SIAM J. COMPUT.
Vol. 38, No. 2, pp. 551–607

SHORT PCPS WITH POLYLOG QUERY COMPLEXITY*

ELI BEN-SASSON† AND MADHU SUDAN‡

Abstract. We give constructions of probabilistically checkable proofs (PCPs) of length $n \cdot \text{polylog}\, n$ proving satisfiability of circuits of size n that can be verified by querying $\text{polylog}\, n$ bits of the proof. We also give analogous constructions of locally testable codes (LTCs) mapping n information bits to $n \cdot \text{polylog}\, n$ bit long codewords that are testable with $\text{polylog}\, n$ queries. Our constructions rely on new techniques revolving around properties of codes based on relatively *high*-degree polynomials in *one* variable, i.e., Reed–Solomon codes. In contrast, previous constructions of short PCPs, beginning with [L. Babai, L. Fortnow, L. Levin, and M. Szegedy, *Checking computations in polylogarithmic time*, in Proceedings of the 23rd ACM Symposium on Theory of Computing, ACM, New York, 1991, pp. 21–31] and until the recent [E. Ben-Sasson, O. Goldreich, P. Harsha, M. Sudan, and S. Vadhan, *Robust PCPs of proximity, shorter PCPs, and applications to coding*, in Proceedings of the 36th ACM Symposium on Theory of Computing, ACM, New York, 2004, pp. 13–15], relied extensively on properties of *low*-degree polynomials in *many* variables. We show how to convert the problem of verifying the satisfaction of a circuit by a given assignment to the task of verifying that a given function is close to being a Reed–Solomon codeword, i.e., a univariate polynomial of specified degree. This reduction also gives an alternative to using the "sumcheck protocol" [C. Lund, L. Fortnow, H. Karloff, and N. Nisan, *J. ACM*, 39 (1992), pp. 859–868]. We then give a new PCP for the special task of proving that a function is close to being a Reed–Solomon codeword. The resulting PCPs are not only shorter than previous ones but also arguably simpler. In fact, our constructions are also more natural in that they yield locally testable codes first, which are then converted to PCPs. In contrast, most recent constructions go in the opposite direction of getting locally testable codes from PCPs.

Key words. probabilistically checkable proofs (PCPs), PCPs of proximity, locally testable codes, Reed–Solomon codes

AMS subject classification. 68Q17

DOI. 10.1137/050646445

1. Introduction.

Probabilistically checkable proof (PCP) systems as formulated in [20, 3, 2] are proof systems that allow efficient probabilistic verification based on querying a few bits of a proof. Formally, a PCP system is given by a PCP-verifier that probabilistically queries a few bits of a purported proof of a claimed theorem and accepts valid proofs of true theorems with probability 1, while accepting any claimed proof of false assertions with low probability, say, at most 1/2. The celebrated PCP theorem [3, 2] asserts that for any language in NP there exists a PCP-verifier that reads just a constant number of bits from a proof of polynomial length. Subsequently, it was shown in [28, 26] that the number of queries can be made as small as *three* bits, while rejecting proofs of false assertions with probability arbitrarily close to (but

*Received by the editors November 30, 2005; accepted for publication (in revised form) December 5, 2006; published electronically May 23, 2008.

http://www.siam.org/journals/sicomp/38-2/64644.html

†Computer Science Department, Technion—Israel Institute of Technology, Haifa, 32000, Israel (eli@cs.technion.ac.il). This author is a Landau Fellow who was supported by the Taub and Shalom Foundations. This author's work was also supported by an Alon Fellowship of the Israeli Council for Higher Education, an International Reintegration Grant from the European Community, and grants from the Israeli Science Foundation and the US-Israel Binational Science Foundation. This work was done while the author was at the Radcliffe Institute for Advanced Study, Cambridge, MA.

‡Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139 (madhu@mit.edu). The work of this author was supported in part by NSF Award CCR-0312575. This work was done while the author was at the Radcliffe Institute for Advanced Study, Cambridge, MA.

larger than) 1/2. Such query-efficient proofs translate to strong inapproximability results for many combinatorial optimization problems; see [7, 8, 26, 28, 38].

Somewhat surprisingly, PCPs are rarely appreciated for their positive properties, i.e., as methods of transforming proofs into extremely efficiently verifiable formats. Instead their negative implications for combinatorial optimization dominate their study. In principle, PCPs could form the semantic analogue of error-correcting codes: Error-correcting codes are used to preserve data for long periods of time; PCPs may be used to preserve data, with a promise of integrity with respect to any fixed Boolean property, for long periods of time. However such uses seemed infeasible using current PCP constructions, which are *too long* and *too complex*. This forms the motivation of our work, which tries to find *shorter and simpler PCPs*.

A number of works [5, 37, 27, 24, 11, 9] have been focused on optimizing the length of the PCP. In addition to the inherent motivation mentioned above, the length of PCPs also plays an important role in their use in cryptography (e.g., in CS proofs [30, 35] and their applications [6, 13]) and is closely related to the construction of locally testable codes [24, 11, 9]. Simplifying PCP constructions has long been a goal within the study of PCPs, though little progress had been achieved in this direction until Dinur's recent surprising proof of the PCP theorem by gap amplification [18] continuing the combinatorial approach taken in [19]. Although we also construct simpler PCPs, our approach by contrast relies on adding algebraic structure instead of combinatorics.

PCPs. Our main result, Theorem 2.2, is a PCP construction that blows up the proof length by only a polylogarithmic factor resulting in a PCP of *quasilinear* length. (Throughout this paper, a function $f : \mathbb{N}^+ \to \mathbb{N}^+$ is said to be *quasilinear* if $f(n) = n \cdot \text{polylog}\, n$.) These short proofs can be verified by querying a polylogarithmic number of bits of the proof. By way of comparison, the recent results of Ben-Sasson et al. [9] give proofs of length $n \cdot \exp(\text{poly}\log\log n)$ with a query complexity of $\text{poly}\log\log n$. Thus, while the query complexity of our PCPs is higher than that of most recent results, the proof size is smaller.

PCPs of proximity. The results of [9] are actually for a stronger notion of PCPs, called *PCPs of proximity* (PCPPs). This notion was simultaneously introduced (under the name *assignment testers*) in [19] and a similar notion also appeared earlier in [40]. Informally, a PCPP-verifier's input includes two oracles, a "claimed theorem" and the "proof," and the verifier confirms that the claimed theorem is *close* in, say, Hamming distance, to a true theorem. It does so by making few oracle queries into the theorem and the proof. In contrast, recall that a PCP-verifier had unlimited access to the "claimed theorem" but verified that it was true *exactly* as stated. Theorem 2.10 gives a construction of PCPPs for all languages in NP with shorter proofs of proximity, though with larger query complexity than that of [9].

Locally testable codes. PCPs typically go hand-in-hand with *locally testable codes* (LTCs); for a detailed discussion of LTCs, see [24, 23] and references therein. Briefly, LTCs are error-correcting codes with relatively large rate and distance. Additionally, the amount of noise in a received word can be bounded from above by querying only a sublinear number of positions of the received word. Specifically, these codes have an associated tester that reads very few symbols of a received word and accepts codewords with probability 1, while rejecting words that are far from all codewords with constant probability (say, 1/2). Theorem 2.13 constructs LTCs with parameters similar to those of our PCPs. Namely, the codes have linear distance while the codeword to message ratio and the query complexity of the tester are polylogarithmic.

We highlight the fact that our work first constructs LTCs with polylogarithmic rate and query complexity, after which PCPs with the same parameters are derived as a consequence. While the early work of Babai et al. [5] also had this feature, constructions of smaller LTCs (in particular, those in [24, 11, 9]) reverse this direction, getting PCPs first and then deriving LTCs as a consequence. Our work thus achieves one of the goals associated with LTCs, namely, offering benefits and insights into PCPs via direct construction of LTCs.

Our techniques. Although our construction is algebraic as in prior PCP constructions, our techniques are significantly different and thus interesting in their own right. All previous algebraic PCPs (i.e., those excluding the combinatorial construction of [19]) start with a PCP based on the properties of multivariate polynomials over some finite field. Some key ingredients in such constructions are the following: (1) a *low-degree test*, i.e., a method to test if a function given by an oracle is close to being a low-degree multivariate polynomial, (2) a *self-corrector*, i.e., a procedure to compute the value of a multivariate polynomial at a given point, given oracle access to a polynomial that is close to this polynomial, (3) a *zero-tester*, i.e., an efficient procedure to verify if a function given by an oracle is close to a multivariate polynomial that is zero on every point in a prespecified subset of its domain, and (4) a reduction from verifying satisfiability to zero-testing. Typical solutions to the above problems yield a query complexity that is polynomial in the number of variables and the degree of the multivariate polynomial. This query complexity can then be reduced using a set of techniques referred to as *proof composition*.

Our solution follows a similar outline (though we do not need a self-corrector) except that, for the most part, we work only with univariate polynomials. This forms the essence of our technical advantage, giving PCPs with smaller proof length. The length of PCPs is well known to grow with the number of variables in the polynomials used to construct them, and reducing this number was an obvious way to try to reduce PCP length. However, reducing the number of variables increases the degree of the associated polynomials, and since solutions to steps (1)–(3) above had query complexity polynomial in the degree, previous solutions needed to use a large number of variables to significantly reduce the number of queries. In our case, we propose analogous questions for *univariate* polynomials and give query-efficient solutions for them, leading to short PCPs. We describe our solutions to the steps (1)–(4) in reverse order.

We start with the reduction from satisfiability to testing zero polynomials, which is step (4) above. The usual reduction is a transformation from a Boolean formula ϕ to a constraint C on pairs of polynomials along with subsets S_1 and S_2 of the multivariate domains with the following property: ϕ is satisfiable iff there exist polynomials P_1, P_2 that are zero on S_1, S_2, respectively, and furthermore $C(P_1, P_2)$ holds. (To enable "easy verification," $C(P_1, P_2)$ needs to be of a special form, but we will not get into this now.) In general, these reductions are simple, and our version of the reduction is as well. However in our case, the reductions appear particularly natural since we deal with a very small number of variables. In section 5 we describe our natural way of reducing NP-complete problems to problems about testing zeros of polynomials. In the end we use a somewhat more complex solution due only to our goal of extreme length efficiency; even in this case the full proof is only a few pages long.

Next we move to the zero-testing problem, which is step (3) above. We reduce this to two univariate low-degree testing questions, along with a natural consistency test between the two polynomials. The query complexity is a *constant* independent of

the degrees of the polynomials we are working with. Furthermore, it directly reduces zero-testing to low-degree testing while most previous solutions relied on some form or other of the self-correcting question. Put together, our solutions for steps (3) and (4) give a short and simple reduction from verifying NP statements to testing the degree of a univariate function. Furthermore, these reductions add only a *constant* number of queries to the query complexity of the low-degree testing protocol. This highlights the importance of the low-degree testing problem for univariate polynomials, which we describe below.

Reed–Solomon codes and proofs of proximity. The problem at the heart of our PCPs is the following: Given a finite field $\mathbb{F}$, a degree bound d, and oracle access to a function $f : \mathbb{F} \to \mathbb{F}$, test if f is close to a polynomial of degree at most d. Specifically, if f is a degree-d polynomial, then the test must always accept. On the other hand, if f is δ-far from every degree-d polynomial, i.e., the value of f needs to be changed on at least δ-fraction of the points in $\mathbb{F}$ to get a degree-d polynomial, then the test must reject with high probability. The objective is to do this while querying the oracle for f as few times as possible. The functions derived by evaluating polynomials of a specified degree over a field are known as *Reed–Solomon codes*, which we sometimes refer to by the name *RS-codes*. Our goal is thus to provide an efficient test for membership in these codes.

It is easy to see that, as such, the problem above allows no very efficient solutions: A tester that accepts all degree-d polynomials with probability 1 must probe the value of f in at least $d+2$ places before it can reject any function. This is too many queries for our purpose. This is where the notion of PCPPs comes to the rescue. Whereas it is hard to test if function f described by the oracle represents a degree-d polynomial with fewer than $\Omega(d)$ queries, it is conceivable (and indeed implied by previous works, for example, by [9]) that one can use an auxiliary proof oracle π to "prove" that f is close to the evaluations of a degree-d polynomial. More formally, our new task is thus to design a PCPP-verifier that makes a few queries to a pair of oracles (f, π), where we allow π to return elements of $\mathbb{F}$ as answers, and the following holds: If f is a degree-d polynomial, there exists a valid proof π so that (f, π) is always accepted by the tester. If f is δ-far from every degree-d polynomial, then for every π, the pair (f, π) must be rejected with high probability.

Since the property of being a degree-d polynomial over $\mathbb{F}$ can be efficiently verified (in time $|\mathbb{F}| \cdot \operatorname{polylog} |\mathbb{F}|$), we can apply the final theorem of Ben-Sasson et al. [9], which gives length-efficient proofs for any property relative to the time it takes to verify the property deterministically, to get moderately efficient solutions to this problem. Unfortunately, such a solution would involve proof oracles of length $|\mathbb{F}| \cdot \exp(\operatorname{poly} \log \log |\mathbb{F}|)$, which is longer than we can allow. Their solution would also not satisfy our (subjective) simplicity requirement. However it does confirm that our goal of making $o(d)$ queries is attainable.

Our main technical result is a PCPP for Reed–Solomon codes. This proof of proximity has length $O(n \cdot \operatorname{polylog} n)$ and query complexity $\operatorname{polylog} n$ for RS-codes over a field $\mathbb{F}$ of cardinality n and characteristic 2. We also describe some variations, such as PCPPs for RS-codes over certain prime fields, but these are not needed for our final PCP results. Our proof of proximity consists of an encoding of an efficient FFT-like evaluation of the low-degree polynomial. Our analysis makes crucial (black-box) use of Polishchuk and Spielman's [37] analysis of a natural low-degree test for bivariate polynomials.

We remark that almost all ingredients in the construction of our PCPs, including

the PCPPs for RS-codes, are simple. The simplicity of the PCP also means that the "hidden constants" in the construction are relatively small and the building blocks we use can be implemented with relative ease. In fact, our main building blocks, namely, the PCP of proximity for Reed–Solomon codes and its verifier described in Theorem 3.2, have been recently implemented successfully in code [12], resulting in PCPPs of length $\approx \frac{1}{4}n \cdot \log^4 n$ for RS-codes over binary fields of size n.

Recent developments. One of the main problems left open by this work was obtaining quasilinear PCPs and PCPPs with *constant* query complexity. This was recently solved by Dinur in [18] by applying her novel proof of the PCP theorem by gap amplification to our Theorem 2.2. Dinur provides a general transformation that takes any PCP of length $\ell(n)$ where the verifier makes $q(n)$ queries and converts it into a PCP of length $\ell(n) \cdot q(n)^{O(1)}$ where the verifier makes $O(1)$ queries. Applying this to the trivial PCP that makes $O(n)$ queries yields a simple proof of the PCP theorem, though with long proofs. On the other hand, applying this transformation to our PCP yields a quasilinear PCP with constant query complexity.

With the exception of [5], the running time of all previously known PCP- and PCPP-verifiers, including ours, is polynomial in the size of the input. Recently, it was shown in [10] that the running time of our PCPP-verifier can be reduced to be polylogarithmic, maintaining the query complexity and proof length of our PCPP construction in Theorem 2.10.

Organization of this paper. In section 2 we present formal definitions of the notions of PCPs, LTCs, and PCPPs, and we present the formal statements of our main theorems about these concepts. In section 3 we introduce the main technical notions used in this paper, namely, Reed–Solomon codes, some computationally important subclasses of Reed–Solomon codes, and algebraic satisfiability problems. We state our technical results about these problems and then show how our main theorems (i.e., the ones stated in section 2) follow from these technical results. In sections 4–7, we prove our technical results. A more detailed breakdown of these results is given at the end of section 3.

2. Definitions and main results.

Preliminaries. Unless specified otherwise, our alphabet of choice is $\Sigma = \{0,1\}$ and all logarithms are taken to base 2. For a function $t : \mathbb{N}^+ \to \mathbb{N}^+$, recall that NTIME($t(n)$) is the class of languages $L \subseteq \Sigma^*$ decidable in nondeterministic time $t(n)$ on inputs of length n.

2.1. PCPs.

The following is a variant of the standard definition of PCPs [3], where the running time of the verifier is allowed to grow exponentially with the randomness. This is done following [10] to allow a statement of results about languages whose nondeterministic decision time is superpolynomial. Recall that an oracle machine is said to be *nonadaptive* if its queries do not depend on previous oracle answers. We stress that all oracle machines considered in this paper, and, in particular, the following PCP-verifier and the PCPP-verifier of Definition 2.4, are nonadaptive.

DEFINITION 2.1 (PCP). *For functions $r, q : \mathbb{N}^+ \to \mathbb{N}^+$ an $(r(n), q(n))$-*PCP-verifier *is a probabilistic machine V with oracle access to a* probabilistically checkable proof*, or simply, a* proof*, denoted π. On input x of length n, V runs in time $2^{O(r(n))}$, tosses $r(n)$ coins, makes $q(n)$ nonadaptive queries to the proof, and outputs either* `accept` *or* `reject`*. We denote by $V^{\pi}[x;R]$ the output of V on input x, proof π, and random coins R.*

For constant $s \in [0,1]$, a language $L \subseteq \Sigma^*$ is said to belong to the class of

languages

$$\mathbf{PCP}_s[\mathsf{randomness}\ r(n), \mathsf{query}\ q(n)]$$

if there exists an $(r(n), q(n))$-PCP-verifier V_L such that the following hold:

- Perfect completeness: *If $x \in L$ then $\exists \pi$ such that* $\Pr_R[V_L^\pi[x;R] = \texttt{accept}] = 1$.
- Soundness: *If $x \notin L$ then $\forall \pi$ we have* $\Pr_R[V_L^\pi[x;R] = \texttt{accept}] \leq s$.

Our first main result is the following. Recall the definition of a *proper complexity function* from [36, Definition 7.1], where a function $f(n)$ is proper if it can be computed in time $\text{polylog}\, n$.

THEOREM 2.2 (quasilinear PCPs). *For any proper complexity function $t : \mathbb{N}^+ \to \mathbb{N}^+$,*

$$\text{NTIME}(t(n)) \subseteq \mathbf{PCP}_{\frac{1}{2}}\mathsf{randomness}\ \log(t(n) \cdot \text{polylog}\, t(n)), \mathsf{query}\ \text{polylog}\, t(n)].$$

Remark 2.3. The parameters of Theorem 2.2 have been recently improved. In particular, [18] reduced the query complexity to $O(1)$ and [10] reduced the verifier's running time to $\text{poly}\, n + \text{polylog}\, t(n)$ (as opposed to $t(n) \cdot \text{polylog}\, t(n)$). In both cases all other parameters remain unchanged.

Since without loss of generality the proof is of size at most $2^{\mathsf{randomness}} \times \mathsf{query}$ the previous theorem implies that the probabilistically checkable proof for $x \in L$ is quasilinear in the running time of the nondeterministic machine deciding L.

In contrast, the recent results of [9] give proofs of length $n \cdot \exp(\text{poly}\log\log n)$ with a query complexity of $\text{poly}\log\log n$ and slightly longer proofs with constant query complexity. Thus, while the query complexity of our PCPs is higher than that of the previous state of the art, their length is shorter.

2.2. Proximity and proofs of proximity. We now formalize the notion of verifying proofs of theorems where even the theorem is not known but rather is provided as an oracle to the verifier. The verifier, in such a case, can hope only to certify that the theorem is "close" to one that is true. To define this notion we first need to formalize the notion of "closeness," or proximity.

We will work with a variety of distance measures $\Delta : \Sigma^N \times \Sigma^N \to [0,1]$, where a distance measure satisfies the properties (1) $\Delta(x,x) = 0$, (2) $\Delta(x,y) = \Delta(y,x)$, and (3) $\Delta(x,z) \leq \Delta(x,y) + \Delta(y,z)$. The most common/natural one, and the target of most of our theorems, will be *relativized Hamming distance* over the alphabet Σ, denoted $\mathsf{Hamming}_\Sigma(\cdot,\cdot)$. Formally, for $y = (y_1, \ldots, y_N), y' = (y'_1, \ldots, y'_N) \in \Sigma^N$,

$$\mathsf{Hamming}_\Sigma(y,y') = |\{i : y_i \neq y'_i\}|/N.$$

For our proofs we use other distance measures on strings which may weigh different coordinates differently. For example, given a set $I \subseteq [N]$ we may consider the distance $\mathsf{Hamming}_{\Sigma,I}(y,y') = |\{i \in I : y_i \neq y'_i\}|/|I|$. Note that a convex combination of distance measures is also a distance measure, and this describes many other distance measures we use later.

Given a distance measure $\Delta : \Sigma^N \times \Sigma^N \to [0,1]$ and a set $S \subseteq \Sigma^N$ we define the distance of an element $y \in \Sigma^N$ from S to be

$$\Delta(y,S) = \begin{cases} \min_{s \in S} \Delta(y,s), & S \neq \emptyset, \\ 1, & S = \emptyset. \end{cases}$$

We are now ready to describe PCPs of proximity (PCPPs)/assignment testers [9, 19]. We follow the general formulation as appearing in [9]. In this formulation, the input comes in two parts (x, y), where $x \in \Sigma^*$ is given explicitly to the verifier and $y \in \Sigma^*$ is given as oracle. In addition, the verifier is given oracle access to a proof. The verifier is allowed to read x in its entirety, but its queries to y are counted as part of its query complexity, i.e., together with the queries to the proof. Throughout this paper we assume without loss of generality the explicit input of a pair instance includes a specification of the length of the implicit input. If unspecified we set the length to be $t(|x|)$. Formally, we assume the explicit input is of the form $x = (x', N)$, where $N = |y|$. The size of the explicit input is the size of x'.

DEFINITION 2.4 (PCPP-verifier). *For functions $r, q : \mathbb{N}^+ \to \mathbb{N}^+$ an $(r(n), q(n))$-*PCPP-verifier *is a probabilistic machine V with oracle access to an* implicit input y *and a* proof of proximity*, or simply, a* proof*, denoted π. On explicit input $x = (x', N)$ with $|x'| = n$, and N an integer, verifier V runs in time $2^{O(r(n))}$, tosses $r(n)$ coins, makes at most $q(n)$ nonadaptive queries in total to the two oracles, y of size N and π, and outputs either* `accept` *or* `reject`*. We denote by $V^{(y,\pi)}[x; R]$ the output of the PCPP-verifier on input x and random coins R.*

PCPPs refer to languages consisting of pairs of strings where the elements in these pairs refer to the two parts of the input in Definition 2.4. Thus, we define a *pair language* to be subset of $\Sigma^* \times \Sigma^*$. It is useful for us to measure the complexity of a pair language as a function of its first input. So PAIR-TIME$(t(n))$ is the set of languages L such that there exists a machine M that takes time $t(|x|)$ on input (x, y) such that $L = \{(x, y) : M(x, y) = \texttt{accept}\}$. One notable pair language in PAIR-TIME$(n \cdot \text{polylog}\, n)$ is CKTVAL, the language of pairs (C, w), where C is a Boolean circuit with N inputs and w is an assignment satisfying C. PAIR-NTIME$(t(n))$ is defined similarly, this time allowing M to be a nondeterministic machine.

For a pair language L and $x \in \Sigma^*$, $x = (x', N)$, let

$$L_x \triangleq \{y \in \Sigma^N : (x, y) \in L\}.$$

PCPP-verifiers are intended to accept implicit inputs in L_x and reject implicit inputs that are far from being in L_x. This gives rise to classes of pair languages defined in terms of PCPPs.

DEFINITION 2.5 (PCPP). *For functions $r, q : \mathbb{N}^+ \to \mathbb{N}^+$,* soundness parameter *$s \in [0, 1]$, family of* distance measures *$\Delta = \{\Delta_N : \Sigma^N \times \Sigma^N \to [0, 1]\}_{N \in \mathbb{N}^+}$, and* proximity parameter *$\delta \in [0, 1]$ we say the pair language L belongs to the class of languages*

$$\mathbf{PCPP}_{s,\delta}\left[\begin{array}{l}\mathsf{randomness}\ r(n),\\ \mathsf{query}\ q(n),\\ \mathsf{distance}\ \Delta\end{array}\right]$$

if there exists an $(r(n), q(n))$-PCPP-verifier V_L such that the following hold for all $(x, y), |y| = N$:

- Perfect completeness: *If $(x, y) \in L$ then $\exists \pi$ such that* $\Pr_R[V_L^{(y,\pi)}[x; R] = \texttt{accept}] = 1$.
- Soundness: *If $\Delta_N(y, L_x) \geq \delta$, then $\forall \pi$* $\Pr_R[V_L^{(y,\pi)}[x; R] = \texttt{accept}] \leq 1 - s$.

Remark 2.6. As mentioned earlier, our main results (for example, Theorem 2.10) target the relative Hamming distance. However, to prove these we shall need to use PCPPs with different distance measures (see subsection 3.4).

Our constructions of PCPPs come naturally with a somewhat different soundness condition than the one required in Definition 2.5. On the one hand, they do not achieve a soundness error of an absolute constant. On the other hand, they satisfy the additional property that a PCPP-verifier for L_x rejects *every* string y with probability proportional to the distance of y from L_x. We formalize this "strong" soundness condition below and then state a general transformation from PCPPs with strong soundness to the weaker version above. (The term "strong" is derived from the analogous definition of *strong locally testable codes* [24, Definition 2.1].)

DEFINITION 2.7 (strong PCPP). *For r, q, Δ as in Definition* 2.5 *and* soundness function $s : (0,1] \times \mathbb{N}^+ \to (0,1]$, *we say language L belongs to the class*

$$\mathbf{Strong\text{-}PCPP}_{s(\delta,n)} \left[\begin{array}{l} \mathsf{randomness}\ r(n), \\ \mathsf{query}\ q(n), \\ \mathsf{distance}\ \Delta \end{array} \right]$$

if there exists an $(r(n), q(n))$-PCPP-verifier V_L with perfect completeness as in Definition 2.5 *and for all (x, y), $|y| = N$, the following holds:*

- Strong soundness: $\forall \pi \quad \Pr_R[V_L^{(y,\pi)}[x; R] = \mathtt{accept}] \leq 1 - s(\Delta_N(y, L_x), n)$.

Remark 2.8. Naturally, one expects the soundness function to be nondecreasing. Formally, we say $s : (0,1] \times \mathbb{N}^+ \to [0,1]$ is *nondecreasing* if for all $n \in \mathbb{N}^+$ the function $s(\cdot, n) : (0,1] \to (0,1]$ is nondecreasing. This implies that the farther y is from L_x, the higher the rejection probability or soundness. Indeed, all soundness functions considered in this paper are nondecreasing.

Notice that a "weak" PCPP, with soundness parameter s_0 and distance parameter s_0, is also a "strong" PCPP with a threshold soundness function $s(\delta, n)$ that evaluates to 0 on $\delta' < \delta_0$ and to s_0 on $\delta' \geq \delta_0$. A converse of this is also true. To see this one needs only to amplify the soundness error from $s(\delta, n)$ to some fixed desired constant s'. The now standard application of randomness efficient sampling allows such amplification with little additional cost in randomness. Indeed, using the expander-neighborhood sampler of [25] (see also [22, section C.4]) we get the following proposition, given here without proof. (For a proof see [9, Lemma 2.11].)

PROPOSITION 2.9 (strong PCPPs imply "weak" ones). *Let $s : (0,1] \times \mathbb{N}^+ \to (0,1]$ be a nondecreasing soundness function as defined in Remark* 2.8. *If a pair language L belongs to*

$$\mathbf{Strong\text{-}PCPP}_{s(\delta,n)} \left[\begin{array}{l} \mathsf{randomness}\ r(n), \\ \mathsf{query}\ q(n), \\ \mathsf{distance}\ \Delta \end{array} \right],$$

then, for every $s', \delta \in (0,1)$, the language L belongs to

$$\mathbf{PCPP}_{s',\delta} \left[\begin{array}{l} \mathsf{randomness}\ r(n) + O(\frac{1}{s(\delta,n)} \cdot \log \frac{1}{s'}), \\ \mathsf{query}\ O\left(\frac{q(n) \cdot \log 1/s'}{s(\delta,n)}\right), \\ \mathsf{distance}\ \Delta \end{array} \right].$$

Furthermore, the proof queried by the "weak" PCPP-verifier is of the same length as that queried by the "strong" one.

We are now ready to state our main result for PCPPs.

THEOREM 2.10 (quasilinear PCPPs). *For any proper complexity function t :*

$\mathbb{N}^+ \to \mathbb{N}^+$,

$$\text{PAIR-NTIME}(t(n)) \subseteq \mathbf{Strong\text{-}PCPP}_{\delta/\operatorname{polylog} t(n)} \left[\begin{array}{l} \mathsf{randomness}\ \log(t(n) \cdot \operatorname{polylog} t(n)), \\ \mathsf{query}\ \operatorname{polylog} t(n), \\ \mathsf{distance}\ \mathsf{Hamming}_\Sigma \end{array} \right].$$

Consequently, as implied by Proposition 2.9*, for any* $s, \delta \in (0,1)$*,*

$$\text{PAIR-NTIME}(t(n)) \subseteq \mathbf{PCPP}_{s,\delta} \left[\begin{array}{l} \mathsf{randomness}\ \log(t(n) \cdot \operatorname{polylog} t(n)), \\ \mathsf{query}\ \operatorname{polylog} t(n), \\ \mathsf{distance}\ \mathsf{Hamming}_\Sigma \end{array} \right].$$

Furthermore, the length of the proof queried by the PCPP-verifier (in both the strong and weak cases) is $t(n) \cdot \operatorname{polylog} t(n)$*.*

Remark 2.11. As in the case of Theorem 2.2, the parameters of Theorem 2.10 have been recently improved. In particular, [18] reduced the query complexity to $O(1)$ and [10] reduced the verifier running time to $\operatorname{poly} n + \operatorname{polylog} t(n)$. In both cases all other parameters remain unchanged.

The previous state of the art with respect to PCPPs [9] gave proofs of length $n \cdot \exp(\operatorname{poly} \log \log n)$ with a query complexity of $\operatorname{poly} \log \log n$ (and slightly longer proofs with constant query complexity). Once again, our query complexity is somewhat higher but our proofs are somewhat shorter.

2.3. Locally testable codes. We now move to the third notion addressed by this paper—that of LTCs.

For field $\mathbb{F}$ and integers n, k, d, a linear $[n,k,d]_{\mathbb{F}}$-code is an injective linear map $C : \mathbb{F}^k \to \mathbb{F}^n$ such that for every pair $x \neq y \in \mathbb{F}^k$, $\mathsf{Hamming}_{\mathbb{F}}(C(x), C(y)) \geq d/n$. We point out that all codes considered in this paper are linear. The *alphabet* of C is $\mathbb{F}$, the *blocklength* is n, the *dimension* is k, the *rate* is k/n, and the *distance* is d. The *image* of C is the linear space $\text{Image}(C) = \{C(a) : a \in \mathbb{F}^k\}$. Often a code is identified with its image.

Loosely speaking, a linear $[n,k,d]_{\mathbb{F}}$-code is said to be *locally testable* if a *tester*, i.e., a randomized machine with oracle access to the supposed codeword, can distinguish with high probability between words in the code and words that are far from it, while making only $o(k)$ random queries into a purported codeword. The following is essentially Definition 2.1 from [24].

