


Fast Reed-Solomon Interactive Oracle Proofs of Proximity

Eli Ben-Sasson

Technion, Haifa, Israel

eli@cs.technion.ac.il

 <https://orcid.org/0000-0002-0708-0483>

Iddo Bentov

Cornell University, Ithaca, NY, USA

iddo333@gmail.com

Yinon Horesh

Technion – Israel Institute of Technology, Haifa, Israel

ynon980@gmail.com

Michael Riabzev

Technion – Israel Institute of Technology, Haifa, Israel

riabzevmichael@gmail.com

Abstract

The family of Reed-Solomon (RS) codes plays a prominent role in the construction of quasilinear probabilistically checkable proofs (PCPs) and interactive oracle proofs (IOPs) with perfect zero knowledge and polylogarithmic verifiers. The large concrete computational complexity required to prove membership in RS codes is one of the biggest obstacles to deploying such PCP/IOP systems in practice.

To advance on this problem we present a new interactive oracle proof of proximity (IOPP) for RS codes; we call it the *Fast RS IOPP* (FRI) because (i) it resembles the ubiquitous Fast Fourier Transform (FFT) and (ii) the arithmetic complexity of its prover is strictly linear and that of the verifier is strictly logarithmic (in comparison, FFT arithmetic complexity is quasi-linear but not strictly linear). Prior RS IOPPs and PCPs of proximity (PCPPs) required super-linear proving time even for polynomially large query complexity.

For codes of block-length N , the arithmetic complexity of the (interactive) FRI prover is less than $6 \cdot N$, while the (interactive) FRI verifier has arithmetic complexity $\leq 21 \cdot \log N$, query complexity $2 \cdot \log N$ and constant soundness – words that are δ -far from the code are rejected with probability $\min\{\delta \cdot (1 - o(1)), \delta_0\}$ where δ_0 is a positive constant that depends mainly on the code rate. The particular combination of query complexity and soundness obtained by FRI is better than that of the quasilinear PCPP of [Ben-Sasson and Sudan, SICOMP 2008], even with the tighter soundness analysis of [Ben-Sasson et al., STOC 2013; ECCO 2016]; consequently, FRI is likely to facilitate better concretely efficient zero knowledge proof and argument systems.

Previous concretely efficient PCPPs and IOPPs suffered a constant *multiplicative* factor loss in soundness with each round of “proof composition” and thus used at most $O(\log \log N)$ rounds. We show that when δ is smaller than the unique decoding radius of the code, FRI suffers only a negligible *additive* loss in soundness. This observation allows us to increase the number of “proof composition” rounds to $\Theta(\log N)$ and thereby reduce prover and verifier running time for fixed soundness.

2012 ACM Subject Classification Theory of computation → Interactive proof systems

Keywords and phrases Interactive proofs, low degree testing, Reed Solomon codes, proximity testing



© Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev;
licensed under Creative Commons License CC-BY

45th International Colloquium on Automata, Languages, and Programming (ICALP 2018).

Editors: Ioannis Chatzigiannakis, Christos Kaklamani, Dániel Marx, and Donald Sannella;

Article No. 14; pp. 14:1–14:17



Leibniz International Proceedings in Informatics

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



Digital Object Identifier 10.4230/LIPIcs.ICALP.2018.14

Related Version Electronic Colloquium on Computational Complexity, report TR17-134 [10].

Funding Supported by the European Research Council under POC grant OMIP – DLV-693423, an Israel Science Foundation grant 1501/14 and the USA–Israel binational science fund, grant # 2014359.

Acknowledgements We thank Justin Drake, Peter Manohar and Nicholas Spooner for helping clarify the presentation and for pointing out and correcting errors in earlier manuscripts.

1 Introduction

The family of Reed-Solomon (RS) codes is a fundamental object of study in algebraic coding theory and theoretical computer science [56]. For an evaluation set S of N elements in a finite field \mathbb{F} and a rate parameter $\rho \in (0, 1]$, the code $\text{RS}[\mathbb{F}, S, \rho]$ is the space of functions $f : S \rightarrow \mathbb{F}$ that are evaluations of polynomials of degree $d < \rho N$ [56]. The *RS proximity problem* assumes a verifier has oracle access to $f : S \rightarrow \mathbb{F}$, and asks that verifier to distinguish, with “large” confidence and “small” query complexity, between the case that f is a codeword of $\text{RS}[\mathbb{F}, S, \rho]$ and the case that f is δ -far in relative Hamming distance from all codewords. This problem has been addressed in several different computational models (surveyed next and summarized in Table 1), and is also the focus of this paper.

RS proximity testing: When no additional data is provided to the verifier, the RS proximity problem is commonly called a *testing* problem, and has been first defined and addressed by Rubinfeld and Sudan in [58] (cf. [32]). In this case, one can see that $d + 1$ queries are necessary and sufficient to solve the problem: codewords are accepted by their tester with probability 1 whereas functions that are δ -far from the code are rejected with probability $\geq \delta$. Since no additional information is provided to the verifier in this model, we may say that a prover attempting to convince the verifier that $f \in \text{RS}[\mathbb{F}, S, \rho]$ spends zero computational effort, zero rounds of interaction and produces a proof of length zero.

RS proximity verification – PCPP model: Probabilistically checkable proofs of proximity (PCPP) [21, 30] relax the testing problem to a setting in which the verifier is given oracle access also to an auxiliary proof, called a PCPP and denoted π . This PCPP is produced by the prover, which is given $f \in \text{RS}[\mathbb{F}, S, \rho]$ as input. The time required to produce π is the *prover complexity* and $|\pi|$ is called the *proof length*¹; similarly, *verifier complexity* is the total time required to generate queries and check query-answers. The techniques used to prove the celebrated PCP Theorem [2, 3] also show that the proximity problem can be solved with constant query complexity and proof length and prover complexity $N^{O(1)}$, or with proof length $N^{1+\epsilon}$ and query complexity $(\log N)^{O(1/\epsilon)}$ [5]. The current state of the art in the PCPP model gives proofs of length $\tilde{O}(N) \triangleq N \cdot \log^{O(1)} N$ with constant query complexity [23, 28] and prover complexity $\tilde{O}(N)$ [16]; verifier complexity is $\text{poly log } N$ [20, 50].

RS proximity verification – IOPP model: Interactive oracle proofs of proximity (IOPP), formally introduced in [13] and, independently, in [57] (under the name “probabilistically checkable interactive proofs of proximity”), generalize IPs, PCPs and interactive PCPs (IPCP) [42]. As in an IP and IPCP, several rounds of interaction are used in which the prover sends messages $\pi_1, \pi_2, \dots, \pi_r$ in response to successive verifier messages. As in a PCP and

¹ Typically π is a sequence of elements in \mathbb{F} . Therefore, proof length is measured over the alphabet \mathbb{F} .

