2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)

Local Proofs Approaching the Witness Length
[Extended Abstract]

Noga Ron-Zewi
Department of Computer Science
University of Haifa
Haifa, Israel
Email: noga@cs.haifa.ac.il

Ron D. Rothblum
Department of Computer Science
Technion
Haifa, Israel
Email: rothblum@cs.technion.ac.il

***Abstract*—Interactive oracle proofs (IOPs) are a hybrid between interactive proofs and PCPs. In an IOP the prover is allowed to interact with a verifier (like in an interactive proof) by sending relatively long messages to the verifier, who in turn is only allowed to query a few of the bits that were sent (like in a PCP). Efficient IOPs are at the core of leading practical implementations of highly efficient proof-systems.**

In this work we construct, for a large class of NP relations, IOPs in which the communication complexity approaches the witness length. More precisely, for any NP relation for which membership can be decided in polynomial-time and bounded polynomial space (e.g., SAT, Hamiltonicity, Clique, Vertex-Cover, etc.) and for any constant $\gamma > 0$, we construct an IOP with communication complexity $(1 + \gamma) \cdot n$, where n is the original witness length. The number of rounds, as well as the number of queries made by the IOP verifier, are constant.

This result improves over prior works on short IOPs/PCPs in two ways. First, the communication complexity in these short IOPs is proportional to the complexity of *verifying* the NP witness, which can be polynomially larger than the witness size. Second, even ignoring the difference between witness length and non-deterministic verification time, prior works incur (at the very least) a large constant multiplicative overhead to the communication complexity.

In particular, as a special case, we also obtain an IOP for CircuitSAT with communication complexity $(1+\gamma) \cdot t$, for circuits of size t and any constant $\gamma > 0$. This improves upon the prior state-of-the-art work of Ben Sasson *et al.* (ICALP, 2017) who construct an IOP for CircuitSAT with communication length $c \cdot t$ for a large (unspecified) constant $c \geq 1$.

Our proof leverages the local testability and (relaxed) local correctability of high-rate tensor codes, as well as their support of a sumcheck-like procedure. In particular, we bypass the barrier imposed by the low rate of *multiplication codes* (e.g., Reed-Solomon, Reed-Muller or AG codes) - a key building block of all known short PCP/IOP constructions.

***Keywords*-computational complexity**

I. Introduction

The celebrated PCP Theorem, established in the early 90's [2]–[5], shows that it is possible to encode any NP witness in such a way that the veracity of the witness can be verified by reading only a constant number of bits from the encoding. This foundational result has had a transformative effect on TCS with diverse applications in cryptography and complexity theory.

A basic and natural question that has drawn a great amount of interest is what is the minimal overhead in encoding that is needed to allow for such local checking of an NP witness. While the original PCP theorem only guarantees a polynomial overhead, a beautiful line of work has culminated in remarkably short PCPs. More precisely, the works of Ben Sasson and Sudan [6] combined with that of Dinur [7] yield PCPs of length $t \cdot \mathrm{polylog}(t)$ for any time t non-deterministic computation.

A central open question in the area is whether such poly-logarithmic overhead is indeed necessary, or can one construct truly linear length PCPs. Actually, since the question is highly dependent on the computational model (since transitions between standard computational models usually involve at the very least a logarithmic overhead), a cleaner formalization of this question is the following:

Does the NP*-complete language*[1] CircuitSAT *have a* PCP *whose length is linear in the given circuit?*

Beyond its intrinsic interest, this question also has implications to the construction of efficient proof-

A full version of this paper is available online at ECCC [1].

[1] Recall that the language CircuitSAT consists of the set of all satisfiable Boolean circuits, and that there is an $O(t \cdot \log t)$-time reduction from NTIME(t) to CircuitSAT [8].

2575-8454/20/$31.00 ©2020 IEEE
DOI 10.1109/FOCS46700.2020.00083

systems that build on PCPs [9]–[11] as well as to the field of hardness of approximation [12]–[14]. The state-of-the-art is a result of [15] who construct a linear length PCP for CircuitSAT albeit with only n^{ε} query complexity, for any desired constant $\varepsilon > 0$. This result falls well short of the desired goal of *constant* query complexity (and has the additional drawback that the verification is non-uniform).

Motivated by the goal of constructing such short PCPs and their applications to the construction of efficient proof-systems, Ben Sasson *et al.* [11] recently proposed a natural generalization of PCPs called *interactive oracle proofs*, or IOPs for short.[2] An IOP is an interactive protocol (similarly to an interactive proof), but at each round the prover can send a *long* message from which the verifier is allowed to read only a few bits (i.e., PCP-style access to the prover messages). Ben Sasson *et al.* [17] later constructed an IOP for CircuitSAT in which the communication complexity (which we will also refer to as *proof length*) is $O(t)$, thereby demonstrating that local proofs with constant overhead exist, if one allows for interaction.

A recent exciting sequence of works [18]–[21] has leveraged the efficiency of IOPs to construct highly efficient succinct argument schemes which are now at the core of leading *practical* implementations [19].

A. Our results

In this work we construct IOPs with nearly optimal proof length. As our first main result, which actually follows from a more general statement that will be elaborated on in Section I-A1, we construct an IOP for CircuitSAT in which the proof length is $(1+\gamma)\cdot t$, for circuits of size t and for any constant $\gamma > 0$.

Theorem 1 (Informally Stated, see Corollary II.3 and Remark I.2)**.** *For every constants $\gamma, \varepsilon > 0$, CircuitSAT has a constant-round and constant-query IOP in which the proof-length is $(1+\gamma)\cdot t$, where t is the circuit size. The IOP has perfect completeness and soundness error ε. The verifier runs in time $\tilde{O}(t)$ and the prover runs in time $\mathrm{poly}(t)$ (given the satisfying assignment).*

Theorem 1 should be contrasted with the main result of [17], who give an IOP for CircuitSAT with proof length $c\cdot t$, for a constant $c \geq 1$ that is left unspecified.[3] While the proof length in Theorem 1 is shorter than that in [17], the round complexity is slightly larger (Indeed, while we have not attempted to optimize the round complexity, a naive implementation seems to require 6 rounds, whereas [17] has only 3 rounds).