DEFINITION 2.12 (locally testable codes). *A randomized polynomial time oracle machine* T *is called a* (δ, q, γ)-tester *for the linear* $[n,k,d]_{\mathbb{F}}$ *code* C *if it satisfies the following two conditions:*

- *For any* $w \in \text{Image}(C)$*,*

$$\Pr[T^w[R] = \texttt{accept}] = 1,$$

  where $T^w[R]$ *denotes the output of the tester on oracle* w *and random coins* R*.*
- *For any* $w \in \mathbb{F}^n$ *such that* $\mathsf{Hamming}_{\mathbb{F}}(w, \text{Image}(C)) \geq \delta$*,*

$$\Pr[T^w[R] = \texttt{reject}] \geq \gamma.$$

A code is said to be (δ, q, γ)-locally testable *if it has a* (δ, q, γ)*-tester.*

THEOREM 2.13 (locally testable codes with polylogarithmic rate). *Let $\delta, \gamma \in (0,1)$, $\Sigma = \mathbb{F}_2$ be the field of two elements and let n be any power of 2. Then, there exists a linear $[N = n \cdot \text{polylog}\, n, K = n/8, D = N/8]_\Sigma$-code that is $(\delta, \text{polylog}\, n, \gamma)$-locally testable. Furthermore, encoding, decoding, and testing can be performed in time polynomial in n.*

Remark 2.14. As with Theorems 2.2 and 2.10, the query complexity of Theorem 2.13 has been recently improved in [18] to $O(1)$, leaving all other parameters unchanged.

We remark that we also give LTCs over a variety of other fields and other choices of n (see Theorems 3.2 and 3.4 for details). Also if we relax the requirement that the code be linear, then the theorem above follows immediately from Theorem 2.10 (and [9, section 4.1]) without any restrictions on the choice of n.

3. Technical ingredients of our constructions. In this section we introduce the main technical ingredients of our paper and prove the three main theorems (Theorems 2.2, 2.10, and 2.13) of our paper, assuming these ingredients. Recall that these theorems promise short PCPs, PCPPs, and LTCs. We stress that while the construction of PCPs and LTCs follows easily from the PCPP construction, this is not the approach in our paper.

We start by constructing an LTC, based on one of the most popular codes, namely, the Reed–Solomon code. We give a PCPP for a language whose elements are essentially Reed–Solomon codewords. Recalling the fact that Reed–Solomon codes are evaluations of univariate polynomials of bounded degree, this result shows how it is possible to prove that a function given as an oracle is close to some polynomial of bounded degree. Subsection 3.1 below describes the actual language based on Reed–Solomon codes and states the PCPP construction that we obtain for this language. This immediately leads to a proof of Theorem 2.13.

We then move to the constructions of PCPs and PCPPs for general NTIME languages. These constructions are obtained by first reducing the NTIME language under consideration to an algebraic version of SAT that we call an algebraic constraint satisfaction problem, and then giving PCPs (and PCPPs) for algebraic constraint satisfaction problems. We define algebraic constraint satisfaction problems and state their completeness for NTIME in subsection 3.2.

The advantage of algebraic constraint satisfaction problems is that the natural "classical" proofs of satisfiability for these problems come in the form of two univariate polynomials of bounded degree, say, f, g, that satisfy some simple constraints. For example, in the PCP construction, the verifier knows some set $H \subseteq \mathbb{F}$ and would like to verify that $g(x) = 0$ for every $x \in H$. The PCPPs for Reed–Solomon codes already show how to prove/verify that the functions f and g are close to some polynomials of bounded degree. In subsection 3.3 we augment this PCPP so as to test that it vanishes on the set H, and this leads us to a proof of Theorem 2.2. Finally, in subsection 3.4 we describe the additional ingredients needed to get a PCPP for NTIME and prove Theorem 2.10 modulo these ingredients.

3.1. PCPPs for Reed–Solomon codes. We start by defining the Reed–Solomon codes and a pair language based on these codes. We then describe two cases of Reed–Solomon codes where we can obtain PCPPs for membership in the language. This yields our main theorem (Theorem 2.13) on LTCs.

DEFINITION 3.1 (Reed–Solomon codes and pair language). *The* evaluation *of a polynomial $P(z) = \sum_{i=0}^{d} a_i z^i$ over $S \subseteq \mathbb{F}, |S| = n$ is the function $p : S \to \mathbb{F}$ defined by*

$p(s) = P(s)$ for all $s \in S$. The formal sum $P(z)$ is called the polynomial corresponding *to (the function) p. The* Reed–Solomon code *of degree at most d over $\mathbb{F}$, evaluated at S, is*

$$\mathrm{RS}(\mathbb{F}, S, d) \triangleq \{p : S \to \mathbb{F} \mid p \text{ is an evaluation of a polynomial of degree } \leq d \text{ over } S\}.$$

The pair language PAIR-RS *is defined as follows. The explicit input is a triple $(\mathbb{F}, S, d)$, where $\mathbb{F}$ is a description of a finite field,*[1] *$S \subseteq \mathbb{F}$, and d is an integer. The size of the explicit input is assumed to be $|S| + O(1)$ field elements because in all our applications both d and $\mathbb{F}$ can be described using $\log |\mathbb{F}|$ bits. The implicit input is a function $p : S \to \mathbb{F}$. The size of the implicit input is $|S|$ field elements. A pair $((\mathbb{F}, S, d), p)$ is in* PAIR-RS *iff $p \in \mathrm{RS}(\mathbb{F}, S, d)$ and the explicit input is in the format described above.*

Notice that $\mathrm{RS}(\mathbb{F}, S, d)$ is the image of a linear $[n, d+1, n-d]_{\mathbb{F}}$-code. To see this, set $S = \{\xi_1, \ldots, \xi_n\}$ and consider the linear map sending $(a_0, \ldots, a_d) \in \mathbb{F}^{d+1}$ to the codeword $(P(\xi_1), \ldots, P(\xi_n))$ for $P(z) = \sum_{i=0}^{d} a_i z^i$.

Next we state our main technical results, namely, quasilinear length proofs of proximity for Reed–Solomon codes. Our results hold for certain "well-behaved" fields and evaluation sets, including fields of characteristic 2 (Theorem 3.2) and multiplicative subgroups that are sufficiently smooth (Theorem 3.4). As is customary when discussing Reed–Solomon codes, our distance measure is the relative Hamming distance over alphabet $\mathbb{F}$, denoted $\mathsf{Hamming}_{\mathbb{F}}$, and our alphabet is the underlying field. In particular, queries are answered by field elements.

3.1.1. Fields of characteristic two.

THEOREM 3.2 (PCPPs for RS-codes over fields of characteristic 2). *Let* PAIR-ADDITIVE-RS *be the restriction of* PAIR-RS *to pairs $((\mathrm{GF}(2^\ell), S, d), p)$, where $\mathrm{GF}(2^\ell)$ is the Galois field of size $n = 2^\ell$ and characteristic 2 and $S \subseteq \mathbb{F}$ is $\mathrm{GF}(2)$-linear. (Recall that S is $\mathrm{GF}(2)$-linear iff for all $\alpha, \beta \in S$ we have $\alpha + \beta \in S$.) Then,*

$$\text{PAIR-ADDITIVE-RS} \in \mathbf{Strong\text{-}PCPP}_{\delta/\operatorname{polylog} n} \left[\begin{array}{l} \mathsf{randomness}\ \log(n \cdot \operatorname{polylog} n), \\ \mathsf{query}\ O(1), \\ \mathsf{distance}\ \mathsf{Hamming}_{\mathrm{GF}(2^\ell)} \end{array} \right].$$

Consequently (using Proposition 2.9), for any $s, \delta \in (0, 1)$,

$$\text{PAIR-ADDITIVE-RS} \in \mathbf{PCPP}_{s,\delta} \left[\begin{array}{l} \mathsf{randomness}\ \log(n \cdot \operatorname{polylog} n), \\ \mathsf{query}\ \operatorname{polylog} n, \\ \mathsf{distance}\ \mathsf{Hamming}_{\mathrm{GF}(2^\ell)} \end{array} \right].$$

Furthermore, the proof queried by the "weak" PCPP-verifier is of the same length as that queried by the "strong" one.

Remark 3.3. The proof of Theorem 3.2 can be modified to obtain (strong) PCPPs with parameters as above for some other fields also. In particular, we can get PCPPs for $\mathbb{F}$ of characteristic $\leq \operatorname{polylog} n$ as long as the evaluation set S is linear over a subfield of $\mathbb{F}$ of size $\operatorname{polylog} n$. For simplicity, and since this suffices for our applications, we prove the result only for characteristic 2.

We prove Theorem 3.2 in section 6. Here we note that Theorem 3.2 immediately leads to a construction of LTCs. In particular, we use it to prove Theorem 2.13 later in this section. But before doing so, we describe a different collection of fields $\mathbb{F}$ and sets S where we can derive PCPPs for Reed–Solomon codes.

[1] An explicit description for such a field could be via a prime a and an irreducible polynomial $g(x)$ over $\mathrm{GF}(a)$.

3.1.2. RS-codes over smooth fields. For this part, and throughout the rest of this paper, let $\mathbb{F}^*$ denote the multiplicative group of a finite field $\mathbb{F}$. The *order* of $\omega \in \mathbb{F}^*$, denoted $\mathrm{ord}(\omega)$, is the smallest positive integer n such that $\omega^n = 1$. The multiplicative group generated by ω is $\langle\omega\rangle \triangleq \{\omega^0, \omega^1, \ldots, \omega^{n-1}\}$.

THEOREM 3.4 (PCPPs for smooth RS-codes). *Let* PAIR-SMOOTH-RS *be the restriction of* PAIR-RS *to pairs* $((\mathbb{F}, \langle\omega\rangle, d), p)$*, where* $\mathrm{ord}(\omega) = n$ *is a power of* 2*. Then,*

$$\text{PAIR-SMOOTH-RS} \in \mathbf{Strong\text{-}PCPP}_{\delta/\operatorname{polylog} n} \left[\begin{array}{l} \mathsf{randomness}\ \log(n \cdot \operatorname{polylog} n), \\ \mathsf{query}\ O(1), \\ \mathsf{distance}\ \mathsf{Hamming}_{\mathbb{F}} \end{array} \right].$$

Consequently (using Proposition 2.9*), for any* $s, \delta \in (0, 1)$*,*

$$\text{PAIR-SMOOTH-RS} \in \mathbf{PCPP}_{s,\delta} \left[\begin{array}{l} \mathsf{randomness}\ \log(n \cdot \operatorname{polylog} n), \\ \mathsf{query}\ \operatorname{polylog} n, \\ \mathsf{distance}\ \mathsf{Hamming}_{\mathbb{F}} \end{array} \right].$$

Furthermore, the proof queried by the "weak" PCPP-verifier is of the same length as that queried by the "strong" one.

Remark 3.5. Examination of the proof of Theorem 3.4 shows that it can be extended to $\langle\omega\rangle$ of size n that is $\operatorname{polylog} n$-smooth; i.e., all prime factors of n are at most $\operatorname{polylog} n$. For simplicity, we state and prove our theorem only for the multiplicative case of a 2-smooth n.

While not immediately evident, prime fields satisfying the requirements of the previous theorem abound. In section 7 we discuss this and provide an alternative proof of the quasilinear PCP Theorem 2.2 that relies on such prime fields. Notice the intersection of PAIR-ADDITIVE-RS and PAIR-SMOOTH-RS is empty. Indeed, a field with a multiplicative subgroup of size 2^k must be of size $c \cdot 2^k + 1$ for integer c, whereas the size of a field of characteristic 2 is a power of 2. Next we show how to construct LTCs using the PCPPs for RS-codes over fields of characteristic two.

3.1.3. Proof of quasilinear LTC—Theorem 2.13.

Proof of Theorem 2.13. Given an integer $n = 2^t$ we use Theorem 3.2 above applied to the field $\mathbb{F}$ of size n, with $S = \mathbb{F}$ and $d = n/8$. The resulting Reed–Solomon code has rate $\Omega(1)$ and relative distance at least $7/8$. We then convert the PCPP for this code into an LTC over $\mathbb{F}$ using a standard conversion. Here we simply sketch this step. For a formal proof, see [9, Proposition 4.1].

The codewords of the LTC are in one-to-one correspondence with the codewords of the Reed–Solomon code. The codeword of the LTC corresponding to a polynomial p consists of two parts. The first part is simply the Reed–Solomon encoding of p repeated sufficiently often so that the first part takes at least, say, half of the coordinates of the LTC. The second part consists of the PCPP that p is a member of the language. The LTC-verifier simply simulates the PCPP-verifier using the first half as the oracle for the implicit input and the second half as the proof oracle, along with some spot-checks to verify that the first part repeats the same codeword several times. It is straightforward to see that the rate of this LTC is asymptotically bounded by the length of the Reed–Solomon codewords divided by the length of their PCPP, and the query complexity is similar to that of the PCPP-verifier. It is easy to see that the LTC so obtained has a relative distance of at least $7/16$ (i.e., half the relative distance of the Reed–Solomon code).

It remains to convert this code into a binary code. This is also straightforward using the idea of concatenation of codes. We pick a small error-correcting code with $|\mathbb{F}|$ codewords of length $\ell = O(\log |\mathbb{F}|)$ and distance, say, at least $.4\ell$ and represent elements of $\mathbb{F}$ as codewords of this code. This converts the LTC obtained in the previous paragraph into a binary code with relative distance at least .4 times the distance of that code, which yields a relative distance of at least $7/40 > 1/8$. The verifier of the LTC above can now be simulated on this binary code with a multiplicative increase in the query complexity by a factor of $O(\log |\mathbb{F}|)$. □

Thus the PCPP for Reed–Solomon codes immediately leads to short LTCs. Additionally as we discuss in the upcoming sections, it also forms the central ingredient in our PCP and PCPP constructions.

3.2. Algebraic constraint satisfaction problems. To obtain length-efficient PCPs and PCPPs we reduce $L \in \text{NTIME}(t(n))$ to an *algebraic* constraint satisfaction problem. We describe this problem by comparing it to a combinatorial analogue, namely, 3SAT. A 3-CNF formula ψ with n variables and m clauses can be viewed as a mapping $\psi : \{0,1\}^n \to \{0,1\}^m$, sending an assignment to the characteristic vector of the set of clauses satisfied by it. The "natural" proof of satisfiability of a 3-CNF formula is a vector a of n bits. The proof proves the satisfiability of ψ if $\psi(a) = \vec{1}$. The typical advantage of this proof is that verification is a sequence of local steps, i.e., to verify that the jth coordinate of $\psi(a)_j = 1$, we need only examine three coordinates of a.

An instance of an algebraic constraint satisfaction ϕ similarly can be viewed as a mapping $\phi : \mathbb{F}[x] \to \mathbb{F}[x]$ from polynomials to polynomials. A candidate proof for the algebraic problem is a low-degree (univariate) polynomial $A \in \mathbb{F}[x]$ over finite field $\mathbb{F}$, called the *proof polynomial*. The map ϕ would map A to a polynomial P of slightly larger degree. ϕ would be considered satisfiable if $P = \phi(A)$ vanishes on a prespecified subset H of $\mathbb{F}$. Finally, for "local verifiability," we will expect that computing $P(x_0)$ requires knowledge of A at very few places, denoted k. But here we place some very strong restrictions on the local neighborhoods. Whereas in 3SAT, there was no simple relationship between a clause index j and the variables participating in the clause, in algebraic constraint satisfaction problems, we expect $P(x_0)$ to depend on A on some set of points of the form $\{\text{AFF}_1(x_0), \ldots, \text{AFF}_k(x_0)\}$, where $\text{AFF}_i(x) = a_i x + b_i$ is an affine map. Moreover, we insist that the computation of $P(x_0)$ from x_0 and $A(\text{AFF}_1(x_0)), \ldots, A(\text{AFF}_1(x_0))$ itself be algebraically simple. Combining all these ingredients leads to the following definition.

DEFINITION 3.6 (univariate algebraic constraint satisfaction problem (CSP)). *Instances of the language* ALGEBRAIC-CSP *are tuples of the form* $\phi = (\mathbb{F}, \{\text{AFF}_1, \ldots, \text{AFF}_{k'}\}, H, C)$, *where* $\mathbb{F}$ *is a field,* $\text{AFF}_i(x) \triangleq a_i x + b_i$ *is an affine map over* $\mathbb{F}$ *specified by* $a_i, b_i \in \mathbb{F}$, $H \subseteq \mathbb{F}$, *and* $C : \mathbb{F}^{k'+1} \to \mathbb{F}$ *is a polynomial of degree at most* $|H|$ *in its first variable. The* size *of* ϕ *is* $|\mathbb{F}|$.

A polynomial $A \in \mathbb{F}[x]$ *is said to* satisfy *the instance* $\phi \in$ ALGEBRAIC-CSP *iff* $\deg(A) \le |H| - 1$ *and for all* $x \in H$,

$$C(x, A(\text{AFF}_1(x)), \ldots, A(\text{AFF}_{k'}(x))) = 0. \tag{3.1}$$

The instance ϕ *is in* ALGEBRAIC-CSP *iff there exists a polynomial satisfying it.*

For integers k, d, *let* ALGEBRAIC-CSP$_{k,d}$ *be the restriction of* ALGEBRAIC-CSP, *to instances as above where* $k' \le k$ *and the degree of* C *in all but the first variable is at most* d.

Our main theorem on PCPs is obtained by reducing NTIME languages to ALGEBRAIC-CSP, while preserving the length of instances to within polylogarithmic factors.

THEOREM 3.7 (ALGEBRAIC-CSP is NTIME-complete). *There exist integers k, d such that for any proper complexity function $t : \mathbb{N}^+ \to \mathbb{N}^+$ and $L \in \mathrm{NTIME}(t(n))$ the following hold.*

1. *L is reducible to* ALGEBRAIC-CSP$_{k,d}$ *in time* $\mathrm{poly}\, t(n)$.
2. *The reduction maps an instance of L of size n to an instance of* ALGEBRAIC-CSP$_{k,d}$ *over field* $\mathrm{GF}(2^\ell)$ *of size* $2^\ell \le t(n)\,\mathrm{polylog}\, t(n)$ *and characteristic* 2, *where* $100(kd+1)(|H|-1) < 2^\ell \le 200(kd+1)(|H|-1)$.

The proof of this theorem is given in section 5.

Remark 3.8. Inspection of the proof of Theorem 3.7 gives $k = 10$ and $d = 8$. More careful optimization can give $k = 9$ and $d = 1$; i.e., C is multilinear in all but the first variable. Favoring simplicity over constant optimization, we omit this proof. Additionally, one can obtain the theorem for any $\mathbb{F}$ as long as $|\mathbb{F}| > |H|$. However, to derive Theorems 2.2 and 2.10 we need $|\mathbb{F}| \gg |H|$.

Very similar algebraic reductions are prevalent in many previous PCPs [5, 3, 2, 37, 39, 11, 9], starting with [5], and our reduction follows that of Polishchuk and Spielman [37]. However, all previous reductions used multivariate polynomials to perform degree reduction. Namely, a message (or assignment) of length n is encoded by an m-variate polynomial of degree $\approx m \cdot n^{1/m}$ (allowing proximity testing with $n^{1/m}$ queries). In contrast, our reduction does not reduce the degree at all; in fact it slightly increases it. The PCPPs for the RS-code described earlier allow us to tolerate this and verify proximity to high-degree polynomials with very small query complexity—logarithmic in the degree.

For our PCPP construction we need to modify the reduction above so that it works appropriately for pair languages. Suppose we wish for a reduction R from a pair language L to a pair language L'. Note that such a reduction can only work with the explicit input of pair languages. Furthermore, the reduction should say something about (the proximity of) the implicit input to an accepting pair. The following definition of a "systematic reduction" (borrowing a phrase from coding theory) specifies our needs.

DEFINITION 3.9 (systematic reduction). *A* systematic reduction *from a pair language L to L' is given by a pair of functions (R, m), $R : \Sigma^* \to \Sigma^*$, and $m : \Sigma^* \times \mathbb{Z}^+ \to \mathbb{Z}^+$ satisfying the following properties:*

- *For every x, the function $m'(i) = m(x, i)$ restricted to the domain $\{1, \ldots, N\}$ is injective and maps to the range $\{1, \ldots, N'\}$, where N denotes the length of the implicit input associated with x, and N' denotes the length of the implicit input associated with $R(x)$.*
- *If $y \in L_x$, then there exists a $y' \in \Sigma^*$ such that $y' \in L'_{R(x)}$ and $y_i = y'_{m(i)}$ for every $i \in \{1, \ldots, N\}$.*
- *If $y' \in L'_{R(x)}$, then $y \in L_x$, where y is the string given by $y_i = y'_{m(i)}$ for $i \in \{1, \ldots, N\}$.*

The running time of the reduction is the maximum of the computation times of $R(x)$ and $m(x, i)$, measured as a function $|x|$.

We next state a variant of Theorem 3.7 giving systematic reductions from pair languages to a language related to ALGEBRAIC-CSP. To this end, we define PAIR-ALGEBRAIC-CSP to be the pair language whose explicit inputs are instances ϕ as in Definition 3.6 and whose implicit inputs are mappings $y : \mathbb{F} \to \mathbb{F}$. A pair

(ϕ, y) is in PAIR-ALGEBRAIC-CSP iff the polynomial corresponding to y satisfies ϕ. We now state our theorem about the completeness of PAIR-ALGEBRAIC-CSP for PAIR-NTIME.

THEOREM 3.10. *There exist integers k, d such that for any proper complexity function $t : \mathbb{N}^+ \to \mathbb{N}^+$ and $L \in$ PAIR-NTIME$(t(n))$ the following hold.*

1. *L is reducible to* PAIR-ALGEBRAIC-CSP$_{k,d}$ *by a systematic reduction, given by the pair (R, m), in time* poly $t(n)$.
2. *For $x \in \{0,1\}^n$, the instance $R(x)$ is an instance of* ALGEBRAIC-CSP$_{k,d}$ *over field* GF(2^ℓ) *of size $2^\ell \le t(n)\,\mathrm{polylog}\, t(n)$ and characteristic 2, where $100(kd+1)(|H|-1) < 2^\ell \le 200(kd+1)(|H|-1)$.*

Since Theorem 3.10 is proved by a minor modification of the proof of Theorem 3.7, we prove them together in section 5.

3.3. Vanishing RS-codes and the PCP construction. The completeness of ALGEBRAIC-CSP for NP, combined with the PCPPs for Reed–Solomon codes, suggests a natural approach to building PCPs. In order to prove that some input instance x belongs to some NP language L, the verifier transforms x into an instance ϕ of ALGEBRAIC-CSP. To prove that ϕ is satisfiable, the prover can write a table of the assignment function $A : \mathbb{F} \to \mathbb{F}$. Furthermore, the prover can also write a table of the transformed polynomial $P = \phi(A)$. In order to verify that this is a valid proof of the satisfiability of ϕ, the verifier need only verify the following three properties: (1) The degrees of A and P are as specified; (2) A and P satisfy the relationship $P = \phi(A)$; and (3) P vanishes on the set H. The PCPP for Reed–Solomon codes solves the problem in (1) above. The locality in the definition of ALGEBRAIC-CSP turns out to lead to a simple solution to step (2) above as well. This leaves us to solve the problem in step (3). In this section we abstract this problem, calling it the vanishing RS-code problem, and state our result showing how to verify this. We then formalize the argument above to get a formal proof of Theorem 2.2.

We remark that the problem in step (3) is a special case of a common problem in all previous algebraic PCPs [4, 5, 20, 3, 2, 37, 39, 11, 9], where the goal is to test whether an m-variate function f, given as an oracle, is close to some polynomial p that *vanishes* on a set H^m for some prespecified subset $H \subset \mathbb{F}$. Our setting specializes this to the case where the functions are univariate (i.e., $m = 1$) as opposed to multivariate in the above mentioned results. This motivates the following pair language.

DEFINITION 3.11 (vanishing RS-codes). *A polynomial $P(z)$ over field $\mathbb{F}$ is said to* vanish *over $H \subseteq \mathbb{F}$ iff for all $h \in H, P(h) = 0$. For field $\mathbb{F}$, subsets S, H of $\mathbb{F}$, and integer d, the H-vanishing RS-code is*

$$\mathrm{VRS}(\mathbb{F}, S, H, d) \triangleq \{p \in \mathrm{RS}(\mathbb{F}, S, d) : \text{ The polynomial correspoding to } p \text{ vanishes on } H\}.$$

The pair language PAIR-VRS *is defined as follows. The explicit input is a quadruple $(\mathbb{F}, S, H, d)$, where $\mathbb{F}$ is a description of a finite field, $S, H \subseteq \mathbb{F}$, and d is an integer. The implicit input is a function $p : S \to \mathbb{F}$. The size of both the implicit and explicit inputs is $O(|S|)$. A pair $((\mathbb{F}, S, H, d), p)$ is in* PAIR-VRS *iff $p \in \mathrm{VRS}(\mathbb{F}, S, H, d)$.*

Note that in the above definition we do not require H to be a subset of S. The following lemma reduces testing proximity to the vanishing RS-code to testing proximity to the standard RS-code.

LEMMA 3.12 (PCPPs for PAIR-VRS). *Suppose a field $\mathbb{F}$, $S \subseteq \mathbb{F}$, and integer d*

are such that

$$\mathrm{RS}(\mathbb{F}, S, d) \in \mathbf{Strong\text{-}PCPP}_{s(\delta)} \left[\begin{array}{l} \mathsf{randomness}\ r, \\ \mathsf{query}\ q, \\ \mathsf{distance}\ \mathsf{Hamming}_{\mathbb{F}} \end{array} \right].$$

Then, for any $H \subset \mathbb{F}$ *and* $s'(\delta) = \min\{\delta/2, s(\delta/2)\}$,

$$\mathrm{VRS}(\mathbb{F}, S, H, d) \in \mathbf{Strong\text{-}PCPP}_{s'(\delta)} \left[\begin{array}{l} \mathsf{randomness}\ \max\{r, \log |S|\}, \\ \mathsf{query}\ q + 2, \\ \mathsf{distance}\ \mathsf{Hamming}_{\mathbb{F}} \end{array} \right].$$

Applying the previous lemma to Theorems 3.2 and 3.4 we immediately get the following corollary.

COROLLARY 3.13 (quasilinear PCPPs for vanishing RS-codes). *Let* PAIR-ADDITIVE-VRS *be the restriction of* PAIR-VRS *to pairs* $((\mathbb{F}, S, H, d), p))$, *where* $\mathbb{F}, S$ *are as defined in Theorem* 3.2. *Similarly, let* PAIR-SMOOTH-VRS *be the restriction of* PAIR-VRS *to pairs where* $\mathbb{F}, S$ *are as defined in Theorem* 3.4. *Let* $|S| = n$. *Then,*

$$\begin{array}{l}\text{PAIR-ADDITIVE-VRS,} \\ \text{PAIR-SMOOTH-VRS}\end{array} \in \mathbf{Strong\text{-}PCPP}_{\delta/\operatorname{polylog} n} \left[\begin{array}{l} \mathsf{randomness}\ \log(n \cdot \operatorname{polylog} n), \\ \mathsf{query}\ O(1), \\ \mathsf{distance}\ \mathsf{Hamming}_{\mathbb{F}} \end{array} \right].$$

Consequently (using Proposition 2.9*), for any* $s, \delta \in (0, 1)$,

$$\begin{array}{l}\text{PAIR-ADDITIVE-VRS,} \\ \text{PAIR-SMOOTH-VRS}\end{array} \in \mathbf{PCPP}_{s,\delta} \left[\begin{array}{l} \mathsf{randomness}\ \log(n \cdot \operatorname{polylog} n), \\ \mathsf{query}\ \operatorname{polylog} n, \\ \mathsf{distance}\ \mathsf{Hamming}_{\mathbb{F}} \end{array} \right].$$

Furthermore, the proof queried by the "weak" PCPP-verifier is of the same length as that queried by the "strong" one.

Lemma 3.12 generalizes to the case of multivariate polynomials and can replace the sumcheck-based protocols in previous PCP constructions [5, 3, 2, 37, 27, 24, 11, 9]. We describe this problem and our solution to it in subsection 4.4. We remark that our solution is both simpler and somewhat more efficient than the previous solutions (alas we do not need it for any of our own constructions). For our PCP construction the univariate version above suffices, as we show next.

3.3.1. Proof of quasilinear PCP—Theorem 2.2. We now show how to use the PCPP for (vanishing) RS-codes to prove Theorem 2.2, which gives short PCPs for all NTIME languages.

Proof of 2.2. *Overview.* We need to show that $L \in \mathrm{NTIME}(t(n))$ has short PCPs. We start by reducing an instance $\psi, |\psi| = n$ of L to an instance ϕ of ALGEBRAIC-CSP$_{k,d}$ of quasilinear size in n. As our proof, we request an evaluation p_0 of the polynomial A satisfying ϕ. Additionally, we request an evaluation p_1 of the polynomial from (3.1) appearing in Definition 3.6. To verify that A satisfies ϕ, we need only test that (i) p_0 is of sufficiently low degree, (ii) p_0, p_1 are consistent, i.e., p_1 is the evaluation of the polynomial from (3.1), and (iii) the polynomial corresponding to p_1 vanishes on H. We test (i) using Theorem 3.2, (iii) using Corollary 3.13, and (ii) using an additional consistency test with constant query complexity. Details follow.

Let $\phi = \{\mathrm{GF}(2^\ell), \{\textsc{Aff}_1, \ldots, \textsc{Aff}_k\}, H, C\}$ be the instance of ALGEBRAIC-CSP$_{k,d}$ that ψ is reduced to via Theorem 3.7. Let $\mathbb{F} = \mathrm{GF}(2^\ell)$ and notice that $|\mathbb{F}| = \Theta(t(n) \operatorname{polylog} t(n))$.

Probabilistically checkable proof. The verifier expects oracle access to a proof comprised of

- function $p_0 : \mathbb{F} \to \mathbb{F}$ and proof of proximity π_0 to $\mathrm{RS}(\mathbb{F}, \mathbb{F}, |H| - 1)$ as per Theorem 3.2 and
- function $p_1 : \mathbb{F} \to \mathbb{F}$ and proof of proximity π_1 to $\mathrm{VRS}(\mathbb{F}, \mathbb{F}, H, (kd+1)(|H|-1))$ as per Corollary 3.13.

Notice that Theorem 3.2 and Corollary 3.13 imply that the size of the proof is quasilinear in $|\mathbb{F}|$, which is quasilinear in $t(n)$.

Verifier operation. The verifier tosses $\log(t(n) \cdot \mathrm{polylog}\, t(n))$ coins and runs the following subtests using the same random coins in all and accepting iff all subtests accept.

- Invoke the PCPP-verifier for the Reed–Solomon code from the second part of Theorem 3.2 on explicit input $(\mathbb{F}, \mathbb{F}, |H| - 1)$, implicit input p_0, and proof π_0, using proximity parameter $\delta = 1/10k$ and soundness half.
- Invoke the PCPP-verifier for vanishing Reed–Solomon code from the second part of Corollary 3.13 on explicit input $(\mathbb{F}, \mathbb{F}, H, |H| - 1)$, implicit input p_1, and proof π_1, using proximity parameter $1/100$ and soundness half.
- Select random $x \in \mathbb{F}$, query $p_0(x), p_0(\textsc{Aff}_1(x)), \ldots, p_0(\textsc{Aff}_k(x))$ and $p_1(x)$; accept iff

$$p_1(x) = C(x, p_0(x), p_0(\textsc{Aff}_1(x)), \ldots, p_0(\textsc{Aff}_k(x))).$$

Basic parameters. Randomness and query complexity are as claimed, by Theorem 3.2 and Corollary 3.13.