■ **Table 1** Comparison of RS proximity protocols. For concreteness, all results are stated for binary additive RS codes with rate $\rho = 1/8$ evaluated over a sufficiently large set S , $|S| = N$ satisfying $N/|\mathbb{F}| < 0.001$ with proximity parameter $\delta < \delta_0$ (cf. Theorem 2) and soundness at least 0.99δ ; i.e., the rejection probability of δ -far words is at least 0.99δ for $\delta < \delta_0$ (in particular, smaller δ leads to smaller soundness). Exponents for the 4th row taken from [16]; the various exponents c in the 5th and 6th row have not been estimated in prior works but are greater than the respective exponents in the 4th row.

	prover comp.	proof length	verifier comp.	query comp.	round comp.
1. Testing [58]	0	0	$\tilde{O}(\rho N)$	ρN	0
2. PCP [2, 3]	$N^{O(1)}$	$N^{O(1)}$	$N^{O(1)}$	$O\left(\frac{1}{\delta}\right)$	1
3. PCP [6, 5]	$N^{1+\epsilon}$	$N^{1+\epsilon}$	$\frac{1}{\delta} \log^{O(1/\epsilon)} N$	$\frac{1}{\delta} \log^{O(1/\epsilon)} N$	1
4. PCPP [23, 21, 16]	$\geq N \log^{1.5} N$	$\geq N \log^{1.5} N$	$\geq \frac{1}{\delta} \log^{5.8} N$	$\frac{1}{\delta} \log^{5.8} N$	1
5. PCPP [28, 50]	$N \log^c N$	$N \log^c N$	$\frac{1}{\delta} \log^c N$	$O\left(\frac{1}{\delta}\right)$	1
6. IOPP [12, 9]	$N \log^c N$	$> 4 \cdot N$	$\frac{1}{\delta} \log^c N$	$O\left(\frac{1}{\delta}\right)$	$\log \log N$
7. This work	$< 6 \cdot N$	$< \frac{N}{3}$	$\leq 21 \cdot \log N$	$2 \log N$	$\frac{\log N}{2}$

IPCP, the verifier is not required to read prover messages in entirety but rather may query them at random locations (in an IPCP, verifier must read the full messages π_2, \dots but may query π_1 randomly); the query complexity is the total number of entries read from f and $\pi_1, \pi_2, \dots, \pi_r$. The prover is provided with $f \in \text{RS}[\mathbb{F}, S, \rho]$ as input and *prover complexity* is the total time required to produce all (prover) messages², while *proof length* is generalized from the PCPP setting to the IOPP setting and defined as $|\pi_1| + \dots + |\pi_r|$. IOPPs can be used to “replace” PCPP proof composition with more rounds of interaction, and thereby reduce proof length and prover complexity without compromising soundness (see Section 1.3). In particular, the IOPP version of the aforementioned PCPP constructions reduces proof length to $O(N)$ with no change to soundness and/or query complexity [8, 13]. In spite of the shorter proof length, prover complexity in prior works was $\Theta(N \text{poly log } N)$ due to a limitation on the number of proof-composition rounds, explained in Section 2.1.

1.1 Main results

We present a new IOPP for RS codes, called the Fast RS IOPP (FRI) because of its resemblance to the Fast Fourier Transform (FFT) [26]; its analysis relies on the quasi-linear RS-PCPP [23] (see Section 2.1). FRI is the first RS-IOPP to have (i) *strictly linear* arithmetic complexity for the prover with (ii) *strictly logarithmic* arithmetic complexity for the verifier and (iii) *constant* soundness. We start by recalling IOPP systems as described in [12, Section 3.2], after informally summarizing the main complexity parameters of IOPs (introduced and discussed thoroughly in [19]).

1.1.1 IOP

An *Interactive Oracle Proof (IOP)* system S is defined by a pair of interactive randomized algorithms $S = (P, V)$, where P denotes the prover and V the verifier. On input x of length N , the number of rounds of interaction is denoted by $r(N)$ and called the *round complexity* of the system. During a single round the prover sends a message to which the verifier is

² Notice that prover complexity does not include the time needed to produce f .

given oracle access, and the verifier responds with a message to the prover. The *proof length*, denoted $\ell(N)$, is the sum of lengths of all messages sent by the prover. The *query complexity* of the protocol, denoted $q(N)$, is the number of entries read by V from the various prover messages; since the verifier has oracle access to those messages, typically $q(N) \ll \ell(N)$. (For the FRI system $q(N) = O(\log \ell(N))$). We denote by $\langle P \leftrightarrow V \rangle(x)$ the output of V after interacting with P on input x ; this output is either `accept` or `reject`. An IOP is said to be *transparent* (or have *public randomness*) if all messages sent from the verifier are public random coins and all queries are determined by public coins, which are broadcast to the prover (such protocols are also known as Arthur-Merlin protocols [4]).

1.1.2 IOPP

As its name suggests, an *IOP of proximity (IOPP)* is the natural generalization of a PCP of Proximity (PCPP) to the IOP model. An IOPP for a family of codes³ \mathcal{C} is a pair (P, V) of randomized algorithms, called *prover* and *verifier*, respectively. Both parties receive as common input a specification of a code $C \in \mathcal{C}$ which we view as a set of functions $C = \{f : S \rightarrow \Sigma\}$ for a finite set S and alphabet Σ . We also assume that the verifier has oracle access to a function $f^{(0)} : S \rightarrow \Sigma$ and that the prover receives the same function as explicit input. The number of rounds of interaction, or *round complexity*, is denoted by r , *query complexity* is denoted by q .

► **Definition 1** (Interactive Oracle Proof of Proximity (IOPP) [12]). An r -round Interactive Oracle Proof of Proximity (IOPP) $S = (P, V)$ is a $(r + 1)$ -round IOP. We say S is an $(r$ -round) IOPP for the error correcting code $C = \{f : S \rightarrow \Sigma\}$ with soundness $s^- : (0, 1] \rightarrow [0, 1]$ with respect to distance measure Δ , if the following conditions hold:

- **First message format:** the first prover message, denoted $f^{(0)}$, is a purported codeword of C , i.e., $f^{(0)} : S \rightarrow \Sigma$
- **Completeness:** $\Pr [\langle P \leftrightarrow V \rangle = \text{accept} \mid \Delta(f^{(0)}, C) = 0] = 1$; this means that for every $f^{(0)} \in C$ the protocol terminates in acceptance.
- **Soundness:** For any P^* , $\Pr [\langle P^* \leftrightarrow V \rangle = \text{reject} \mid \Delta(f^{(0)}, C) = \delta] \geq s^-(\delta)$

The sum of lengths of all prover messages, except for $f^{(0)}$, is the IOPP *proof length*; the time required to generate all messages except for $f^{(0)}$ is the *prover complexity*. The IOPP *query complexity* is the total number of queries to all messages, including $f^{(0)}$ and the *decision complexity* is the computational complexity (see following remark) required by the verifier to reach its verdict, once the queries and query answers are provided as inputs.

► **Remark** (Computational model for decision complexity). The computational model in which decision complexity is computed is left undefined. A natural default is to use *boolean circuit complexity*. However, later we study families of linear codes in which each IOPP query is answered by a field element. The natural computational model in this case is that of *arithmetic complexity*, i.e., for a linear code C over a finite field \mathbb{F} , it is the number of arithmetic operations over \mathbb{F} made by the verifier to reach its decision.

1.1.3 Main Theorem

The finite field of size q is denoted here by \mathbb{F}_q ; when q is clear from context we omit it. A field is called *binary* if $q = 2^m$, $m \in \mathbb{N}$. A subset S of a binary field is an *additive coset* if it is a coset of a subgroup of the additive group \mathbb{F}^+ , i.e., if S is an additive shift of an

³ The definition of an IOPP can be generalized to arbitrary languages; we study an IOPP for a specific family of codes so prefer to limit the scope of our definition accordingly.

\mathbb{F}_2 -linear space contained in \mathbb{F}_q . The *binary additive RS code family* is the collection of codes $\text{RS}[\mathbb{F}, S, \rho]$ where \mathbb{F} is a binary field and S an additive coset. This family of codes is one for which quasilinear PCPP were defined in [23], and our main theorem is stated for it (see Table 1).