Remark I.1. *The IOP in Theorem 1 can be extended to* sub-constant *values of $\gamma = \gamma(t) > 0$ and $\varepsilon = \varepsilon(t) > 0$ (simultaneously) at the cost of increasing the query complexity to $\mathrm{poly}\left(\frac{1}{\gamma}, \log(1/\varepsilon)\right)$ (and no overhead to the round complexity).[4] This leads to an IOP with communication complexity $(1+o(1))\cdot t$ and negligible soundness error, but with poly-logarithmic query complexity.*

1) Approaching the witness length: When considering a generic NP relation $\mathcal{R}$, the result of Theorem 1, the main result in [17], and essentially all short PCP constructions, only yield constructions in which the communication scales with the *verification complexity* of $\mathcal{R}$, rather than with the length of the original NP witness. Essentially, this is because PCPs — in order to facilitate local checking — typically encode the entire computation (rather than just the witness) via a suitable error correcting code.

For example, consider checking the 3-Colorability of a (connected) graph $G = (V, E)$. Using any of the known PCPs or IOPs in the literature would give proof length $\Omega(|E|)$, which can be quadratically larger than the length of the natural NP witness (which has length $O(|V|)$). Alternatively, checking the satisfiability of a given CNF formula ϕ with m clauses and n variables is only known to have PCPs and IOPs of length $\Omega(m+n)$, but has a standard NP witness of length n (which again can be significantly smaller than $m+n$).

This limitation actually turns out to be inherent for PCPs. Fortnow and Santhanam [22] showed that constructing PCPs whose length is a *fixed* polynomial of the witness length is impossible, unless NP $\subseteq$ co-NP/poly.

Interestingly however, Kalai and Raz [23] show that the picture changes drastically if one allows for interaction. In particular, their work yields IOPs with proof length that is polynomial in the witness size for a

[2] The same notion was proposed independently by Reingold *et al.* [16] (who used the term *probabilistically checkable interactive proof*) but for a different motivation - facilitating the construction of doubly-efficient interactive proofs.

[3] We estimate that c is at least $3\cdot 6^3 = 648$. This is since the [17] IOP includes three codewords, each of which is a tensor (of dimension ≥ 3) of an AG code. The AG code has rate $1-\frac{1}{q-1}$, using an alphabet of size q^2 (for any prime power q). However, since they encode *binary* messages, the effective rate is $(1-\frac{1}{q-1})/\log(q^2)$, which achieves its maximum (over prime powers) at $q = 4$.

[4] The fact that the soundness error can be reduced is non-trivial: straightforward error reduction by repetition can significantly increase the proof length.

large class of NP relations, with poly-logarithmic query complexity.[5] We improve upon the result of [23] by constructing IOPs, also for a large class of NP relations, in which the proof length is $(1+\gamma)\cdot n$, where n is the length of the original NP witness, rather than $\mathrm{poly}(n)$ as in [23], and with constant query and round complexities.

The class of relations that we can support consists of all NP relations that can be verified in polynomial-time and bounded polynomial space.[6] In particular, we obtain IOPs with communication approaching the witness length for many natural NP problems, such as: SAT, Hamiltonicity, TSP, all 21 of Karp's original NP-complete problems[7] [29], etc.

Theorem 2 (Informally Stated, see Corollary II.3)**.** *Let $\mathcal{R}$ be an* NP *relation which can be verified in polynomial-time and bounded polynomial space (i.e., space n^{ξ} for some sufficiently small constant $\xi > 0$). Then for any constants $\gamma, \varepsilon > 0$, the relation $\mathcal{R}$ has a constant-round and constant-query* IOP *with proof-length $(1+\gamma)\cdot n$, where n is the original witness length. The* IOP *has perfect completeness and soundness error ε. The verifier runs in quasi-linear time and the prover runs in polynomial-time (given the* NP *witness).*

Remark I.2. *It is worth emphasizing that* every *language in* NP *has a corresponding* NP *relation which can be verified in small space. Indeed, this follows directly from the Cook-Levin theorem. However, the Cook-Levin transformation incurs a polynomial blowup to the witness size (corresponding to the non-deterministic verification time of the relation). The point of Theorem 2 is that for* NP *relations which* a priori *can be verified in small space, we do not need to pay this additional (potentially large) overhead.*

Still, by applying Theorem 2 to CircuitSAT *using the* NP *relation arising from the Cook-Levin theorem (in which the witness includes, in addition to the satisfying assignment, the values of all of the gates in the circuit), we obtain an* IOP *for* CircuitSAT *with proof length $(1+\gamma)\cdot t$, where t is the circuit size. Thus, Theorem 1 follows as an immediate corollary of Theorem 2.*

Similarly to Theorem 1, we can extend Theorem 2 to sub-constant values of $\gamma = \gamma(n) > 0$ and $\varepsilon = \varepsilon(n) > 0$, at the cost of increasing the query complexity to $\mathrm{poly}(1/\gamma, \log(1/\varepsilon))$ (and no overhead to the round complexity), leading to communication complexity $(1+o(1))\cdot n$ and negligible soundness error, but with poly-logarithmic query complexity.

The communication complexity in Theorem 2 is very close to optimal, under the following plausible complexity theoretic conjecture. Loosely speaking, the *randomized strong exponential time hypothesis* (RSETH) states that SAT is not contained in $\mathsf{BPTIME}\big(2^{(1-\gamma)\cdot n}\big)$ for any constant $\gamma > 0$ (where n here is the number of variables in the formula). The work of Goldreich and Håstad [30] shows that only languages in $\mathsf{BPTIME}(2^{b}\cdot \mathrm{poly}(n))$ have constant-round public-coin interactive proofs (let alone IOPs) in which the prover sends at most b bits. Put together, these two facts imply that SAT does not have a (constant-round and public-coin) IOP in which the prover sends less than $(1-\gamma)\cdot n$ bits, unless RSETH is false. For a more precise statement and further details, see the full version [1].

We also mention that the limitation in Theorem 2 to relations computable in space n^{ξ} (for a sufficiently small constant $\xi > 0$) is inherited from the work of [16], on which we build. In this work we have not made an attempt to optimize the constant ξ. However, we believe that ξ could potentially be any constant smaller than $1/2$ (i.e., leading to a space bound of $n^{0.499}$). Whether or not the space bound can be improved or altogether eliminated is an interesting open problem.