Completeness. Suppose $\psi \in L$. Then $\phi \in \text{ALGEBRAIC-CSP}_{k,d}$ by Theorem 3.7. Suppose A satisfies ϕ as per Definition 3.6. Let p_0 be the evaluation of A on $\mathbb{F}$. Let

$$B(x) \triangleq C(x, A(\textsc{Aff}_1(x)), \ldots, A(\textsc{Aff}_k(x))). \tag{3.2}$$

Notice that $\deg(B) \le \deg_{x_0}(C) + \sum_{i=1}^{k} \deg_{x_i}(C) \cdot \deg(A) \le (kd+1)(|H|-1)$. Furthermore, B vanishes on H because A satisfies ϕ. Let p_1 be the evaluation of B on $\mathbb{F}$. We conclude that $p_0 \in \mathrm{RS}(\mathbb{F}, \mathbb{F}, |H| - 1)$ and $p_1 \in \mathrm{VRS}(\mathbb{F}, \mathbb{F}, H, (kd+1)(|H|-1))$, so by the completeness property of Theorem 3.2 and Corollary 3.13 there exist proofs π_0, π_1 causing the first two subtests of the verifier to accept. Finally, for all $x \in \mathbb{F}$ we have by construction

$$\begin{aligned} p_1(x) = B(x) &= C(x, A(\textsc{Aff}_1(x)), \ldots, A(\textsc{Aff}_k(x))) \\ &= C(x, p_0(\textsc{Aff}_1(x)), \ldots, p_0(\textsc{Aff}_1(x))). \end{aligned}$$

We conclude that the last subtest also accepts with probability 1, completing the proof of the completeness statement.

Soundness. Suppose $\psi \notin L$. Then $\phi \notin$ ALGEBRAIC-CSP by Theorem 3.7. There are several cases to consider:

- If p_0 is $(1/10k)$-far from $\mathrm{RS}(\mathbb{F}, \mathbb{F}, |H| - 1)$ then Theorem 3.2 implies the first subtest rejects with probability $1/2$. Similarly, if p_1 is not $(1/100)$-close to $\mathrm{VRS}(\mathbb{F}, \mathbb{F}, H, (kd+1)(|H|-1))$ then Corollary 3.13 implies the second subtest rejects with probability $1/2$.
- Otherwise, let A be the unique polynomial of degree $\le |H| - 1$ that is closest to p_0 and let $B(x)$ be as defined in (3.2). Let $p_2 : \mathbb{F} \to \mathbb{F}$ be defined by

$$p_2(x) = C(x, p_0(\textsc{Aff}_1(x)), \ldots, p_0(\textsc{Aff}_k(x))).$$

A union bound implies p_2 is $(1/10)$-close to the valuation of B on $\mathbb{F}$ because p_0 is $(1/10k)$-close to the valuation of A and $\text{AFF}_i(x)$ is uniformly distributed on $\mathbb{F}$ when x is uniformly distributed on $\mathbb{F}$.
Let B' be the (unique) polynomial closest to p_1. Notice that $B \neq B'$ because if $B = B'$, then B vanishes on H, implying A satisfies ϕ. Summing up, we have p_1 is $(1/100)$-close to B' and p_2 is $(1/10)$-close to B, where $B \neq B'$ are polynomials of degree $\leq |\mathbb{F}|/100$, so they agree on at most $1/100$ fraction of their entries. Thus, the third subtest accepts with probability at most $(1/10) + (1/100) + (1/100) < 1/2$. The soundness analysis is complete and with it we have proved Theorem 2.2. □

3.4. Systematic RS-codes and quasilinear PCPPs. We now turn to the task of building PCPPs for PAIR-NTIME languages. It is relatively straightforward to convert the PCP-verifier for ALGEBRAIC-CSP constructed in the previous section into a PCPP-verifier for PAIR-ALGEBRAIC-CSP. Unfortunately, this is not sufficient to imply a PCPP-verifier for all PAIR-NTIME languages, despite the systematic reduction given by Theorem 3.10.[2]

To get to the underlying issue, consider pair language L in PAIR-NTIME, and consider the task of proving/verifying if $(x, y) \in L$, where x is explicit and y is given as an oracle. Using the systematic reduction of Theorem 3.10, a PCPP-verifier could convert x to an instance $\phi = R(x)$ of ALGEBRAIC-CSP. It now demands proof oracles for a polynomial A satisfying ϕ, along with other ingredients as in the proof of Theorem 2.2 that prove that A satisfies ϕ. The PCPP-verifier can now verify that A satisfies ϕ with few queries. It still needs to verify that A is consistent with the implicit input y. Using the "systematic" nature of the given reduction, it also knows that y ought to be "contained" in A. More specifically, it knows that there is some subset H of A such that the evaluation of the polynomial A on the set H should be equal to the string y. In what follows we abstract this problem as that of building a PCPP-verifier for "systematic" Reed–Solomon codes. Such a verifier is given two oracles, one for a function $f : H \to \mathbb{F}$ (representing the implicit input y above), and the other for a (supposedly polynomial) function $p : S \to \mathbb{F}$, and attempts to verify that p is a polynomial of the appropriate degree that agrees with the function f. We formalize the task below and state our main result for this task.

DEFINITION 3.14 (systematic RS-codes). *For field $\mathbb{F}$, subsets $S, H \subseteq \mathbb{F}, |H| \leq |S|/2$, and integer $d \leq |S|/2$ let $\text{RS}_{\text{sys}}(\mathbb{F}, S, H, d)$ be the set of pairs of functions $(f : H \to \mathbb{F}, p : S \to \mathbb{F})$, such that $p \in \text{RS}(\mathbb{F}, S, d)$ and the polynomial corresponding to p agrees with f on H. The pair language* PAIR-RS_{sys} *is the set of pairs with explicit input $(\mathbb{F}, S, H, d)$ as above and implicit input $(f, p) \in \text{RS}_{\text{sys}}(\mathbb{F}, S, H, d)$. Similarly, the pair language* PAIR-ADDITIVE-RS_{sys} *(*PAIR-SMOOTH-RS_{sys}*, respectively) is the restriction of* PAIR-RS_{sys} *to pairs with $\mathbb{F}, S$ as in Theorem* 3.2 *(Theorem* 3.4*, respectively).*

Notice in Definition 3.14 we do not require H to be a subset of S, nor do we assume $H \cap S = \emptyset$. Furthermore, we allow d to be greater than $|H| - 1$, in which case there are several polynomials of degree d that all agree with f on H.

Recall that when building PCPP-verifiers we need to specify our distance measure. Since typically $|H| \ll |S|$, the standard Hamming distance is not a good measure because under this measure (f, p) is close to RS_{sys} whenever p is low degree, regardless

[2]Indeed, we do not know of a generic reduction that, when given a pair language L with a systematic reduction to L' and a PCPP-verifier for L', can construct an efficient PCPP-verifier for L.

of the amount of agreement between p and f. To amend this we use the following weighted Hamming distance:

$$\mathsf{Hamming}^{\frac{1}{2}}_{\mathbb{F}}((f,p),(f',p')) = \frac{1}{2}(\mathsf{Hamming}_{\mathbb{F}}(f,f') + \mathsf{Hamming}_{\mathbb{F}}(p,p')).$$

The main theorem of this section gives an efficient PCPP for the language of systematic Reed–Solomon codes. Its proof is deferred to subsection 4.3.

THEOREM 3.15 (PCPPs for systematic RS-codes).

$$\begin{array}{l}\text{PAIR-ADDITIVE-RS}_{\text{sys}},\\ \text{PAIR-SMOOTH-RS}_{\text{sys}}\end{array} \in \mathbf{Strong\text{-}PCPP}_{\delta/\operatorname{polylog} n}\left[\begin{array}{l}\mathsf{randomness}\ \log(n\cdot\operatorname{polylog} n),\\ \mathsf{query}\ \operatorname{polylog} n,\\ \mathsf{distance}\ \mathsf{Hamming}^{\frac{1}{2}}_{\mathbb{F}}\end{array}\right].$$

3.4.1. Proof of quasilinear PCPP Theorem 2.10. We now show how to use Theorems 3.10 and 3.15 to construct efficient PCPP-verifiers for all PAIR-NTIME languages.

Proof of Theorem 2.10. Let (x,y) be an instance of PAIR-$L \in \text{NTIME}(t(n))$ with $|y| = N$. The PCPP-verifier starts by reducing x to an instance $\phi = (\mathbb{F}, \{\text{AFF}_1, \ldots, \text{AFF}_k\}, H, C\}$ of ALGEBRAIC-CSP as in Theorem 3.10. Let $m : \{1,\ldots,N\} \to \mathbb{F}$ be the efficiently computable injective mapping as per Definition 3.9. Let $m([N]) = \{m(1),\ldots,m(N)\} \subseteq \mathbb{F}$. From here on we view y as a function from $m([N])$ to $\mathbb{F}$ by associating $\{0,1\}$ with the same elements in $\mathbb{F}$. Verifier expects oracle access to a proof of proximity comprised of the following:

- A PCP π for x as described in the proof of Theorem 2.2. In particular, the PCP is comprised of functions $p_0, p_1 : \mathbb{F} \to \mathbb{F}$ and subproofs π_0, π_1.
- A proof of proximity for PAIR-ADDITIVE-$\text{RS}_{\text{sys}}(\mathbb{F},\mathbb{F},m([N]),|H|-1)$, denoted π_2.

Verifier operation. The verifier invokes the PCP-verifier described in the proof of Theorem 2.2 on explicit input x and proof π. Reusing randomness, the verifier invokes the PCPP-verifier for the pair language PAIR-ADDITIVE-RS_{sys} described in Theorem 3.15 on explicit input $(\mathbb{F},\mathbb{F},m([N]),|H|-1)$, implicit input pair (y,p_0), and proof π_2. The verifier accepts iff both subverifiers accept. Notice that Theorems 2.2 and 3.15 imply the randomness and query complexity are as claimed.

Completeness. Suppose $(x,y) \in$ PAIR-L and let ϕ be the instance of ALGEBRAIC-CSP that x is reduced to as per Theorem 3.7. By Theorem 3.10, y agrees with p_0 on $m([N])$ and p_0 is an evaluation of a polynomial satisfying ϕ. Completeness now follows from Theorems 2.2 and 3.15.

Soundness. Suppose y is δ-far from L_x in distance measure $\mathsf{Hamming}^{\frac{1}{2}}_{\mathbb{F}}$. There are several cases to consider. First, if $x \notin L$ in which case L_x is empty and $\delta = 1$, then Theorem 2.2 implies the first subtest of our verifier rejects with probability $\delta/\operatorname{polylog} n$. From here on we assume $x \in L$. If p_0 is not $(1/100)$-close to an evaluation of a polynomial of degree $\le |H|-1$ that satisfies ϕ, then the soundness part of the proof of Theorem 2.2 implies the first subtest of our verifier rejects with probability $\ge 1/(100\cdot\operatorname{polylog} n) \ge \delta/\operatorname{polylog} n$. Finally, suppose p_0 is $(1/100)$-close to evaluation of a polynomial A satisfying ϕ and let $y' : m([N]) \to \mathbb{F}$ be the evaluation of A on $m([N])$. Theorem 3.10 implies y' satisfies x, so by assumption y is δ-far from y'. Thus, the pair of functions (y,p_0) is $(\delta/2)$-far from $\text{RS}_{\text{sys}}(\mathbb{F},\mathbb{F},m([N]),|H|-1)$, so

Theorem 3.15 implies that the second subtest rejects (y, p_0) with probability at least $\delta/\operatorname{polylog} n$, completing the proof of Theorem 2.10. □

3.5. Organization of the rest of the paper. We have thus far proved our main theorems (Theorems 2.2, 2.10, and 2.13) assuming the NTIME-completeness result for ALGEBRAIC-CSP (Theorems 3.7 and 3.10) and PCPPs for RS-codes (Theorem 3.2), for vanishing Reed–Solomon codes (Lemma 3.12), and for systematic Reed–Solomon codes (Theorem 3.15). In the following sections we give proofs for these claims. We remark that the sections may be read in any order. The order in which they are sequenced here merely reflects our opinion of the complexity of the proofs.

In section 4 we assume a PCPP for Reed–Solomon codes and give PCPPs for vanishing and systematic Reed–Solomon codes. We also show how to verify vanishing properties of multivariate polynomials in this section (see Lemma 4.7 in subsection 4.4), a result that may be of independent interest. In section 5 we prove the NTIME completeness of ALGEBRAIC-CSP and PAIR-ALGEBRAIC-CSP. In section 6 we give the PCPP for RS-codes over fields of characteristic two, which is our central technical result. Finally, in section 7, we give an analogous PCPP for RS-codes over smooth fields.

4. PCPPs for vanishing and systematic Reed–Solomon codes. In this section, we show how one can test various properties of polynomials given by an oracle, once we have the ability to test that an oracle is close to a polynomial.

The first such property is to verify that a univariate function f is close to some polynomial P that *vanishes* on a prespecified set H. This property was used crucially in building a PCP for NP languages in subsection 3.3.

The PCPP for vanishing RS-codes immediately leads to a PCPP to verify if two given oracles f_1 and f_2 are close to polynomials that agree on the prespecified set of inputs. (This task reduces to verifying whether $f_1 - f_2$ represents a vanishing RS-code.) We refer to this property as "agreeing" Reed–Solomon codes.

We then use the PCPP for agreeing Reed–Solomon codes to get a PCPP for systematic RS-codes. Recall that here, our goal was to take two oracles for functions $f : H \to \mathbb{F}$ and $p : S \to \mathbb{F}$ and verify that p is close to a polynomial P that agrees with f on H. This PCPP was crucial to our PCPP Theorem 2.10.

Finally, we show how to extend our PCPP for vanishing RS-codes to a PCPP for vanishing multivariate polynomial codes even though we do not use this result anywhere in the paper. (However, it was extensively used in previous PCP constructions.)

We note that all the constructions are quite simple and differ from previous such "tests" in a crucial way. Whereas previous tests of properties as above attempt to use the fact that the oracle being tested is close to a polynomial, they do not seem to explicitly use the fact that a low-degree test is available and can be used to test other functions that may be provided by the prover. Our constructions exploit this additional feature to simplify many known tests.

4.1. PCPPs for vanishing Reed–Solomon—Proof of Lemma 3.12. Recall the notion of a vanishing RS-code from Definition 3.11.

LEMMA 4.1 (Lemma 3.12, restated). *Suppose a field $\mathbb{F}$, $S \subseteq \mathbb{F}$, and an integer d are such that*

$$\mathrm{RS}(\mathbb{F}, S, d) \in \mathbf{Strong\text{-}PCPP}_{s(\delta)} \left[\begin{array}{l} \mathsf{randomness}\ r, \\ \mathsf{query}\ q, \\ \mathsf{distance}\ \mathsf{Hamming}_{\mathbb{F}} \end{array} \right].$$

Then, for any $H \subset \mathbb{F}$ and $s'(\delta) = \min\{\delta/2, s(\delta/2)\}$,

$$\mathrm{VRS}(\mathbb{F}, S, H, d) \in \mathbf{Strong\text{-}PCPP}_{s'(\delta)} \left[\begin{array}{l} \mathsf{randomness}\ \max\{r, \log|S|\}, \\ \mathsf{query}\ q+2, \\ \mathsf{distance}\ \mathsf{Hamming}_{\mathbb{F}} \end{array} \right].$$

The key observation in our proof is that a degree-d polynomial $P(z)$ vanishes on H iff the polynomial $g_H(z) \triangleq \prod_{h \in H}(z-h)$ divides $P(z)$, i.e., iff there exists a polynomial $\tilde{P}, \deg(\tilde{P}) \leq d - |H|$ such that $P(z) = g_H(z) \cdot \tilde{P}(z)$.

Proof. The PCPP-verifier for $\mathrm{VRS}(\mathbb{F}, S, H, d)$ has oracle access to implicit input $p : S \to \mathbb{F}$ and a proof combined of two parts: (i) a function $\tilde{p} : S \to \mathbb{F}$ (a supposed evaluation of $\tilde{P}$ on S) and (ii) a proof of proximity $\tilde{\pi}$ to $\mathrm{RS}(\mathbb{F}, S, d - |H|)$. Notice that the proof length is $|S| + |\tilde{\pi}|$. The verifier operates as follows.

- Toss $\max\{r, \log|S|\}$ random coins. Let R denote the random outcome.
- Invoke an assumed RS-verifier using randomness R on explicit input $(\mathbb{F}, S, d - |H|)$, implicit input $\tilde{p}$, and proof $\tilde{\pi}$. Reject if the verifier rejects. Otherwise,
- pick random $\alpha \in S$ (using randomness R); read $p(\alpha)$ and $\tilde{p}(\alpha)$; accept iff $p(\alpha) = g_H(\alpha) \cdot \tilde{p}(\alpha)$.

Notice that $g_H(\alpha)$ can be computed in time $\mathrm{poly}(|H| \cdot \log|\mathbb{F}|)$ by the verifier because H is given as an explicit input. Thus, the running time is as claimed, and so are the randomness and query complexity, by construction. Completeness follows by taking $\tilde{p}$ to be the evaluation of $\tilde{P}(z) \triangleq P(z)/g_H(z)$ and taking $\tilde{\pi}$ to be $\tilde{p}$'s proof of proximity to $\mathrm{RS}(\mathbb{F}, S, d - |H|)$.

As to the soundness, assume p is δ-far from $\mathrm{VRS}(\mathbb{F}, S, H, d)$. If $\tilde{p}$ is $\delta/2$-far from $\mathrm{RS}(\mathbb{F}, S, d - |H|)$, then by assumption the RS-verifier rejects $\tilde{p}$ with probability at least $s(\delta/2)$ and we are done. Otherwise, $\tilde{p}$ is $\delta/2$-close to some polynomial Q of degree $\leq d - |H|$. Let $q : S \to \mathbb{F}$ be the evaluation of Q on S. Notice that the function $\tilde{p} \cdot g_H$ is $\delta/2$-close to $q \cdot g_H$ and the latter function is, by construction, a codeword of $\mathrm{VRS}(\mathbb{F}, S, H, d)$. By assumption, p is $\delta/2$-far from $\tilde{p} \cdot g_H$; hence the last subtest of the verifier rejects with probability at least $\delta/2$. This completes our proof. ∎

4.2. Agreeing Reed–Solomon codes. We now show how to extend the PCPP of the previous section to test if two polynomials agree on a given set. Two polynomials $P_1(z), P_2(z)$ are said to *agree* on $H \subseteq \mathbb{F}$ if $P_1(z) = P_2(z)$ for all $z \in H$. Below we formalize the problem of testing agreement.

DEFINITION 4.2 (agreeing RS-codes). *For field $\mathbb{F}$, subsets $S, H \subseteq \mathbb{F}$, and integers $|S|/2 \geq d_1 \geq d_2$ let $\mathrm{RS}_{\mathrm{agr}}(\mathbb{F}, S, H, d_1, d_2)$ be the set of pairs of functions $p_1, p_2 : S \to \mathbb{F}$, such that $p_1 \in \mathrm{RS}(\mathbb{F}, S, d_1), p_2 \in \mathrm{RS}(\mathbb{F}, S, d_2)$, and the polynomial corresponding to p_1 agrees with the polynomial corresponding to p_2 on H. The pair language* PAIR-$\mathrm{RS}_{\mathrm{agr}}$ *is the set of pairs with explicit input $(\mathbb{F}, S, H, d_1, d_2)$ as above and implicit input $(p_1, p_2) \in \mathrm{RS}_{\mathrm{agr}}(\mathbb{F}, S, H, d_1, d_2)$. Similarly, the pair language* PAIR-ADDITIVE-$\mathrm{RS}_{\mathrm{agr}}$ *(*PAIR-SMOOTH-$\mathrm{RS}_{\mathrm{agr}}$*, respectively) is the restriction of* PAIR-$\mathrm{RS}_{\mathrm{agr}}$ *to pairs with $\mathbb{F}, S$ as in Theorem* 3.2 *(Theorem* 3.4*, respectively).*

LEMMA 4.3. *Assume a field $\mathbb{F}$, $S \subseteq \mathbb{F}$, and integers $d_2 \le d_1 \le |S|/2$ satisfy*

$$\begin{matrix}\mathrm{RS}(\mathbb{F}, S, d_1),\\ \mathrm{RS}(\mathbb{F}, S, d_2),\\ \mathrm{VRS}(\mathbb{F}, S, d_1)\end{matrix} \in \mathbf{Strong\text{-}PCPP}_{s(\delta,|S|)} \left[\begin{array}{l}\text{randomness } r,\\ \text{query } q,\\ \text{distance Hamming}_{\mathbb{F}}\end{array}\right],$$

where s is monotone nondecreasing in δ. Then for any $H \subset \mathbb{F}$,

$$\mathrm{RS}_{\mathrm{agr}}(\mathbb{F}, S, H, d_1, d_2) \in \mathbf{Strong\text{-}PCPP}_{s(\delta/8,|S|)} \left[\begin{array}{l}\text{randomness } r,\\ \text{query } q+2,\\ \text{distance Hamming}_{\mathbb{F}}\end{array}\right].$$

Proof. The main idea is that $P_1(z)$ agrees with $P_2(z)$ on H iff $P_1(z) - P_2(z)$ vanishes on H, so we apply the PCPP from Lemma 3.12 to this difference. Details follow.

The verifier for $\mathrm{RS}_{\mathrm{agr}}(\mathbb{F}, S, H, d_1, d_2)$ has oracle access to implicit inputs $p_1, p_2 : S \to \mathbb{F}$ and proof of proximity comprised of

- proof of proximity π_1 to $\mathrm{RS}(\mathbb{F}, S, d_1)$,
- proof of proximity π_2 to $\mathrm{RS}(\mathbb{F}, S, d_2)$, and
- proof of proximity π_3 to $\mathrm{VRS}(\mathbb{F}, S, d_1)$.

The verifier's operation is to invoke the following three subtests using the same randomness across all tests and accepting iff all of them accept:

- Invoke verifier for $\mathrm{RS}(\mathbb{F}, S, d_1)$ on implicit input p_1 and proof π_1.
- Invoke verifier for $\mathrm{RS}(\mathbb{F}, S, d_2)$ on implicit input p_2 and proof π_2.
- Invoke verifier for $\mathrm{RS}_H(\mathbb{F}, S, d_1)$ on implicit input $p_1 - p_2$ and proof π_3. Querying $p_1 - p_2$ on $\alpha \in S$ is performed by querying $p_1(\alpha), p_2(\alpha)$ and taking their difference.

All properties can be checked as in the proof of Lemma 3.12. For an illustration, consider the soundness. Suppose the pair of functions (p_1, p_2) is δ-far from $\mathrm{RS}_{\mathrm{agr}}(\mathbb{F}, S, H, d_1, d_2)$. If either one of p_1, p_2 is $\delta/8$-far from $\mathrm{RS}(\mathbb{F}, S, d_1), \mathrm{RS}(\mathbb{F}, S, d_2)$, respectively, then (the first/second subtest of) the verifier rejects with probability $s(\delta/8, |S|)$. Otherwise, $q = (p_1 - p_2)$ is $\delta/4$-close to a polynomial of degree d_1 that by assumption does not vanish on H. Since $d_1 \le |S|/2$ and $\delta \le 1$ we conclude q is $1/4$-far from $\mathrm{VRS}(\mathbb{F}, S, H, d_1)$ in which case the third subtest of the verifier rejects with probability $\ge s(1/4, |S|) \ge s(\delta/8, |S|)$. The last inequality follows from monotonicity of s. ∎

The previous lemma combined with Lemma 3.12 and Theorems 3.2 and 3.4 immediately implies the following.

COROLLARY 4.4 (PCPPs for agreeing RS-codes).

$$\begin{matrix}\text{PAIR-ADDITIVE-RS}_{\mathrm{agr}},\\ \text{PAIR-SMOOTH-RS}_{\mathrm{agr}}\end{matrix} \in \mathbf{Strong\text{-}PCPP}_{\delta/\operatorname{polylog} n} \left[\begin{array}{l}\text{randomness } \log(n \cdot \operatorname{polylog} n),\\ \text{query } O(1),\\ \text{distance Hamming}_{\mathbb{F}}\end{array}\right].$$

4.3. PCPP for systematic Reed–Solomon codes: Proof of Theorem 3.15. Recall the definition of the systematic RS-code (Definition 3.14) and the related notation presented in subsection 3.4. We now prove Theorem 3.15, restated below.

THEOREM 4.5 (Theorem 3.15, restated).

$$\begin{array}{l}\text{PAIR-ADDITIVE-RS}_{\text{sys}},\\ \text{PAIR-SMOOTH-RS}_{\text{sys}}\end{array} \in \mathbf{Strong\text{-}PCPP}_{\delta/\operatorname{polylog} n}\left[\begin{array}{l}\text{randomness } \log(n\cdot\operatorname{polylog} n),\\ \text{query } \operatorname{polylog} n,\\ \text{distance Hamming}_{\mathbb{F}}^{\frac{1}{2}}\end{array}\right].$$

To prove the theorem we apply induction to the following lemma, the proof of which appears below. Roughly, the lemma says that if we have PCPPs for systematic and agreeing RS-codes of size $n/2$, then we can construct systematic codes of size n. Intuitively, the proof is as follows. To verify that a message is indeed encoded by a codeword, we split the message into two parts of equal length. We ask for the encoding of each of these submessages and verify agreement of the subencodings with the encoding of the large message. This part uses PCPPs for agreeing codes described in the previous section. Then we pick one of the submessages at random and verify that it is consistent with its supposed subencoding and for this part we use induction. Details follow.

LEMMA 4.6. *If $H, S \subseteq \mathbb{F}$, and $d \le |S|/2$ satisfy the following conditions:*

1. *there exist $S_0, S_1 \subseteq S, |S_0|, |S_1| = |S|/2$, and a partition $H_0 \cup H_1 = H, |H_0|, |H_1| = |H|/2$ and these sets are computable in time t_0,*
2. *we have*

$$\mathrm{RS}(\mathbb{F}, S, d) \in \mathbf{Strong\text{-}PCPP}_{s_1(\delta,|S|)}\left[\begin{array}{l}\text{randomness } r_1,\\ \text{query } q_1,\\ \text{distance Hamming}_{\mathbb{F}}\end{array}\right],$$

3. *for $i = 0, 1$ we have*

$$\mathrm{RS}_{\text{agr}}(\mathbb{F}, S, H_i, d, |H_i|-1) \in \mathbf{Strong\text{-}PCPP}_{s_2(\delta,|S|)}\left[\begin{array}{l}\text{randomness } r_2,\\ \text{query } q_2,\\ \text{distance Hamming}_{\mathbb{F}}\end{array}\right],$$

4. *for $i = 0, 1$ we have*

$$\mathrm{RS}_{\text{sys}}(\mathbb{F}, S_i, H_i, |H_i|-1) \in \mathbf{Strong\text{-}PCPP}_{s_3(\delta,|S_i|)}\left[\begin{array}{l}\text{randomness } r_3,\\ \text{query } q_3,\\ \text{distance Hamming}_{\mathbb{F}}^{\frac{1}{2}}\end{array}\right],$$

and s_3 is subadditive, i.e., $s_3(\delta_0, |S_0|) + s_3(\delta_1, |S_1|) \ge s_3(\delta_0 + \delta_1, |S_i|)$,

then for any $0 < \alpha < 1/16$,

$$\mathrm{RS}_{\text{sys}}(\mathbb{F}, S, H, d) \in \mathbf{Strong\text{-}PCPP}_{s(\delta,|S|)}\left[\begin{array}{l}\text{randomness } r,\\ \text{query } q,\\ \text{distance Hamming}_{\mathbb{F}}^{\frac{1}{2}}\end{array}\right],$$

where

$$\begin{aligned} s(\delta, |S|) &= \min\left\{s_1(\alpha\delta, |S|), s_2(\alpha\delta, |S|)/2, \frac{1}{2}s_3((2-\alpha)\delta, |S|/2)\right\},\\ r &= \max\{r_1, r_2, r_3\} + 1,\\ q &= q_1 + q_2 + q_3. \end{aligned}$$

Proof of Theorem 3.15. Consider first the case of PAIR-ADDITIVE-$\mathrm{RS}_{\mathrm{sys}}$. We need to prove there exists a constant $c \geq 1$ that will be specified later such that for any $\mathbb{F}$ of characteristic 2, linear space $S \subseteq \mathbb{F}$, $|S| = n$, set $H \subset \mathbb{F}, |H| = 2^\ell \leq |S|/2$, and $d \leq |S|/2$ we have

$$\mathrm{RS}_{\mathrm{sys}}(\mathbb{F}, S, H, d) \in \mathbf{Strong\text{-}PCPP}_{\delta/(\log n)^c} \left[\begin{array}{l} \mathsf{randomness}\ \log(n \cdot \mathrm{polylog}\, n), \\ \mathsf{query}\ \mathrm{polylog}\, n, \\ \mathsf{distance}\ \mathsf{Hamming}_{\mathbb{F}} \end{array} \right].$$

Our proof is by induction on n, using Lemma 4.6. The base case of constant n follows because a verifier can query all entries in the implicit input and reject any pair of functions (f, p) that is not in $\mathrm{RS}_{\mathrm{sys}}(\mathbb{F}, S, H, d)$.

As to the inductive step, partition H into two equal sets H_0, H_1 arbitrarily. Let $S_0 = S_1$ be a $(k-1)$-dimensional space. Part 1 of Lemma 4.6 holds by construction. Parts 2 and 3 follow from Theorem 3.2 and Corollary 4.4, respectively, with

$$s_1(\delta, n), s_2(\delta, n) \geq \delta/(\log n)^{c'}, \quad q_1, q_2 = \mathrm{polylog}\, n, \quad r_1, r_2 = \log(n \cdot \mathrm{polylog}\, n),$$

where c' is a constant. Part 4 holds by induction with

$$s_3(\delta, n/2) = \delta/((\log n) - 1)^c, \quad q_3 = \mathrm{poly}((\log n) - 1), \quad r_3 \leq \log(n \cdot \mathrm{polylog}\, n).$$

Notice that s_3 is subadditive. Apply Lemma 4.6 with $\alpha = 1/\log n$. Randomness and query complexities follow immediately and the verifier's running time is polynomial. As to soundness, notice that $\frac{1}{2} s_3((2-\alpha)\delta, n/2) \geq (1 - 1/\log n)\delta/(\log n - 1)^c \geq \delta/\log^c n$. Thus, by selecting $c > c' + 1$ we conclude that

$$s(\delta, n) \geq \min\{\delta/\log^{c'+1} n, \delta/\log^c n\} \geq \delta/\log^c n.$$

This completes the proof.

Regarding PAIR-SMOOTH-$\mathrm{RS}_{\mathrm{sys}}$, change $S = \langle \omega \rangle$ and $S_0 = S_1 = \langle \omega^2 \rangle$, and use Theorem 3.4. The rest of the proof is identical. □

Proof of Lemma 4.6. To prove that p is an evaluation of a polynomial P that agrees with f on H, we request that the prover provide evaluations of the polynomials that agree with P on the two partitions of H. We test agreement of p with these two polynomials, denoted p_0, p_1, and then split f to two corresponding parts and recurse. Details follow. We describe the proof of proximity, followed by the verifier's operation, and conclude with completeness and soundness analysis.

Proof of proximity. The proof for the implicit input pair $f : H \to \mathbb{F}, p : S \to \mathbb{F}$ is defined recursively. In the base case ($|S| = O(1)$) the proof is empty. Otherwise, it is comprised of

- one proof of proximity π to $\mathrm{RS}(\mathbb{F}, S, d)$,
- two functions $p_0, p_1 : S \to \mathbb{F}$,
- two proofs of proximity π_0, π_1 to $\mathrm{RS}_{\mathrm{agr}}(\mathbb{F}, S, H_0, d, |H_0| - 1)$ and $\mathrm{RS}_{\mathrm{agr}}(\mathbb{F}, S, H_1, d, |H_1| - 1)$, respectively, and
- two proofs of proximity π'_0, π'_1 to $\mathrm{RS}_{\mathrm{sys}}(\mathbb{F}, S_0, H_0, |H_0| - 1)$ and to $\mathrm{RS}_{\mathrm{sys}}(\mathbb{F}, S_1, H_1, |H_1| - 1)$, respectively, defined recursively.