► **Theorem 2 (Main – FRI properties).** *The binary additive RS code family of rate $\rho = 2^{-\mathcal{R}}$, $\mathcal{R} \geq 2$, $\mathcal{R} \in \mathbb{N}$ has an IOPP (FRI) with the following properties, where $N = |S|$ denotes block-length (which equals the prover-side input length for a fixed $\text{RS}[\mathbb{F}, S, \rho]$ code) and $\rho N > 16$:*

- **Prover Complexity** is less than $6N$ arithmetic operations in \mathbb{F} ; proof length is less than $N/3$ field elements and round complexity is at most $\frac{\log N}{2}$;
- **Verifier complexity** Query complexity is $2 \log N$; the verifier decision is computed using at most $21 \log N$ arithmetic operations over \mathbb{F} ;
- **Soundness:** There exists $\delta_0 \geq \frac{1}{4}(1 - 3\rho) - \frac{1}{\sqrt{N}}$ such that every f that is δ -far in relative Hamming distance from the code, is rejected with probability at least $\min\{\delta, \delta_0\} - \frac{3N}{|\mathbb{F}|}$;
- **Parallelization:** Each prover-message can be computed in $O(1)$ time on a Parallel Random Access Machine (PRAM) with common read and exclusive write (CREW), assuming a single \mathbb{F} arithmetic operation takes unit time.

► **Remark (Space complexity).** Given the i th prover message as input, each symbol of the $(i + 1)$ th prover message can be computed with space complexity $O(\log |\mathbb{F}|)$, i.e., the space required to hold a constant number of field elements.

This follows immediately from the fact that each prover message is computed in $O(1)$ arithmetic operations on a parallel machine.

Generalizing Theorem 2 to arbitrary rate $\rho \in (0, 1]$ can be done as described in [23, Proposition 6.13] (cf. remark 6.2 there); this leads to slightly larger constants in the prover and verifier complexity. For practical applications like ZK-IOPs [14, 12], rates of the form stated in the theorem above suffice.

► **Remark (FRI for “smooth codes”).** We call a multiplicative group $H \subset \mathbb{F}_q$ *smooth* if its order ($|H|$) is 2^k for $k \in \mathbb{N}$. The family of *smooth RS codes* of rate ρ is the set of $\text{RS}[\mathbb{F}_q, H, \rho]$ codes in which H is a smooth multiplicative group. Theorem 2 holds *also* with respect to the family of smooth RS codes, with somewhat smaller constants than 6 and 21 for the prover and verifier arithmetic complexity (see full version of this paper [10]); see Section 2.1 for a high-level overview of the smooth case and full version of this paper [10] for more details on modifying the protocol to this case. The protocol can be further generalized to groups of order c^k for constant c (perhaps with different arithmetic complexity constants), details omitted.

The soundness bound of Theorem 2 is nearly tight for $\delta \leq \delta_0$. We conjecture that a similar bound holds for all δ . See full version of this paper [10] for a more detailed version of the conjecture that implies it, and a discussion of Equation (1)

► **Conjecture 3.** *The soundness limit δ_0 of Theorem 2 approaches $1 - \rho$. Specifically, for all $\delta \leq 1 - \rho$, the rejection probability of any f that is δ -far from the RS-code of rate ρ and block-length N over \mathbb{F} , is at least*

$$\delta - \frac{2 \log N}{\sqrt{|\mathbb{F}|}}. \tag{1}$$

1.2 Applications to transparent zero knowledge implementations

Prover-efficient IOPPs of the kind presented here are crucially needed to facilitate *practical* ZK argument systems that are (i) *transparent* (public randomness), (ii) *universal* – apply to any computation – and (iii) *(doubly) scalable* – have quasi-linear proving time *and* polylogarithmic verification time, simultaneously. In a follow-up paper we use the FRI protocol (among other things) to realize in code the first ZK system (called a ZK-STARK there) that achieves the three properties listed above [11]. The concrete efficiency of that protocol, which relies to a large degree on the efficiency of the FRI protocol presented here, allows one to construct ZK arguments of knowledge (ZK-ARKs) for computational statements, where verifying the computational integrity of the statement using the ZK-STARK verifier is strictly faster than naïve verification via re-execution, and the communication complexity is strictly smaller than the size of the non-deterministic witness supporting the claim. The ZK-STARK prover is $\approx 50\times$ faster than the previous state-of-the-art transparent system, code-named SCI [7] (that system does not have ZK), and $\approx 10\times$ faster than state-of-the-art ZK-SNARKs [17] (which are not transparent); see [11, Figure 5] for details.

In the remainder of this section we explain, briefly, how our system could be incorporated in a larger practical ZK system (like the ZK-STARK mentioned earlier). In Section 1.3 we discuss the range of block-lengths that are relevant in applications, and the resulting communication complexity arising from their use.

The seminal works of Babai et al. [6, 5] showed that verifying the correctness of an arbitrary nondeterministic computation running for $T(N)$ steps can be achieved by a verifier running in time $\text{poly}(N, \log T(N))$ in the PCP model. Kilian’s construction transforms such PCPs into a 4-round ZK argument in which the total communication complexity and verifier running time are bounded by $\text{poly} \log T(N)$ [43] (cf. [44, 40, 41]), assuming a family of collision-resistant hash functions. Micali further compressed this system into a non-interactive computationally sound (CS) proof system, assuming both prover and verifier share access to the same random function [48]; this is typically realized in practice using a hash function like SHA2 and relying on the Fiat-Shamir heuristic [31]. No implementation of these marvelous techniques has appeared during the quarter century that has passed since they were first published. This is explained, in part, by concerns about the efficiency of these constructions for concrete programs and run-times. Among the numerous components involved in building these systems, a significant computational bottleneck is that of computing solutions to the Reed-Muller (RM) proximity problem, also known as “low degree testing” of multivariate polynomials.

Quasilinear PCPs based on RS codes have prover complexity that is asymptotically more efficient than RM codes which lead to PCPs with super-quasi-linear length, and a number of works have explored the concrete efficiency of these RS-based protocols [16, 8]. Recently, Ben-Sasson et al. suggested an IOP with perfect zero knowledge (PZK) for NP [14], later extended to NEXP [12], in which prover complexity is quasilinear and verifier complexity is $\text{poly}(N, \log T(N))$; this PZK-IOP can be compiled, using Kilian’s technique, into an interactive ZK argument with succinct⁴ communication complexity, or, using Micali’s technique (cf. [61]), into a *non-interactive random oracle proof (NIROP)* as defined in [19]. In light of this, the practicality of Kilian- and Micali-type ZK argument systems with polylogarithmic verifiers should be reconsidered.

To add motivation, a number of interesting practical succinct argument systems (with and without zero-knowledge) have been reported recently (see [62] for an excellent updated survey

⁴ Here, as in past works, “succinct” is synonymous to “polylogarithmic”.

of the subject and [7] for a comparison of PCP/IOP-based solutions to other approaches). A particular system based on the *quadratic span programs* (QSP) of Gennaro et al. [33] (cf. [17]) has been used by Ben-Sasson et al. to build a decentralized anonymous payment (DAP) system called “Zerocash” [15], later deployed as a practical commercial crypto-currency called “Zcash” [52, 38]. However, the QSP based ZK system used in Zerocash/Zcash, called a “preprocessing SNARK” [24], requires a setup phase that involves *private* randomness; additionally, it relies on rather strong cryptographic “knowledge of exponent” assumptions, and quantum computers can create pseudo-proofs of falsities in polynomial time for such systems [60] (cf. [54]). In contrast, the aforementioned succinct interactive and non-interactive (NIROP) systems based on quasilinear PZK-IOPs require only public randomness for their setup, and the only cryptographic assumption required to realize them⁵ is the existence of a family of collision resistant hash functions [43], in particular, they are not known to be breakable by quantum computers in polynomial time. Therefore, there is great interest in understanding whether succinct (interactive and non-interactive) ZK argument systems which require only public randomness (and resistant to known polynomial time quantum algorithms) can be *practically* built and used, say, by Zcash. Ben-Sasson et al. [7] describe such an implemented system, called “succinct computational integrity (SCI)” which is not ZK and has comparatively large communication complexity⁶. As mentioned above, the RS proximity solution described in Theorem 2 is already used within an implemented ZK system [11].

