace each vertex $v \in V$ by a “cloud” of ℓ vertices, which we denote by $[v]$, and consider binary assignments to these new vertices. For every edge $(u, v) \in E$ we add a cloud of $\ell' = O(1)$ vertices (where ℓ' will be determined later), which we denote by $[(u, v)]$, and consider assignments over Σ_0 to these new vertices. We transform each constraint $c \in \mathcal{C}$ and its corresponding edge $(u, v) \in E$ as follows.

Let S_c be the set of pairs of assignments to the vertices in the clouds $[u]$ and $[v]$ that are valid C -encodings of assignments to vertices u and v that satisfy the constraint c . Consider a non-adaptive 2-query PCPP system for S_c , with respect to proximity parameter $\delta' < \delta$, answer alphabet Σ_0 , and with unique valid proofs for valid statements (i.e., for every, and only for, $x \in S_c$ there exists a unique proof π_x that is accepted with probability 1). Denote this PCPP's proof length by ℓ' , and note that $\ell' = O(1)$ since the proof refers to statements of length $\ell = O(1)$. The corresponding PCPP verifier naturally induces a constraint graph over $O(1)$ vertices (i.e., the vertices in $[u]$, $[v]$, and $[(u, v)]$) and alphabet Σ_0 , whose edges and constraints we add to G_4 . It is easy to verify that the Hadamard-based PCPP in [3] (with standard query reduction) is a PCPP system that satisfies the foregoing conditions (see discussion at the end of the subsection).

We show that G_3 and G_4 have the same number of satisfying assignments. Let φ be the natural mapping of a satisfying assignment $a_3: V \rightarrow \Sigma$ of G_3 to a satisfying assignment $a_4: (\cup_{v \in V} [v]) \cup (\cup_{(u, v) \in E} [(u, v)]) \rightarrow \Sigma_0$ of G_4 , by assigning to each $[v]$ -cloud in G_4 the C -encoding of $a_3(v)$, and to each $[(u, v)]$ -cloud the assignment corresponding to the unique PCPP oracle that asserts the assignment to $[u]$ and $[v]$ is in S_c .

By construction, and since C is one-to-one, the function φ is one-to-one as well. We show that every satisfying assignment of G_4 is in the range of φ . Let $a_4: (\cup_{v \in V} [v]) \cup (\cup_{(u, v) \in E} [(u, v)]) \rightarrow \Sigma_0$

be a satisfying assignment of G_4 . We show a satisfying assignment $a_3 : V \rightarrow \Sigma$ of G_3 such that $a_4 = \varphi(a_3)$.

Since the PCPP accepts only valid statements with probability 1, the assignment a_4 must assign valid C -encodings to each $[v]$ -cloud in G_4 . Let $a_3 : V \rightarrow \Sigma$ be the assignment that gives each v in G_3 the value that is encoded by $[v]$ in G_4 (i.e., $a_3(v) = C^{-1}(a_4([v]))$ for every $v \in V$, where C^{-1} is well-defined because C is one-to-one and the assignment to each $[v]$ is a valid C -encoding). Since the PCPP has unique valid proofs (i.e., for each valid statement there exists a unique proof that is accepted with probability 1), every $[(u, v)]$ -cloud must be assigned with this unique PCPP proof. Therefore $a_4 = \varphi(a_3)$.

On the features of the Hadamard-based PCPP. We briefly recall the construction of the Hadamard-based PCPP for quadratic equations over $\text{GF}(2)$, due to Arora et al. [3]. We assume familiarity with this construction (see, e.g., [12, Sec. 9.3.2.1] for a detailed exposition), and argue that it has unique valid proofs for valid statements. The other required features of this PCPP (i.e., having a non-adaptive verifier, constant query complexity and alphabet size, perfect completeness, and constant soundness) are well established. We remark that since we actually need a 2-query PCPP, we will also apply a standard query reduction transformation.²¹

Recall that for every system of quadratic equations over n variables, the corresponding Hadamard PCPP verifier has a hard-coded description of the system and query access to a purported satisfying assignment $\alpha = (\alpha_1, \dots, \alpha_n)$ (the statement) and to a PCPP proof-oracle that consists of two parts: (1) Hadamard encoding of a satisfying assignment $\beta = (\beta_1, \dots, \beta_n)$, and (2) Hadamard encoding of the sequence of all 2-way products $\beta_i \beta_j$.

Checking that α , the assignment presented in the PCPP input-oracle, satisfies the given quadratic system reduces to (1) checking that the proof-oracle properly encodes some string β , (2) checking that a random linear combination of the equations is satisfied by β , and (3) checking that β equals α . Specifically, for (1), we locally test the Hadamard encoding used in both parts of the proof-oracle, and check their consistency (by employing self-correction on the second part). For (2), we check that a random linear combination of the equations is satisfied by the assignment encoded in the second part of the proof-oracle. Finally, for (3), we check that the input-oracle α is consistent with β by sampling bits of α and using self-correction on the first

To see that this PCPP system has unique valid proofs (for valid statements), observe that Steps (1) and (3) in the foregoing description accept with probability 1 if and only if all the corresponding local conditions hold, where the conjunction of these local conditions mandate that the proof-oracle encodes $\beta = \alpha$ (i.e., its first part is the Hadamard encoding of β and its second part is the Hadamard encoding of the matrix $(\beta_i \beta_j)_{i,j \in [n]}$).