Verifier operation. Let $f_i : H_i \to \mathbb{F}$ be the restriction of the function f to domain H_i and let p'_i be the restriction of the function p_i to domain S_i for $i = 0, 1$. The verifier tosses $r = \max\{r_1, r_2, r_3\} + 1$ coins, sets $i \in \{0, 1\}$ according to the first coin, and performs the following subtests reusing the remaining $r - 1$ coins across different tests:

- Invoke verifier for $\mathrm{RS}(\mathbb{F}, S, d)$ on input p and proof π.
- Invoke verifier for $\mathrm{RS}_{\mathrm{agr}}(\mathbb{F}, S, H_i, d, |H_i| - 1)$ on input pair (p, p_i) and proof π_i.
- Invoke verifier for $\mathrm{RS}_{\mathrm{sys}}(\mathbb{F}, S_i, H_i, |H_i| - 1)$ on input pair (f_i, p'_i) and proof π'_i.
- Accept iff all aforementioned tests accept.

Basic properties. The randomness is r, by construction. The query complexity is the sum of queries made by the various subtests, as claimed.

Completeness. Assume $(f, p) \in \mathrm{RS}_{\mathrm{sys}}(\mathbb{F}, S, H, d)$. Since p is of degree $\leq d$ there exists a proof π accepted by the first subtest of the verifier with probability one. Let p_i be the polynomial of degree $|H_i| - 1$ that agrees with f_i (on H_i). By construction p agrees with p_i on H_i. Thus, there exist proofs π_i accepted by the second subtest of the verifier with probability 1. Finally, notice that $(f_i, p'_i) \in \mathrm{RS}_{\mathrm{sys}}(\mathbb{F}, S_i, H_i, |H_i| - 1)$, so there exist subproofs π'_i causing the third test of the verifier to accept with probability one.

Soundness. Assume the distance of (f, p) from $\mathrm{RS}_{\mathrm{agr}}(\mathbb{F}, S, H, d)$ is exactly δ. There are several cases to consider. (i) If p is $\alpha\delta$-far from $\mathrm{RS}(\mathbb{F}, S, d)$, then the first test of the verifier rejects with probability $s_1(\alpha\delta, |S|)$. (ii) If for some $i \in \{0, 1\}$ the distance of (p, p_i) from $\mathrm{RS}_{\mathrm{agr}}(\mathbb{F}, S, H_i, d, |H_i| - 1)$ is greater than $\alpha\delta$, then the second test of the verifier rejects with probability $s_2(\alpha\delta, |S|)/2$. The factor half decrease in rejection probability is due to the random selection of i. (iii) Otherwise, because (i) does not hold and $d \leq |S|/2$ and $\alpha < 1/16$, we conclude that p is $\alpha\delta$-close to a unique polynomial P, so f is $((2 - \alpha)\delta)$-far from the evaluation of P on H. Similarly, because (ii) does not hold, we conclude that each of p_0, p_1 is 1/8-close to the unique polynomial agreeing with P on H_i.

For $i = 0, 1$, let δ_i be the distance of (f_i, p'_i) from $\mathrm{RS}_{\mathrm{sys}}(\mathbb{F}, S_i, H_i, |H_i| - 1)$ using measure $\mathsf{Hamming}^{\frac{1}{2}}_{\mathbb{F}}$. Notice that $\delta_0 + \delta_1 \geq (2 - \alpha)\delta$ because p_i is 1/8-close to the evaluation of P on H, so p'_i is 1/4-close to the evaluation of P on H_i, while f is $((2 - \alpha)\delta)$-far from it. By induction, the rejection probability of the third subtest in this case is at least

$$\frac{1}{2}(s_3(\delta_0, |S|/2) + s_3(\delta_1, |S|/2)) \geq \frac{1}{2} s_3(\delta_0 + \delta_1, |S|/2) \geq \frac{1}{2} s_3((2 - \alpha)\delta, |S|/2).$$

Summing up, our rejection probability is at least as claimed and this completes our proof. □

4.4. PCPPs for multivariate polynomials and vanishing Reed–Muller codes. Finally, we give a generalization of Lemma 3.12 to the case of multivariate polynomials. This generalization would suffice to replace the sumcheck-based protocols in previous PCP constructions [5, 3, 2, 37, 27, 24, 11, 9].

In the multivariate problem we are given sets $S, H \subset \mathbb{F}$ and oracle access to a multivariate function $f : S^m \to \mathbb{F}$. We are asked to verify that f is close to a polynomial of degree $\leq d$ in each variable that evaluates to zero on H^m. Once again, we do not need to assume $H \subset S$. Recall that evaluations of low-degree multivariate polynomials form the well-known *Reed–Muller* code. We denote by $\mathrm{RM}(\mathbb{F}, S, d, m)$ the set of functions $p : S^m \to \mathbb{F}^m$ that are evaluations of m-variate polynomials of maximal individual degree d. We denote by $\mathrm{VRM}(\mathbb{F}, S, H, d, m)$ its subcode consisting of all evaluations of polynomials that vanish on H^m. Our main lemma of this section is the following.

LEMMA 4.7 (multivariate zero testing). *Suppose field $\mathbb{F}$, set $S \subseteq \mathbb{F}$, and integers d, m satisfy*

$$\mathrm{RM}(\mathbb{F}, S, d, m) \in \mathbf{Strong\text{-}PCPP}_{s(\delta)} \left[\begin{array}{l} \mathsf{randomness}\ r, \\ \mathsf{query}\ q, \\ \mathsf{distance}\ \mathsf{Hamming}_{\mathbb{F}} \end{array} \right].$$

Then, for any $H \subset \mathbb{F}$ and $s'(\delta) = \min\{s(\delta), 1 - ((m+1)\delta + (\frac{d}{|S|})^m)\}$,

$$\mathrm{VRM}(\mathbb{F}, S, H, d, m) \in \mathbf{Strong\text{-}PCPP}_{s'(\delta)} \left[\begin{array}{l} \mathsf{randomness}\ r + m \log |S|, \\ \mathsf{query}\ (m+1)(q+1), \\ \mathsf{distance}\ \mathsf{Hamming}_{\mathbb{F}} \end{array} \right].$$

Notice that the query complexity of previous solutions to this problem depended polynomially on the size of H. Our solution has query complexity that is independent of H and is based on a straightforward characterization of VRM that resembles Alon's combinatorial Nullstellensatz [1]). Before proving the lemma we first recall some (relatively well-known) results on testing proximity to Reed–Muller codes.

Testing proximity to multivariate polynomials. It is easy to extend the PCPP for the RS-code into one for the Reed–Muller code (based on multivariate polynomials), given the extensive literature on testing multivariate polynomials using axis parallel lines [4, 5, 20, 3, 37, 21].

For a set $S \subseteq \mathbb{F}$ and an m-variate function $f : S^m \to \mathbb{F}$, let $\delta_m^d(f)$ be the fractional distance of f from $\mathrm{RM}(\mathbb{F}, S, d, m)$. Let $\delta_{m,i}^d(f)$ denote the fractional distance of f from a polynomial of degree d in the ith variable, and an unbounded degree in all other variables. Finally, let $\mathbb{E}[\delta_{m,i}^d(f)]$ be the expectation of $\delta_{m,i}^d$ over random $i \in [m]$. The following lemma is a rephrasing of [3, Lemma 5.2.1]. Notice that Lemma 6.13 is a special case of it with tighter parameters.

LEMMA 4.8 (see [3]). *There exists a universal constant c such that for every $S \subset \mathbb{F}$ such that $|S| \geq \mathrm{poly}(m, d)$,*

$$\delta_m^d(f) \leq c \cdot m \cdot \mathbb{E}[\delta_{m,i}^d(f)].$$

This lemma and Theorem 3.2 imply short PCPPs for Reed–Muller codes.

LEMMA 4.9 (RM PCP of proximity). *Let $S \subset \mathbb{F}$ and d, m be integers such that $|S| \geq \mathrm{poly}(m, d)$ for the polynomial of Lemma 4.8 and suppose*

$$\mathrm{RS}(\mathbb{F}, S, d) \in \mathbf{Strong\text{-}PCPP}_{s(\delta)} \left[\begin{array}{l} \mathsf{randomness}\ r, \\ \mathsf{query}\ q, \\ \mathsf{distance}\ \mathsf{Hamming}_{\mathbb{F}} \end{array} \right].$$

Then

$$\mathrm{RM}(\mathbb{F}, S, d, m) \in \mathbf{Strong\text{-}PCPP}_{s(\delta)/m} \left[\begin{array}{l} \mathsf{randomness}\ r + \log(m \cdot |S|^{m-1}), \\ \mathsf{query}\ q, \\ \mathsf{distance}\ \mathsf{Hamming}_{\mathbb{F}} \end{array} \right].$$

Proof. The proof for a purported RM-codeword is the collection of proofs of proximity for each axis parallel line (to the RS-code). A line parallel to the ith axis is $\{(b_1, \ldots, b_{i-1}, x_i, b_{i+1}, \ldots, b_m) : x_i \in S\}$, where $b_1, \ldots, b_m \in S$. The verifier selects a random axis parallel line and invokes the RS-verifier of Definition 6.7 on the line and its proof. The proof follows from Lemma 4.8. □

Remark 4.10. A more query-efficient test can be constructed when $S = \mathbb{F}$. Instead of axis parallel lines, we use an ϵ-biased set of directions as in [11]. This results in proofs of similar length and query complexity and slightly larger randomness, but the soundness is as large as $\Omega(s(\delta))$ and independent of m.

Testing proximity to vanishing multivariate polynomials. We now move to the proof of Lemma 4.7. The catch in immediately extending the univariate verifier of Lemma 3.12 to even the bivariate case is that the "factoring" concept does not extend immediately. Specifically, if we are given that a bivariate polynomial $Q(x, y)$ has a zero at (α, β) this does not imply that $Q(x, y)$ has some nice factors. However, one can abstract a nice property about Q from this zero. Specifically, we can say that there exist polynomials $A(x, y), B(x, y)$ (of the right degree) such that $Q(x, y) = A(x, y) \cdot (x - \alpha) + B(x, y) \cdot (y - \beta)$. Thus to prove that $Q(\alpha, \beta) = 0$, we may ask the prover to give an evaluation of $Q(x, y)$, $A(x, y)$, and $B(x, y)$. We can then test that Q, A, and B are of low degree and that they satisfy the identity above. Extending this idea to m-variate polynomials that are zero on an entire generalized rectangle is straightforward. The technical lemma giving the identity is included below. The lemma is also a key ingredient in Alon's combinatorial Nullstellensatz [1]. We include a proof for completeness.

LEMMA 4.11. *Let $Q(x_1, \ldots, x_m)$ be a polynomial over $\mathbb{F}_Q$ of degree d in each of m variables. Let $H \subseteq \mathbb{F}_Q$ and let $g_H(z) \stackrel{\text{def}}{=} \prod_{\beta \in H}(z - \beta)$. Then Q evaluates to 0 on H^m iff there exist m-variate polynomials $A_1, \ldots, A_m$ of individual degree at most d such that $Q(\vec{x}) = \sum_{i=1}^{m} A_i(\vec{x}) \cdot g_H(x_i)$.*

Remark 4.12. The lemma above is intentionally sloppy with degree bounds. While tighter degree bounds on A_i's can be obtained, this will not be needed for our PCPs.

Proof. One direction is immediate. If $Q(\vec{x}) = \sum_{i=1}^{m} A_i(\vec{x}) \cdot g_H(x_i)$ then $Q(\vec{\alpha}) = 0$ for every $\vec{\alpha} \in H^m$. The other direction is proved in three steps. First, we show that for any polynomial $P(x_1, \ldots, x_m)$ of degree d_j in x_j, and any $i \in \{1, \ldots, m\}$, there exist polynomials $B(x_1, \ldots, x_m)$ and $C(x_1, \ldots, x_m)$ of degree at most d_j in x_j, with the degree of C in x_i being at most $\min\{d_j, |H|-1\}$, such that $P(\vec{x}) = B(\vec{x}) \cdot g_H(x_i) + C(\vec{x})$. Second, we show that there exist polynomials $A_1, \ldots, A_m$ and R with the A_i's having degree at most d in each variable and R having degree at most $|H|-1$ in each variable such that $Q(\vec{x}) = \sum_{i=1}^{m} A_i(\vec{x}) \cdot g_H(x_i) + R(\vec{x})$, where Q is the polynomial from the lemma statement. In the final step, we show that $R(\vec{x}) = 0$, concluding the proof.

Step 1. Recall that any polynomial $f(x_i)$ can be written as $q(x_i) \cdot g_H(x_i) + r(x_i)$, where r has degree less than $|H|$. Applying this fact to the monomials x_i^D for nonnegative D we find that there exist polynomials $q_D(x_i)$ and $r_D(x_i)$, with degree of q_D being at most D and degree of r_D being less than $|H|$, such that $x_i^D = q_D(x_i) \cdot g_H(x_i) + r_D(x_i)$. Now consider any polynomial $P(x_1, \ldots, x_m)$ of degree d_i in x_i. Suppose $P(\vec{x}) = \sum_{D=0}^{d_i} P_i(\vec{x}') \cdot x^D$, where $\vec{x}' = (x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_m)$. Writing the monomials x_i^D in terms of the q_D's and r_D's, we get

$$P(\vec{x}) = \left(\sum_{D=0}^{d_i} P_i(\vec{x}') q_D(x_i) \right) \cdot g_H(x_i) + \left(\sum_{D=0}^{d_i} P_i(\vec{x}') r_D(x_i) \right).$$

Letting $B(\vec{x}) = \sum_{D=0}^{d_i} P_i(\vec{x}') q_D(x_i)$ and $C(\vec{x}) = (\sum_{D=0}^{d_i} P_i(\vec{x}') r_D(x_i))$ yields the polynomials as claimed. In particular, the degrees of B and C in any variable are no more than that of P, and the degree of C in x_i is smaller than $|H|$.

Step 2. We now claim that there exist polynomials $A_1, \ldots, A_m$ and $R_0, \ldots, R_m$ such that for every $j \in \{0, \ldots, m\}$, $Q(\vec{x}) = \sum_{i=0}^{j} A_i(\vec{x}) \cdot g_H(x_i) + R_j(\vec{x})$, with A_i's being of degree at most d in each variable and R_j being of degree less than $|H|$ in $x_1, \ldots, x_j$ and of degree at most d in the remaining variables. The proof is straightforward by induction on j, with the induction step using Step 1 on the polynomial $P() = R_j()$ and the variable x_{j+1}. The final polynomials $A_1, \ldots, A_m$ and $R = R_m$ are the polynomials as required to yield the subclaim of this step.

Step 3. Finally, we note that for every $\vec{\alpha} \in H^m$, we have $R(\vec{\alpha}) = Q(\vec{\alpha}) - \sum_{i=1}^{m} A_i(\vec{\alpha}) \cdot g_H(\alpha_i) = 0 - \sum_{i=1}^{m} 0 = 0$. But R is a polynomial of degree less than $|H|$ in each variable and is zero on the entire box H^m. This can happen only if $R \equiv 0$. Thus we get that $Q(\vec{x}) = \sum_{i=1}^{m} A_i(\vec{x}) \cdot g_H(x_i)$, with the A_i's being of degree at most d in each variable, as required in the completeness condition. ∎

Proof of Lemma 4.7. As a proof of the proximity of $q \in \mathbb{F}^{S^m}$ to the code $\mathrm{VRM}(\mathbb{F}, S, d, m)$ our verifier expects (i) the evaluations of $A_1, \ldots, A_m$ from Lemma 4.11 on S^m, denoted $a_1, \ldots, a_m$, and (ii) for each of $q, a_1, \ldots, a_m$, a proof of proximity of A_i to $\mathrm{RM}(\mathbb{F}, S, d, m)$. Proof length is as claimed. The verifier operates as follows. First, it tests proximity of each of $q, a_1, \ldots, a_m$ to $\mathrm{RM}(\mathbb{F}, S, d, m)$. Then, a random $\langle \alpha_1, \ldots, \alpha_m \rangle \in S^m$ is selected and the verifier accepts iff $q(\vec{\alpha}) = \sum_{i=1}^{m} g_H(\alpha_i) \cdot a_i(\vec{\alpha})$. The query complexity is as claimed. Completeness follows from Lemma 4.11. As to the soundness, if any of $q, a_1, \ldots, a_m$ is δ-far from $\mathrm{RM}(\mathbb{F}, S, d, m)$, the verifier rejects with probability $s(\delta)$. Otherwise, q is δ close to a polynomial Q that does not vanish on H^m. If $A_1, \ldots, A_m$ are the polynomials closest to $a_1, \ldots, a_m$, respectively, then by Lemma 4.11 we get $Q(\vec{x}) \neq \sum_i A_i(\vec{x}) \cdot g_H(x_i)$ and Q has degree at most d in each variable. Thus, the two polynomials agree on $\leq d^m$ points, so the acceptance probability of the verifier is $\leq (m+1)\delta + (\frac{d}{|S|})^m$ as claimed. ∎

5. Quasilinear reductions of NTIME(n) to ALGEBRAIC-CSP. In this section we show the completeness of ALGEBRAIC-CSP for NTIME classes, thereby proving Theorem 3.7 (restated below). We also show how to modify this proof to get a proof of Theorem 3.10, which shows the completeness of PAIR-ALGEBRAIC-CSP for PAIR-NTIME classes under systematic reductions.

THEOREM 5.1 (Theorem 3.7, restated). *There exist integers k, d such that for any proper complexity function $t : \mathbb{N}^+ \to \mathbb{N}^+$ and $L \in \mathrm{NTIME}(t(n))$ the following hold:*

1. *L is reducible to $\mathrm{ALGEBRAIC\text{-}CSP}_{k,d}$ in time $\mathrm{poly}\, t(n)$.*
2. *An instance of L of size n is reduced to an instance of $\mathrm{ALGEBRAIC\text{-}CSP}_{k,d}$ over field $\mathrm{GF}(2^\ell)$ of size $2^\ell \leq t(n)\,\mathrm{polylog}\, t(n)$ and characteristic 2, where $100(kd+1)(|H|-1) < 2^\ell \leq 200(kd+1)(|H|-1)$.*

5.1. Warmup—quadratic size reduction. To illustrate the ideas used in the proof of Theorem 3.7, we start with a simpler proof of a weaker version of it, where the size blowup is quadratic rather than quasilinear. Our starting point is the following NP-complete language essentially from Cook's theorem [14]. (See also [36, proof of Theorem 8.2]).

DEFINITION 5.2 (domino tiling). *A domino tiling instance over alphabet Σ is a tuple of constraints $\psi = \{\hat{C}_{ij} : i, j \in \{0, \ldots, n-2\}\}$, where each constraint is a mapping $\hat{C}_{ij} : \Sigma^3 \to \{\mathtt{accept}, \mathtt{reject}\}$. An instance is satisfiable iff there exists a mapping $\hat{A} : \{0, n^2 - 1\} \to \Sigma$ such that for all $i, j \in \{0, \ldots, n-2\}$*

$$\hat{C}_{i,j}(\hat{A}(in+j), \hat{A}(in+j+1), \hat{A}((i+1)n+j) = \mathtt{accept}.$$

The language DOMINO-TILING$_\Sigma$ *is the set of all satisfiable instances over alphabet* Σ.

THEOREM 5.3 (DOMINO-TILING is NTIME-complete [14]). *There exists a finite size alphabet* Σ *such that if* $L \in \text{NTIME}(t(n))$ *for a proper complexity function* $t : \mathbb{N}^+ \to \mathbb{N}^+$, *then* L *is reducible to* DOMINO-TILING$_\Sigma$ *under quadratic size reductions.*

Our warmup version of Theorem 3.7 is the following.

THEOREM 5.4. *For every finite alphabet* Σ, *the language* DOMINO-TILING$_\Sigma$ *is reducible under linear sized reductions to* ALGEBRAIC-CSP$_{4,|\Sigma|}$.

Notice that although the reduction from DOMINO-TILING to ALGEBRAIC-CSP is linear, the reduction from a language $L \in \text{NTIME}(t(n))$ to DOMINO-TILING incurs a quadratic size blowup.

Proof. We reduce an instance ψ of DOMINO-TILING$_\Sigma$ to an instance $\phi = (\mathbb{F}, \{\text{AFF}_1, \ldots, \text{AFF}_4\}, H, C)$ as in Definition 3.6. We will make crucial use of the fact that the constraint $\hat{C}_{ij}$ depends on assignment entries whose indices are *linear functions* of i and j.

Fix $\mathbb{F}$ to be any finite field satisfying $100n^2 < |\mathbb{F}| \leq 200n^2$. Let ω be a generator of $\mathbb{F}^*$. Associate Σ with arbitrary elements of $\mathbb{F}$. View an assignment to ψ as a mapping $\hat{A} : \{w^{in+j} : i, j \in \{0, \ldots, n-1\}\} \to \Sigma$ where the domain and range of this mapping are subsets of $\mathbb{F}$. The arithmetized instance ϕ will be satisfied only by polynomials A that are a low-degree extension of an assignment $\hat{A}$ that satisfies ψ. Thus, the constraint polynomial C will ensure that (i) A takes only values in Σ on $I = \{w^{in+j} : i, j \in \{0, \ldots, n-1\}\}$ and (ii) the evaluation of A on I produces an assignment $\hat{A}$ that satisfies ψ. Details follow.

Define

$$\text{AFF}_1(x) = x;\ \text{AFF}_2(x) = x \cdot \omega^n;\ \text{AFF}_3(x) = x \cdot \omega, \text{AFF}_4(x) = x \cdot \omega^{-n^2},$$

$$H = I \cup \omega^{n^2} \cdot I = \{\{w^{in+j} : i \in \{0, \ldots, 2n-1\}, j \in \{0, \ldots, n-1\}\}.$$

Notice that $|I| = n^2$ and $|H| = 2n^2$. We now define the constraint polynomial C.

Notice that $\hat{C}_{i,j}$ can be interpreted as a function from $\Sigma^3 \subset \mathbb{F}^3$ to $\{0,1\} \subset \mathbb{F}$ and we associate 0 with `accept` and 1 with `reject`. Arithmetize this constraint by a trivariate polynomial $C_{i,j} : \mathbb{F}^3 \to \mathbb{F}$ of degree at most $|\Sigma| - 1$ in each variable, satisfying

$$C_{i,j}(\sigma_1, \sigma_2, \sigma_3) = \hat{C}_{i,j}(\sigma_1, \sigma_2, \sigma_3) \quad \forall \sigma_1, \sigma_2, \sigma_3 \in \Sigma. \tag{5.1}$$

For $\omega^{in+j} \in H$, let $P_{i,j}(x)$ be the unique polynomial of degree $|H| - 1$ that evaluates to 1 on ω^{in+j} and to 0 on every other element in H. Finally, let $P_\Sigma(x) = \prod_{\sigma \in \Sigma}(x - \sigma)$ be the unique monic nonzero polynomial of degree $|\Sigma|$ whose set of roots is precisely Σ. The constraint polynomial is

$$C(x, y_1, \ldots, y_4) = \sum_{i,j=0}^{n-2} P_{i,j}(x) \cdot C_{i,j}(y_1, y_2, y_3) + \sum_{i=n}^{2n-1} \sum_{j=0}^{n-1} P_{i,j}(x) \cdot P_\Sigma(y_4). \tag{5.2}$$

The polynomial $P_{i,j}$ is often used to "bundle" together many constraints and verify that all of them are satisfied, forming the algebraic analogue of an AND gate. The second summand on the right-hand side of (5.2) corresponds to the set of constraints (i) mentioned above, and the first summand corresponds to (ii).

Notice that C has degree $|H|-1$ in its first variable and degree $|\Sigma|$ in the remaining variables. We conclude that ϕ is a legal instance of ALGEBRAIC-CSP$_{4,|\Sigma|}$.

Completeness. Suppose $\psi \in$ DOMINO-TILING$_\Sigma$ and let $\hat{A}$ be a proof for ψ. Let A be the low-degree extension of $\hat{A}$; i.e., A is a polynomial of degree $\leq n^2 - 1$ satisfying $A(\omega^{in+j}) = \hat{A}(in + j)$ for all $i, j \in \{0, \ldots, n-1\}$. We now prove for all $x \in H$

$$C(x, A(x), A(\omega^n x), A(\omega x), A(\omega^{-n^2} x)) = 0.$$

If $x = \omega^{in+j} \in H$ then by definition of $P_{i,j}$ at most one summand of (5.2) can be nonzero. There are two cases to consider:

- $i \leq n-2$: The summand to consider is $P_{i,j}(\omega^{in+j}) \cdot \hat{C}_{i,j}(\hat{A}(\omega^{in+j}), \hat{A}(\omega^{in+j+1}), \hat{A}(\omega^{(i+1)n+j}))$. This summand vanishes because $\hat{A}$ satisfies $\hat{C}_{i,j}$.
- $i \geq n$: The summand to consider is $P_{i,j}(\omega^{in+j}) \cdot P_\Sigma(\hat{A}(\omega^{in+j}))$, which vanishes because $\hat{A}$ evaluates to Σ on I.

We conclude that $\phi \in$ ALGEBRAIC-CSP$_{4,|\Sigma|}$.

Soundness. Suppose $\phi \in$ ALGEBRAIC-CSP$_{4,|\Sigma|}$ and let A witness this. Let $\hat{A} : \{0, \ldots, n^2 - 1\}$ be defined by $\hat{A}(in + j) = A(\omega^{in+j})$. We claim $\hat{A}$ satisfies ψ. First, notice that the range of A on inputs from I is Σ. If this is not the case and $A(\omega^{in+j}) \notin \Sigma$, then $P_\Sigma(A(\omega^{in+j})) \neq 0$, so (5.2) does not vanish on $x = \omega^{n^2+in+j} \in H$.

Since A evaluates to Σ on I and for $\sigma_1, \sigma_2, \sigma_3 \in \Sigma$ (5.1) implies $C_{i,j}(\sigma_1, \sigma_2, \sigma_3) = 0$ iff $\hat{C}(\sigma_1, \sigma_2, \sigma_3) =$ `accept`, we conclude that $\hat{A}$ satisfies ψ so $\psi \in$ DOMINO-TILING$_\Sigma$. This completes our proof. □

5.2. Quasilinear size reduction. In this section we prove Theorem 3.7 and show that ALGEBRAIC-CSP is NTIME($t(n)$)-complete under quasilinear size reductions. Our proof is similar to that of Polishchuk and Spielman [37]; however, our ending point is a problem over univariate polynomials.

Overview. The reason we chose DOMINO-TILING as our starting point in the previous section was because this language was NP-complete and additionally had "nice" structure, in the sense that each constraint ($\hat{C}_{i,j}$) depended on assignment entries whose indices are *linear functions* of the constraint index. The problem with DOMINO-TILING is that the reduction from an arbitrary language in NTIME($t(n)$) to it results in instances of size $t^2(n)$. Thus, we are looking for an NP-complete language that has a similar "nice" structure, yet whose blowup factor, when reducing from a language in NTIME($t(n)$), is only quasilinear.

One such language is DE BRUIJN COLORING, first presented by Polishchuk and Spielman [37], based on a construction of [5]. First we will describe this language and state its completeness. Then we will arithmetize it and reduce it to ALGEBRAIC-CSP. The crucial observation in the arithmetization, given in Proposition 5.11, is that the de Bruijn graph can be embedded in an "affine" graph over a finite field (see Definition 5.9).

DE BRUIJN COLORING. Let $\sigma : \{0,1\}^k \to \{0,1\}^k$ be the cyclic permutation operator; i.e., for $w \in \{0,1\}^k, w = (w_1, \ldots, w_k)$ let $\sigma(w) = (w_k, w_1, \ldots, w_{k-1})$. Let $u \oplus v$ denote the bitwise xor of $u, v \in \{0,1\}^k$ and let $e_i \in \{0,1\}^k$ be the sequence that is zero on all but the ith coordinate, where it is one.

DEFINITION 5.5 (wrapped de Bruijn graph [39]). *The k-dimensional* wrapped de Bruijn graph *is the following directed graph $B_k = (V, E)$. Let m be the smallest power of 2 satisfying $m > 5k$. The vertex set is*

$$V = \{(w, i) : w \in \{0,1\}^k, i \in \{0, \ldots, m-1\}\}.$$

Each vertex $v = (w, i) : w \in \{0,1\}^k, i \in \{0, \ldots, m-1\}$, has two neighbors:

$$N_0(v) = (\sigma(w), (i + 1 \bmod m)), \quad N_1(v) = ((\sigma(w)) \oplus e_1, (i + 1 \bmod m)).$$

Remark 5.6. The definition in [39, section 4.3.2] is slightly different from the above; namely, it fixes $m = 5k + 1$. However, Theorem 5.8 holds for any $m > 5n$, as inspection of [39, section 4.3] reveals.

DEFINITION 5.7 (DE BRUIJN COLORING). *Let* $\hat{\Sigma} = \{0,1\}^4$. *The language* DE BRUIJN COLORING *has as its space of instances tuples of the form* $\psi = \{B_k, \hat{C}\}$, *where* $B_k = (V, E)$ *is a* k*-dimensional wrapped de Bruijn graph, and* $\hat{C} = \{\hat{C}_v : v \in V\}$ *is a set of constraints, where* $\hat{C}_v : \hat{\Sigma}^3 \to \{\texttt{accept}, \texttt{reject}\}$.

An instance is in the language DE BRUIJN COLORING *iff there exists an assignment* $\hat{A} : V \to \hat{\Sigma}$ *such that for all* $v \in V$ *we have* $\hat{C}_v(\hat{A}(v), \hat{A}(N_0(v)), \hat{A}(N_1(v))) = \texttt{accept}$.

THEOREM 5.8. DE BRUIJN COLORING *is NP-complete. Moreover, for any proper complexity function* $t : \mathbb{N}^+ \to \mathbb{N}^+$, *a language* $L \in \mathrm{NTIME}(t(n))$ *is reducible in time* $\mathrm{poly}\, t(n)$ *to an instance* $\psi = \{B_k, \hat{C}\}$ *of* DE BRUIJN COLORING, *where* $k = \lceil \log(t(n) \cdot O(\log^2 t(n))) \rceil$.

Proof. $L \in \mathrm{NTIME}(t(n))$ is reducible in time $\mathrm{poly}\, t(n)$ to an instance of CKTSAT of size $O(t(n) \log t(n))$ [29, 15]. This instance is reducible in time $\mathrm{poly}\, t(n)$ to an instance of DE BRUIJN COLORING of size $t(n)\,\mathrm{polylog}\, t(n)$ [37]. (See [39, section 4.3] for details.) ∎

To arithmetize an instance of DE BRUIJN COLORING we embed B_k in an *affine graph* as defined below. Recall an injective graph homomorphism of G to H is an injective mapping $f : V(G) \to V(H)$ such that if $(u, v) \in E(G)$ then $(f(u), f(v)) \in E(H)$. Further recall an affine map $\mathrm{AFF} : \mathbb{F} \to \mathbb{F}$ is of the form $\mathrm{AFF}(z) = az + b$ for $a, b \in \mathbb{F}$.

DEFINITION 5.9 (affine graph). *Let* $\mathcal{A}$ *be a set of affine maps over a field* $\mathbb{F}$. *The* affine graph $G(\mathbb{F}, \mathcal{A})$ *over* $\mathbb{F}$, *generated by* $\mathcal{A}$, *is the directed graph over vertex set* $\mathbb{F}$, *where each vertex* $v \in \mathbb{F}$ *is connected to* $\mathrm{AFF}(v)$ *for all* $\mathrm{AFF} \in \mathcal{A}$. *Notice that the outdegree of this graph is at most* $|\mathcal{A}|$.

We will use the following elementary properties of primitive polynomials (see [32, section 3.1]).

PROPOSITION 5.10. *Let* $S(x)$ *be a primitive polynomial of degree* s *over* GF(2). *Then, denoting* $\xi_i = x^i (\mathrm{mod}\ S(x))$, *we have that* $\xi_1, \ldots, \xi_{2^s} = \xi_0$ *are distinct polynomials over* GF(2) *of degree less than* s.