Remark I.3. *Our results can also be interpreted as* interactive PCPs *(*IPCP*) [23], a more restricted model than* IOP*s in which the prover first sends a single long message to which the verifier has oracle access (like a* PCP*), followed by short interactive protocol with sublinear communication during which the verifier reads the prover's messages in full (like an interactive proof).*

Specifically, Theorem 2 (see Theorem II.1) gives an IPCP *in which the first message has length $(1+\gamma)\cdot n$ and constant query complexity, and the rest of the communication is of length n^{β}, for arbitrarily small constants*

[5] Kalai and Raz [23] consider a restricted model (in fact a predecessor) of IOPs called *interactive* PCPs. Their result, combined with followup works [16], [24], yields IOPs with length that is polynomial in the witness size for all NP relations that can be verified in either bounded depth, or bounded space. Jumping ahead, we remark that our results can also be interpreted as interactive PCPs (see Remark I.3).

[6] We emphasize that machine verifying the relation is allowed read-many access to the witness (in contrast to the much more restricted complexity class NL in which the verifier has read-once access to the witness). For further discussion, see [25, Section 5.3.1].

[7] For all but 2 of Karp's 21 problems, it is straightforward to see that their natural NP relation can be verified in bounded space. The two exceptions are: *feedback vertex set* and *feedback arc set*. In these two problems, given a graph G one needs to decide whether there is a small set of vertices (resp., edges) whose removal makes the graph acyclic. The natural NP witness for these problems is a specification of the set of vertices (resp., edges) to be removed. While we are unaware of a *deterministic* bounded space algorithm for checking whether a (directed) graph is a-cyclic, it is straightforward to show that this problem is in *co-$\mathcal{NL}$*. Using an unpublished extension [26] of the [16] protocol to $\mathcal{NL}$ (and therefore also *co-$\mathcal{NL}$* [27], [28]) one can obtain short IOPs also for these two problems.

$\gamma, \beta > 0$. *Alternatively, optimizing the communication complexity, we can obtain an* IPCP *in which the first message has length* $(1 + 2^{-(\log m)^{1-\varepsilon_0}}) \cdot n$ *and query complexity* $2^{(\log m)^{1-\varepsilon_0}}$*, and the rest of the communication has length* $2^{(\log m)^{1-\varepsilon_0}}$*, where* m *is the input length,* n *is the witness length, and* $\varepsilon_0 > 0$ *is a small absolute constant (see Corollary II.4). This should be contrasted with the main result of [23] that gives* IPCP *in which the first message has length* $\mathrm{poly}(n)$ *with constant query complexity (in fact, one query suffices), and the rest of the communication has length* $\mathrm{polylog}(n)$.

2) Interactive oracle proofs of proximity with sublinear proof length: Loosely speaking, proofs of proximity are proof-systems in which the verifier runs in *sublinear* time. Since the verifier cannot even read its own input, we only require that she rejects inputs that are *far* from the language. Various models of proofs of proximity have been considered in the literature, depending on the type of communication with the prover. In particular, PCP-style access [31], [32], interactive proofs [33], [34], non-interactive proofs [35], as well as an IOP variant [11], [16].

As a technical step toward proving Theorem 2, we also construct short *interactive oracle proofs of proximity* (IOPP). In an IOPP, the verifier is given oracle access to an implicit input w and is allowed to communicate with an all powerful but untrusted prover (who sees all of w). In each round of the interaction the prover sends a long message and the verifier can choose to read a few of the bits of the message, as well as a few bits of w. At the end of the interaction the verifier should accept if w belongs to the language and should reject (with high probability) if w is *far* from the language (no matter what the prover does).

Theorem 3 (Informally Stated, see Corollary II.6)**.** *Let $\mathcal{L}$ be a language computable in polynomial-time and bounded-polynomial space. Then, for any constant* $\beta, \gamma > 0$ *there exists an* IOPP *for $\mathcal{L}$ with communication complexity* $\gamma \cdot n$ *and constant query and round complexities. The verifier's running time is* n^{β}*, and the prover's running time is* $\mathrm{poly}(n)$.

We remark that the communication complexity in Theorem 3 is strictly less than the input length n. At first glance this may seem quite surprising as, by the aforementioned work [30] (see also [36]), we do not expect IOPs for NP with communication that is shorter than the witness length. Indeed, the key difference which enables such short communication is the fact that Theorem 3 is for *deterministic* languages.

We remark that Theorem 3 is optimal in several ways. First, the work of Kalai and Rothblum [37] implies that there exists a language $\mathcal{L}^* \in \mathsf{Logspace}$ such that any IOP for $\mathcal{L}^*$ with communication complexity $o(n)$ must have query complexity $\omega(1)$, under a strong but plausible cryptographic assumption (i.e., exponentially hard cryptographic pseudorandom generators computable in logarithmic space). Second, a bound on the space complexity of $\mathcal{L}$ is inherent under the widely believed assumption that P is not contained in $\mathsf{SPACE}(\tilde{O}(n))$ (c.f., [38, Theorem 1.4]).

Remark I.4. *Interestingly, since the proof length is strictly less than the input size, Theorem 3 is non-trivial even if we ignore the fact the verifier only reads a small part of the proof. Thus, the result can also be viewed as an* interactive proof of proximity *(*IPP*) [33], [34]. As a point of comparison, note that [16], [34] also construct* IPP*s with sublinear communication for bounded-space computations, but in a different parameter regime: the result in [16], [34] has much smaller communication complexity (e.g.,* $O(\sqrt{n})$*) but on the other hand it does not support constant query complexity as in Theorem 3.*

B. Techniques

Next we give an overview of the proof of Theorem 2 (from which Theorem 1 follows, see Remark I.2). As a warmup, we focus here on a high level overview of a short IOP for 3SAT (i.e., rather than all bounded space relations) and with only some non-trivial query complexity (i.e., $O(\sqrt{n})$ rather than constant). Later, in Section I-B1, we discuss the additional steps needed to generalize to any bounded-space relation and with constant query complexity.