1.3 Concrete degree, communication, and round complexity

In this section we *briefly* discuss the “size” of RS codes that would be needed for various practical applications and the effect of logarithmic round complexity on security. Due to space limitations, and because the focus of this paper is *theoretical* (within the information theoretic IOP model), we omit implementation details and point the interested reader to full version of this paper [10]; cf. [7, 14].

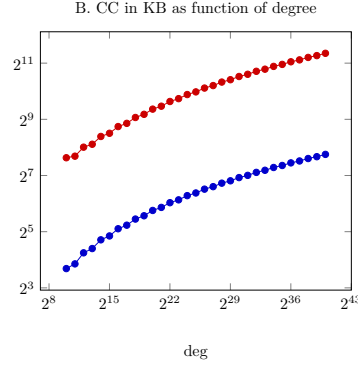
The message length of RS codes of degree $d = \rho \cdot N - 1$ is precisely d , so we start by recounting the range of degrees (message sizes) that seem practically relevant. Later we calculate the communication complexity arising from using the FRI protocol to argue proximity to codes of practically relevant block-lengths, and end by discussing the practical implications of an IOPP with $\log d$ rounds. Throughout this section $\rho = 1/8$ ($N = 8 \cdot d$) because this setting is used in prior [7] and future [11] works.

1.3.1 RS block-length of systems realized in code

The recently realized IOP-based argument system called SCI (“Scalable Computational Integrity”) reduces computational statements, like “*the output of program P on input x equals y after T steps*” to a *pair* of RS-proximity testing problems. SCI uses an IOP version of the quasilinear PCP of [23], which could be replaced with FRI. Programs bench-marked by SCI were executed on a simple MIPS-like virtual machine called *TinyRAM* [18]. Generally speaking, RS degree increases in size with the number of TinyRAM machine cycles T .

⁵ To reach a (non-interactive) computationally sound (CS) proof [49], the “random oracle” is assumed, and realized in practice by relying on the Fiat-Shamir heuristic. In particular, this approach as well is not known to be breakable by quantum computers in polynomial time.

⁶ Communication complexity in SCI is on the order of tens of megabytes long, compared with QSP based zk-SNARKs that are shorter than 300 Bytes.



■ **Figure 1** A. Degree of RS code arising from the *exhaustive subset sum* program [7, Appendix C], as a function of the number of TinyRAM machine cycles. B. Communication complexity (CC) as a function of degree, using $\lambda = 160$ bits, field size 2^{64} , soundness error $\epsilon = 2^{-80}$, and maximal proximity parameter $\delta = 1 - \rho$. The higher (red) graph corresponds to proven soundness (see full version of this paper [10]) and the lower (blue) corresponds to conjectured soundness (Conjecture 3). Both plots use code rate $\rho = 1/8$.

Figure 1.A plots the degree d as a function of T for a specific simple program, showing that $d \approx T \cdot 2^{21}$.

For crypto-currency applications requiring zero knowledge, block-length will be dominated by the type of cryptographic primitives required, and the number of times they are invoked within a computational statement. For instance, *ZK contingent payments* [47] require a single hash, and Zerocash’s *Pour circuit* [15] uses 64 hash invocations, leading in that work to RS codewords (over a prime field) with degree (=number of gates) approximately 2^{22} . Our new work in progress shows that a single hash invocation requires RS block-length between $2^{12} = 4096$ (for a Davies–Meyer hash based on AES128) to 2^{19} (for SHA2), meaning that degrees in the range $d \in [2^{12}, 2^{26}]$ are relevant for existing crypto-currency (ZK) applications [11].

1.3.2 Estimated communication complexity and argument length

The practical realization of interactive proof systems (see Section 1.2) into interactive argument systems [43] and CS proofs [49] can be extended to the IOPP model, in which multiple rounds of interaction are used [19]. Using Kilian’s scheme [43], during the i th round the prover sends the root $\text{root}^{(i)}$ of a Merkle hash tree $\text{Tree}^{(i)}$ whose leaves are labeled by entries of $f^{(i)}$, and the verifier replies with randomness. Using Micali’s scheme [49], the (non-interactive) prover queries the random oracle with $\text{root}^{(i)}$ to “simulate” the verifier’s i th message. When verifier queries to $f^{(i)}$ are answered by the prover, each answer is accompanied by an *authentication path* (AP) that shows the query answer is consistent with $\text{root}^{(i)}$. Let $\text{CC}_{\delta,\epsilon}(N)$ denote the prover-side communication complexity (in bits) of an argument/CS proof realized by applying the Kilian/Micali scheme to FRI, where δ is the proximity parameter and ϵ is the error bound, i.e., words that are δ -far from the RS code are rejected with probability $< \epsilon$. Then

$$\text{CC}_{\delta,\epsilon}(N) = \mathbf{q}_{\delta,\epsilon} \cdot \log |\mathbb{F}| + \text{AP}_{\delta,\epsilon} \cdot \lambda \quad (2)$$

where $\mathbf{q}_{\delta,\epsilon}$ denotes total query complexity in the IOP model to reach soundness $\geq 1 - \epsilon$ for proximity parameter δ , $\text{AP}_{\delta,\epsilon}$ is the number of nodes in the sub-forest of the Merkle trees $\text{Tree}^{(0)}, \dots, \text{Tree}^{(r)}$ induced by all authentication paths, and λ is the number of output bits of

the hash function used to construct the Merkle trees. In our preliminary results [11] we use $\lambda = 160$, $\epsilon = 2^{-80}$, $|\mathbb{F}| = 2^{64}$ and $\rho = 1/8$. Figure 1.B plots the communication complexity for this setting under the proven soundness of Theorem 2 and the (better) soundness of Conjecture 3. In both cases we use maximally large distance $\delta = 1 - \rho = 7/8$ to show the concrete difference in communication complexity between the proven and conjectured soundness. This plot also motivates the quest for improving the soundness analysis of Theorem 2.

1.3.3 Round complexity considerations

Assuming that a crypto-currency block-chain serves as a time-stamping service for public messages and a public beacon of randomness, one may use block-chains to simulate verifier messages. Several block-chains (including Zcash) generate blocks every 2.5 minutes, which means that a FRI proof for $d = 2^k$ will take roughly $k \cdot \frac{5}{4}$ minutes to complete, or less than 1 hour⁷ for $d < 2^{40}$.

For fixed d , the round complexity stated in Theorem 2 is $\frac{1}{2} \log d$, but the more refined version (see [10]) gives a trade-off between query (q) and round (r) complexity, of the form $r = \log d / \log q$, allowing further reduction in round complexity in exchange for larger communication complexity.

Finally, the Random Oracle model used by Micali to “compress” interactive argument systems (like Kilian’s) into CS proofs applies equally to multi-round IOPs like FRI, with negligible impact on argument length; see [19, Remark 1.6] for a detailed discussion. Practically speaking, those who treat hash functions like SHA2 as realizations of the RO model (a position taken by Bitcoin and other crypto-currency miners), might feel comfortable compiling IOP protocols like FRI into succinct non-interactive arguments, as described in [19].

1.4 Related works

High-rate LTCs Locally testable codes (LTCs) are error correcting codes for which – by definition – prover complexity and proof length equal 0 (as stated for the case of RS codes by Rubinfeld and Sudan [58]); in other words, when focusing solely on prover complexity, LTCs offer an optimal solution (zero complexity). Nevertheless, as discussed in Section 1.2, the specific question of small prover complexity for RS codes is highly relevant because of the its applications to practical ZK-IOPs.