We now define a graph homomorphism injecting B_k to an affine graph of outdegree eight over $\mathrm{GF}(2^\ell)$ for any $\ell > k + \log 5k + 2$. Briefly, a vertex $(w, i) \in B_k$ will be mapped to a polynomial $p_{(w,i)} \in \mathrm{GF}(2^\ell)$. We will show that the polynomials corresponding to $(w, i+1)$ and $(w \oplus e_i, i+1)$ can be obtained by applying two out of eight possible affine shifts to $p_{(w,i)}$. In what follows, addition and multiplication are in $\mathrm{GF}(2^\ell)$ and we identify $\{0,1\}$ with GF(2).

PROPOSITION 5.11. *Let* m *be the smallest power of* 2 *satisfying* $m > 5k$. *Let* $\mathrm{GF}(2^\ell) = \mathrm{GF}(2)[x]/q(x)$, *where* $q(x)$ *is an irreducible polynomial of degree* ℓ. *Let* $S(x)$ *be a primitive polynomial of degree* $s = \log m$ *(note that* s *is an integer), and let* ξ_i *be as defined in Proposition* 5.10. *For* $((w_1, \ldots, w_k), i), w_j \in \{0,1\}, i \in [m]$, *let*

$$g(w) = x^s \cdot \sum_{j=1}^{k} w_j x^j \ ; \quad h(i) = \xi_i \ ; \quad f(w,i) = g(w) + h(i). \tag{5.3}$$

Then, the mapping $f : V(B_k) \to \mathrm{GF}(2^\ell)$ *is an injective homomorphism of* B_k

into the affine graph $G(\mathrm{GF}(2^\ell), \mathcal{A})$*, where*

(5.4)
$$\mathcal{A} = \left\{ \textsc{Aff}_b(\alpha) \triangleq x \cdot \alpha + b_1 S(x) + b_2 x^{s+1} + b_3 x^{s+k+1} : b = (b_1, b_2, b_3) \in \{0,1\}^3 \right\}.$$

Proof. Our mapping is injective. Note that $\deg(h(i)) < s$ for all $i \in [m]$, whereas the minimal degree of a nonzero term of $g(w)$ is $s+1$. Thus, $f(w,i) = f(w',i')$ iff $g(w) = g(w')$ and $h(i) = h(i')$. The former happens by definition iff $w = w'$ and the latter happens iff $i = i'$ because Proposition 5.10 implies $\xi_i \neq \xi_{i'}$ for all $i \neq i' \in [m]$.

To prove f is a homomorphism, we need to show that if $((w,i),(w',(i+1 \bmod m)))$ is an edge of $B_{k,m}$ then $f(w',(i+1 \bmod m)) = \textsc{Aff}_b(f(w,i))$ for some $b \in \{0,1\}^3$. There are eight cases to consider. Recall w' is either $\sigma(w)$ or $(\sigma(w)) \oplus e_1$. Note that $\ell > k+s+1$ so for all $w \in \{0,1\}^k$ we have that $x \cdot g(w) \bmod (q(x))$ is equal to $x \cdot g(w)$ as polynomials over GF(2). By definition of g we get

$$g(\sigma(w)) = \begin{cases} x \cdot g(w), & w_k = 0, \\ x \cdot g(w) + x^{s+k+1} + x^{s+1}, & w_k = 1. \end{cases}$$

Similarly,

$$g((\sigma(w)) \oplus e_1) = \begin{cases} x \cdot g(w) + x^{s+1}, & w_k = 0, \\ x \cdot g(w) + x^{s+k+1}, & w_k = 1. \end{cases}$$

Finally, by definition of h we get

$$h(i+1 \bmod m) = \begin{cases} x \cdot h(i), & \deg(h(i)) < s-1, \\ x \cdot h(i) + S(x), & \deg(h(i)) = s-1. \end{cases}$$

Our claim follows from the definition of $\textsc{Aff}_b$ and the previous equations. ∎

Proof of Theorem 3.7. We prove Theorem 3.7 for $k = 10$ and $d = |\hat{\Sigma}| = 16$, where $\hat{\Sigma}$ is from Definition 5.7. By Theorem 5.8 it suffices to show a polynomial time reduction sending an instance $\psi = \{B_k, \hat{C}\}$ of de Bruijn Coloring to an instance of ALGEBRAIC-CSP over a field of size $2^k \cdot \operatorname{poly} k$. We reduce in time $\operatorname{poly} 2^\ell$ to an instance over $\mathrm{GF}(2^\ell)$ for any $\ell > k + (\log 5k) + 2$ and the reduced instance is of the form

$$\phi = \{\mathrm{GF}(2^\ell), \{\textsc{Aff}', \textsc{Aff}''\} \cup \mathcal{A}, H, C(x, y_0, y_1, z_{000}, \ldots, z_{111})\},$$

where $\textsc{Aff}'(x) = x$, $\textsc{Aff}''(x) = \zeta - x$ (for ζ to be defined later), and $\mathcal{A}$ is as in (5.4).

Embed $\hat{\Sigma}$ in $\mathrm{GF}(2^\ell)$ arbitrarily and associate `accept` with 0 and `reject` with 1. As in the proof of Theorem 5.4, we view the constraint $\hat{C}_v$ as a mapping from $\hat{\Sigma} \subset \mathrm{GF}(2^\ell)$ to $\{0,1\}$. Recall that Proposition 5.11 showed $V(B_k)$ can be embedded in $G(\mathrm{GF}(2^\ell), \mathcal{A})$ via the embedding f from (5.3). Notice that the outdegree of $G(\mathrm{GF}(2^\ell), \mathcal{A})$ is greater than the outdegree of B_k; thus when arithmetizing $\hat{C}_v$ we must take into account which of the eight neighbors of $f(v)$ in $G(\mathrm{GF}(2^\ell), \mathcal{A})$ are maps of the neighbors of v in B_k. Let $b_0(v), b_1(v) \in \{0,1\}^3$ denote the two relevant neighbors of $f(v)$ in $G(\mathrm{GF}(2^\ell), \mathcal{A})$ satisfying $f(N_0(v)) = \textsc{Aff}_{b_0(v)}(f(v))$ and $f(N_1(v)) = \textsc{Aff}_{b_1(v)}(f(v))$. Let $C_v(y, z_{b_0}, z_{b_1})$ be the trivariate polynomial of degree at most $|\hat{\Sigma}| - 1$ in each variable, agreeing with $\hat{C}_v$ on inputs in $\hat{\Sigma}^3$. Let $I = \{f(v) : v \in V\}$. Let $\zeta \in \mathrm{GF}(2^\ell)$ satisfy $(\zeta + I) \cap I = \emptyset$, where $\zeta + I = \{\zeta + \xi : \xi \in I\}$ and set $H = I \cup (\zeta + I)$. Such ζ exists because viewing elements of $\mathrm{GF}(2^\ell)$ as polynomials over GF(2) modulo an irreducible polynomial of degree ℓ, all elements in I

have degree at most $k+s$. Now, let $P_h(x)$ be the polynomial of degree $|H|-1$, that is, 1 when $x=h$ and 0 for all $x=h' \in H, h' \neq h$. Finally, let $P_{\hat{\Sigma}}(y)$ be the nonzero polynomial of degree $|\hat{\Sigma}|$ whose roots are precisely the elements of $\hat{\Sigma}$. We are ready to define the constraint polynomial of ϕ:

$$C(x, y_0, y_1, z_{000}, \ldots, z_{111}) = \sum_{v \in I} P_v(x) \cdot C_v(y_0, z_{b_0(v)}, z_{b_1(v)}) + \sum_{h \in H \setminus I} P_h(x) \cdot P_{\hat{\Sigma}}(y_1). \tag{5.5}$$

The second summand on the right-hand side checks that all vertices receive colors in $\hat{\Sigma}$ and the first summand checks that all coloring constraints are satisfied. The polynomials P_v, P_h are used to "bundle" all constraints into one polynomial.

Note that $\deg_x(C) \leq |H|-1$ and the degree in the remaining variables is at most $|\hat{\Sigma}|$. Thus, ϕ is a legal instance of ALGEBRAIC-CSP$_{k,d}$.

Completeness. Suppose $\psi \in$ DE BRUIJN COLORING and let $\hat{A} : V \to \hat{\Sigma}$ witness this. Let A be the polynomial of degree $\leq |V|-1$ satisfying $A(f(v)) = \hat{A}(v)$ for all $v \in V$. We claim A satisfies ϕ. We need to show for all $x \in H$

$$C\left(x, A(x), A(\zeta - x), A(\textsc{Aff}_{000}(x)), \ldots, A(\textsc{Aff}_{111}(x))\right) = 0.$$

As in the proof of Theorem 5.4, when $x \in H$, at most one summand of (5.5) may be nonzero, by definition of P_v. We split the proof into cases.

- $x \in I$: Let $v = x$. The summand to consider is $P_v(v) \cdot C_v(A(v), A(\textsc{Aff}_{b_0(v)}(v)), A(\textsc{Aff}_{b_1(v)}(v)))$, which vanishes because $\hat{C}(\hat{A}(v), \hat{A}(N_0(v)), \hat{A}(N_1(v)) = \texttt{accept}, f(N_0(v)) = \textsc{Aff}_{b_0(v)}(f(v))$, and $f(N_1(v)) = \textsc{Aff}_{b_1(v)}(f(v))$.
- $x \in H \setminus I$: The summand to consider is $P_v(x) \cdot P_{\hat{\Sigma}}(A(\zeta - x))$. By construction of H and selection of ζ we have $\zeta - x \in I$. By construction A takes on values in $\hat{\Sigma}$ on $\zeta - x$, so the summand vanishes.

Soundness. Suppose $\phi \in$ ALGEBRAIC-CSP$_{k,d}$ and let A witness this. Let $\hat{A} : I \to \mathrm{GF}(2^\ell)$ be the evaluation of A on I. First we claim that the range of $\hat{A}$ is $\hat{\Sigma}$. Indeed, assume $A(x) \notin \hat{\Sigma}$ for $x \in I$. Let $x' = \zeta + x$ and notice that $x' \in H \setminus I$. Then the second summand of (5.5) does not vanish on x'. Since all other summands vanish by construction of P_v, we reach a contradiction. We conclude that $\hat{A}$ is a legal assignment to ψ.

Next, we claim that $\hat{A}$ satisfies ψ. Consider the constraint $\hat{C}_v$. Equation (5.5) holds for v and $P_{v'}(v) = 0$ for all $v' \neq v, v' \in H$, implying $C_v(A(v), A(\textsc{Aff}_{b_0(v)}(v)), A(\textsc{Aff}_{b_1(v)}(v))) = 0$. Recall from the previous paragraph that $A(v), A(\textsc{Aff}_{b_0(v)}(v)), A(\textsc{Aff}_{b_1(v)}(v)) \in \hat{\Sigma}$. By construction of C_v we conclude that $\hat{C}_v(\hat{A}(v), \hat{A}(N_0(v)), \hat{A}(N_1(v))) = \texttt{accept}$. This completes our proof. ∎

5.3. Systematic reduction to PAIR-ALGEBRAIC-CSP. We now show how to modify the reduction of the previous section to apply it to pair languages and get a systematic reduction, thus proving Theorem 3.10.

Proof. Consider the sequence of reductions applied to an instance x of L and resulting in an instance ϕ of ALGEBRAIC-CSP. First, we reduce x to an instance C of CKTSAT along the lines of [29, 15]. Inspection reveals that this reduction is systematic. Indeed, the implicit input y is embedded into the inputs of C, and C accepts only inputs $y \in L_x$.

In the next step we reduce C to an instance ψ of DE BRUIJN COLORING. Once again, inspection of this reduction shows it is systematic [39, section 4.3]. In particular, this latter reduction embeds all nodes of C including its inputs in the first layer of the wrapped de Bruijn graph and each input node is mapped to a unique vertex. By construction, a coloring of the resulting de Bruijn graph is legal only if the colors of the vertices corresponding to inputs form an assignment satisfying C. Similarly, any assignment satisfying C can be extended to a coloring satisfying the constraints of ψ.

Finally, consider the reduction from DE BRUIJN COLORING to ALGEBRAIC-CSP. Notice that each vertex of the de Bruijn graph is mapped to a distinct element of $\mathbb{F}$ (Proposition 5.11). Additionally, by construction we map the colors of the de Bruijn coloring problem to distinct elements of $\mathbb{F}$. By construction, ϕ is satisfied by A iff A is the low-degree extension of a coloring that satisfies ψ. We have seen that all steps of our reduction are systematic; hence so is their concatenation. This completes our proof. □

6. PCPPs for Reed–Solomon codes over fields of characteristic 2. In this section we give a PCPP-verifier for RS-codes when the field is of characteristic 2 and the set of evaluation points is a linear subspace of the field over GF(2), thereby proving Theorem 3.2 (restated below).

An overview of the proof appears in subsection 6.1. This is followed by a formal description of the proof of proximity and verifier in subsection 6.2 and the analysis of its basic properties in subsection 6.3. The analysis of the soundness follows in subsection 6.4. We conclude with a formal proof of Theorem 3.2 in subsection 6.5.

THEOREM 6.1 (Theorem 3.2, restated). *Let* PAIR-ADDITIVE-RS *be the restriction of the language* PAIR-RS *to pairs* $((\mathrm{GF}(2^\ell), L, d), p)$*, where* $\mathrm{GF}(2^\ell)$ *is the Galois field of size* $n = 2^\ell$ *(and characteristic* 2*) and* $L \subseteq \mathbb{F}$ *is* GF(2)*-linear. Then,*

$$\text{PAIR-ADDITIVE-RS} \in \mathbf{Strong\text{-}PCPP}_{\delta/\operatorname{polylog} n} \begin{bmatrix} \mathsf{randomness}\ \log(n \cdot \operatorname{polylog} n), \\ \mathsf{query}\ O(1), \\ \mathsf{distance}\ \mathsf{Hamming}_{\mathrm{GF}(2^\ell)} \end{bmatrix}.$$

Remark 6.2. For simplicity, we first prove the theorem for the special case of degree $d = |L|/8 - 1$. Then we show in Proposition 6.14 that this implies that the theorem holds for all degrees.

6.1. Sketch of proof of Theorem 3.2. At a high level, we attempt a reduction from the task of testing a univariate polynomial to the task of testing a bivariate polynomial of significantly smaller degree. We then invoke an analysis of a "bivariate low-degree test" by Polishchuk and Spielman [37], which reduces the task of testing bivariate polynomials back to the task of testing univariate polynomials, of much smaller degree than the original. Recursing on this idea leads to the full test. We note that crucial to our obtaining short PCPPs is the evaluation of the bivariate polynomial on a carefully selected, algebraically structured, subset of points. This set is very different from the sets typically used in previous PCP constructions, e.g., in [5, 2, 17], which are product sets usually consisting of the whole field.

We start by considering the polynomial $P(z)$ of degree $< n/8$ evaluated on the linear space $L \subset \mathrm{GF}(2^\ell)$ of cardinality n and address the task of "testing" it. Our starting point is that for any polynomial $q(z)$ of degree $\approx \sqrt{n}$, we can define a bivariate polynomial $Q(x, y)$ of degree $\approx \sqrt{n}$ in each variable that "captures" all the information of P. Specifically, we can reconstruct P from Q using the identity $P(z) = Q(z, q(z))$.

PROPOSITION 6.3. *Given any pair of polynomials $P(z), q(z)$, there exists a unique bivariate polynomial $Q(x,y)$ with $\deg_x(Q) < \deg(q)$ and $\deg_y(Q) = \lfloor \deg(P)/\deg(q) \rfloor$, such that $P(z) = Q(z, q(z))$.*

Proof. We use division over the ring of bivariate polynomials $\mathbb{F}[z,y]$ (see [16] for more details). Fix the lexicographic ordering on terms where $z > y$; i.e., terms are ordered first by their degree in z and then by their degree in y. Divide $P(z)$ by $(y - q(z))$, obtaining

$$P(z) = Q'(z,y) \cdot (y - q(z)) + Q(z,y). \tag{6.1}$$

By the basic properties of division in this ring Q is uniquely defined, and $\deg_y(Q) = \lfloor \deg(P)/\deg(q) \rfloor$ and $\deg_z(Q) < \deg(q)$. To complete the proof set $y = q(z)$ and notice that the first summand on the right-hand side of (6.1) vanishes. □

The presentation of P of degree $\approx n$ as a bivariate polynomial Q of individual degree $\approx \sqrt{n}$ is useful, because testing of bivariate polynomials reduces to testing of univariate polynomials of roughly the same degree using well-known "low-degree tests" and their analysis. This leads us to the hope that Q might provide a good "proof" that P is of low degree. More to the point, to prove that a table of evaluations of P corresponds to the evaluations of a polynomial of low degree, the prover can provide a table of evaluations of a bivariate polynomial Q, prove that Q has degree $\sqrt{n}$ in each variable, and then prove that Q is consistent with the table of evaluations of P.

To completely describe the above approach, all we need to do is describe which set of points we will specify Q on so as to achieve both tasks: (i) verifying that Q has low degree, and (ii) that it is consistent with P. However, this leads to conflicting goals. To test that Q has low degree, using a bivariate verifier, we need to know its values on some subset $X \times Y$, where $X, Y \subseteq \mathrm{GF}(2^\ell)$. To make this efficient, we need to make $|X|, |Y| \approx \sqrt{n}$. On the other hand, to test its consistency with P, the natural approach is to ask for its values on the set

$$T = \{(z, q(z)) | z \in L\}.$$

Unfortunately the set T, which depends on the selection of $q(z)$, is far from being of the form $X \times Y$. For starters, the projection of T onto its first coordinate has cardinality n while we would like this projection to be of cardinality $O(\sqrt{n})$.

Our solution is to ask the prover to provide the evaluation on *both* sets of points. This leads to a problem of checking consistency between the two sets and to do so we pick $q(z)$ in a way that will ensure T is compatible with $X \times Y$. In particular, we choose $q(z)$ to be a special *linearized* polynomial as defined in [32, Chapter 3, section 4]. A polynomial $q(z)$ over $\mathrm{GF}(2^\ell)$ is said to be linearized if $q(x+y) = q(x) + q(y)$ for every $x, y \in \mathrm{GF}(2^\ell)$. A linearized polynomial defines a linear map over $\mathrm{GF}(2^\ell)$ and we abuse notation and use q to denote this map. For $S \subset \mathrm{GF}(2^\ell)$, let $q(S) = \{q(s) : s \in S\}$. The linearized polynomial we use and its useful properties are listed below.

PROPOSITION 6.4. *For L a linear subspace of $\mathrm{GF}(2^\ell)$ that is a direct sum of the linear spaces L_0, L_1, let*

$$q(z) = q_{L_0}(z) \triangleq \prod_{\alpha \in L_0} (z - \alpha).$$

- *The polynomial $q(z)$ is linearized.*
- *The kernel of (the linear map defined by) q is L_0.*
- *$q(L) = q(L_1)$ and $q(L_1)$ is a linear space of dimension $\dim(L_1)$.*

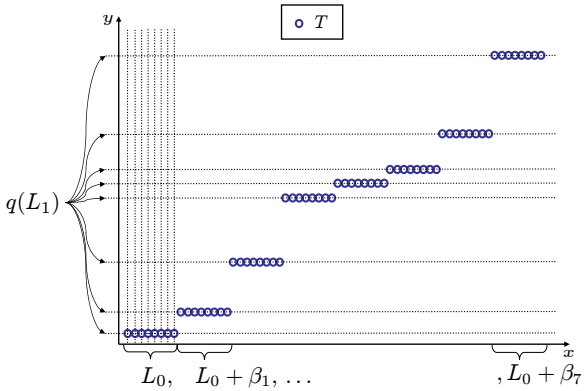

FIG. 1. *Here* $\mathbb{F} = \mathrm{GF}(2^6)$ *is the field with* 64 *elements and* q *is a linearized polynomial of degree* 8. *We plot the set of points* $T \subset \mathbb{F} \times \mathbb{F}$ *defined by* $T = \{(z, q(z)) : z \in \mathbb{F}\}$. *Notice* T *can be partitioned into eight product sets, each set being a product of an affine shift of* L_0 *and some* $\beta \in L_1$.

- q *is a one-to-one map from* L_1 *to* $q(L_1)$*; i.e., for* $\beta \neq \beta'$ *we get* $q(\beta) \neq q(\beta')$.
- q *is an* $|L_0|$ *to one map on* L*, where, for* $\beta \in L_1$*, the affine space* $L_0 + \beta \triangleq \{\alpha + \beta : \alpha \in L_0\}$ *is mapped to* $q(\beta)$.

Proof. The first part is proved by induction on the dimension of L. The base case (dimension zero) is easy, as $q_{L_0}(z) = z$ is clearly linearized. For the inductive step, let $L_0 = \mathrm{span}(\hat{L}, \alpha)$, where $\dim(\hat{L}) = k - 1$ and $\alpha \in \mathrm{GF}(2^\ell)$. Let $\hat{q}(z) = q_{\hat{L}}(z)$ be the linearized polynomial whose set of roots is $\hat{L}$. Clearly, $q_{L_0}(z) = \hat{q}(z) \cdot \hat{q}(\alpha + z)$ because addition and subtraction are the same in fields of characteristic 2. So

$$\begin{aligned} q_{L_0}(x+y) &= \hat{q}(x+y) \cdot \hat{q}(\alpha + x + y) = \hat{q}^2(x) + \hat{q}^2(y) + \hat{q}(\alpha)(\hat{q}(x) + \hat{q}(y)) \\ &= \hat{q}(x) \cdot \hat{q}(\alpha + x) + \hat{q}(y) \cdot \hat{q}(\alpha + y) = q_{L_0}(x) + q_{L_0}(y). \end{aligned}$$

We conclude that q_{L_0} is a linearized polynomial. The second part follows because $\deg(q) = |L_0|$ and the elements of L_0 are all roots of q.

The last three parts follow via basic linear algebra from our previous assertions that q defines a linear map with kernel L_0. ∎

With Proposition 6.4 in hand, we return to the task of providing a proof of proximity for the evaluation of a polynomial on the set of points L. Write L as the direct sum of L_0, L_1, with $\dim(L_0) = \lfloor \dim(L)/2 \rfloor$ and $\dim(L_1) = \lceil \dim(L)/2 \rceil$ (so $|L_0|, |L_1| \approx \sqrt{|L|}$), and take $q(z) = q_{L_0}(z)$ as described above. The last part of Proposition 6.4 implies that q partitions T into the disjoint union of $|L_1|$ lines, where each line is a product of a set of size $\approx \sqrt{|L|}$ with a singleton set (see Figure 1):

$$T = \bigcup_{\beta \in L_1} \{\{L_0 + \beta\} \times \{q(\beta)\}\}.$$

This suggests requesting the evaluation of Q on the set of points $(L_0 \times q(L_1)) \cup T$, the cardinality of which is $\leq 2n$. With such an evaluation in hand we can use the

subset $L_0 \times q(L_1)$ to perform a bivariate low-degree test, by testing proximity to the RS-code of degree $\approx \sqrt{n}$ of a random row/column of this product set. The consistency of Q's evaluation on the product set $L_0 \times q(L_1)$ and on the set T can also be addressed, by reading $Q(x, q(\beta))$ for all points $x \in L_0 \cup (L_0 + \beta)$ for $\beta \in L_1$. This consistency is precisely what is needed to connect the evaluation of P on the set L, that is isomorphic to T, to the evaluation of the bivariate Q on the product set $L_0 \times q(L_1)$. We have reduced our original problem of size n to $O(\sqrt{n})$ identical problems of size $O(\sqrt{n})$.

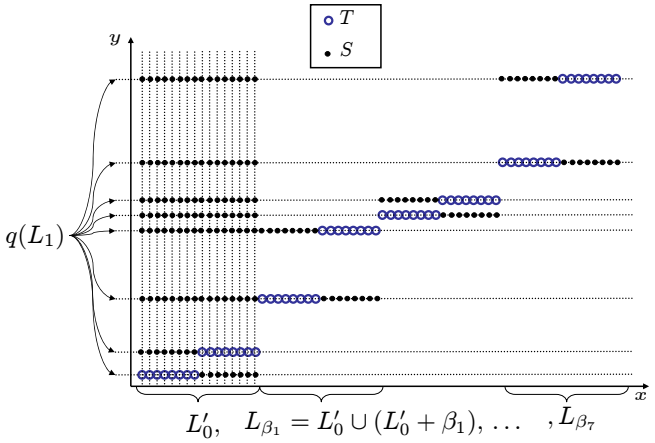

FIG. 2. *The proof of proximity for P is the evaluation of Q on the set of points denoted S. Notice it has a large subset that is the product set $L'_0 \times q(L_1)$, allowing for bivariate low degree testing. Additionally, $S \cup T$ can be partitioned into eight rows and each row is a linear space.*

Our description so far leads to a proof of proximity of size $O(n)$ that can be tested by making $O(\sqrt{n})$ queries. However, the *robustness* of our tests can be used to decrease the query complexity further, at the price of increasing the proof length. Informally, robustness means the following. If a function $f : (L_0 \times q(L_1)) \cup T \to \mathrm{GF}(2^\ell)$ is δ-far from being a low-degree bivariate polynomial, then the expected distance of a random row/column of f from a low-degree univariate polynomial is $\Omega(\delta)$. To apply recursion, notice that all of our tests verify proximity to Reed–Solomon codewords evaluated on *linear subspaces* of $\mathrm{GF}(2^\ell)$. To see this notice that L_0 and $q(L_1)$ are linear spaces, and so is $L_0 \cup (L_0 + \beta) = \mathrm{span}(L_0, \beta)$. Using recursion we conclude that to test proximity to the RS-code of size n it suffices to test proximity to RS-codes of size $\approx \sqrt{n}$, which can be done by testing proximity to the RS-code of size $\approx n^{1/4}$, etc. Applying this recursion a $\log \log n$ number of times reduces the degree to a constant and gives us our proofs of length $n \cdot \mathrm{polylog}\, n$.

From intuition to proof. Our rigorous analysis follows the intuition above, with one technical difference regarding the degree of the bivariate polynomial Q. To use the bivariate low-degree test on Q, we need its evaluation on a product set of points $X \times Y$, where $|X| \gg \deg_x(Q)$ and $|Y| \gg \deg_y(Q)$. In our case Proposition 6.3 gives us only $|X| > \deg_x(Q)$. As to y, we get $|Y| > 8 \deg_y(Q)$, which is sufficient. So we need to enlarge X. This is done by taking a linear space $L'_0 \supset L_0$ of dimension $\dim(L_0) + 2$ and asking for the evaluation of Q on $(L'_0 \times q(L_1)) \cup T$. This causes a new problem, because $L'_0 \cup (L_0 + \beta)$ is not a linear space, as $\dim(L'_0) > \dim(L_0)$. This

problem is fixed by asking for the evaluation of Q on the linear space $L'_0 \cup (L'_0 + \beta)$. The resulting set of points is described in Figure 2.

6.2. The RS proof of proximity and its associated verifier. First we define the structure of the proof of proximity for $\mathrm{RS}(\mathrm{GF}(2^\ell), L, d)$ and then describe the verifier's operation. As explained in the previous section, the proof for a purported low-degree polynomial $p : L \to \mathrm{GF}(2^\ell)$ is an evaluation of a low-degree bivariate polynomial related to p on a carefully chosen subset of $\mathrm{GF}(2^\ell) \times \mathrm{GF}(2^\ell)$, concatenated with a sequence of subproofs for RS-codes of smaller size. To formally define the proof of proximity we use the following notation throughout this section.

Given basis $(b_1, \ldots, b_k)$ for L, let

$$L_0 \triangleq \mathrm{span}(b_1, \ldots, b_{\lfloor k/2 \rfloor}); \quad L'_0 \triangleq \mathrm{span}(b_1, \ldots, b_{\lfloor k/2 \rfloor + 2}); \quad L_1 \triangleq \mathrm{span}(b_{\lfloor k/2 \rfloor + 1}, \ldots, b_k). \tag{6.2}$$

Fix $q(x) \triangleq q_{L_0}(x)$. Notice that $L'_0 \cap L_1 = \mathrm{span}(b_{\lfloor k/2 \rfloor + 1}, b_{\lfloor k/2 \rfloor + 2})$, and, in particular, this intersection is nonempty. For $\beta \in L_1$ let

$$L_\beta \triangleq \begin{cases} \mathrm{span}(L'_0, b_{\lfloor k/2 \rfloor + 3}), & \beta \in L'_0, \\ \mathrm{span}(L'_0, \beta) & \text{otherwise.} \end{cases} \tag{6.3}$$

A *partial* bivariate function f over $\mathrm{GF}(2^\ell)$ is a function with a partial domain $f : S \to \mathrm{GF}(2^\ell)$, where $S \subset \mathrm{GF}(2^\ell) \times \mathrm{GF}(2^\ell)$. The β-row of S is the set $R_\beta = \{\alpha : (\alpha, \beta) \in S\}$ (this set might be empty). The *restriction* of f to the β-row is the univariate function $f|^{\leftrightarrow}_\beta : R_\beta \to \mathrm{GF}(2^\ell)$ that agrees with f on its inputs, i.e., $f|^{\leftrightarrow}_\beta(\alpha) = f(\alpha, \beta)$. Similarly, the α-column of S is $C_\alpha = \{\beta : (\alpha, \beta) \in S\}$, and the restriction of f to it is $f|^{\updownarrow}_\alpha : C_\alpha \to \mathrm{GF}(2^\ell)$ defined by $f|^{\updownarrow}_\alpha(\beta) = f(\alpha, \beta)$.

Definition 6.5 (Reed–Solomon proof of proximity). *The proof of proximity for a purported codeword of the Reed–Solomon code* $\mathrm{RS}(\mathrm{GF}(2^\ell), L, n/8 - 1)$ *is defined by induction on* $k = \dim(L)$. *If* $k \le 6$ *then it is empty. Otherwise, the proof is a pair* $\pi = \{f, \Pi\}$, *where* f *is a partial bivariate function over partial domain* $S \subset \mathrm{GF}(2^\ell) \times \mathrm{GF}(2^\ell)$ *defined next and* Π *is a sequence of proofs of proximity for RS-codes over (smaller) linear spaces.*

Partial domain. *Let*

$$T \triangleq \bigcup_{\beta \in L_1} \{\{L_0 + \beta\} \times \{q(\beta)\}\}; \qquad S \triangleq \left(\bigcup_{\beta \in L_1} \{L_\beta \times \{q(\beta)\}\} \right) \setminus T. \tag{6.4}$$

Auxiliary proofs. *For each* $\beta \in L_1$, *the sequence of proofs* Π *has a unique subproof for an RS-codeword over* L_β *of degree* $|L_\beta|/8 - 1$, *denoted* $\pi^{\leftrightarrow}_\beta$. *For each* $\alpha \in L'_0$, *the sequence* Π *includes a unique subproof for an RS-codeword over* $q(L_1)$ *of degree* $|q(L_1)|/8 - 1$, *denoted* $\pi^{\updownarrow}_\alpha$. *Formally,*

$$\Pi \triangleq \{\pi^{\leftrightarrow}_\beta : \beta \in L_1\} \cup \{\pi^{\updownarrow}_\alpha : \alpha \in L'_0\}.$$

The next proposition shows that $S \cup T$ can be decomposed into rows and columns that are linear spaces (of size $\approx \sqrt{|L|}$). This gives some explanation of our peculiar choice of the set S as described in the previous section and shown in Figure 2.

Proposition 6.6. *The set* $S \cup T$ *is the disjoint union of* $q(\beta)$*-rows for* $\beta \in L_1$. *The* $q(\beta)$*-row of* $S \cup T$ *is the linear space* L_β. *Similarly, for every* $\alpha \in L'_0$, *the* α*-column of* $S \cup T$ *is the linear space* $q(L_1)$.