Let $\phi : \{0,1\}^n \to \{0,1\}$ be a 3CNF formula and let $\gamma > 0$. We construct an IOP for proving that ϕ is satisfiable with communication complexity $(1+\gamma) \cdot n$ and query complexity $O(\sqrt{n})$ (note that the witness here is the satisfying assignment, and so the witness length is n). To construct the IOP, our starting point is the following rough outline shared by most PCP and IOP constructions:

1) The prover provides an error-corrected encoding of the computation (either as part of the PCP or as the first message of the IOP).
2) The error-correcting code is chosen so that there is way for the verifier to check that the given alleged codeword is actually a valid codeword. For example, this can be done if the above code is *locally*

testable and *(relaxed) locally correctable*,[8] or by providing an auxiliary proof that the codeword is close to being valid (as in [6]).

3) Lastly, the PCP/IOP is designed to have a mechanism for ensuring that the message encoded within the codeword - an alleged computation - is indeed a valid and accepting computation.

Our construction also shares this basic schema but departs significantly in the details. Let us focus first on Step 1. The first main difference here is that in our construction we only provide an encoding of the NP *witness* (i.e., the satisfying assignment in the case of 3SAT) rather than the entire computation (which includes also the values of all the clauses in ϕ). Intuitively, this makes our job harder when handling Step 3. The second main difference, on which we elaborate next, is that the code that we use in Step 1 is a *high rate code* which is not a *multiplication code*.

Multiplication codes and how to avoid[9] *them:* Loosely speaking, a multiplication code [39], [40] is a linear error correcting code C, over a finite field $\mathbb{F}$, so that the set $\{c_1 \star c_2 : c_1, c_2 \in C\}$, where $c_1 \star c_2$ denotes entry-wise multiplication (over $\mathbb{F}$), is itself a code (i.e., it has large relative distance). The archtypical example of a multiplication code is the Reed-Solomon code (since the product of two sufficiently low degree polynomials is a low degree polynomial). As elegantly articulated in the works of Meir [39], [40], the multiplication property is leveraged in PCP and IOP constructions to facilitate checking of *non-linear* relations that arise in the computation (e.g., verifying correctness of AND gates).

Unfortunately, all known multiplications codes (e.g., the Reed-Solomon code [41], the Reed-Muller code [42], [43], low rate tensor codes [39], [40] or AG codes [15], [44]) have rate less than $1/2$, and this is inherent [45]. Since we are aiming for rate close to 1, we cannot afford to encode the entire computation using such a code.

Instead, we encode the satisfying assignment w using a *high rate* binary code $C : \{0,1\}^n \to \{0,1\}^{(1+\gamma/2)\cdot n}$ with constant relative distance (which is not a multiplication code). Beyond having high rate and constant relative distance, we will also need for C to be (1) locally testable, (2) (relaxed) locally correctable, and (3) support a certain "sumcheck-like property", elaborated on below.

It turns out that the tensor product of a high-rate binary code of constant relative distance satisfies all the aforementioned properties. Given a linear code $C_0 : \{0,1\}^k \to \{0,1\}^m$, consider the (two-dimensional) *tensor product code* $C_0 \otimes C_0 : \{0,1\}^{k\times k} \to \{0,1\}^{m\times m}$. Recall that the codewords of $C_0 \otimes C_0$ are $m \times m$ matrices with the constraints that both the rows and columns are all codewords of the base code C_0.

It is well-known that the tensor product operation squares the rate and the relative distance of the base code. In particular, if the base code has high-rate and constant relative distance, then so does its tensor product. Moreover, a recent line of work [46]–[50] has established the local testability and relaxed local correctability of high-rate tensor codes.[10] Hence, we take $C : \{0,1\}^n \to \{0,1\}^{(1+\gamma/2)\cdot n}$ to be a tensor product of some high-rate base code C_0. Given a satisfying assignment $w \in \{0,1\}^n$ for ϕ, the prover in our IOP first sends $C(w)$ (which has length $(1+\gamma/2)\cdot n$). Next, addressing Step 2 in the outline, we observe that C is indeed locally testable and relaxed locally correctable with query complexity $O(\sqrt{n})$.

Thus, we are left with the question of how to implement Step 3, which is the key technical challenge. This is where we use the "sumcheck-like" property of our code C. To explain our approach we first take a brief detour into the classical method for constructing PCPs due to [2], [53]. Our presentation follows [54].

A quick recap of classical PCP *techniques:* Imagine momentarily that the prover can provide an encoding of the satisfying assignment w under the *low degree extension code* (a variant of the Reed Muller code). This code is very "PCP-friendly" but has very poor rate.

In more detail, let $\mathbb{F}$ be a finite field of size $|\mathbb{F}| \gg \log(n)$ and let $\hat{w} : \mathbb{F}^{\log(n)} \to \mathbb{F}$ be the unique multilinear polynomial such that for every $i \in \{0,1\}^{\log(n)}$ it holds that $\hat{w}(i) = w_i$, where we identify $\{0,1\}^{\log(n)}$ with $[n]$ in the natural way. The existence of such a (unique) polynomial, referred to as the *multilinear extension of* w, basically follows from interpolation, see the full version [1] for details. Observe that the truth table of

[8] Informally, a code is said to be locally testable if, given a string w, it is possible to determine whether w is a codeword of C, or far from all codewords in C, by reading only a small part of w. A code is said to be locally correctable if, given a codeword c that has been partially corrupted by some errors, it is possible to recover any coordinate of c by reading only a small part of the corrupted version of c. Finally, in relaxed local correction, the local corrector is additionally allowed to abort whenever a corruption is detected.

[9] Actually, our construction *does* use multiplication codes (specifically the Reed-Muller or Low Degree Extension code). What we avoid is explicitly sending an encoding of the computation (or witness) via a multiplication code.

[10] Local testability actually requires the dimension of the tensor product to be at least 3 [51], [52]. For simplicity we ignore this fact in this high-level overview.

$\hat{w}$ has *super-polynomial* length and so we cannot afford for the prover to send $\hat{w}$. Nevertheless, let us assume for now that the verifier also has oracle access to $\hat{w}$.