Classical “direct” constructions of LTCs, such as the Hadamard code studied by Blum, Luby and Rubinfeld [25] and the log N -variate RM codes used in early PCP constructions [1, 5] have sub-constant rate, thus lead to long proofs and large PCP prover complexity.

More recently, there has been remarkable progress on constructing locally testable codes (LTCs) with small query complexity and large soundness. Kopparty et al. obtained such codes with rate approaching 1 [45] and Gopi et al. presented LTCs that reach the Gilbert Varshamov bound [36]. These LTCs have super-polylogarithmic query complexity. Additionally, in contrast to RS codes, we are not aware of PCP constructions with similar parameters nor do we know how to convert these LTCs into PCPs.

PCPs and IOPs: A number of recent works have considered PCP constructions with small proof length and query complexity. In addition to the aforementioned works on quasi-linear PCPs, Moshkovitz and Raz constructed PCPs with optimally small query complexity

⁷ Compare this with Bitcoin’s “best practice” of waiting 1 hour for confirmations, or 3 days required to clear standard checks.

(measured in bits) and proofs of length $N^{1+o(1)}$ [51], where N denotes the length of the NP statement (like a 3CNF) for which the PCP is constructed, achieving better soundness than Håstad's result [37]. A different line of works attempts to optimize the *bit-length* of PCP proofs; the state of the art, due to Ben-Sasson et al., achieves PCPs of bit-length $O(N)$ and query complexity N^ϵ [22]. In the IOP model, which generalizes PCPs by allowing more rounds of interaction, Ben-Sasson et al. presented a 2-round IOP with bit-length $O(N)$, constant query complexity (measured in bits) and constant soundness [13]. (Prover arithmetic complexity in all of these systems is super-linear.)

Soundness amplification: A number of results in the PCP literature have suggested techniques for improving soundness of general PCP constructions, including the parallel repetition theorem of Raz [55], the gap amplification technique of Dinur [28] and direct-product testing, introduced by Goldreich and Safra [34] (cf. [29, 39]). These techniques lead to excellent soundness bounds with small query complexity. The concrete prover complexity of PCPs and PCPPs associated with these methods has not been studied in prior works but prover complexity is at least super-linear, and often polynomially large.

Doubly-efficient “proofs for muggles”: A recent line of works, initiated by Goldwasser, Kalai and Rothblum [35], revisits the IP model which is equivalent to PSPACE [46, 59], focusing on *doubly efficient* systems in which the prover runs in polynomial time (as opposed to polynomial space, as in the aforementioned results) and verifier runs in nearly linear time. The state of the art along this line is due to Reingold et al. [57], they construct doubly-efficient IP protocols with a constant number of rounds for a family of languages in P. Prover complexity in this line of works is at least super-linear, and typically polynomially large and verifier complexity is super-polylogarithmic, and often super-linear as well (cf. [27, 57]).

2 Overview of the FRI IOPP and its soundness

In this section we consider the task of building an IOPP for a “smooth” RS code (defined below). We start in Section 2.1 by considering the *completeness* case, where we describe the interaction between the verifier and an *honest prover* attempting to prove membership in the RS code of a valid codeword $f^{(0)}$. The IOPP protocol is explained in similarity to the Inverse Fast Fourier Transform (IFFT) [26]. Then, in Section 2.2, we consider the *soundness* case, where we assume $f^{(0)}$ is far in relative Hamming distance from the code and need to prove lower bounds on the verifier's rejection probability. Soundness analysis is the most challenging aspect of our work (as it is for all prior PCPP/IOPP works). Our analysis uses the soundness analysis of the quasilinear RS-PCPP [23] for the case of “large” Hamming distance (beyond the unique decoding radius of the code), and presents a novel, tighter, analysis for “small” Hamming distance (below that radius).

2.1 FRI overview and similarity to the Fast Fourier Transform (FFT)

We start by describing the protocol in similarity to the IFFT algorithm; that algorithm is also related to the quasi-linear PCPP for RS codes of [23], and towards the end of this section we explain the connection between FRI and that quasi-linear PCPP.

Let $\omega^{(0)}$ generate a *smooth* multiplicative group of order $N = 2^n$ (see Remark 1.1.3), denoted $L^{(0)}$, that is contained in a field \mathbb{F} ; in signal processing applications $\omega^{(0)}$ is a complex root of unity of order 2^n and \mathbb{F} is the field of complex numbers (we shall use a different setting). Assume the prover claims that $f^{(0)} : L^{(0)} \rightarrow \mathbb{F}$ is a member of $\text{RS}[\mathbb{F}, L^{(0)}, \rho]$, i.e., $f^{(0)}$ is the evaluation of an unknown polynomial $P^{(0)}(X) \in \mathbb{F}[X]$, $\deg(P) < \rho 2^n$; for simplicity we assume $\rho = 2^{-\mathcal{R}}$ and \mathcal{R} is a positive integer. The task of the verifier is to distinguish between

low-degreeness ($f^{(0)} \equiv P^{(0)}$ for some low degree $P^{(0)}$) and cases where $f^{(0)}$ is far from all polynomials of degree $< \rho 2^n$. Recalling the IFFT, if $f^{(0)} \equiv P^{(0)}$ there exist polynomials $P_0^{(1)}, P_1^{(1)} \in \mathbb{F}[Y]$ such that $\max \left\{ \deg \left(P_0^{(1)} \right), \deg \left(P_1^{(1)} \right) \right\} < \frac{1}{2} \rho 2^n$ and

$$\forall x \in L^{(0)} \quad f^{(0)}(x) = P^{(0)}(x) = P_0^{(1)}(x^2) + x \cdot P_1^{(1)}(x^2),$$

or, letting $Q^{(1)}(X, Y) \triangleq P_0^{(1)}(Y) + X \cdot P_1^{(1)}(Y)$ and defining $q^{(0)}(X) \triangleq X^2$, we have

$$P^{(0)}(X) \equiv Q^{(1)}(X, Y) \pmod{Y - q^{(0)}(X)} \quad (3)$$

where $\deg_X(Q^{(1)}) < 2$ and $\deg_Y(Q^{(1)}) < \frac{1}{2} \rho 2^n$. The map $x \mapsto q^{(0)}(x)$ is 2-to-1 on $L^{(0)}$ because $q^{(0)}(x) = q^{(0)}(-x)$, and the output of this map is the multiplicative group generated by $\omega^{(1)} = (\omega^{(0)})^2$, this group has order 2^{n-1} , denote it by $L^{(1)}$. Moreover, for every $x^{(0)} \in \mathbb{F}$ and $y \in L^{(1)}$, the value of $Q^{(1)}(x^{(0)}, y)$ can be computed by querying two entries of $f^{(0)}$ because $\deg_X(Q^{(1)}) < 2$ (the two entries are the two roots of the polynomial $y - q^{(0)}(X)$).