Proof. By construction of S, to prove the claim about the rows of $S \cup T$ it suffices to show that the $q(\beta)$-row of T is a subset of L_β. By the last part of Proposition 6.4 this row is

$$\{\gamma \in L : q(\gamma) = q(\beta)\} = q^{(-1)}(q(\beta)) \cap L = L_0 + \beta \subset L_\beta.$$

The inclusion above follows by definition from (6.3). This completes the proof of the claim about the rows.

Now consider the α-column of $S \cup T$ for $\alpha \in L_0'$. By (6.3) we have $L_0' \subset L_\beta$ for every $\beta \in L_1$ and $L_\beta \times \{q(\beta)\} \subset S$, so $(\alpha, q(\beta)) \in S$ implying $q(L_1)$ is a subset of the α-column. However, by the first part of our proposition, the only nonempty rows of $S \cup T$ are the $q(\beta)$-rows. So we conclude that the α-column of $S \cup T$ is precisely $q(L_1)$. ∎

DEFINITION 6.7 (RS-verifier). *The verifier for proximity to* $\mathrm{RS}(\mathrm{GF}(2^\ell), L, d = |L|/8 - 1)$ *is denoted* $\mathrm{V_{RS}}^{(p,\pi)}(\mathrm{GF}(2^\ell), L, d)$. *It receives as explicit inputs the description of the field* $\mathrm{GF}(2^\ell)$*, a basis* $(b_1, \ldots, b_k)$ *for* L*, and the degree parameter* $d = |L|/8 - 1$*. The implicit input of the verifier is the purported codeword* $p : L \to \mathrm{GF}(2^\ell)$ *and its purported proof is* $\pi = \{f, \Pi\}$ *as described in Definition* 6.5*. The verifier operates as follows.*

Base case ($|L| \leq 64$). *The verifier reads* p *in entirety and accepts iff* $p \in \mathrm{RS}(\mathrm{GF}(2^\ell), L, |L|/8 - 1)$.

Recursion ($|L| > 64$). *Let* $\hat{p} : T \to \mathrm{GF}(2^\ell)$ *be the partial bivariate function corresponding to* p*,*

$$\hat{p}(\gamma, q(\gamma)) = p(\gamma) \text{ for } \gamma \in L. \tag{6.5}$$

Notice that $\hat{p}$ *is well defined because the mapping* $\gamma \mapsto (\gamma, q(\gamma))$ *is a bijection from* L *to* T*. Let*

$$\hat{f} : S \cup T \to \mathrm{GF}(2^\ell) \tag{6.6}$$

be the function that agrees with f *on* S *and with* $\hat{p}$ *on* T*. Notice that* $\hat{f}$ *is well defined because* $S \cap T = \emptyset$*. The verifier sets* $L_0 = \mathrm{span}(b_1, \ldots, b_{\lfloor k/2 \rfloor})$*, computes the coefficients of the polynomial* $q(x) = q_{L_0}(x)$*, and performs one of the following two tests with probability half each.*

Row test. *Pick* $\beta \in L_1$ *at random. Let* L_β *be as in* (6.3)*. Invoke*

$$\mathrm{V_{RS}}^{(\hat{f}|^{\rightarrow}_{q(\beta)}, \pi^{\rightarrow}_{\beta})}(\mathrm{GF}(2^\ell), L_\beta, |L_\beta|/8 - 1).$$

Column test. *Pick* $\alpha \in L_0'$ *at random. Let* L_1 *be as in* (6.2)*. Compute a basis for* $q(L_1)$ *and invoke*

$$\mathrm{V_{RS}}^{(\hat{f}|^{\updownarrow}_{\alpha}, \pi^{\updownarrow}_{\alpha})}(\mathrm{GF}(2^\ell), q(L_1), |q(L_1)|/8 - 1).$$

Remark 6.8. The "inner" verifiers, i.e., the row and column tests, restrict their attention to special subsets of p and π. To simplify our analysis, we assume these special subsets are copied to an "inner oracle" before invocation of an inner test. This assumption can be made without loss of generality because the verifier is nonadaptive; i.e., its operation does not depend on the implicit input given to it. Furthermore, the indices of the queries needed at the bottom of the recursion can be computed efficiently given the random coins used through the recursion as can be verified by inspection of the proof of Proposition 6.9.

The following subsections analyze the performance of V_{RS}. Specifically, the next subsection analyzes the simple properties including the running time, query complexity, randomness/size complexity, and the completeness. The soundness analysis is addressed in subsection 6.4.

6.3. Basic properties.

PROPOSITION 6.9. $\mathrm{V}_{\mathrm{RS}}{}^{(p,\pi)}(\mathrm{GF}(2^\ell), L, |L|/8-1)$ *makes at most* 64 *queries into* p *and* π. *It tosses at most* $k + O(\log k)$ *random coins (recall that* $k = \dim(L)$*) and runs in time* $\operatorname{poly} \ell$. *The size of the proof* π *accessed by* $\mathrm{V}_{\mathrm{RS}}{}^{(p,\pi)}(\mathrm{GF}(2^\ell), L, |L|/8-1)$ *is* $2^k \cdot \operatorname{poly} k = |L| \cdot \operatorname{polylog} |L|$.

Proof. The query complexity is easy to verify. In the base case, the verifier reads 64 field elements. In the inductive case the verifier invokes V_{RS} which by induction makes 64 queries.

Regarding randomness complexity, in the base case the verifier tosses zero coins. In the inductive case, the verifier tosses one coin to determine which test to perform—row or column. It then tosses $k/2 + O(1)$ coins to determine the inner call and then $(k/2 + O(1)) + O(\log(k/2 + O(1)))$ coins in the recursive call. Adding up, we get a total of $k + O(\log k)$ coins. The size of the proof can be similarly analyzed or bounded by $2^{\text{randomness}}$ to get the same bound.

We now analyze the running time, which is the sum of two processes.

The preprocessing time. This is the time required by the *outer* verifier $\mathrm{V}_{\mathrm{RS}}{}^{(p,\pi)}[\mathrm{GF}(2^\ell), L, |L|/8-1]$ to prepare the explicit input for invoking an *inner* verifier on a row/column. Notice that $q(x)$ can be computed and evaluated in polynomial time in $|L_0|$ and ℓ and so can the bases for L_0, L_0', L_β, L_1, and $q(L_1)$. Thus, the preprocessing time is polynomial.

The index translation time. Suppose the outer verifier conducts an inner row test of the form

$$\mathrm{V}_{\mathrm{RS}}{}^{(\hat{f}|^{\leftrightarrow}_{q(\beta)}, \pi^{\leftrightarrow}_{\beta})}[\mathrm{GF}(2^\ell), q(L_\beta), |L_\beta|/8-1].$$

The case of a column test is analogous. A query to $\hat{f}|^{\leftrightarrow}_{q(\beta)}$ by the inner verifier is indexed by an element $\alpha \in L_\beta$. However, this query needs to be *translated* to a query to $\hat{f}$, which is a pair $(\alpha, \beta) \in S \cup T$. This translation is easily seen to be efficient given α and β. Furthermore, translating a query to $\hat{f}$ into a query to $f : S \to \mathbb{F}$ or $p : T \to \mathbb{F}$ is also easy. If $\beta = q(\alpha)$ we query $p(\alpha)$ because $(\alpha, \beta) \in T$, and otherwise we query $f(\alpha, \beta)$. This translation involves evaluating $q(\alpha)$, which can be done efficiently as argued above.

We conclude that for each level of the recursion, the running time of the preprocessing and index translation is at most polynomial in $|L|$ and ℓ. Since there are $O(\log \ell)$ levels of recursion, we conclude that the running time is as stated, completing our proof. ∎

Next we move to the completeness part of the proof.

PROPOSITION 6.10 (perfect completeness). *If* p *is the evaluation of a polynomial* P *of degree* $< |L|/8$, *then there exists a proof that causes the RS-verifier to accept with probability one.*

This part is straightforward given the intuition developed in the proof sketch of Theorem 3.2. If p is indeed low-degree, then there exists a proper low-degree bivariate polynomial Q that is consistent with it on all rows. Looking at Figure 2 we argue that S is a union of linear spaces and the restriction of Q to each row is low-degree and consistent with p. The formal proof follows.

Proof. To prove the proposition inductively, it suffices to construct $f : S \to \mathrm{GF}(2^\ell)$ so that the function $\hat{f} : S \cup T \to \mathrm{GF}(2^\ell)$ is such that for every $\beta \in L_1$, the $q(\beta)$-row of $\hat{f}$ is a codeword of $\mathrm{RS}(\mathrm{GF}(2^\ell), L_\beta, |L_\beta|/8 - 1)$, and for every $\alpha \in L_0'$, the α-column of $\hat{f}$ is a codeword of $\mathrm{RS}(\mathrm{GF}(2^\ell), L_1, |L_1|/8 - 1)$.

Using Proposition 6.3 we get $P(x) = Q(x, q(x))$ for $q(x) = q_{L_0}(x)$, where

$$\deg_x(Q) < |L_0| \quad \text{and} \quad \deg_y(Q) = \lfloor \deg(P)/\deg(q) \rfloor < (|L|/8)/|L_0| = |L_1|/8. \tag{6.7}$$

Set $f(\alpha, \beta') = Q(\alpha, \beta')$ for every $(\alpha, \beta') \in S$. If $(\alpha, \beta') \in T$ we have $\beta' = q(\alpha)$, so

$$\hat{p}(\alpha, \beta') = \hat{p}(\alpha, q(\alpha)) = p(\alpha) = P(\alpha) = Q(\alpha, q(\alpha)) = Q(\alpha, \beta').$$

Thus, $\hat{f}$ is the evaluation of Q on $S \cup T$. Consider the $q(\beta)$-row of $\hat{f}$ for $\beta \in L_1$. By Proposition 6.6 the $q(\beta)$-row of $S \cup T$ is L_β. By (6.3) we have $|L_\beta| = 8 \cdot |L_0|$ because $\dim(L_\beta) = \dim(L_0) + 3$. By (6.7) we have $\deg(Q(x, q(\beta)) \le \deg_x(Q) < |L_0|$. We conclude that the $q(\beta)$-row of $\hat{f}$ is indeed a member of $\mathrm{RS}(\mathrm{GF}(2^\ell), L_\beta, |L_\beta|/8 - 1)$.

Similarly, by Proposition 6.6 the α-column of $\hat{f}$ is $q(L_1)$. By Proposition 6.4 $\dim(q(L_1)) = \dim(L_1)$, so $|q(L_1)| = |L_1|$. By construction the α-column of $\hat{f}$ is the evaluation of $Q(\alpha, y)$ on $q(L_1)$. Equation (6.7) completes our proof, because $\deg(Q(\alpha, y)) \le \deg_y(Q) < |L_1|/8$. ∎

6.4. Soundness. Our analysis of the soundness is by induction. Assume V_{RS} accepts implicit input p and proof $\pi = \{f, \Pi\}$ with high probability. Let $\hat{p}, \hat{f}$ be the partial bivariate functions as defined in (6.5) and (6.6), respectively. We argue by induction that for most $\alpha \in L_0'$ and $\beta \in L_1$, the α-column and $q(\beta)$-row of $\hat{f}$ are close to polynomials of degree roughly $\sqrt{|L|}$. The analysis of Polishchuk and Spielman implies that $\hat{f}$ restricted to the product set $L_0' \times q(L_1)$ is very close to some low-degree bivariate polynomial. Then we claim that $\hat{p}$ is close to an evaluation of the same polynomial on the set of points T. This implies p is close to a degree-$|L|/8$ univariate polynomial, completing the analysis. Formally, we have the following.

LEMMA 6.11 (soundness). *There exists constant $c \ge 1$ such that for every integer k and ϵ, if*

$$\Pr[\mathrm{V}_{\mathrm{RS}}{}^{(p,\pi)}(\mathrm{GF}(2^\ell), \mathrm{span}(b_1, \ldots, b_k), 2^k/8 - 1) = \mathsf{reject}] \le \epsilon,$$

then p is $\left(c^{\log k} \cdot \epsilon\right)$-close to $\mathrm{RS}(\mathrm{GF}(2^\ell), \mathrm{span}(b_1, \ldots, b_k), 2^k/8 - 1)$.

To prove the lemma, we need a version of the analysis of Polishchuk and Spielman of the bivariate test. The following lemma is directly implied by the main theorem in [37]. We defer its proof to section 6.6 below.

DEFINITION 6.12. *For set $S \subseteq \mathbb{F} \times \mathbb{F}$, partial bivariate function $f : S \to \mathbb{F}$, and nonnegative integers d_1, d_2, define $\delta^{(d_1,d_2)}(f)$ to be the fractional distance of f from a polynomial of degree d_1 in its first variable and d_2 in its second variable. Formally,*

$$\delta^{(d_1,d_2)}(f) \triangleq \min_{\{Q : S \to \mathbb{F} \mid \deg_x(Q) \le d_1, \deg_y(Q) \le d_2\}} \{\delta(f, Q)\}.$$

Let $\delta^{(d,*)}(f)$ and $\delta^{(*,d)}(f)$ denote the fractional distances when the degree in one of the variables is unrestricted.

LEMMA 6.13 (bivariate test on product set [37]). *There exists a universal constant $c_0 \ge 1$ such that the following holds. For every $A, B \subseteq \mathbb{F}$ and integers $d_1 \le |A|/4, d_2 \le |B|/8$ and function $f : A \times B \to \mathbb{F}$, it is the case that*

$$\delta^{(d_1,d_2)}(f) \le c_0 \cdot \left(\delta^{(d_1,*)}(f) + \delta^{(*,d_2)}(f)\right).$$

Proof of Lemma 6.11. The proof is by induction on k. Let $L = \mathrm{span}(b_1, \ldots, b_k)$ and let L_0, L_0', L_1, q be as defined in the beginning of subsection 6.2. We use the following constants, where $\hat{c}$ is a parameter to be minimized and c_0 is the universal constant from Lemma 6.13:

$$c_1 \triangleq \hat{c}^{\log(7/6)}/2; \quad c_2 \triangleq \frac{c_1}{3c_0}; \quad c_3 \triangleq \frac{3c_1}{16(3c_0+2)}.$$

We fix c to be the minimal $\hat{c}$ such that $c_3 \geq 2$ and $\frac{1}{c_3} + \frac{16}{c_1} \leq 1$. Notice that c_1, c_2, c_3 are strictly increasing functions of $\hat{c}$, so c is well defined (we do not attempt to minimize it).

The base case $k \leq 6$ is immediate. For the inductive case we assume that the lemma is true by induction for smaller dimension k' and, in particular, for the recursive calls of the RS-verifier, and we now prove it for dimension $k \geq 7$.

Let $\pi = (f, \Pi)$ be as in Definition 6.5 and assume that (p, π) is rejected by the verifier with probability at most ϵ. We assume without loss of generality that $\epsilon \leq c^{-\log k}$, for otherwise there is nothing to prove. We show below that p is within distance $c^{\log k} \cdot \epsilon$ of some RS-codeword. In what follows let $\hat{p}, \hat{f}$ be the partial bivariate functions defined in (6.5) and (6.6), respectively.

Step 1. *Restricting the bivariate function $\hat{f}$ to a product set $L_0' \times q(L_1)$.* Denote by $\epsilon(\alpha)$ the probability that the inner verifier rejects $\hat{f}|_\alpha^\updownarrow$ (and its proof), and similarly let $\epsilon(\beta)$ be the probability verifier rejects $\hat{f}|_{q(\beta)}^\leftrightarrow$. Let ϵ_{col} be the expectation of $\epsilon(\alpha)$ over random $\alpha \in L_0'$ and let ϵ_{row} be the similar expectation of $\epsilon(\beta)$ over random $\beta \in L_1$. By definition of the verifier, we have $\epsilon = \frac{1}{2}(\epsilon_{\mathrm{row}} + \epsilon_{\mathrm{col}})$. Since these quantities are nonnegative we get $\epsilon_{\mathrm{row}}, \epsilon_{\mathrm{col}} \leq 2\epsilon$.

Let $d_1 = |L_0| - 1$ and recall that $|L_\beta| = 8|L_0|$ for every $\beta \in L_1$. This follows from (6.2) and (6.3). First we bound $\delta^{(d_1,*)}(\hat{f})$. This quantity is the expectation over random $\beta \in L_1$ of the fractional distance of $\hat{f}|_{q(\beta)}^\leftrightarrow$ from a degree-d_1 univariate polynomial. Let $\delta^{(d_1)}(\hat{f}|_{q(\beta)}^\leftrightarrow)$ denote this distance. We get

$$\begin{aligned} \delta^{(d_1,*)}(\hat{f}) &= \mathbb{E}_{\beta \in L_1}\left[\delta^{(d_1)}(\hat{f}|_{q(\beta)}^\leftrightarrow)\right] \leq \mathbb{E}_{\beta \in L_1}\left[\epsilon(\beta) \cdot c^{\log(\dim(L_\beta))}\right] \\ &\leq \epsilon_{\mathrm{row}} \cdot c^{\log(\lfloor k/2 \rfloor + 3)} \leq 2\epsilon \cdot c^{\log \frac{6k}{7}} = \frac{\epsilon}{c_1} \cdot c^{\log k}. \end{aligned} \tag{6.8}$$

The first inequality follows by induction, the second follows because $\dim(L_\beta) = \lfloor k/2 \rfloor + 3$ for every $\beta \in L_1$, the third holds for $k \geq 7$, and the last equality is true for our setting of c_1.

Let f' be the restriction of $\hat{f}$ to $L_0' \times q(L_1)$; i.e., $f' : L_0' \times q(L_1) \to \mathrm{GF}(2^\ell)$ is the function that agrees with $\hat{f}$ on its domain. Since $|L_0'| = |L_\beta|/2$ we get from (6.8)

$$\delta^{(d_1,*)}(f') \leq \frac{2\epsilon}{c_1} \cdot c^{\log k}. \tag{6.9}$$

Let $d_2 = |L_1|/8 - 1$. By analogy to (6.8), we get by induction

$$\delta^{(*,d_2)}(f') \leq \epsilon_{\mathrm{col}} \cdot c^{\log(\lfloor k/2 \rfloor + 1)} \leq \frac{\epsilon}{c_1} \cdot c^{\log k}. \tag{6.10}$$

The conditions of Lemma 6.13 hold with respect to f' and $A = L_0', B = q(L_1)$, because $d_1 \leq |L_0'|/4$ and $d_2 \leq |q(L_1)|/8$. Thus, from (6.9), (6.10), Lemma 6.13, and

our setting of c_2, we conclude that f' is "close" to an evaluation of a low degree bivariate polynomial:

$$\delta^{(d_1,d_2)}(f') \le \frac{\epsilon}{c_2} \cdot c^{\log k}. \tag{6.11}$$

Step 2. *Extending the analysis to the bivariate function* $\hat{p}$. Let Q be the degree-(d_1, d_2) polynomial closest to f'. We wish to bound the probability over random $(\alpha, \tilde{\beta}) \in T$ that $\hat{p}(\alpha, \tilde{\beta}) \neq Q(\alpha, \tilde{\beta})$. Let $\tilde{\beta} = q(\beta)$ and notice that $\tilde{\beta} \in q(L_1)$. This follows from Proposition 6.4. Call $\tilde{\beta}$ *good* if the polynomial closest to $\hat{f}|^{\leftrightarrow}_{\tilde{\beta}}$ is $Q(x, \tilde{\beta})$, and otherwise it is *bad*. We bound the probability as follows:

$$\begin{aligned}\Pr_{(\alpha,\tilde{\beta})\in T}\left[\hat{p}(\alpha,\tilde{\beta}) \neq Q(\alpha,\tilde{\beta})\right] \le \Pr_{\tilde{\beta}\in q(L_1)}\left[\tilde{\beta} \text{ is bad }\right] \\ + \Pr_{(\alpha,\tilde{\beta})\in T}\left[\hat{p}(\alpha,\tilde{\beta}) \neq Q(\alpha,\tilde{\beta}) | \tilde{\beta} \text{ is good}\right].\end{aligned} \tag{6.12}$$

- *First summand of* (6.12). We start by bounding the probability of bad $\tilde{\beta}$. Let $\hat{f}|^{\leftrightarrow}_{\tilde{\beta}}{}'$ be the restriction of $\hat{f}|^{\leftrightarrow}_{\tilde{\beta}}$ to domain L_0' and let $Q_{\tilde{\beta}}(x)$ be the degree-d_1 polynomial closest to $\hat{f}|^{\leftrightarrow}_{\tilde{\beta}}$. If $\tilde{\beta}$ is bad, i.e., $Q_{\tilde{\beta}}(x) \neq Q(x, \tilde{\beta})$, then $\hat{f}|^{\leftrightarrow}_{\tilde{\beta}}{}'$ is either (3/8)-far from $Q(x, \tilde{\beta})$ or (3/8)-far from $Q_{\tilde{\beta}}(x)$. This is because $Q_{\tilde{\beta}}(x)$ and $Q(x, \tilde{\beta})$ can agree on at most $|L_0'|/4$ locations in L_0'. Thus, by (6.9) and (6.11), we get

  $$\Pr_{\tilde{\beta}\in q(L_1)}\left[\tilde{\beta} \text{ is bad }\right] \le 2 \cdot \frac{8}{3} \cdot \max\left\{\frac{1}{c_2}, \frac{2}{c_1}\right\} \cdot \epsilon \cdot c^{\log k} \le \frac{\epsilon}{c_3} \cdot c^{\log k}. \tag{6.13}$$

  The last inequality follows by bounding the maximum of two nonnegative numbers by their sum and holds for our setting of c_1, c_2, c_3.
- *Second summand of* (6.12). Let $T_{\text{good}} = \{(\alpha, \tilde{\beta}) \in T : \tilde{\beta} \text{ is good}\}$. Since $\hat{p}$ is a function on a subdomain of $\hat{f}$ we can bound the second summand in (6.12) as follows:

  $$\begin{aligned}\Pr_{(\alpha,\tilde{\beta})\in T}\left[\hat{p}(\alpha,\tilde{\beta}) \neq Q(\alpha,\tilde{\beta}) | \tilde{\beta} \text{ is good}\right] \le \Pr_{(\alpha,\tilde{\beta})\in S}\left[\hat{f}(\alpha,\tilde{\beta}) \neq Q(\alpha,\tilde{\beta})\right] \\ \cdot \frac{|S|}{|T_{\text{good}}|}.\end{aligned} \tag{6.14}$$

  We already showed in (6.8) that $\Pr_S[\hat{f} \neq Q]$ is relatively small, so we need only to argue that $|T_{\text{good}}|$ is large relative to $|S|$. By Proposition 6.4 the $\tilde{\beta}$-row of T is an affine shift of L_0 by $\tilde{\beta}$. From the proof of the first part of Proposition 6.6 we conclude that the $\tilde{\beta}$-row of T is 1/8 fraction subset of the $\tilde{\beta}$-row of S, so (6.13) implies

  $$|T_{\text{good}}|/|S| = \frac{1}{8} \cdot (1 - \Pr[\tilde{\beta} \text{ is bad}]) \ge 1/16. \tag{6.15}$$

  The last inequality follows from (6.13) by our assumption that $\epsilon \cdot c^{\log k} \le 1$ and because we set $c_3 \ge 2$.

Summing up from (6.13), (6.14), and (6.15) we get

$$\Pr_{(\alpha,\tilde{\beta})\in T}\left[\hat{p}(\alpha,\tilde{\beta}) \neq Q(\alpha,\tilde{\beta})\right] \le \left(\frac{1}{c_3} + \frac{16}{c_1}\right) \epsilon \cdot c^{\log k} \le \epsilon \cdot c^{\log k}. \tag{6.16}$$

The last inequality holds because we set c_1 and c_3 such that $\frac{1}{c_3} + \frac{16}{c_1} \le 1$.

Step 3. *From bivariate $\hat{p}$ to univariate p.* Let $P(x) = Q(x, q(x))$. Notice that $\deg(P) \leq |L|/8 - 1$. This follows from the degree of Q and $\deg(q) = |L_0| = |L'_0|/4$. Using (6.16) and the definition $T = \{(\gamma, q(\gamma)) : \gamma \in L\}$ we conclude that for all but a $(\epsilon \cdot c^{\log k})$-fraction of L we have

$$p(\gamma) = \hat{p}(\gamma, q(\gamma)) = Q(\gamma, q(\gamma)) = P(\gamma).$$

The fractional distance of p from a degree-$|L|/8 - 1$ polynomial is as claimed, completing our proof. □

6.5. Proof of Theorem 3.2. In this subsection we complete the formal proof of Theorem 3.2. First consider the case of degree precisely $|L|/8 - 1$, dealt with in the preceding sections. In particular, the proof of proximity and its associated verifier are described in subsection 6.2. The query complexity, randomness, and proof length are argued in Proposition 6.9. Perfect completeness is asserted by Proposition 6.10. Soundness is analyzed in Lemma 6.11. This completes the proof of the special case. The following proposition, Proposition 6.14, generalizes the degree and completes the full proof of Theorem 3.2.

In what follows we say a soundness function $s : [0,1] \times \mathbb{N}^+ \to [0,1]$ is *monotone* if it increases with δ; i.e., for all n we have $\delta \geq \delta' \Rightarrow s(\delta, n) \geq s(\delta', n)$.

PROPOSITION 6.14. *Let L be either of the pair-languages* PAIR-ADDITIVE-RS *and* PAIR-SMOOTH-RS*, and let $L_{\frac{1}{8}}$ be the restriction of L to explicit pairs of the form $(\mathbb{F}, S, |S|/8 - 1)$. Suppose*

$$L_{\frac{1}{8}} \in \mathbf{Strong\text{-}PCPP}_{s(\delta,n)} \left[\begin{array}{l} \mathsf{randomness}\ r(n), \\ \mathsf{query}\ q(n), \\ \mathsf{distance\ Hamming}_{\mathbb{F}} \end{array} \right],$$

where $s(\delta, n)$ is monotone and $r(n) \geq \log n$. Then for $s'(\delta, n) = \min\{\delta/2, s(\delta/64, n)\}$,

$$L \in \mathbf{Strong\text{-}PCPP}_{s'(\delta,n)} \left[\begin{array}{l} \mathsf{randomness}\ r(n), \\ \mathsf{query}\ O(q(n)), \\ \mathsf{distance\ Hamming}_{\mathbb{F}} \end{array} \right].$$

Proof. Let (x, p) be an instance to L with explicit input $x = (\mathbb{F}, S, d')$. Denote $d = \frac{|S|}{8} - 1$ and let $V_{\frac{1}{8}}$ denote the verifier for $L_{\frac{1}{8}}$. We start with the case of $d' < d$. On explicit input $(\mathbb{F}, S, d' < d)$ the verifier expects a (concatenation of) two subproofs for $\mathrm{RS}(\mathbb{F}, S, d)$, denoted π_1, π_2. The verifier operates as follows:

- Toss $r(n)$ coins. Let $\mathcal{R}$ denote the random string.
- Invoke $V_{\frac{1}{8}}$ using randomness $\mathcal{R}$ on explicit input $(\mathbb{F}, S, d)$, implicit input p, and proof π_1.
- Fix $Q(z) \triangleq z^{d-d'}$ and set $p'(z) = p(z) \cdot Q(z)$. Invoke $V_{\frac{1}{8}}$ using randomness $\mathcal{R}$ on explicit input $(\mathbb{F}, S, d)$, implicit input p', and proof π_2.

Notice that querying $p'(\alpha)$ can be simulated by querying $p(\alpha)$ and multiplying the answer by $Q(\alpha)$. Additionally evaluating $Q(\alpha)$ can be done in time polylog $|\mathbb{F}|$. So the running time, query complexity, and randomness are essentially inherited from $V_{\frac{1}{8}}$. Completeness follows by observing that $\deg(p) \leq d'$ implies $\deg(p') \leq d$. As to soundness, there are two cases to consider. If p is $\delta/4$-far from $\mathrm{RS}(\mathbb{F}, S, d)$ then by assumption, the first subtest rejects with probability at least $s(\delta/4, n) \geq s(\delta/64, n)$ (the previous inequality follows from monotonicity). Otherwise, p is within relative distance $\delta/4 \leq 1/4$ of an evaluation of a polynomial P with $d' < \deg(P) \leq d$. In this

case, p' is 1/4-close to the evaluation of $P'(z) = Q(z) \cdot P(z)$, where $d < \deg(P') < |S|/4$. Thus, P' is 3/4-far from $\mathrm{RS}(\mathbb{F}, S, d)$, so the distance of p' from the same code is at least $1/2 > \delta/4$. We conclude that the second subtest rejects with probability $s(\delta/4, n) \geq s(\delta/64, n)$.

Next assume $d' > d$ and notice without loss of generality that $d' \leq 8(d+1)$ because otherwise every implicit input is a codeword. The key observation is that a polynomial $P(z)$ is of degree d' iff it can be written as a sum $P(z) = \sum_{i=0}^{7} z^{i(d+1)} \cdot P_i(z)$, where $\deg(P_i) = d_i \leq d$ can be uniquely and efficiently computed given d and d'. Let $L_{(\leq d)}$ be the restriction of L to instances of degree $\leq d$ and let $V_{(\leq d)}$ denote the verifier for $L_{(\leq d)}$. The proof for explicit input $(\mathbb{F}, S, d')$ consists of eight functions $p_0, \ldots p_7 : S \to \mathbb{F}$ and eight proofs of proximity to $L_{(\leq d)}$ denoted $\pi_0, \ldots, \pi_7$. On explicit input $(\mathbb{F}, S, d')$ and implicit input p, the verifier operates as follows:

- Toss $r(n)$ coins. Let $\mathcal{R}$ denote the random string.
- For $i = 0, \ldots, 7$, invoke $V_{(\leq d)}$ using randomness $\mathcal{R}$ on explicit input $(\mathbb{F}, S, d_i)$, implicit input p_i, and proof π_i.
- Using $\mathcal{R}$, select uniformly at random $\gamma \in S$. Accept iff $p(\gamma) = \sum_{i=0}^{7} \gamma^{i(d+1)} p_i(\gamma)$.

Proof length, randomness, completeness, running time, and query complexity follow from construction. As to soundness, assume that p is δ-far from $\mathrm{RS}(\mathbb{F}, S, d')$. There are two cases to consider. If $p(z)$ disagrees with $\sum_{i=0}^{7} z^{i(d+1)} p_i(z)$ on a $\delta/2$-fraction of $z \in S$, then the second subtest rejects with probability $\geq \delta/2$. Otherwise, $p(z)$ is $\delta/2$-close to $\sum_{i=0}^{7} z^{i(d+1)} p_i(z)$. In this case at least one p_i must be $\delta/16$-far from $\mathrm{RS}(\mathbb{F}, S, d_i)$. So the first part of this proof (for the case $d' < d$) implies the rejection probability is at least $s(\frac{\delta}{4 \cdot 16}, n)$. This completes our proof. ∎

6.6. Proof of Lemma 6.13. The lemma is an immediate corollary of the bivariate testing theorem of Polishchuk and Spielman [37, Theorem 9]. We use here the general version of it appearing in Spielman's thesis.