Consider the polynomial $P : (\mathbb{F}^{\log(n)})^3 \times \mathbb{F}^3 \to \mathbb{F}$ of total degree $O(\log n)$ defined as:

$$\begin{aligned} P(i_1, i_2, i_3, b_1, b_2, b_3) = \hat{I}_\phi(i_1, i_2, i_3, b_1, b_2, b_3) \\ \cdot \Big(\hat{w}(i_1) - b_1\Big) \\ \cdot \Big(\hat{w}(i_2) - b_2\Big) \\ \cdot \Big(\hat{w}(i_3) - b_3\Big), \end{aligned} \tag{1}$$

where $\hat{I}_\phi$ is a multilinear extension of a Boolean function I_ϕ that on input $(i_1, i_2, i_3, b_1, b_2, b_3) \in \{0,1\}^{3\log(n)+3}$ outputs 1 if the clause $(x_{i_1} = b_1) \vee (x_{i_2} = b_2) \vee (x_{i_3} = b_3)$ appears in ϕ and otherwise outputs 0. The significance of P is that it has the following easy-to-verify property: P is identically 0 in the hypercube $\{0,1\}^{3\log(n)+3}$ if and only if $\hat{w}$ corresponds to a satisfying assignment for ϕ.

Thus, we only need to check that indeed $P|_{\{0,1\}^{3\log(n)+3}} \equiv 0$. This can be done using a variant of the classical (interactive) sumcheck protocol [55].[11] More specifically, and without getting into the details, there exists a constant round interactive proof for checking whether a given low degree polynomial $Q : \mathbb{F}^m \to \mathbb{F}$ is identically 0 in $\{0,1\}^m$. Most importantly, the verifier in the protocol only needs oracle access to Q and moreover, only makes a single query to Q.

It is very instructive to think of the above protocol as an *interactive reduction* - a central notion in our work. Generally speaking, in an interactive reduction a (computationally) complex claim about an input w is reduced, by interaction with a prover, to a much simpler claim about w. Completeness means that if the original claim was true then the honest prover will make the verifier generate a true (and simpler) claim, whereas soundness means that if the original claim was false, then no matter what the prover does, with high probability, either the verifier rejects or it generates a false claim. Since the claim has been simplified, intuitively, progress has been made.

We emphasize that in an interactive reduction the verifier doesn't get any form of access to the input w - it merely reduces the complexity of claims about w, without ever seeing it. For example, the above protocol (for checking whether the polynomial Q vanishes on the hypercube) can be viewed as an interactive reduction from a claim about 2^m values of Q to a claim about a single value.

Using this reduction, applied to the polynomial P, the prover and verifier can reduce the satisfiability of the formula ϕ to a claim about a single point of the polynomial P. The verifier can now directly check this claim, via Eq. (1), by making three queries to $\hat{w}$.[12]

Taking a step back, what we started off with was a claim that w is a satisfying assignment for ϕ and we ended up with claims about three particular values of $\hat{w}$. Thus, we can view this entire process itself as an interactive reduction from the claim that the assignment w satisfies the formula ϕ to claims about three specific points of its low degree extension $\hat{w}$. We emphasize that in this interactive reduction, which we denote by Π_{reduce}, the verifier only needs to get ϕ and doesn't need any form of access to w.

Back to our IOP *construction:* If we could afford for the prover to send $\hat{w}$ then at this point we would be done - after sending $\hat{w}$, the prover and verifier run Π_{reduce} and then the verifier checks the three claims about $\hat{w}$ by reading the three corresponding points from the prover's first message (using also the local testability and correctability of the low degree extension).

Alas, in our actual IOP the prover can only send a high-rate encoding $C(w)$ and cannot afford to send $\hat{w}$. Still, we can use Π_{reduce} to our advantage. In particular, after the prover sends $C(w)$, the two parties run Π_{reduce}. The reduction generates claims about three values of $\hat{w}$. We are now faced with a (potentially) simpler task. Given oracle access to $C(w)$, we merely need to check the three claims about $\hat{w}$. For simplicity let us focus on one of these three claims, that is, a claim of the form $\hat{w}(z) = b$ (where $z \in \mathbb{F}^{\log(n)}$ and $b \in \mathbb{F}$).

It is natural to wonder at this point whether checking that $\hat{w}(z) = b$, given oracle access to $C(w)$, is any simpler than checking that $\phi(w) = 1$. We would like to argue that the answer is affirmative. In particular, since the low degree extension is a *linear* code (over the field $\mathbb{F}$), the claim $\hat{w}(z) = b$ is *linear* - i.e., it can be rephrased as a claim of the form $\langle \lambda_z, w \rangle = b$, for some $\lambda_z \in \mathbb{F}^n$ (that depends only on $z \in \mathbb{F}^{\log(n)}$).

Thus, we need a procedure for checking a linear claim about w, given oracle access to $C(w)$. Observing that C is a tensor code, a natural approach is to use the classical sumcheck protocol. Recall that the sumcheck protocol is an interactive proof for computing $\sum_{i \in [n]} w_i$, given oracle access to $C(w)$. While the protocol was originally designed specifically for the

[11] Since we aim for query complexity $O(\sqrt{n})$, we can use a constant-round variant of sumcheck with communication $O(\sqrt{n})$.

[12] Here we also use the fact that the verifier can compute $\hat{I}_\phi$ by itself in $\tilde{O}(n)$ time, see the full version [1] for details.

low degree extension code, it was later abstracted by Meir [39], who showed that it can be applied to any tensor code.

The discussion so far comes close to resolving our problem. The difficulty that remains is that we would like to check that $\langle \lambda_z, w \rangle = b$ whereas the sumcheck protocol supports claims of the form $\langle \mathbf{1}, w \rangle = b$, where $\mathbf{1}$ is the all 1's vector. That is, the sumcheck protocol seems limited to the particular linear claim in which all coefficients are equal to 1. Unfortunately, the linear claim in question (corresponding to the vector λ_z) does not have this form.

Before proceeding, we remark that if C were a *multiplication* code then we could have easily handled this difficulty be applying the sumcheck protocol to the codeword $C(w) \star C(\lambda_z)$.

Sumcheck for rank 1 *tensor coefficients:* While we do not know how to extend the sumcheck protocol to handle linear claims with arbitrary coefficients, we show that it is possible to extend it to handle a particular form of coefficient structure. Luckily, the vector λ_z has a suitable form.