Our verifier thus samples $x^{(0)} \in \mathbb{F}$ uniformly at random and requests the prover to send as its first oracle a function $f^{(1)} : L^{(1)} \rightarrow \mathbb{F}$ that is supposedly the evaluation of $Q^{(1)}(x^{(0)}, Y)$ on $L^{(1)}$. Assuming $f^{(0)} \in \text{RS}[\mathbb{F}, L^{(0)}, \rho]$, the discussion above shows that $f^{(1)} \in \text{RS}[\mathbb{F}, L^{(1)}, \rho]$. Notice that there exists a 3-query test for the consistency of $f^{(0)}$ and $f^{(1)}$, we call it the *round consistency test*:

1. sample a pair of distinct elements $s_0, s_1 \in L^{(0)}$ such that $s_0^2 = s_1^2 = y$; in other words, sample a uniform $y \in L^{(1)}$ and let s_0, s_1 be the two roots of the polynomial $y - X^2$;
2. query $f^{(0)}(s_0), f^{(0)}(s_1)$ and $f^{(1)}(y)$, denote the query answers by α_0, α_1 and β , respectively;
3. interpolate the “line” through (s_0, α_0) and (s_1, α_1) , i.e., find the polynomial $p(X)$ of degree at most 1 that satisfies $p(s_0) = \alpha_0$ and $p(s_1) = \alpha_1$; notice p is unique and well-defined because $s_0 \neq s_1$;
4. accept if and only if $p(x^{(0)}) = \beta$ and otherwise reject;

Tallying the costs of the first round, the verifier sends a single field element ($x^{(0)}$) and the prover responds with a message (oracle) $f^{(1)} : L^{(1)} \rightarrow \mathbb{F}$ evaluated on a domain that is half the size of $L^{(0)}$; testing the consistency of $f^{(0)}$ and $f^{(1)}$ requires three field elements per test (repeating the test boosts soundness). We thus reduced a single proximity problem of size 2^n and rate ρ to a single analogous problem of size 2^{n-1} and same rate. Repeating the process for $r = n - \mathcal{R}$ rounds leads to a function $f^{(r)}$ that is supposedly of constant degree and evaluated over a domain of constant size $2^{\mathcal{R}}$, so at this point the prover sends the single constant that describes the function, and the verifier uses it as $f^{(r)}$ in the last round consistency test, the one that tests consistency of $f^{(r-1)}$ and $f^{(r)}$.

Applying inductive analysis to all r rounds, if $f^{(0)} \in \text{RS}[\mathbb{F}, L^{(0)}, \rho]$ (and the prover is honest) then all r round consistency tests pass with probability 1 and $f^{(r)}$ is indeed a constant function. In other words, the protocol we described has perfect completeness.

► **Remark (FRI as a “biased” version of quasi-linear RS-PCPP).** The quasi-linear PCPP of [23] is quite similar to FRI, including the degree-reduction (from $P^{(0)}$ to $P^{(1)}$) obtained by requesting the prover to evaluate a bivariate polynomial $Q(X, Y)$ on a collection of axis-parallel lines. There are two main differences between FRI and that PCPP:

1. the quasilinear PCPP is non-interactive, and thus the prover evaluates $Q^{(1)}(X, Y)$ on a large subset of $\mathbb{F} \times \mathbb{F}$, whereas the FRI protocol uses interaction to reduce proving time, by requesting the prover to apply recursion only to the axis-parallel lines selected by the verifier.

2. the polynomial $q^{(0)}(X)$ used in [23] has degree $\approx \sqrt{\deg(P^{(0)})}$ and thus $Q^{(1)}(X, Y)$ has degree $\approx \sqrt{\deg(P^{(0)})}$ in each of its variables. In contrast, the polynomial $q^{(0)}$ used by FRI has *constant* degree, and so the degrees of $Q^{(1)}$ are very biased (constant degree in X vs. $\deg(P^{(0)})/2$ in Y). This leads to larger recursion depth for FRI but also avoids the necessity to apply recursive low-degree testing to each of the axes (the X -axis) because of its constant degree.

2.1.1 Differences between informal and actual protocol

The differences between the informal and formal protocols are mostly technical; we list them now. The field \mathbb{F} is finite and *binary*, i.e., of characteristic 2; nevertheless the construction and analysis can be immediately applied to RS codes evaluated over smooth multiplicative groups (of order 2^n), as explained informally above (cf. Remark 1.1.3). In binary fields, the natural evaluation domains (like $L^{(0)}, L^{(1)}$ above) are cosets of *additive* groups (not multiplicative ones), i.e., $L^{(i)}$ is an affine shift of a linear space over \mathbb{F}_2 . The map $q^{(0)}(X) = X^2$ is *not* 2-to-1 on $L^{(0)}$ (in binary fields it is a 1-to-1 map, a Frobenius automorphism of \mathbb{F} over \mathbb{F}_2) so we use a different polynomial $q^{(0)}(X)$ that is many-to-one on $L^{(0)}$ and such that the set $L^{(1)} = \{y = q^{(0)}(x) \mid x \in L^{(0)}\}$ is a coset of an *additive* group, like $L^{(0)}$, but of smaller size ($|L^{(1)}| \ll |L^{(0)}|$); the polynomial $q^{(0)}$ is known as an *affine subspace* polynomial, belonging to the class of *linearized* polynomials. We use $q^{(0)}$ of degree 4 instead of 2 because this reduces the number of rounds from n to $n/2$ with no increase in total query complexity; notice that a similar reduction could be applied in the multiplicative setting by using $q^{(0)} = X^4$ (but we preferred simplicity to efficiency in the informal exposition above). Finally, the actual protocol performs all queries only after the prover has sent all of $f^{(1)}, \dots, f^{(r)}$. Thus, we construct a protocol with two phases. The first phase, called the COMMIT phase, involves r rounds. At the beginning of the i th round the prover has sent oracles $f^{(0)}, \dots, f^{(i-1)}$, and during this (i)th round the verifier samples and sends $x^{(i)}$ and the prover responds by sending the next oracle $f^{(i)}$. During the second phase, called the QUERY phase, the verifier applies the *round consistency test* to all r rounds. To save query complexity *and boost soundness*, the query made to $L^{(i)}$ is used to test *both* consistency of $f^{(i-1)}$ vs. $f^{(i)}$ *and* consistency of $f^{(i)}$ vs. $f^{(i+1)}$.

2.2 Soundness analysis – overview

Proof composition is a technique introduced by Arora and Safra [3] in the context of PCPs, adapted to PCPPs in [21, 30] and optimized for the special case of the RS code in [23]. Informally, it reduces proximity testing problems over a large domain to similar proximity testing problems over significantly smaller domains. The process reducing $f^{(0)}$ to $f^{(1)}$ above is a special case of proof composition, and each invocation of it incurs two costs on behalf of the verifier. The first is the *query complexity* needed to check consistency of $f^{(0)}$ and $f^{(1)}$ (the “round consistency test”) and the second is the reduction in *distance*, which affects the *soundness* of the protocol. Assuming $f^{(0)}$ is $\delta^{(0)}$ -far from all codewords in relative Hamming distance, for proof composition to work one should prove that with high probability $f^{(1)}$ is $\delta^{(1)}$ -far from all codewords where $\delta^{(1)}$ depends on $\delta^{(0)}$; larger values of $\delta^{(1)}$ imply higher (better) soundness and smaller communication complexity. A benefit of the FRI protocol is that with high probability $\delta^{(1)} \geq (1 - o(1))\delta^{(0)}$, i.e., the reduction in distance in our protocol is *negligible*. In contrast, prior RS proximity PCPP and IOPP solutions follow the construction and analysis of [23] which in turn is based on the bivariate testing Theorem of