THEOREM 6.15 (see [39, Theorem 4.2.19]). *Let $\mathbb{F}$ be a field, $S, T \subseteq \mathbb{F}$. Let $R(x, y)$ be a polynomial over $\mathbb{F}$ of degree $(d, |T| - 1)$ and let $C(x, y)$ be a polynomial over $\mathbb{F}$ of degree $(|S| - 1, e)$. If*

$$\Pr_{(x,y) \in S \times T} [R(x, y) \neq C(x, y)] < \gamma^2 \quad \text{and} \quad 2\left(\frac{d}{|S|} + \frac{e}{|T|} + \gamma\right) < 1,$$

then there exists a polynomial $Q(x, y)$ of degree (d, e) such that

$$\Pr_{(x,y) \in S \times T} [R(x, y) \neq Q(x, y) \text{ or } C(x, y) \neq Q(x, y)] < 2\gamma^2.$$

To prove Lemma 6.13 we show the contrapositive form for $c_0 = 128$, making no attempt to optimize constants. We may assume without loss of generality that $\delta^{(d,*)}, \delta^{(*,e)} < 1/c_0$; otherwise the claim is trivial. Correct each row of f to its closest RS-codeword (breaking ties arbitrarily), obtaining a bivariate polynomial $R(x, y)$ of degree $(d, |T| - 1)$. By definition, $\Delta(R(x, y), f) = \delta^{(d,*)}(f)$. Similarly, correct the columns of f to obtain the polynomial $C(x, y)$ of degree $(|S| - 1, e)$ that is within fractional distance $\delta^{(*,e)}(f)$ of f. We get

$$\Pr_{(x,y) \in S \times T} [R \neq C] \leq \delta^{(d,*)}(f) + \delta^{(*,e)}(f) = \gamma^2 < 1/64.$$

Since $\gamma \leq 1/8, d \leq |S|/4$, and $e \leq |T|/8$, both conditions of Theorem 6.15 hold, allowing us to conclude that $R(x, y)$ is $(2\gamma^2)$-close to $\mathrm{RM}(\mathbb{F}, S \times T, (d, e))$. The triangle inequality completes the proof:

$$\delta^{(d,e)}(f) \leq \Delta(f, R) + \Delta(R, \mathrm{RM}(\mathbb{F}, S \times T, (d, e))) \leq 3\delta^{(d,*)}(f) + 2\delta^{(*,e)}(f).$$

7. PCPPs for Reed–Solomon codes over smooth fields. In this section we give a PCPP-verifier for Reed–Solomon codes over smooth fields, when the set S over which the polynomials are evaluated are multiplicative subfields of the field, thereby proving Theorem 3.4 (restated below). We also show it suffices for obtaining quasilinear PCPs (Theorem 2.2). Our presentation mirrors that of the additive case presented in section 6.

THEOREM 7.1 (Theorem 3.4, restated). *Let* PAIR-SMOOTH-RS *be the restriction of* PAIR-RS *to pairs* $((\mathbb{F}, \langle\omega\rangle, d), p)$*, where* $\mathrm{ord}(\omega) = n$ *is a power of* 2*. Then,*

$$\text{PAIR-SMOOTH-RS} \in \mathbf{Strong\text{-}PCPP}_{\delta/\operatorname{polylog} n}\left[\begin{array}{l}\mathsf{randomness}\ \log(n \cdot \operatorname{polylog} n),\\ \mathsf{query}\ O(1),\\ \mathsf{distance}\ \mathsf{Hamming}_{\mathbb{F}}\end{array}\right].$$

7.1. Proof overview. This section should be read as a continuation of subsection 6.1. The crucial property used in our constructions in section 6 was that the linearized polynomial $q(z)$ "nicely partitions" the linear space L. Specifically, $q(z)$ defines a linear map on L, its image is a linear space of size $\approx \sqrt{|L|}$, and for every value in its image, the set of preimages of that value form an affine space of size $\approx \sqrt{|L|}$.

In the smooth case, $\mathbb{F}$ contains a multiplicative subgroup $S = \langle\omega\rangle$ of size n. Assume that $\sqrt{n}$ is an integer and consider a polynomial $P(z)$ evaluated over S. Using Proposition 6.3 with the polynomial $q(z) \triangleq z^{\sqrt{n}}$ we get $P(z) = Q(z, z^{\sqrt{n}})$. Notice that $q(z)$ "nicely partitions" $\langle\omega\rangle$. Specifically, $q(\langle w\rangle) = \langle\omega^{\sqrt{n}}\rangle$ is of size $\sqrt{n}$ (recall $q(S) \triangleq \{q(s) : s \in S\}$), and for every value in the image of q, the set of its preimages is a multiplicative coset of $\langle\omega^{\sqrt{n}}\rangle$.

Thus, to prove proximity of $P(z)$ to $\mathrm{RS}(\mathbb{F}, S, d)$ we may ask for an evaluation of $Q(x, y)$ on the set of points $(X \times Y) \cup Z$, where $Z = \{(z, q(z)) : z \in \langle\omega\rangle\}$ and $X = Y = \langle\omega^{\sqrt{n}}\rangle$. In the additive case we used the fact, implied by Proposition 6.6, that the union of a linear space (L'_0) and an affine shift of it ($L'_0 + \beta$) form a linear space of slightly larger dimension. In the smooth case, it is not true in general that $\langle\omega^{\sqrt{n}}\rangle$ and a coset of it form a small multiplicative group. In fact, the smallest group containing both can be as large as $\langle\omega\rangle$. To overcome this problem, we define the *shifted Reed–Solomon code* (SRS-code), which is formed of evaluations of polynomials over a multiplicative group $\langle\omega\rangle$ and a coset of it of the form $\kappa\langle\omega\rangle \triangleq \{\kappa z : z \in \langle w\rangle\}$.

The crucial observation is that $q(z)$ "nicely partitions" each of $\langle\omega\rangle$ and $\kappa\langle\omega\rangle$ into $\sqrt{n}$ cosets of $\langle\omega^{\sqrt{n}}\rangle$ (see Figure 3). Indeed, the image of $q(\langle\omega\rangle) = \langle\omega^{\sqrt{n}}\rangle$ and $q(\kappa\langle\omega\rangle) = \kappa^{\sqrt{n}}\langle\omega^{\sqrt{n}}\rangle$. Similarly, for an element in $q(\langle\omega\rangle)$ of the form $\omega^{j\sqrt{n}}, j \in [\sqrt{n}]$, we get $q^{(-1)}(\omega^{j\sqrt{n}}) = \omega^j\langle\omega^{\sqrt{n}}\rangle$ and for an element in $q(\kappa\langle\omega\rangle)$ of the form $\kappa^{\sqrt{n}}\omega^{j\sqrt{n}}$ we get $q^{(-1)}(\kappa^{\sqrt{n}}\omega^{j\sqrt{n}}) = \kappa\omega^j\langle\omega^{\sqrt{n}}\rangle$. Thus, we will ask our prover to provide an evaluation of the bivariate polynomial Q on the points (see Figure 4)

$$\{(z, q(z)) : z \in \langle\omega\rangle \cup \kappa\langle\omega\rangle\} \bigcup \left(\langle\omega^{\sqrt{n}}\rangle \times (\langle\omega^{\sqrt{n}}\rangle \cup \kappa^{\sqrt{n}}\langle\omega^{\sqrt{n}}\rangle)\right).$$

By our previous discussion we notice that the restriction of Q to certain rows and columns forms a word of an SRS-code of length $\approx \sqrt{n}$.

This allows us to measure proximity to the SRS-code of length n by measuring proximity to SRS-codes of size $\approx \sqrt{n}$. As in the additive case of section 6 we use the bivariate testing lemma, Lemma 6.13, to apply recursion and obtain quasilinear

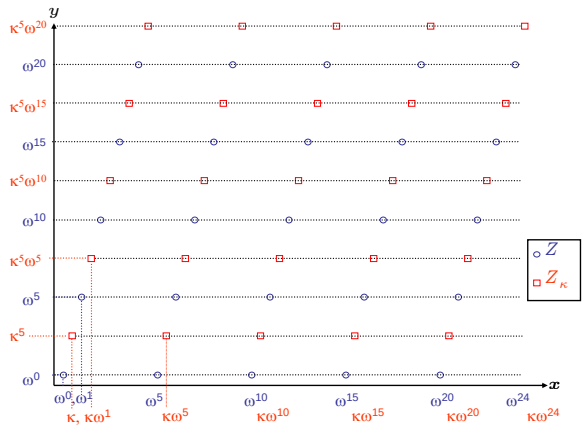

FIG. 3. *In this case,* $\mathbb{F} = \mathbb{Z}_{101}$. *Let* σ *generate* $\mathbb{F}^*$, *let* $\omega = \sigma^4$ *be an element of order* $n = 25$, *and let* $\kappa = \sigma^2$ *and* $q(z) = z^5$. *The elements on each axis are ordered by increasing powers of* σ *and the figure shows the subsets of points* $Z, Z_\kappa \subset \mathbb{F}^* \times \mathbb{F}^*$, *where* $Z = \{(z, q(z)) : z \in \langle\omega\rangle\}$ *and* $Z_\kappa = \{(\kappa z, q(\kappa z)) : z \in \langle\omega\rangle\}$.

sized proofs that can be tested with polylogarithmic query complexity. We need some technical modifications, arising from difficulties similar to the additive case. In particular, the degree of Q in its first variable is too large for applying Lemma 6.13, so we reduce this degree by breaking Q into a sum of several polynomials of sufficiently small degree. Additionally, we will not assume that $\sqrt{n}$ is an integer; rather we use the fact that $n = 2^k$ and work with the multiplicative subgroups generated by $n_0 = 2^{\lceil k/2 \rceil}, n_1 = 2^{\lfloor k/w \rfloor}$ that are of size $\approx \sqrt{n}$.

7.2. The shifted Reed–Solomon code. We prove Theorem 3.4 by proving a stronger statement about testing proximity to shifted RS-codes, defined next.

DEFINITION 7.2 (shifted Reed–Solomon code). *For* $\mathbb{F}$ *a finite field,* $\omega, \kappa \in \mathbb{F}^*, \mathrm{ord}(\omega) = n$, *and integer* d, *the degree-*d shifted Reed–Solomon (SRS)-code *over* $\langle\omega\rangle$ *with* shift κ *is*

$$\mathrm{SRS}(\mathbb{F}, d, \omega, \kappa) \triangleq \mathrm{RS}(\mathbb{F}, \langle\omega\rangle \cup \kappa\langle\omega\rangle, d).$$

Let PAIR-SMOOTH-SRS *be the pair language whose explicit inputs are triples* $(\mathbb{F}, S = \langle\omega\rangle \cup \kappa\langle\omega\rangle, d)$, *where* $\mathrm{ord}(\omega)$ *is a power of* 2 *and whose implicit inputs are functions* $p : S \to \mathbb{F}$. *The size of (explicit and implicit) inputs is* $\mathrm{ord}(\omega)$. *A pair* $((\mathbb{F}, S, d), p)$ *is in* PAIR-*SRS if* $p \in \mathrm{SRS}(\mathbb{F}, \omega, \kappa, d)$.

Notice that $\mathrm{SRS}(\mathbb{F}, \omega, 1, d) = \mathrm{RS}(\mathbb{F}, \langle\omega\rangle, d)$. Thus, Theorem 3.4 follows from the following theorem, the proof of which occupies the rest of the section.

THEOREM 7.3 (SRS PCP of proximity).

$$\text{PAIR-SMOOTH-SRS} \in \mathbf{Strong\text{-}PCPP}_{\delta/\operatorname{polylog} n}\left[\begin{array}{l}\mathsf{randomness}\ \log(n \cdot \operatorname{polylog} n),\\ \mathsf{query}\ O(\log|\mathbb{F}|),\\ \mathsf{distance}\ \mathsf{Hamming}_{\mathbb{F}}\end{array}\right].$$

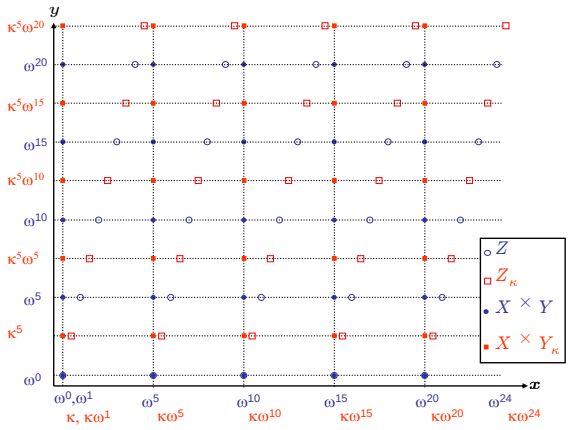

FIG. 4. *The proof of proximity for the smooth RS-code is the evaluation of Q on the set of points $Z \cup Z_\kappa \cup (X \times Y) \cup (X \times Y_\kappa)$. Notice that the restriction of this set to every row and column gives a pair—a multiplicative group of order $\sqrt{n}$ and a coset of it.*

As in the additive case, our proof will focus on the special case of degree $d = n/8 - 1$, and Proposition 6.14 generalizes this to an arbitrary degree.

7.3. The SRS proof of proximity and its associated verifier.

Notation. Recall $\mathrm{ord}(\omega) = n = 2^k$ for integer k. Let $n_0 = 2^{\lceil k/2 \rceil}$ and $n_1 = 2^{\lfloor k/2 \rfloor}$. Note that $n = n_0 \cdot n_1$ and $\sqrt{n/2} \le n_1 \le n_0 \le \sqrt{2n}$. For $r \le \mathrm{ord}(\alpha)$, let $\langle \alpha \rangle_r \triangleq \{\alpha^0, \alpha^1, \ldots, \alpha^{r-1}\}$. When dealing with a purported codeword of $\mathrm{SRS}(\mathbb{F}, \omega, \kappa, d)$ we treat it as a *pair* of functions, $p : \langle \omega \rangle \to \mathbb{F}$ and $p_\kappa : \kappa\langle \omega \rangle \to \mathbb{F}$.

DEFINITION 7.4 (SRS proof of proximity). *The proof of proximity for a purported codeword of the code $\mathrm{SRS}(\mathbb{F}, \omega, \kappa, n/8 - 1)$ is defined by induction on $n = \mathrm{ord}(\omega)$. If $n \le 16$ then it is empty. Otherwise, it is of the form*

$$\pi = (\{f^{(\ell)}, f_\kappa^{(\ell)}, g^{(\ell)}, g_\kappa^{(\ell)}, \{\pi^{(1,\beta,\ell)}, \pi^{(2,\beta,\ell)}\}_{\beta \in \langle \omega \rangle_{n_1}}, \{\pi^{(3,\tilde{\alpha},\ell)}\}_{\tilde{\alpha} \in \langle \omega^{n_1} \rangle}\}_{\ell \in \{0,\ldots,7\}}),$$

where

- $f^{(\ell)}, f_\kappa^{(\ell)} : \langle \omega^{n_1} \rangle \times \langle \omega \rangle_{n_1} \to \mathbb{F}$,
- $g^{(\ell)}, g_\kappa^{(\ell)} : \langle \omega^{n_1} \rangle \times \langle \omega^{n_0} \rangle \to \mathbb{F}$, *and*
- $\pi^{(1,\cdot,\ell)}, \pi^{(2,\cdot,\ell)}, \pi^{(3,\cdot,\ell)}$ *are proofs for* SRS*-codewords (over $\mathbb{F}$) of sizes n_0, n_0, n_1, respectively.*

We are now ready to describe the proximity tester.

DEFINITION 7.5 (SRS-verifier). *The verifier for proximity to $\mathrm{SRS}(\mathbb{F}, \omega, \kappa, d = \mathrm{ord}(\omega)/8 - 1)$ is denoted $\mathrm{V}_{\mathrm{SRS}}^{\langle (p, p_\kappa), \pi \rangle}(\mathbb{F}, \omega, \kappa, d)$. It receives as explicit input the parameters $\mathbb{F}, \omega, \kappa$ as defined in the statement of Theorem 7.3. The implicit input is a pair of functions $p : \langle \omega \rangle \to \mathbb{F}, p_\kappa : \kappa\langle \omega \rangle \to \mathbb{F}$. The proof π is as described in Definition 7.4. The verifier operates as follows.*

Base case ($n \le 16$). *The verifier reads p and p_κ in entirety and accepts iff $(p, p_\kappa) \in \mathrm{SRS}(\mathbb{F}, \omega, \kappa, 1)$.*

Recursion ($n \ge 32$). *The verifier computes $n_0 = 2^{\lceil k/2 \rceil}, n_1 = 2^{\lfloor k/2 \rfloor}$ and performs one of the following four tests with probability $1/4$ each.*

Outer. *Pick* $\tilde{\alpha} \in \langle \omega^{n_1} \rangle, \beta \in \langle \omega \rangle_{n_1}$ *uniformly at random; query* $p(\tilde{\alpha} \cdot \beta)$, $p_\kappa(\tilde{\alpha} \cdot \beta)$ *and* $f^{(\ell)}(\tilde{\alpha}, \beta)$, $f^{(\ell)}_\kappa(\tilde{\alpha}, \beta)$ *for every* $\ell \in \{0, \ldots, 7\}$*; accept iff* $p(\tilde{\alpha} \cdot \beta) = \sum_{\ell=0}^{7} (\tilde{\alpha} \cdot \beta)^{\ell n_0/8} \cdot f^{(\ell)}(\tilde{\alpha}, \beta)$ *and* $p_\kappa(\kappa\tilde{\alpha} \cdot \beta) = \sum_{\ell=0}^{7} (\kappa\tilde{\alpha} \cdot \beta)^{\ell n_0/8} \cdot f^{(\ell)}_\kappa(\tilde{\alpha}, \beta)$.

Inner. *Pick* $\ell \in \{0, \ldots, 7\}, \beta \in \langle \omega \rangle_{n_1}$ *at random and invoke*

$$\mathrm{V}_{\mathrm{SRS}}^{\langle (g^{(\ell)}|^{\leftrightarrow}_{\beta^{n_0}}, f^{(\ell)}|^{\leftrightarrow}_{\beta}), \pi^{(1,\beta,\ell)} \rangle}(\mathbb{F}, \omega^{n_1}, \beta, n_0/8 - 1).$$

Inner$_\kappa$. *Pick* $\ell \in \{0, \ldots, 7\}, \beta \in \langle \omega \rangle_{n_1}$ *at random and invoke*

$$\mathrm{V}_{\mathrm{SRS}}^{\langle (g^{(\ell)}_\kappa|^{\leftrightarrow}_{\beta^{n_0}}, f^{(\ell)}_\kappa|^{\leftrightarrow}_{\beta}), \pi^{(2,\beta,\ell)} \rangle}(\mathbb{F}, \omega^{n_1}, \kappa\beta, n_0/8 - 1).$$

Inner$_c$. *Pick* $\ell \in \{0, \ldots, 7\}, \tilde{\alpha} \in \langle \omega^{n_1} \rangle$ *at random and invoke*

$$\mathrm{V}_{\mathrm{SRS}}^{\langle (g^{(\ell)}|^{\updownarrow}_{\tilde{\alpha}}, g^{(\ell)}_\kappa|^{\updownarrow}_{\tilde{\alpha}}), \pi^{(3,\alpha,\ell)} \rangle}(\mathbb{F}, \omega^{n_0}, \kappa^{n_0}, n_1/8 - 1).$$

The remaining subsections analyze the performance of this verifier, thus yielding Theorem 3.4. Specifically, the next subsection analyzes the simple properties including the query complexity, the randomness/size complexity, and the completeness. The hard part, the soundness analysis, is addressed in subsection 7.5.

7.4. Basic properties.

PROPOSITION 7.6. $\mathrm{V}_{\mathrm{SRS}}^{\langle (p, p_\kappa), \pi \rangle}(\mathbb{F}, \omega, \kappa, n/8 - 1)$ *makes at most* 32 *queries into* p, p_κ, π. *It tosses at most* $\log_2 n + O(\log \log n)$ *random coins and runs in time* $\mathrm{poly}\, n$. *The size of the proof* π *is* $O(n \cdot \mathrm{polylog}\, n)$.

Proof. The proof is straightforward from the definition. The query complexity is easy to verify. In the base case, the verifier reads 32 field elements. In the inductive case, if the verifier chooses to execute the **Outer** step, then it makes $18 < 32$ queries; else it makes a recursive query to $\mathrm{V}_{\mathrm{SRS}}^{\langle\rangle}$ which makes 32 queries by induction.

The randomness complexity is similar. In the base case the verifier tosses 0 coins. In the inductive case, the verifier tosses $O(1)$ coins to determine which step to perform. If it chooses the outer test, it picks $\tilde{\alpha}$ and β at random with $\log n + O(1)$ coins. If it chooses one of the inner tests, it tosses $\log \sqrt{n} + O(1)$ coins to determine the inner call, and then $\log \sqrt{n} + O(\log \log \sqrt{n})$ coins in the recursive call. Adding up, we get a total of $\log n + O(\log \log n)$ coins in all. Notice that all computations are simple and can be performed in time $\mathrm{poly}\, n$. Finally, the size of the proof is bounded by $2^{\mathrm{randomness}}$. ∎

Next we move to the completeness part of the proof. This part is straightforward given the intuition developed in subsection 7.1. We first generalize Proposition 6.3 and express a univariate polynomial as a sum of bivariate polynomials of low degree. We then use this to describe a proof π that is accepted with probability 1 when accompanying an SRS-codeword.

PROPOSITION 7.7. *Given positive integers* d_1, d_2, L, *and* d *such that* $d_1 \cdot d_2 \cdot L \geq d$, *the following holds: For every univariate polynomial* $P(x)$ *of degree less than* d *there exists a sequence of* L *bivariate polynomial* $Q^{(0)}(y, z), \ldots, Q^{(L-1)}(y, z)$, *of degree less than* d_1 *in* y *and* d_2 *in* z, *such that*

$$P(x) = \sum_{\ell=0}^{L-1} x^{\ell \cdot d_1} Q^{(\ell)}(x, x^{L \cdot d_1}).$$

Furthermore, such a sequence is unique if $d_1 \cdot d_2 \cdot L = d$.

Proof. Let the a_i's be the coefficients of P; i.e., $P(x) = \sum_{i=0}^{d-1} a_i x^i$. Now let

$$Q^{(\ell)}(y,z) = \sum_{i=0}^{d_1-1} \sum_{j=0}^{d_2-1} a_{i+\ell\cdot d_1 + j\cdot d_1 \cdot L} y^i z^j,$$

where a_i is defined to be 0 if $i \geq d$. It can be verified by inspection that we have

$$P(x) = \sum_{\ell=0}^{L-1} x^{\ell \cdot d_1} Q^{(\ell)}(x, x^{L\cdot d_1}).$$

Uniqueness follows from a counting argument: the set of sequences of polynomials $Q^{(0)}, \ldots, Q^{(L-1)}$ forms a vector space of dimension $L \cdot d_1 \cdot d_2 = d$, the dimension of the space of polynomials of degree less than d. ∎

PROPOSITION 7.8 (completeness). *If* (p, p_κ) *equal the SRS encoding of some polynomial* P *of degree less than* $n/8$*, then there exists a proof that causes the SRS proximity tester to accept with probability one.*

Proof. The proof is by induction on n. Let $Q^{(0)}, \ldots, Q^{(7)}$ be the polynomials as given by Proposition 7.7 applied to P with integers $d_1 = n_0/8$, $d_2 = n_1/8$, $L = 8$, and $d = n/8$. Note that we have $d_1 \cdot d_2 \cdot L = d$, since $n_0 \cdot n_1 = n$. For every $\ell \in \{0, \ldots, 7\}$, we let $f^{(\ell)}(\tilde{\alpha}, \beta) = Q^{(\ell)}(\tilde{\alpha}\beta, \beta^{n_0})$, $f_\kappa^{(\ell)}(\tilde{\alpha}, \beta) = Q^{(\ell)}(\kappa\tilde{\alpha}\beta, \kappa^{n_0}\beta^{n_0})$, $g^{(\ell)}(\tilde{\alpha}, \tilde{\beta}) = Q^{(\ell)}(\tilde{\alpha}, \tilde{\beta})$, and $g_\kappa^{(\ell)}(\tilde{\alpha}, \tilde{\beta}) = Q^{(\ell)}(\tilde{\alpha}, \kappa^{n_0}\tilde{\beta})$ for every $\tilde{\alpha} \in \langle \omega^{n_1} \rangle$, $\beta \in \langle \omega \rangle_{n_1}$, and $\tilde{\beta} \in \langle \omega^{n_0} \rangle$.

Note that the above choice of table $f^{(\ell)}$, $f_\kappa^{(\ell)}$, $g^{(\ell)}$, $g_\kappa^{(\ell)}$ is such that the **Outer** test accepts with probability one. Specifically, we have

$$\begin{aligned} p(\tilde{\alpha}\cdot\beta) &= P(\tilde{\alpha}\cdot\beta) \\ &= \sum_{\ell \in \{0,\ldots,7\}} (\tilde{\alpha}\cdot\beta)^{\ell n_0/8} Q^{(\ell)}(\tilde{\alpha}\beta, \tilde{\alpha}^{n_0}\beta^{n_0}) \\ &= \sum_{\ell \in \{0,\ldots,7\}} (\tilde{\alpha}\cdot\beta)^{\ell n_0/8} Q^{(\ell)}(\tilde{\alpha}\beta, \beta^{n_0}) \\ &= \sum_{\ell \in \{0,\ldots,7\}} (\tilde{\alpha}\cdot\beta)^{\ell n_0/8} f^{(\ell)}(\tilde{\alpha}, \beta). \end{aligned}$$

Similarly we get $p_\kappa(\kappa\tilde{\alpha}\cdot\beta) = \sum_{\ell=0}^{7} (\kappa\tilde{\alpha}\cdot\beta)^{\ell n_0/8} \cdot f_\kappa^{(\ell)}(\tilde{\alpha}, \beta)$.

Now we describe how to set up the rest of the subproofs $\pi^{(\cdot,\cdot,\cdot)}$ such that the inner tests accept. For this part, note that the recursive calls to the SRS proximity verifiers access implicit input pairs that satisfy the completeness condition on smaller inputs. Consider, for example, the invocation

$$\mathrm{V}_{\mathrm{SRS}}^{\langle (g^{(\ell)}|_{\beta^{n_0}}^{\leftrightarrow}, f^{(\ell)}|_{\beta}^{\leftrightarrow}), \pi^{(1,\beta,\ell)} \rangle}(\mathbb{F}, \omega^{n_1}, \beta, n_0/8 - 1)$$

by **Inner** for some $\ell \in \{0, \ldots, 7\}$ and $\beta \in \langle \omega \rangle_{n_1}$. We may relate these implicit inputs to the polynomial $Q^{(\ell)}$ as follows: We have $g^{(\ell)}|_{\beta^{n_0}}^{\leftrightarrow}(\tilde{\alpha}) = g^{(\ell)}(\tilde{\alpha}, \beta^{n_0}) = Q^{(\ell)}(\tilde{\alpha}, \beta^{n_0})$, $f^{(\ell)}|_{\beta}^{\leftrightarrow}(\tilde{\alpha}) = f^{(\ell)}(\tilde{\alpha}, \beta) = Q(\tilde{\alpha}\cdot\beta, \beta^{n_0})$. Thus, if we let $P'(\tilde{\alpha}) = Q^{(\ell)}(\tilde{\alpha}, \beta^{n_0})$ and $\omega' = \omega^{n_1}$, then the pair $f^{(\ell)}|_{\beta}^{\leftrightarrow}, g^{(\ell)}|_{\beta^{n_0}}^{\leftrightarrow}$ is a codeword of the SRS-code $\mathrm{SRS}(\mathbb{F}, \omega', \beta, n_0/8-1)$ corresponding to the encoding of P', and thus (by induction) there exists a proof $\pi^{(1,\beta,\ell)}$ that causes the recursive verifier to accept with probability one. Similar reasoning shows that the verifier also accepts with probability one when invoking $\mathbf{Inner}_\kappa$ or $\mathbf{Inner}_c$. ∎

7.5. Soundness. We now argue the soundness of the SRS-verifier as follows. By induction, for most $\tilde{\alpha}$ and β, the functions $g^{(\ell)}|^{\updownarrow}_{\tilde{\alpha}}$, $g^{(\ell)}|^{\leftrightarrow}_{\beta^{n_0}}$, $g^{(\ell)}_{\kappa}|^{\updownarrow}_{\tilde{\alpha}}$, and $g^{(\ell)}_{\kappa}|^{\leftrightarrow}_{\beta^{n_0}}$ are close to polynomials of degree roughly $\sqrt{n}$. The bivariate testing lemma, Lemma 6.13, implies that $g^{(\ell)}$ and $g^{(\ell)}_{\kappa}$ are very close to some low-degree bivariate polynomials $Q^{(\ell)}$ and $Q^{(\ell)}_{\kappa}$. Furthermore, we will show $Q^{(\ell)} \equiv Q^{(\ell)}_{\kappa}$. Next, we claim the function $f^{(\ell)}(z, z^{n_0})$ is close to the function $Q^{(\ell)}(z, z^{n_0})$ and similarly $f^{(\ell)}_{\kappa}$ is close to $Q^{(\ell)}(\kappa z, (\kappa z)^{n_0})$. Finally, we claim that $p(z)$ is close to $\sum_{\ell=0}^{7} z^{\ell n_0/8} \cdot Q^{(\ell)}(z, z^{n_0})$, i.e., p is close to a low-degree univariate polynomial. Similarly p_κ is close to *the same* low-degree polynomial, where the consistency of p and p_κ follows from the equivalence of Q and Q_κ.

LEMMA 7.9 (soundness). *There exists a constant c such that for every ϵ the following holds. If*

$$\Pr\left[\mathrm{V}_{\mathrm{SRS}}^{\langle (p,p_\kappa),\pi\rangle}(\mathbb{F}, \omega, \kappa, \mathrm{ord}(\omega)/8 - 1) = \texttt{reject}\right] \leq \epsilon,$$

then (p, p_κ) is $(c^{\log\log \mathrm{ord}(\omega)} \cdot \epsilon)$-close to $\mathrm{SRS}(\mathbb{F}, \omega, \kappa, \mathrm{ord}(\omega)/8 - 1)$.

Proof. Let c_0 be as in Lemma 6.13. Let $c_1 = 128 \cdot c_0$, $c_2 = (320 + 2c_1)$, and $c_3 = 8c_2 + 4$. We prove the lemma for $c = c_3^2$, which is a (large) constant. Note that the conditions imply $c > 1$ and $c > (2 \cdot (256 + 4c_1))^2$ as will be used later.

We assume the lemma is true by induction for smaller n and in particular for the recursive calls to the various **Inner** tests, and we now prove it for n. Assume $c^{\log\log n} \cdot \epsilon \leq 1$ or else the claim is vacuously true. We use below the fact that $c^{\log\log n_0} \leq c^{\log\log \sqrt{2n}} \leq c^{\log\log n - \frac{1}{2}}$ for every $c \geq 1$ and $n \geq 16$.

Denote by $\epsilon_O(\tilde{\alpha}, \beta)$ the probability that the **Outer** verifier rejects (p, p_κ, π) on random choice $\tilde{\alpha}$ and β. Let ϵ_O denote the expectation of $\epsilon_O(\tilde{\alpha}, \beta)$ over the choice of $\tilde{\alpha}$ and β. Similarly let $\epsilon_I(\ell, \beta)$, $\epsilon_\kappa(\ell, \beta)$, and $\epsilon_c(\ell, \tilde{\alpha})$ denote the probability that **Inner**, **Inner**$_\kappa$, and **Inner**$_c$ reject on random choice ℓ, β, and $\tilde{\alpha}$. Let $\epsilon_I(\ell)$, $\epsilon_\kappa(\ell)$, and $\epsilon_c(\ell)$ denote the expectations of these quantities over β and $\tilde{\alpha}$, and let ϵ_I, ϵ_κ, and ϵ_c denote the expectations over β, $\tilde{\alpha}$, and ℓ. By definition of the tester, we have $\epsilon = \frac{1}{4} \cdot (\epsilon_O + \epsilon_I + \epsilon_\kappa + \epsilon_c)$. Since these quantities are nonnegative, we get $\epsilon_O, \epsilon_I, \epsilon_\kappa, \epsilon_c \leq 4\epsilon$. Similarly, we have $\epsilon_O(\ell), \epsilon_I(\ell), \epsilon_\kappa(\ell), \epsilon_c(\ell) \leq 32\epsilon$ for every $\ell \in \{0, \ldots, 7\}$.