More specifically, we show how to extend the sumcheck protocol to computing linear claims of the form $\langle \lambda, w \rangle$ for any $\lambda \in \mathbb{F}^n$ which corresponds to a rank 1 matrix of dimension $\sqrt{n} \times \sqrt{n}$ (or more generally to any rank 1 tensor). That is, we assume that there exist $\lambda^{(1)}, \lambda^{(2)} \in \mathbb{F}^{\sqrt{n}}$ such that $\lambda = \lambda^{(1)} \otimes \lambda^{(2)}$ (where $\otimes$ denotes the tensor product, and we view λ simultaneously as a vector in $\mathbb{F}^n$ and as a matrix in $\mathbb{F}^{\sqrt{n}} \times \mathbb{F}^{\sqrt{n}}$ in the natural way). The fact that λ_z has this structure follows from the fact that the low degree extension is itself a tensor code.

Thus, we would like to use the sumcheck protocol to compute $\langle \lambda^{(1)} \otimes \lambda^{(2)}, w \rangle$. Let $C_0 : \{0,1\}^{n_0} \to \{0,1\}^{n'_0}$ be a systematic linear code such that $C = C_0 \otimes C_0$. Thus, $n_0 = \sqrt{n}$ and $n'_0 = \sqrt{(1+\gamma/2) \cdot n}$. Let $c = C(w)$. We view c as an $n'_0 \times n'_0$ dimensional matrix in the natural way, and denote its (i,j)-th entry by $c_{i,j}$.

In our sumcheck variant, the (honest) prover sends the message $\pi \in \mathbb{F}^{n'_0}$ which is defined as $\pi_i = \sum_{j \in [n_0]} \lambda^{(2)}_j \cdot c_{i,j}$, for every $i \in [n'_0]$. In other words, π is computed by taking a linear combination of the first n_0 columns of c, with coefficients corresponding to $\lambda^{(2)}$.

At first glance it may seem as though π is a codeword of C_0. This is actually not true since C_0 is linear over the field $\mathbb{GF}(2)$ whereas we are using coefficients in a different (and larger) field $\mathbb{F}$. Nevertheless, if we choose $\mathbb{F}$ to be an extension field of $\mathbb{GF}(2)$, then with some elementary algebraic manipulations, we can show that π can be decomposed into $\log_2(|\mathbb{F}|)$ codewords of C_0.

The verifier, given a string $\tilde{\pi}$ which is allegedly equal to π, first checks that $\tilde{\pi}$ indeed consists of the aforementioned $\log(|\mathbb{F}|)$ codewords and rejects otherwise. Since both π and $\tilde{\pi}$ are composed of codewords, this test ensures us that if they differ then they must differ on a constant fraction of coordinates.

The verifier then chooses a random $i^* \in [n'_0]$ and checks that $\tilde{\pi}_{i^*} = \sum_{j \in [n_0]} \lambda^{(2)}_j \cdot c_{i^*,j}$ by reading the i^*-th row of C. Assuming that the prover sent $\tilde{\pi} \neq \pi$, with constant probability over the choice of i^* it holds that $\tilde{\pi}_{i^*} \neq \pi_{i^*} = \sum_{j \in [n_0]} \lambda^{(2)}_j \cdot c_{i^*,j}$ and so the verifier rejects. This probability can be amplified by choosing a suitably large constant number of random i^*'s. Note that verifier only reads a constant number of rows from c and so the number of queries to c is $O(n'_0) = O(\sqrt{n})$.

Since we can now assume that the prover actually sent π, the verifier can simply output $\sum_{i \in [n_0]} \lambda^{(1)}_i \cdot \pi_i = \sum_{i,j \in [n_0]} \lambda^{(1)}_i \cdot \lambda^{(2)}_j \cdot c_{i,j} = \langle \lambda^{(1)} \otimes \lambda^{(2)}, w \rangle$ as desired.

This concludes the description of the warmup. Observe that the communication complexity is $(1+\gamma/2) \cdot n + \tilde{O}(\sqrt{n}) \leq (1+\gamma) \cdot n$ and the query complexity is $O(\sqrt{n})$ as promised.

A brief digest: Interactive code switching: The key idea in the above proof is that, using interaction, we are able to "switch" between different tensor codes during the protocol. More specifically, we implicitly use a (low-rate) multiplication code to check correctness of the computation, but are able to use a high-rate tensor code for actually encoding the witness. The key facilitator for switching between these codes is the power of the sumcheck protocol.

This approach is reminiscent of Meir's [39] combinatorial proof of the IP = PSPACE theorem. In Meir's protocol, which is an abstraction of the [24] protocol, claims about larger tensor codewords are reduced to claims about smaller tensor codewords via the sumcheck protocol.

1) Additional steps from warmup to Theorem 2:

Constant query complexity: The approach outlined so far only yields an IOP with $O(\sqrt{n})$ query complexity. To obtain a result with *constant* query complexity we follow the usual route - query reduction via composition [4]. In more detail, rather than actually performing the queries we will ask the prover to provide a constant query PCP, or more precisely a PCP of proximity (PCPP), that the verifier would have accepted had it read these queries. Since the PCPP is applied to an input of length $O(\sqrt{n})$ (or more accurately to a computation of size $\tilde{O}(\sqrt{n})$) we can afford to use existing constant-query PCPP constructions (e.g., those with poly-logarithmic overhead [6], [7]). A similar type of IOP composition was used also in [17] and as in their work, we utilize interaction to pay only linearly in the randomness complexity of the so-called "outer" IOP (rather than exponentially as in standard PCP composition).

Actually making this idea go through is somewhat more technically invloved. For example, we need to ensure that our base IOP (with $O(\sqrt{n})$ query complexity) is *robust* (i.e., the verifier cannot be made to accept even if the prover is allowed to flip a constant fraction of its answers a posteriori). We also need the outer IOP verifier to run in sublinear time. We achieve this by making also the *outer* IOP an IOPP. We defer the details to the technical sections.

Extending to general bounded-space computations: When trying to extend our approach past 3SAT, we observe that the main property that we used is that 3SAT has an interactive reduction to a linear claim about the witness, where the linear claim has a rank 1 tensor structure.