Polischuk and Spielman [53] and incur a *constant multiplicative loss* in distance per round of proof composition ($\delta^{(1)} \leq \delta^{(0)}/2$). This loss limited the number of proof composition rounds to $\leq \log N$ and thus required replacing $q^{(0)}(X) = X^2$ with a higher degree polynomial, like $q^{(0)}(X) = X^{2^{n/2}}$. The higher degree of $q^{(0)}$ results in $Q^{(1)}(X, Y)$ having *balanced* X - and Y -degrees, namely

$$\deg_X(Q^{(1)}) \approx \deg_Y(Q^{(1)}) \approx 2^{n/2}.$$

Moving to $q^{(0)}(X)$ of *constant* degree as in FRI gives a *biased RS-IOPP* (because $\deg_X(Q^{(1)}) \ll \deg_Y(Q^{(1)})$). The main benefit of this bias is that one side of the recursive process (that of X) terminates immediately and consequently *removes* the constant multiplicative soundness loss incurred in prior works, replacing it with a negligible additive loss. More to the point, we show that for $\delta^{(0)}$ less than the unique decoding radius of the code ($\delta^{(0)} < (1 - \rho)/2$), with high probability (namely, $1 - \frac{O(1)}{|\mathbb{F}|}$) the sum of (i) the round consistency error and (ii) the “new” distance $\delta^{(1)}$ is at least as large as the “old” distance $\delta^{(0)}$. This statement is relatively straightforward to prove in case the prover is *honest*, i.e., when $f^{(1)}(y) = Q^{(1)}(x^{(0)}, y)$ for all $y \in L^{(1)}$ (in this case there is no round consistency error). The challenging part of the proof is to show this also holds for non-honest provers and arbitrary $f^{(1)}$; see full version of this paper [10] for more details.

References

- 1 Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and hardness of approximation problems. In *Proceedings of the 33rd Annual Symposium on Foundations of Computer Science*, pages 14–23, 1992.
- 2 Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *Journal of the ACM*, 45(3):501–555, 1998. Preliminary version in FOCS ’92.
- 3 Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: a new characterization of NP. *Journal of the ACM*, 45(1):70–122, 1998. Preliminary version in FOCS ’92.
- 4 László Babai. Trading group theory for randomness. In *Proceedings of the 17th Annual ACM Symposium on Theory of Computing*, STOC ’85, pages 421–429, 1985.
- 5 László Babai, Lance Fortnow, Leonid A. Levin, and Mario Szegedy. Checking computations in polylogarithmic time. In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing*, STOC ’91, pages 21–32, 1991.
- 6 László Babai, Lance Fortnow, and Carsten Lund. Nondeterministic exponential time has two-prover interactive protocols. In *Proceedings of the 31st Annual Symposium on Foundations of Computer Science*, FOCS ’90, pages 16–25, 1990.
- 7 Eli Ben-Sasson, Iddo Bentov, Alessandro Chiesa, Ariel Gabizon, Daniel Genkin, Matan Hamilis, Evgenya Pergament, Michael Riabzev, Mark Silberstein, Eran Tromer, and Madars Virza. Computational integrity with a public random string from quasi-linear PCPs. *IACR Cryptology ePrint Archive*, 2016:646, 2016. URL: <http://eprint.iacr.org/2016/646>.
- 8 Eli Ben-Sasson, Iddo Bentov, Ariel Gabizon, and Michael Riabzev. A security analysis of probabilistically checkable proofs. *Electronic Colloquium on Computational Complexity (ECCC)*, 23:149, 2016. URL: <http://eccc.hpi-web.de/report/2016/149>.
- 9 Eli Ben-Sasson, Iddo Bentov, Ariel Gabizon, and Michael Riabzev. A security analysis of probabilistically checkable proofs. *Electronic Colloquium on Computational Complexity (ECCC)*, 23:149, 2016. URL: <http://eccc.hpi-web.de/report/2016/149>.

- 10 Eli Ben-Sasson, Iddo Bentov, Ynon Horesh, and Michael Riabzev. Fast Reed-Solomon interactive oracle proofs of proximity (2nd revision). *Electronic Colloquium on Computational Complexity (ECCC)*, 24:134, 2017. URL: <https://eccc.weizmann.ac.il/report/2017/134>.
- 11 Eli Ben-Sasson, Iddo Bentov, Ynon Horesh, and Michael Riabzev. Scalable, transparent, and post-quantum secure computational integrity, 2017. Unpublished manuscript.
- 12 Eli Ben-Sasson, Alessandro Chiesa, Michael A. Forbes, Ariel Gabizon, Michael Riabzev, and Nicholas Spooner. On probabilistic checking in perfect zero knowledge. *Electronic Colloquium on Computational Complexity (ECCC)*, 23:156, 2016. URL: <http://eccc.hpi-web.de/report/2016/156>.
- 13 Eli Ben-Sasson, Alessandro Chiesa, Ariel Gabizon, Michael Riabzev, and Nicholas Spooner. Short interactive oracle proofs with constant query complexity, via composition and sumcheck. *Electronic Colloquium on Computational Complexity (ECCC)*, 23:46, 2016.
- 14 Eli Ben-Sasson, Alessandro Chiesa, Ariel Gabizon, and Madars Virza. Quasilinear-size zero knowledge from linear-algebraic PCPs. In *Proceedings of the 13th Theory of Cryptography Conference*, TCC '16, pages 33–64, 2016.
- 15 Eli Ben-Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. Zerocash: Decentralized anonymous payments from Bitcoin. In *Proceedings of the 2014 IEEE Symposium on Security and Privacy*, SP '14, 2014.
- 16 Eli Ben-Sasson, Alessandro Chiesa, Daniel Genkin, and Eran Tromer. On the concrete efficiency of probabilistically-checkable proofs. In *Proceedings of the 45th ACM Symposium on the Theory of Computing*, STOC '13, pages 585–594, 2013.
- 17 Eli Ben-Sasson, Alessandro Chiesa, Daniel Genkin, Eran Tromer, and Madars Virza. SNARKs for C: Verifying program executions succinctly and in zero knowledge. In *Proceedings of the 33rd Annual International Cryptology Conference*, CRYPTO '13, pages 90–108, 2013.
- 18 Eli Ben-Sasson, Alessandro Chiesa, Daniel Genkin, Eran Tromer, and Madars Virza. Tinyram architecture specification v2. 00, 2013. Technical report, SCIPR Lab, 2013. URL: <http://www.scipr-lab.org/doc/TinyRAM-spec-0.991.pdf>.
- 19 Eli Ben-Sasson, Alessandro Chiesa, and Nicholas Spooner. *Interactive Oracle Proofs*, pages 31–60. Springer Berlin Heidelberg, Berlin, Heidelberg, 2016. doi:10.1007/978-3-662-53644-5_2.
- 20 Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil Vadhan. Short PCPs verifiable in polylogarithmic time. In *Proceedings of the 20th Annual IEEE Conference on Computational Complexity*, CCC '05, pages 120–134, 2005.
- 21 Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil P. Vadhan. Robust PCPs of proximity, shorter PCPs, and applications to coding. *SIAM Journal on Computing*, 36(4):889–974, 2006.
- 22 Eli Ben-Sasson, Yohay Kaplan, Swastik Kopparty, Or Meir, and Henning Stichtenoth. Constant rate pcps for circuit-sat with sublinear query complexity. *J. ACM*, 63(4):32:1–32:57, 2016. doi:10.1145/2901294.
- 23 Eli Ben-Sasson and Madhu Sudan. Short PCPs with polylog query complexity. *SIAM Journal on Computing*, 38(2):551–607, 2008. Preliminary version appeared in STOC '05.
- 24 Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, ITCS '12, pages 326–349, 2012.
- 25 Manuel Blum, Michael Luby, and Ronitt Rubinfeld. Self-testing/correcting with applications to numerical problems. *J. Comput. Syst. Sci.*, 47(3):549–595, 1993. doi:10.1016/0022-0000(93)90044-w.