For $\ell \in \{0, \ldots, 7\}$, denote by $Q^{(\ell)}(x, y)$ the polynomial of degree at most $n_0/8$ in x and $n_1/8$ in y that is closest to $g^{(\ell)}$ (on the domain $\langle \omega^{n_1} \rangle \times \langle \omega^{n_0} \rangle$), where ties are broken arbitrarily. Similarly let $Q^{(\ell)}_{\kappa}$ be the closest polynomial to $g^{(\ell)}_{\kappa}$. Let $P(z) = \sum_{\ell=0}^{7} z^{\ell n_0/8} \cdot Q^{(\ell)}(z, z^{n_0})$ and let $P_\kappa(z) = \sum_{\ell=0}^{7} z^{\ell n_0/8} \cdot Q^{(\ell)}_{\kappa}(\kappa z, z^{n_0})$. We show below that (p, p_κ) is close to the evaluation of P on $\langle \omega \rangle \cup \kappa \langle \omega \rangle$. (Among other facts, we also show that $P_\kappa(z) \equiv P(\kappa \cdot z)$.)

Step 1. *The functions $Q^{(\ell)}$ (and $Q^{(\ell)}_{\kappa}$).* By the inductive hypothesis applied to **Inner**(ℓ, β), we have that $(g^{(\ell)}|^{\leftrightarrow}_{\beta^{n_0}}, f^{(\ell)}|^{\leftrightarrow}_{\beta})$ is $(c^{\log\log n_0} \cdot \epsilon_I(\ell, \beta))$-close to the SRS encoding of some degree $n_0/8$ polynomial. Thus $g^{(\ell)}|^{\leftrightarrow}_{\beta^{n_0}}$ is at most $(2 \cdot c^{\log\log n_0} \cdot \epsilon_I(\ell, \beta))$-close to the RS encoding of some degree-$n_0/8$ polynomial. Averaging over β, we get that $g^{(\ell)}$ is $(2 \cdot c^{\log\log n_0} \cdot \epsilon_I(\ell))$-close to some bivariate polynomial of degree $n_0/8$ in x and arbitrary degree in y. A similar argument based on the **Inner**$_c$ tests yields that $g^{(\ell)}$ is $(2 \cdot c^{\log\log n_1} \cdot \epsilon_c(\ell))$-close to some bivariate polynomial of degree $n_1/8$ in y and arbitrary degree in x. Now applying Lemma 6.13, we get that $g^{(\ell)}$ is close to some polynomial of degree $n_0/8$ in x and $n_1/8$ in y. More specifically, we

have

$$\begin{aligned}\delta^{(n_0/8,n_1/8)}(g^{(\ell)}) &\le c_0 \cdot \left(\delta^{(n_0/8,*)}(g^{(\ell)}) + \delta^{(*,n_1/8)}(g^{(\ell)})\right)\\ &\le c_0 \cdot \left(2 \cdot c^{\log\log n_0} \cdot \epsilon_I(\ell) + 2 \cdot c^{\log\log n_1} \cdot \epsilon_c(\ell)\right)\\ &\le 64 \cdot c_0 \left(c^{\log\log n_0} + c^{\log\log n_1}\right) \cdot \epsilon\\ &\le 128 \cdot c_0 \cdot c^{\log\log n_0} \cdot \epsilon.\end{aligned}$$

Letting $c_1 \stackrel{\text{def}}{=} 128 \cdot c_0$, we have that $\delta(g^{(\ell)}, Q^{(\ell)}) \le c_1 \cdot c^{\log\log n_0} \cdot \epsilon$. A similar argument shows that $\delta(g_\kappa^{(\ell)}, Q_\kappa^{(\ell)}) \le c_1 \cdot c^{\log\log n_0} \cdot \epsilon$.

Step 2. *The functions* $f^{(\ell)}$ *and* $f_\kappa^{(\ell)}$. Next we move to the functions $f^{(\ell)}$ (for any $\ell \in \{0,\ldots,7\}$) and show that for most $\tilde{\alpha}, \beta$ $f^{(\ell)}(\tilde{\alpha},\beta) = Q^{(\ell)}(\tilde{\alpha}\cdot\beta, \beta^{n_0})$ (and similarly for most $\tilde{\alpha}, \beta$, $f_\kappa^{(\ell)}(\tilde{\alpha},\beta) = Q_\kappa^{(\ell)}(\kappa\cdot\tilde{\alpha}\cdot\beta, \beta^{n_0})$.

We first describe the argument informally. Consider a β such that $g^{(\ell)}|^{\leftrightarrow}_{\beta^{n_0}}$ and $f^{(\ell)}|^{\leftrightarrow}_{\beta}$ pass the **Inner** test with high probability *and* the SRS-codeword correspond to the encoding of $Q(\cdot,\beta^{n_0})$. For such β, we have $f^{(\ell)}|^{\leftrightarrow}_{\beta}(\tilde{\alpha},\beta) = Q(\tilde{\alpha}\cdot\beta,\beta^{n_0})$ for most $\tilde{\alpha}$. It remains to make this argument quantitative, and we do so below.

Define a β to be *good* if the fractional distance between $(g^{(\ell)}|^{\leftrightarrow}_{\beta^{n_0}}, f^{(\ell)}|^{\leftrightarrow}_{\beta})$ and the $\mathrm{SRS}(\mathbb{F}, n_0/8, \omega^{n_1}, \beta)$ encoding of $Q^{(\ell)}(\cdot,\beta^{n_0})$ is at most $1/8$. Let $\delta(\beta)$ denote the relative distance of $f^{(\ell)}|^{\leftrightarrow}_{\beta}$ to the projection of the SRS-codeword nearest to $(g^{(\ell)}|^{\leftrightarrow}_{\beta^{n_0}}, f^{(\ell)}|^{\leftrightarrow}_{\beta})$ onto the second half of the coordinates. Note that

$$\begin{aligned}&\Pr_{\tilde{\alpha},\beta}[f^{(\ell)}(\tilde{\alpha},\beta) \ne Q^{(\ell)}(\tilde{\alpha}\cdot\beta,\beta^{n_0})]\\ &\le \mathbb{E}_\beta[\delta(\beta)|\beta \text{ is good}] \cdot \Pr_\beta[\beta \text{ is good}] + \Pr_\beta[\beta \text{ is not good}]\\ &\le \mathbb{E}_\beta[\delta(\beta)] + \Pr_\beta[\beta \text{ is not good}].\end{aligned}$$

Note that the first term above is easily estimated as in Step 1. We get $\mathbb{E}_\beta[\delta(\beta)] \le (2 \cdot c^{\log\log n_0} \cdot \epsilon_I(\ell)) \le 64 \cdot c^{\log\log n_0} \cdot \epsilon$.

Next we describe two sets that cover the case where β is not good. Let S_1 be the set of all β such that the distance of $(g^{(\ell)}|^{\leftrightarrow}_{\beta^{n_0}}, f^{(\ell)}|^{\leftrightarrow}_{\beta})$ from every SRS-codeword is more than $\frac{1}{8}$. For every $\beta \in S_1$ note that the $\epsilon_I(\ell,\beta) \ge \frac{1}{8c^{\log\log n_0}}$. Thus, the probability that $\beta \in S_1$ is at most $8 \cdot c^{\log\log n_0} \cdot \epsilon_I(\ell) \le 256 \cdot c^{\log\log n_0} \cdot \epsilon$. Next, let S_2 be the set of β for which $(g^{(\ell)}|^{\leftrightarrow}_{\beta^{n_0}}, f^{(\ell)}|^{\leftrightarrow}_{\beta})$ is $\frac{1}{8}$-close to an SRS-codeword, but the SRS-codeword is not the encoding of $Q^{(\ell)}(\cdot,\beta^{n_0})$. For every $\beta \in S_2$, we have that $Q^{(\ell)}(\tilde{\alpha},\beta^{n_0})$ and $g^{(\ell)}(\tilde{\alpha},\beta^{n_0})$ disagree for at least $\frac{5}{8}$ fraction of the $\tilde{\alpha}$'s (since $Q^{(\ell)}(\cdot,\beta^{n_0})$ and the other SRS-codeword can agree on at most $n_0/8$ values of the $\tilde{\alpha}$'s). Since the distance between $g^{(\ell)}$ and $Q^{(\ell)}$ is at most $c_1 \cdot c^{\log\log n_0} \cdot \epsilon$, we get that the probability that $\beta \in S_2$ is at most $\frac{8}{5} \cdot c_1 \cdot c^{\log\log n_0} \cdot \epsilon \le 2c_1 \cdot c^{\log\log n_0} \cdot \epsilon$. Finally, we note that if β is not good, then $\beta \in S_1 \cup S_2$. Thus we get

$$\Pr_\beta[\beta \text{ is not good}] \le (256 + 2c_1) \cdot c^{\log\log n_0} \cdot \epsilon.$$

Putting the above together, and recalling $c_2 = (320+2c_1)$, we get $\Pr_{\tilde{\alpha},\beta}[f^{(\ell)}(\tilde{\alpha},\beta) \ne Q^{(\ell)}(\tilde{\alpha}\cdot\beta,\beta^{n_0})] \le c_2 \cdot c^{\log\log n_0} \cdot \epsilon$. Similarly we also get $\Pr_{\tilde{\alpha},\beta}[f_\kappa^{(\ell)}(\tilde{\alpha},\beta) \ne Q_\kappa^{(\ell)}(\kappa\cdot\tilde{\alpha}\cdot\beta,\beta^{n_0})] \le c_2 \cdot c^{\log\log n_0} \cdot \epsilon$.

Step 3. *The functions p and p_κ.* Next we move to the functions p and show that $p(z)$ usually equals $P(z) = \sum_{\ell=0}^{7} z^{\ell \cdot n_0/8} Q^{(\ell)}(z, z^{n_0})$ for $z \in \langle\omega\rangle$. Note that $\langle\omega\rangle$ is in one-to-one correspondence with $\{\tilde{\alpha} \cdot \beta\}$, where $\tilde{\alpha} \in \langle\omega^{n_1}\rangle$ and $\beta \in \langle\omega\rangle_{n_1}$, and so we are interested in estimating the probability that

$$p(\tilde{\alpha} \cdot \beta) \neq \sum_{\ell=0}^{7} (\tilde{\alpha}\beta)^{\ell \cdot n_0/8} Q^{(\ell)}(\tilde{\alpha}\beta, \beta^{n_0}).$$

We consider the following events: For $\ell \in \{0, \ldots, 7\}$, let E_ℓ be the event that $f^{(\ell)}(\tilde{\alpha}, \beta) \neq Q^{(\ell)}(\tilde{\alpha}\beta, \beta^{n_0})$. Further, let E' be the event that $p(\tilde{\alpha}\cdot\beta) \neq \sum_{\ell=0}^{7} (\tilde{\alpha}\beta)^{\ell \cdot n_0/8} \cdot f^{(\ell)}(\tilde{\alpha}\beta, \beta^{n_0})$. For any ℓ, we have that E_ℓ happens with probability at most $c_2 \cdot c^{\log\log n_0} \cdot \epsilon$. Further, E' happens with probability at most $\epsilon_O \leq 4\epsilon \leq 4 \cdot c^{\log\log n_0} \cdot \epsilon$, using $c \geq 1$. Furthermore, if none of the events E', $\{E_\ell\}_\ell$ occurs, then we do have $p(\tilde{\alpha} \cdot \beta) = \sum_{\ell=0}^{7} (\tilde{\alpha}\beta)^{\ell \cdot n_0/8} Q^{(\ell)}(\tilde{\alpha}\beta, \beta^{n_0})$. Thus, recalling $c_3 = 8c_2 + 4$, we get that $\delta(p, P) \leq c_3 \cdot c^{\log\log n_0} \cdot \epsilon$. Similarly, we get $\delta(p_\kappa, P_\kappa) \leq c_3 \cdot c^{\log\log n_0} \cdot \epsilon$. Combining, we get that $\delta((p, p_\kappa), (P, P_\kappa)) \leq c_3 \cdot c^{\log\log n_0} \cdot \epsilon$. By the definition of $c = c_3^2$ and the condition $c^{\log\log n_0} \leq c^{\log\log n - \frac{1}{2}}$, we get that the final proximity above is at most $c^{\log\log n} \cdot \epsilon$, as desired.

All that remains to be shown is that P and P_κ are consistent, i.e., that $P_\kappa(z) = P(\kappa \cdot z)$.

Step 4. *Consistency of the κ shifts.* We prove this part by showing that for every ℓ, Q and Q_κ are consistent, i.e., $Q_\kappa^{(\ell)}(x, y) = Q^{(\ell)}(x, \kappa^{n_0} y)$. This suffices, since we will then have

$$P_\kappa(z) = \sum_\ell z^{\ell n_0/8} Q_\kappa^{(\ell)}(\kappa z, z^{n_0}) = \sum_\ell z^{\ell n_0/8} Q^{(\ell)}(\kappa z, \kappa^{n_0} z^{n_0}) = P(\kappa z).$$

Fix $\ell \in \{0, \ldots, 7\}$. Define $\tilde{\alpha} \in \langle\omega^{n_1}\rangle$ to be *good* if $(g^{(\ell)}|_{\tilde{\alpha}}^{\uparrow}, g_\kappa^{(\ell)}|_{\tilde{\alpha}}^{\uparrow})$ is 1/8 close to some SRS-codeword *and* $g^{(\ell)}|_{\tilde{\alpha}}^{\uparrow}$ is 1/4 close to the evaluations of $Q^{(\ell)}(\tilde{\alpha}, \cdot)$, *and* $g_\kappa^{(\ell)}|_{\tilde{\alpha}}^{\uparrow}$ is 1/4 close to the evaluations of $Q_\kappa^{(\ell)}(\tilde{\alpha}, \cdot)$. It is straightforward to see that if $\tilde{\alpha}$ is good, then $Q_\kappa^{(\ell)}(\tilde{\alpha}, y) = Q^{(\ell)}(\tilde{\alpha}, \kappa^{n_0} y)$. Furthermore, if the fraction of good $\tilde{\alpha}$'s is more than 1/8, then we will have $Q_\kappa^{(\ell)}(x, y) = Q^{(\ell)}(x, \kappa^{n_0} y)$ as desired. So it suffices to bound the probability of $\tilde{\alpha}$ being not good (to be less than 7/8).

The three conditions above can be analyzed in a manner similar to the analysis of the probability of β not being *good* in Step 2. Specifically, we have the following: The probability that $(g^{(\ell)}|_{\tilde{\alpha}}^{\uparrow}, g_\kappa^{(\ell)}|_{\tilde{\alpha}}^{\uparrow})$ is not 1/8 close to some SRS-codeword is at most $8 \cdot c^{\log\log n_1} \cdot \epsilon_c(\ell) \leq 256 \cdot c^{\log\log n_0} \cdot \epsilon$. The probability that $g^{(\ell)}|_{\tilde{\alpha}}^{\uparrow}$ is 1/8 close to some SRS-codeword and not 1/4 close to the evaluations of $Q^{(\ell)}(\tilde{\alpha}, \cdot)$ is at most $2 \cdot c_1 \cdot c^{\log\log n_0} \cdot \epsilon$. Finally, the probability that $g_\kappa^{(\ell)}|_{\tilde{\alpha}}^{\uparrow}$ is 1/8 close to some SRS-codeword and not 1/4 close to the evaluations of $Q_\kappa^{(\ell)}(\tilde{\alpha}, \cdot)$ is at most $2 \cdot c_1 \cdot c^{\log\log n_0} \cdot \epsilon$. Combining the above we get that the probability that $\tilde{\alpha}$ is not good is at most $(256 + 4 \cdot c_1) \cdot c^{\log\log n_0} \cdot \epsilon$. In turn the final quantity is at most $(256 + 4 \cdot c_1) \cdot c^{\log\log n - \frac{1}{2}} \cdot \epsilon \leq \frac{1}{2} c^{\log\log n} \cdot \epsilon \leq \frac{1}{2} < \frac{7}{8}$ as desired. The first inequality follows from the fact that we have $c > (2 \cdot (256 + 4 \cdot c_1))^2$. This concludes the proof that Q and Q_κ and hence P and P_κ are consistent. Combined with Step 3, this concludes the soundness analysis. □

7.6. Proof of Theorem 3.4.

Proof of smooth SRS PCPP Theorem 7.3 (for special case of $d = n/8 - 1$). The verifier is formally defined in subsection 7.3. Its query complexity, randomness, and

proof length are given by Proposition 7.6. Its completeness is asserted by Proposition 7.8. Its soundness is analyzed in Lemma 7.9. □

Proof of Theorem 3.2. The statement for $d = n/8 - 1$ follows from Theorem 7.3 by setting $\kappa = 1$. The generalization to arbitrary degree d follows from Proposition 6.14. □

7.7. Proving Theorem 2.2 using smooth RS-codes. In this section we briefly outline the modifications needed to prove Theorem 2.2 using PCPPs for *smooth* RS-codes (Theorem 3.4). Our motivation is to present a proof of Theorem 2.2 in as general a setting as possible and in particular show that we do not require the underlying field to be of characteristic 2. Our exposition follows that of subsection 3.3.1.

Our first challenge is to show the existence and abundance of fields with a multiplicative subgroup of order that is a power of 2. A second problem is that we cannot embed de Bruijn graphs in an affine graph of constant degree (Proposition 5.11), because our fields are not of characteristic 2. To solve this problem we embed the de Bruijn graph in an affine graph of logarithmic degree. Thus, we end with a weaker version of Theorem 3.7 and we need to prove quasilinear PCPs for this weaker version, using Theorem 3.4. We now elaborate on each of these three issues.

Prime fields with 2*-smooth subgroups.* Theorem 3.4 holds only for Reed–Solomon codes $\mathrm{RS}(\mathbb{F}, \langle\omega\rangle, d)$, where $\mathrm{ord}(\omega)$ is a power of 2. The following (special case of a) theorem due to Linnik [33] shows that there is a polynomial time computable sequence $\{\mathbb{F}_n\}_{n\in\mathbb{N}}$ such that $n \le |\mathbb{F}_n| \le n^{O(1)}$ and $\mathbb{F}_n^*$ has an element ω the order of which is a power of 2.

THEOREM 7.10 (Linnik's theorem [33]). *There exists a constant* $1 < L < 6$ *such that for any sufficiently large* d*, there exists a prime of size* $\le d^L$ *such that* $d|(p-1)$.

Remark 7.11. The general statement of Linnik's theorem says that there exists a universal constant L such that for every pair of integers $0 < a < n$, there exists a prime $p < n^L$ such that $n|(p-a)$. The case stated above is derived from the general statement by setting $a = 1$.

Suppose we wish to find a field $\mathbb{F}_n$ of size $n^{O(1)}$ that has an element ω of order $\Theta(n)$ that is a power of 2. Let d be a power of 2 such that $n < d \le O(n)$. Let $\mathbb{F}_n$ be the prime field $\mathbb{Z}_p$ for p as in Linnik's theorem, Theorem 7.10. We have $|\mathbb{F}_n^*| = p-1 = k\cdot d$. Let σ be a generator of $\mathbb{F}^*$ and set $\omega = \sigma^k$. Then $\mathrm{ord}(\omega) = \Theta(n)$ is a power of 2. Notice that p and ω can be found in polynomial time (in n) by an exhaustive search. Finally, each element of $\mathbb{F}_n$ is represented by $O(\log n)$ bits.

Algebraic constraint satisfaction problems for PAIR-SMOOTH-RS. We now sketch a proof of a weaker version of Theorem 3.7. The weakness of this version refers to the fact that the number of affine functions is not constant but polylogarithmic. However, we will be able to prove this theorem without relying on fields of characteristic 2. Rather, we need our field only to be sufficiently large.

THEOREM 7.12 (ALGEBRAIC-CSP is NP-complete (weak version)). *There exists an integer* d *such that for any proper complexity function* $t : \mathbb{N}^+ \to \mathbb{N}^+$ *and* $L \in \mathrm{NTIME}(t(n))$*, the following hold.*

1. L *is reducible to* ALGEBRAIC-CSP *in time* $\mathrm{poly}\, t(n)$.
2. *Given any field* $\mathbb{F}$ *of size* $\Omega(t(n)\,\mathrm{polylog}\, t(n))$*, an instance of* L *of size* n *is reduced to an instance of* $\mathrm{ALGEBRAIC\text{-}CSP}_{\mathrm{polylog}\, n, d}$ *over* $\mathbb{F}$.

Proof. The reduction underlying Theorem 3.7 and described in subsection 5.2 relied on the existence of a homomorphism of the wrapped de Bruijn graph B_k (see Definition 5.5) into an affine graph (as per Definition 5.9) of constant degree over a

field of characteristic 2. This homomorphism, in turn, relies on the additive structure of the field (see the proof of Proposition 5.11).

As in the proof of Theorem 5.4, we will assume only that the underlying field is sufficiently large. We use the existence of an efficiently computable homomorphism of B_k into the hypercube of dimension $k + O(\log k)$ (for details see [31]). Next we notice the hypercube of dimension k' can be embedded into an affine graph over any finite field $\mathbb{F}, |\mathbb{F}| > 2^{k'}$. Indeed, fix $\omega \in \mathbb{F}^*$ with $\mathrm{ord}(\omega) \geq 2^{k'}$. Consider the affine graph G over vertex set $\langle\omega\rangle$ and edge set generated by $\{\omega^{(-1)^b \cdot 2^\ell}\}_{\ell \in [k'], b \in \{0,1\}}$. To see that the hypercube can be embedded into G, let $\bar{i} \in \{0, \ldots, 2^{k'} - 1\}$ denote the integer with binary representation $i \in \{0,1\}^{k'}$. Associate with i the element $\omega^{\bar{i}} \in \langle\omega\rangle$. We claim that the elements associated with i and $i + e_\ell$ (in the hypercube) are adjacent in G. Indeed, let b denote the ℓth bit of i and notice that i is associated with $\omega^{\bar{i}}$ whereas $i + e_\ell$ is associated with $\omega^{\bar{i}+(-1)^b 2^\ell} = \omega^{(-1)^b \cdot 2^\ell} \cdot \omega^{\bar{i}}$, so the corresponding vertices are adjacent in G.

From here on we follow the proof of Theorem 3.7, using the above defined affine graph G of degree $\mathrm{polylog}\, n$ instead of the constant degree graph used there. All other details are identical. Thus, our reduction results in an instance of ALGEBRAIC-CSP$_{\mathrm{polylog}\, n, O(1)}$. ∎

Quasilinear PCPs via PCPPs for smooth RS-codes. We now provide efficient PCPs for the instances of ALGEBRAIC-CSP given by Theorem 7.12 and thus provide an alternative proof of Theorem 2.2.

Proof of quasilinear PCP Theorem 2.2. Let ψ be an instance of $L \in \mathrm{NTIME}(t(n))$ of size n. Using Theorem 7.12 we reduce ψ to an instance $\phi = \{\mathbb{F}, \{\textsc{Aff}_1, \ldots, \textsc{Aff}_k\}, H, C\}$ of ALGEBRAIC-CSP$_{k,d}$ of size $n' = n \cdot \mathrm{polylog}\, n$, where $k = \mathrm{polylog}\, n, d = O(1)$ and $\mathbb{F}$ is the smallest prime field containing an element ω with $100kdn' < \mathrm{ord}(\omega) \leq 200kdn'$, where $\mathrm{ord}(\omega)$ is a power of 2. Linnik's theorem, Theorem 7.10, implies that $\mathbb{F}$ and ω exist and can be found in polynomial time (by an exhaustive search).

From here on our proof is essentially identical to the proof presented in subsection 3.3.1 and we use the notation given there. Notice that since $k = \mathrm{polylog}\, n$, the first subtest invokes an RS-verifier with proximity parameter $1/\mathrm{polylog}\, n$. However, Proposition 2.9 implies that the query complexity increases only by a factor of $\mathrm{polylog}\, n$. All other details are exactly as in subsection 3.3.1, and this completes the alternative proof of Theorem 2.2. ∎

Acknowledgments. We thank Oded Goldreich, Prahladh Harsha, Salil Vadhan, and Chris Umans for helpful discussions. We thank Don Coppersmith for pointing us to Linnik's theorem, which appears here as Theorem 7.10. We thank Venkatesan Guruswami, Subhash Khot, Jaikumar Radhakrishnan, and an anonymous referee for pointing out errors in previous versions of the paper. Finally, we thank the editor and anonymous referees for remarks that helped improve the clarity of the presentation.

REFERENCES

[1] N. Alon, *Combinatorial Nullstellensatz*, Combin. Probab. Comput., 8 (1999), pp. 7–29.
[2] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy, *Proof verification and the hardness of approximation problems*, J. ACM, 45 (1998), pp. 501–555.
[3] S. Arora and S. Safra, *Probabilistic checking of proofs: A new characterization of NP*, J. ACM, 45 (1998), pp. 70–122.
[4] L. Babai, L. Fortnow, and C. Lund, *Nondeterministic exponential time has two-prover interactive protocols*, Comput. Complexity, 1 (1991), pp. 3–40.

[5] L. Babai, L. Fortnow, L. Levin, and M. Szegedy, *Checking computations in polylogarithmic time*, in Proceedings of the 23rd ACM Symposium on Theory of Computing, ACM, New York, 1991, pp. 21–31.
[6] B. Barak, *How to go beyond the black-box simulation barrier*, in Proceedings of the 42nd ACM Symposium on Theory of Computing, ACM, New York, 2001, pp. 106–115.
[7] M. Bellare, O. Goldreich, and M. Sudan, *Free bits, PCPs, and nonapproximability—towards tight results*, SIAM J. Comput., 27 (1998), pp. 804–915.
[8] M. Bellare, S. Goldwasser, C. Lund, and A. Russell, *Efficient probabilistically checkable proofs and applications to approximation*, in Proceedings of the 25th ACM Symposium on Theory of Computing, ACM, New York, 1993, pp. 294–304.
[9] E. Ben-Sasson, O. Goldreich, P. Harsha, M. Sudan, and S. Vadhan, *Robust PCPs of proximity, shorter PCPs, and applications to coding*, in Proceedings of the 36th ACM Symposium on Theory of Computing, ACM, New York, 2004, pp. 13–15.
[10] E. Ben-Sasson, O. Goldreich, P. Harsha, M. Sudan, and S. Vadhan, *Short PCPs verifiable in polylogarithmic time*, in Proceedings of the 20th IEEE Conference on Computational Complexity, San Jose, CA, 2005, pp. 120–134.
[11] E. Ben-Sasson, M. Sudan, S. Vadhan, and A. Wigderson, *Randomness-efficient low degree tests and short PCPs via epsilon-biased sets*, in Proceedings of the 35th ACM Symposium on Theory of Computing, ACM, New York, 2003, pp. 612–621.
[12] A. Bhattacharyya, *Implementing Probabilistically Checkable Proofs of Proximity*, Technical report MIT-CSAIL-TR-2005-051 (MIT-LCS-TR-998), MIT, Cambridge, MA, 2005. Available online from http://publications.csail.mit.edu/2005trs.shtml.
[13] R. Canetti, O. Goldreich, and S. Halevi, *The random oracle methodology, revisited*, J. ACM, 51 (2004), pp. 557–594.
[14] S. Cook, *The complexity of theorem-proving procedures*, in Proceedings of the 3rd IEEE Symposium on Foundations of Computer Science, IEEE Computer Society, Los Alamitos, CA, 1971, pp. 151–158.
[15] S. Cook, *Short propositional formulas represent nondeterministic computations*, Inform. Process. Lett., 26 (1988), pp. 269–270.
[16] D. Cox, J. Little, and D. O'Shea, *Ideals, Varieties, and Algorithms*, Springer-Verlag, Berlin, 1992.
[17] I. Dinur, E. Fischer, G. Kindler, R. Raz, and S. Safra, *PCP characterizations of NP: Towards a polynomially-small error-probability*, in Proceedings of the 31st ACM Symposium on Theory of Computing, ACM, New York, 1999, pp. 29–40.
[18] I. Dinur, *The PCP theorem by gap amplification*, in Proceedings of the 38th ACM Symposium on Theory of Computing, ACM, New York, 2006, pp. 241–250.
[19] I. Dinur and O. Reingold, *Assignment testers: Towards a combinatorial proof of the PCP theorem*, SIAM J. Comput., 36 (2006), pp. 975–1024.
[20] U. Feige, S. Goldwasser, L. Lovász, S. Safra, and M. Szegedy, *Interactive proofs and the hardness of approximating cliques*, J. ACM, 43 (1996), pp. 268–292.
[21] K. Friedl, Z. Hátsági, and A. Shen, *Low-degree tests*, in Proceedings of the 5th ACM-SIAM Symposium on Discrete Algorithms, ACM, New York, SIAM, Philadelphia, 1994, pp. 57–64.
[22] O. Goldreich, *A Sample of Samplers—a Computational Perspective on Sampling*, Tech. rep. TR97-020, Electronic Colloquium on Computational Complexity, 1997.
[23] O. Goldreich, *Short Locally Testable Codes and Proofs (Survey)*, Tech. rep. TR05-014, Electronic Colloquium on Computational Complexity, 2005.
[24] O. Goldreich and M. Sudan, *Locally testable codes and PCPs of almost linear length (second revision)*, initial version appeared in Proceedings of the 43rd ACM Symposium on Theory of Computing, ACM, New York, 2002, pp. 13–22.
[25] O. Goldreich and A. Wigderson, *Tiny families of functions with random properties: A quality–size trade–off for hashing*, Random Structures Algorithms, 11 (1997), pp. 315–343.
[26] V. Guruswami, D. Lewin, M. Sudan, and L. Trevisan, *A tight characterization of NP with 3-query PCPs*, in Proceedings of the 39th IEEE Symposium on Foundations of Computer Science, IEEE Computer Society, Los Alamitos, CA, 1998, pp. 18–27.
[27] P. Harsha and M. Sudan, *Small PCPs with low query complexity*, Comput. Complexity, 9 (2000), pp. 157–201.
[28] J. Håstad, *Some optimal inapproximability results*, J. ACM, 48 (2001), pp. 798–859.
[29] F. C. Hennie and R. E. Stearns, *Two-tape simulation of multitape turing machines*, J. ACM, 13 (1966), pp. 533–546.

[30] J. Kilian, *A note on efficient zero-knowledge proofs and arguments (extended abstract)*, in Proceedings of the 24th ACM Symposium on Theory of Computing, ACM, New York, 1992, pp. 723–732.
[31] F. T. Leighton, *Introduction to Parallel Algorithms and Architectures*, Morgan Kaufmann Publishers, San Mateo, CA, 1992.
[32] R. Lidl and H. Niederreiter, *Finite Fields*, 2nd ed., Cambridge University Press, Cambridge, UK, 1997.
[33] Y. Linnik, *On the least prime in an arithmetic progression.* I. *The basic theorem*, Mat. Sb. N. S., 15 (57), 1944, pp. 139–178 (in Russian).
[34] C. Lund, L. Fortnow, H. Karloff, and N. Nisan, *Algebraic methods for interactive proof systems*, J. ACM, 39 (1992), pp. 859–868.
[35] S. Micali, *Computationally sound proofs*, SIAM J. Comput., 30 (2000), pp. 1253–1298.
[36] C. Papadimitriou, *Computational Complexity,* Addison–Wesley, Reading, MA, Longman, Harlow, UK, 1994.
[37] A. Polishchuk and D. Spielman, *Nearly-linear size holographic proofs*, in Proceedings of the 26th ACM Symposium on Theory of Computing, ACM, New York, 1994, pp. 194–203.
[38] A. Samorodnitsky and L. Trevisan, *A PCP characterization of NP with optimal amortized query complexity*, in Proceedings of the 32nd ACM Symposium on Theory of Computing, ACM, New York, 2000, pp. 191–199.
[39] D. Spielman, *Computationally Efficient Error-Correcting Codes and Holographic Proofs*, Ph.D. thesis, MIT, Cambridge, MA, 1995.
[40] M. Szegedy, *Many-valued logics and holographic proofs*, in Proceedings of the 26th International Colloquium on Automata, Languages, and Programming, Prague, Czech Republic, 1999, pp. 676–686.