Using the doubly-efficient interactive proofs of Reingold *et al.* [16] we show that a similar statement holds for any NP relation computable in polynomial-time and bounded polynomial-space. This basically follows from the fact that the verifier in [16] runs in sublinear time given access to the low degree extension of its input. Plugging in the [16] protocol instead of Π_{reduce} lets us obtain a high-rate IOP for any non-deterministic bounded space computation, thereby proving Theorem 2.

Remark I.5. *One can replace the [16] protocol with the doubly efficient interactive proof-system of Goldwasser et al. [24] to obtain an* IOP *approaching the witness length also for* NP *relations that can be verified in small* depth *(rather than small space). However, the resulting* IOP *only has poly-logarithmic query complexity due to the poly-logarithmic number of rounds in [24].*

C. Open problems

Shorter PCP*s:* As mentioned above, the work of [22] shows that SAT does not have a PCP with length that is a fixed polynomial in the witness size (let alone proof length approaching the witness length), unless NP $\subseteq$ co-NP/poly. This still leaves open the possibility that an interesting sub-class of NP relations has such short proofs. Somewhat along this vein, a recent work of Ben Eliezer *et al.* [56] constructs a PCPP with length $n \cdot (\log n)^{o(1)}$ but for a very specific problem. In particular, while we have constructed nearly optimal IOPs for CircuitSAT, the question of whether CircuitSAT has a linear length constant-query PCP remains wide open.

A major difficulty in trying to adapt our approach (as well as previous approaches) is that we apply an *interactive* version of the sumcheck protocol that has sub-linear communication. In contrast, the only way in which we know how to make the sumcheck protocol non-interactive (i.e., by specifying the answers of all possible queries of the verifier) leads to super-linear proof length.

A starting point may be to try to obtain PCPs for CircuitSAT of proof length approaching t with *any* non-trivial query complexity (we note that while [15] constructed a PCP for CircuitSAT of length $O(t)$ with non-trivial query complexity, the hidden constant in the $O(\cdot)$ notation is very large). A positive answer to this question would have to bypass the use of multiplication codes in a fundamentally different way than in our protocol (which capitalizes on interaction). Moreover, we note that for the case of locally testable and decodable codes, a key for reducing the query complexity of linear length locally testable and decodable codes was indeed to first construct such codes of rate approaching 1 and non-trivial sublinear query complexity [49].

Hardness of approximation: A major application of PCPs is showing *hardness of approximation* for central optimization problems (see, e.g., [3], [57]). A fascinating open question, pointed out by [17], [58], is whether IOPs may have similar implications. In particular, a major barrier in applying the traditional PCP methodology for obtaining hardness of approximation results in *fine-grained complexity* is the overhead in the length of existing PCPs. For example, to show that GapSAT is $2^{(1-\varepsilon)\cdot n}$ hard, one would need a PCP with overhead $1+\varepsilon'$. (In particular, note that a PCP of length, say, $100n$ would only roughly yield hardness of $2^{n/100}$.) Interestingly, recent breakthrough results bypassed this barrier by relying on interactive proof machinery [58]–[60]. It is natural to ask whether short IOPs such as those constructed in this work can be used to establish new or improved hardness of approximation results.

II. Formal statement of our results

We first state our IOP results in Section II-A. Then, in Section II-B, we state our IOPP results.

A. IOP *results*

Our main result is an IOP for every NP relation that can be verified in bounded space. The general statement is given in Theorem II.1 below, which gives tradeoffs depending on parameters β (which offers a tradeoff between number of rounds and total communication) and γ (which offers a tradeoff between the rate and the query, communication and verification complexities). Since the statement of Theorem II.1 is somewhat involved, it may be useful for the first reading to skip directly to Corollary II.3 which considers a particularly interesting setting of the parameters.

In the following we say that a function $\alpha = \alpha(n) \in (0,1)$ is nice if it is computable in time $\mathrm{polylog}(n)$ and $\alpha(\Theta(n)) = \Theta(\alpha(n))$ (e.g., $\alpha(n) = 1/\log(n)$ is nice).

Theorem II.1 (IOP for NP)**.** *Let $\mathcal{L} \in \mathsf{NP}$ with corresponding relation $\mathcal{R}_{\mathcal{L}}$ in which the instances have length m and witnesses have length n, where n and m are polynomially related, and such that $\mathcal{R}_{\mathcal{L}}$ can be decided in time* $\mathrm{poly}(n)$ *and space $s \geq \log(n)$. Also, we assume that $m \geq n$ (i.e., instances are not shorter than their corresponding witnesses).*[13]

Let $\gamma = \gamma(m) \in (0,1)$ and $\beta = \beta(m) \in (0,1)$ be nice functions such that $\mathrm{poly}(1/\beta) \leq \log(n)$ *and $\gamma \geq m^{-O(\beta)}$. Then, there exists a $\beta^{-O(1/\beta)}$-round* IOP *for $\mathcal{L}$ with soundness error $1/2$. The query complexity is $(\gamma\beta)^{-O(1/\beta)}$ and the communication consists of a first (deterministic) message sent by the prover of length $(1+\gamma) \cdot n$ bits followed by* $\mathrm{poly}\left(m^{\beta}, (\gamma\beta)^{-1/\beta}, s\right)$ *additional communication. The* IOP *verifier runs in time* $\tilde{O}(m) + \mathrm{poly}\left(m^{\beta}, (\gamma\beta)^{-1/\beta}, s\right)$ *and the* IOP *prover runs in time* $\mathrm{poly}(m)$.

Remark II.2. *The soundness error in Theorem II.1 can be reduced by parallel repetition, while observing that since the first prover message is deterministic, it does not to be repeated (note that in a typical setting of parameters the first prover message in Theorem II.1 is by far the largest part of the communication).*

A particularly interesting setting of parameters is when $\gamma > 0$ is an arbitrarily small constant, and $\beta > 0$ is a sufficiently small constant. In this regime we obtain the following corollary from Theorem II.1.

Corollary II.3. *There exists a fixed constant $\xi > 0$ such that the following holds. Let $\mathcal{L} \in \mathsf{NP}$ be as in Theorem II.1 with $s = s(n) \leq n^{\xi}$. Then, for any constant $\gamma, \varepsilon > 0$ there exists an* IOP *for $\mathcal{L}$ with communication complexity $(1+\gamma) \cdot n$, constant query complexity, constant round complexity, and soundness error ε. The verifier runs in time $\tilde{O}(m)$ and the prover runs in time* $\mathrm{poly}(m)$.