²¹The standard transformation of PCPPs with query complexity $q = 0(1)$ to PCPPs with query complexity 2 preserves the number of oracles that are accepted with probability 1. To see this, recall that this transformation augments the PCPP oracle with each of the $(q\text{-symbol})$ local views of the oracle that the verifier may see (when making the corresponding q queries), where each local view is written as a single symbol over a larger alphabet. The new verifier reads the (large) symbol corresponding to the local view it seeks, checks that the original verifier would have accepted it, and in addition compares this large symbol to one of the corresponding original symbols (selected at random). Since the verifier checks the acceptability of the large symbols and their consistency with the original symbols, only a sequence of large symbols that is consistent with the unique valid proof for the original verifier would be accepted with probability 1.

A.5 Gap amplification for PCPs of Proximity

We first recall how the gap amplification transformation is extended to the setting of PCPs of proximity (closely following [10]).²² Recall that a PCPP has two oracles (an input-oracle and a proof-oracle) and the conversion to 2-variable CSPs calls for introducing vertices for the symbols of both oracles. Given a PCPP system, we first view it as a 2-variable CSP over a constant non-Boolean alphabet Σ , represented as a constraint graph $G = \langle (X \cup V, E), \Sigma, \mathcal{C} \rangle$, where the *vertices in X correspond to the main input* (the statement) *and the vertices in V correspond to the proof oracle*.

The issue is that the standard (PCP) gap amplification does *not* preserve the “input vertex set” X . To resolve this, the idea is to perform the first three steps of gap amplification that we outlined above, and then add a copy of (the unmodified) X to the vertices of the constraint graph that we obtain, and add a test that the assignment to X is consistent with the assignment to the rest of the graph. Alphabet reduction is then performed essentially as before. Details follow.

We apply the degree reduction, expanderization, and powering stages to G , as before, obtaining a constraint graph G_3 over a larger (constant-size) alphabet. Recall that (by the degree reduction step) the vertices V_3 of G_3 are obtained by replacing each vertex $u \in X \cup V$ in G with an expander graph H_u of $\deg_G(u)$ vertices in G_3 , and hence $X \cap V_3 = \emptyset$. Also recall that the assignments to vertices in G_3 correspond to assignments to $t/2$ -neighborhoods in G_2 . Then, we transform G_3 to a new constraint graph G'_3 by adding the vertex set X and randomly selected “consistency edges” with projection constraints between the vertices in X and V_3 (i.e., such a constraint mandates that the Σ -assignment to $x \in X$ fits the $\Sigma^{d^{t/2}}$ -assignment of any vertex v that contains x within its $t/2$ -neighborhood). We stress that this randomized procedure connects each vertex $x \in X$ to at least one vertex in the corresponding expander H_x of G_3 (and hence of G'_3).²³

Finally, we perform alphabet reduction, essentially the same as in Appendix A.4, except for the following issue. Recall that during the PCP alphabet reduction, each vertex v of the constraint graph is replaced with a “cloud” of vertices whose assignment is the encoding of the assignment to v by a binary code. However, in the setting of PCPPs, we cannot perform any modification to the “input vertex set” X . This is resolved by noting that the alphabet size of the assignment to vertices in X did not grow, and so we can perform alphabet reduction as before, except that instead of replacing each $x \in X$ with a cloud of vertices $[x]$, we keep X intact and emulate each query to a vertex in $[x]$ by querying x , computing the encoding of the assignment to x and answering with the relevant bit.

Note that the transformation from G_3 to G'_3 is the only difference between PCP and PCPP gap amplification that is relevant to arguing that the number of satisfying assignments is preserved. The argument is straightforward, we present it next. Consider the natural transformation φ that maps a satisfying assignment $\alpha: V_3 \rightarrow \Sigma^{d^{t/2}}$ of G_3 into a satisfying assignment $\alpha': X \cup V_3 \rightarrow \Sigma \cup \Sigma^{d^{t/2}}$ of G'_3 by assigning $\alpha'(v) = \alpha(v)$ for every $v \in V_3$, and $\alpha'(x)$ with the “ x -assignment” of α to an arbitrary vertex in H_x (i.e., $\alpha'(x) = \alpha(v)|_x$ for any $v \in H_x$); note that this is well-defined, since as argued in Appendices A.1, A.2, and A.3, every satisfying assignment of G_3 assigns each $v \in V_3$ the same value in all the vertices in its $t/2$ -neighborhood (i.e., for every satisfying assignment $\alpha: V_3 \rightarrow \Sigma^{d^{t/2}}$, every $v \in V_3$ and u, w in the $t/2$ -neighborhood of v , it holds that $\alpha(u)|_v = \alpha(w)|_v$).

²²As will be clarified below, the correction of [17] may be ignored here, let alone that it need not be applied at all at the current context in which the constraint graphs are regular (with respect to the “input variables”).

²³We do not present the exact procedure that generates these consistency edges (see [10, 17] for a precise description), as the details are irrelevant to our argument.

Clearly, φ is one-to-one, since it is a systematic transformation. To see that every satisfying assignment of G'_3 is in the range of φ , observe that the assignment of X is forced by the assignment to V_3 .