- 26 James W Cooley and John W Tukey. An algorithm for the machine calculation of complex fourier series. *Mathematics of computation*, 19(90):297–301, 1965.
- 27 Graham Cormode, Michael Mitzenmacher, and Justin Thaler. Practical verified computation with streaming interactive proofs. In *Proceedings of the 4th Symposium on Innovations in Theoretical Computer Science*, ITCS '12, pages 90–112, 2012.
- 28 Irit Dinur. The PCP theorem by gap amplification. *Journal of the ACM*, 54(3):12, 2007.
- 29 Irit Dinur and Elazar Goldenberg. Locally testing direct product in the low error range. In *49th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2008, October 25-28, 2008, Philadelphia, PA, USA*, pages 613–622, 2008. doi:10.1109/FOCS.2008.26.
- 30 Irit Dinur and Omer Reingold. Assignment testers: Towards a combinatorial proof of the PCP theorem. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '04, pages 155–164, 2004.
- 31 Amos Fiat and Adi Shamir. How to prove yourself: practical solutions to identification and signature problems. In *Proceedings of the 6th Annual International Cryptology Conference*, CRYPTO '86, pages 186–194, 1986.
- 32 K. Friedl and M. Sudan. Some improvements to total degree tests. In *Proceedings Third Israel Symposium on the Theory of Computing and Systems*, pages 190–198, Jan 1995. doi:10.1109/ISTCS.1995.377032.
- 33 Rosario Gennaro, Craig Gentry, Bryan Parno, and Mariana Raykova. Quadratic span programs and succinct NIZKs without PCPs. In *Proceedings of the 32nd Annual International Conference on Theory and Application of Cryptographic Techniques*, EUROCRYPT '13, pages 626–645, 2013.
- 34 Oded Goldreich and Shmuel Safra. A combinatorial consistency lemma with application to proving the pcg theorem. *SIAM Journal on Computing*, 29(4):1132–1154, 2000. doi:10.1137/S0097539797315744.
- 35 Shafi Goldwasser, Yael Tauman Kalai, and Guy N. Rothblum. Delegating computation: Interactive proofs for Muggles. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, STOC '08, pages 113–122, 2008.
- 36 Sivakanth Gopi, Swastik Kopparty, Rafael Mendes de Oliveira, Noga Ron-Zewi, and Shubhangi Saraf. Locally testable and locally correctable codes approaching the gilbert-varshamov bound. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 2073–2091, 2017. doi:10.1137/1.9781611974782.135.
- 37 Johan Hr astad. Some optimal inapproximability results. *Journal of the ACM*, 48(4):798–859, 2001.
- 38 Daira Hopwood, Sean Bowe, Taylor Hornby, and Nathan Wilcox, March 2017. URL: <https://github.com/zcash/zips/blob/master/protocol/protocol.pdf>.
- 39 Russell Impagliazzo, Valentine Kabanets, and Avi Wigderson. New direct-product testers and 2-query pcps. *SIAM Journal on Computing*, 41(6):1722–1768, 2012. doi:10.1137/09077299X.
- 40 Yuval Ishai, Mohammad Mahmoody, Amit Sahai, and David Xiao. On zero-knowledge PCPs: Limitations, simplifications, and applications, 2015. Available at <http://www.cs.virginia.edu/~mohammad/files/papers/ZKPCPs-Full.pdf>.
- 41 Yuval Ishai and Mor Weiss. Probabilistically checkable proofs of proximity with zero-knowledge. In *Theory of Cryptography - 11th Theory of Cryptography Conference, TCC 2014, San Diego, CA, USA, February 24-26, 2014. Proceedings*, pages 121–145, 2014. doi:10.1007/978-3-642-54242-8_6.
- 42 Yael Kalai and Ran Raz. Interactive PCP. In *Proceedings of the 35th International Colloquium on Automata, Languages and Programming*, ICALP '08, pages 536–547, 2008.

- 43 Joe Kilian. A note on efficient zero-knowledge proofs and arguments. In *Proceedings of the 24th Annual ACM Symposium on Theory of Computing, STOC '92*, pages 723–732, 1992.
- 44 Joe Kilian, Erez Petrank, and Gábor Tardos. Probabilistically checkable proofs with zero knowledge. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing, STOC '97*, pages 496–505, 1997.
- 45 Swastik Kopparty, Or Meir, Noga Ron-Zewi, and Shubhangi Saraf. High-rate locally-correctable and locally-testable codes with sub-polynomial query complexity. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 202–215, 2016. doi:10.1145/2897518.2897523.
- 46 Carsten Lund, Lance Fortnow, Howard J. Karloff, and Noam Nisan. Algebraic methods for interactive proof systems. *Journal of the ACM*, 39(4):859–868, 1992.
- 47 Gregory Maxwell. Zero knowledge contingent payment, 2011. [Online; accessed 13-October-2017].
- 48 Silvio Micali. Computationally sound proofs. *SIAM Journal on Computing*, 30(4):1253–1298, 2000. Preliminary version appeared in FOCS '94.
- 49 Silvio Micali. Computationally sound proofs. *SIAM J. Comput.*, 30(4):1253–1298, 2000. doi:10.1137/S0097539795284959.
- 50 Thilo Mie. Short PCPPs verifiable in polylogarithmic time with $o(1)$ queries. *Annals of Mathematics and Artificial Intelligence*, 56:313–338, 2009.
- 51 Dana Moshkovitz and Ran Raz. Two-query PCP with subconstant error. *J. ACM*, 57(5):29:1–29:29, 2010. doi:10.1145/1754399.1754402.
- 52 M. Peck. A blockchain currency that beat s bitcoin on privacy [news]. *IEEE Spectrum*, 53(12):11–13, December 2016. doi:10.1109/MSPEC.2016.7761864.
- 53 Alexander Polishchuk and Daniel A. Spielman. Nearly-linear size holographic proofs. In *Proceedings of the 26th Annual ACM Symposium on Theory of Computing, STOC '94*, pages 194–203, 1994.
- 54 John Proos and Christof Zalka. Shor's discrete logarithm quantum algorithm for elliptic curves. *Quantum Info. Comput.*, 3(4):317–344, 2003. URL: <http://dl.acm.org/citation.cfm?id=2011528.2011531>.
- 55 Ran Raz. A parallel repetition theorem. In *Proceedings of the 27th Annual ACM Symposium on Theory of Computing, STOC '95*, pages 447–456, 1995.
- 56 I. S. Reed and G. Solomon. Polynomial codes over certain finite fields. *Journal of the Society for Industrial and Applied Mathematics*, 8(2):300–304, 1960. doi:10.1137/0108018.
- 57 Omer Reingold, Guy N. Rothblum, and Ron D. Rothblum. Constant-round interactive proofs for delegating computation. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 49–62, 2016. doi:10.1145/2897518.2897652.
- 58 Ronitt Rubinfeld and Madhu Sudan. Self-testing polynomial functions efficiently and over rational domains. In *Proceedings of the Third Annual ACM/SIGACT-SIAM Symposium on Discrete Algorithms, 27-29 January 1992, Orlando, Florida.*, pages 23–32, 1992. URL: <http://dl.acm.org/citation.cfm?id=139404.139410>.
- 59 Adi Shamir. IP = PSPACE. *Journal of the ACM*, 39(4):869–877, 1992.
- 60 P. W. Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pages 124–134, Nov 1994. doi:10.1109/SFCS.1994.365700.
- 61 Paul Valiant. Incrementally verifiable computation or proofs of knowledge imply time/space efficiency. In *Proceedings of the 5th Conference on Theory of Cryptography, TCC'08*, pages 1–18, Berlin, Heidelberg, 2008. Springer-Verlag. URL: <http://dl.acm.org/citation.cfm?id=1802614.1802616>.

- 62 Michael Walfish and Andrew J. Blumberg. Verifying computations without reexecuting them. *Commun. ACM*, 58(2):74–84, 2015. doi:10.1145/2641562.