We also state another corollary of Theorem II.1, focusing on minimizing the communication complexity following the first prover's message. Specifically, letting $\beta(m) = \frac{1}{(\log m)^{\Theta(\varepsilon_0)}}$ and $\gamma(m) = 2^{-(\log m)^{1-\Theta(\varepsilon_0)}}$ we obtain the following.

Corollary II.4. *There exists an absolute constant $\varepsilon_0 > 0$ such that the following holds. Let $\mathcal{L} \in \mathsf{NP}$ be as in Theorem II.1 with $s = s(n) \leq 2^{(\log m)^{1-\varepsilon_0}}$. Then, there exists a $2^{(\log m)^{\varepsilon_0}}$-round* IOP *for $\mathcal{L}$ with soundness error $1/2$. The query complexity is $2^{(\log m)^{1-\varepsilon_0}}$, and the communication consists of a first (deterministic) message sent by the prover of length $(1 + 2^{-(\log m)^{1-\varepsilon_0}}) \cdot n$, followed by $2^{(\log m)^{1-\varepsilon_0}}$ additional communication. The* IOP *verifier runs in time $\tilde{O}(m)$ and the* IOP *prover runs in time* $\mathrm{poly}(m)$.

B. IOPP *results*

We next state our IOPP results. Our main result is an IOPP for bounded space computations in which the communication complexity is slightly less than n.

Theorem II.5 (IOPP for bounded-space computations)**.** *Let $\mathcal{L}$ be a language computable in time* $\mathrm{poly}(n)$ *with space $s = s(n) \geq \log n$. Then for every $\delta = \delta(n) \in (0,1)$, $\beta = \beta(n) \in (0,1)$ and $\gamma = \gamma(n) \in (0,1)$ such that* $\mathrm{poly}(1/\beta) \leq \log(n)$ *and* $\gamma = \gamma(n) \geq \frac{200 \cdot 4^{1/\beta} \cdot \log n}{n^{\beta/2}}$ *the following holds.*

There exists a $\beta^{-O(1/\beta)}$-round IOPP *for $\mathcal{L}$ with respect to proximity parameter δ, and with soundness error $1/2$. The query complexity is* $\mathrm{poly}\left((\gamma\beta)^{-1/\beta}, 1/\delta\right)$, *and the communication consists of a first (deterministic) message sent by the prover of length $\gamma \cdot n$ bits followed by* $\mathrm{poly}\left(n^{\beta}, (\gamma\beta)^{-1/\beta}, 1/\delta, s\right)$ *additional communication. The* IOP *verifier runs in time* $\mathrm{poly}\left(n^{\beta}, (\gamma\beta)^{-1/\beta}, 1/\delta, s\right)$, *and the* IOP *prover runs in time* $\mathrm{poly}(n)$.

As in the IOP case, letting $\delta, \gamma > 0$ be arbitrary constants and $\beta > 0$ be a sufficiently small constant, we obtain the following constant query IOPP.

Corollary II.6. *For any $\beta > 0$ there exists a constant $\varepsilon_0 > 0$ such that the following holds. Let $\mathcal{L}$ be a language computable in time* $\mathrm{poly}(n)$ *and space n^{ε_0}. Then, for any constants $\delta, \gamma, \varepsilon > 0$ there exists an* IOPP *for $\mathcal{L}$, wrt proximity parameter δ, with communication complexity $\gamma \cdot n$, constant query complexity, constant round complexity, and soundness error ε. The verifier runs in time n^{β}, and the prover runs in time* $\mathrm{poly}(n)$.

Acknowledgements

We thank Ale Chiesa, Yuval Ishai, Swastik Kopparty, Or Meir and Eylon Yogev for useful discussions.

[13] This requirement can be handled by simply padding the input with 0's if necessary. This increases the input size by at most n.

The authors are supported by the Milgrom family grant for Collaboration between the Technion and University of Haifa. Noga Ron-Zewi is also supported by the Israeli Science Foundation (Grant No. 735/20). Ron D. Rothblum is also supported by the Israeli Science Foundation (Grants No. 1262/18 and 2137/19), and the Technion Hiroshi Fujiwara cyber security research center and Israel cyber directorate.

REFERENCES

[1] N. Ron-Zewi and R. Rothblum, "Local proofs approaching the witness length," *Electronic Colloquium on Computational Complexity (ECCC)*, vol. 26, p. 127, 2019. [Online]. Available: https://eccc.weizmann.ac.il/report/2019/127

[2] L. Babai, L. Fortnow, L. A. Levin, and M. Szegedy, "Checking computations in polylogarithmic time," in *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing, May 5-8, 1991, New Orleans, Louisiana, USA*, 1991, pp. 21–31. [Online]. Available: https://doi.org/10.1145/103418.103428

[3] U. Feige, S. Goldwasser, L. Lovász, S. Safra, and M. Szegedy, "Interactive proofs and the hardness of approximating cliques," *J. ACM*, vol. 43, no. 2, pp. 268–292, 1996. [Online]. Available: https://doi.org/10.1145/226643.226652

[4] S. Arora and S. Safra, "Probabilistic checking of proofs: A new characterization of NP," *J. ACM*, vol. 45, no. 1, pp. 70–122, 1998. [Online]. Available: https://doi.org/10.1145/273865.273901

[5] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy, "Proof verification and the hardness of approximation problems," *J. ACM*, vol. 45, no. 3, pp. 501–555, 1998. [Online]. Available: https://doi.org/10.1145/278298.278306

[6] E. Ben-Sasson and M. Sudan, "Short PCPs with polylog query complexity," *SIAM J. Comput.*, vol. 38, no. 2, pp. 551–607, 2008. [Online]. Available: https://doi.org/10.1137/050646445

[7] I. Dinur, "The PCP theorem by gap amplification," *J. ACM*, vol. 54, no. 3, p. 12, 2007. [Online]. Available: https://doi.org/10.1145/1236457.1236459

[8] N. Pippenger and M. J. Fischer, "Relations among complexity measures," *J. ACM*, vol. 26, no. 2, pp. 361–381, 1979. [Online]. Available: https://doi.org/10.1145/322123.322138

[9] J. Kilian, "A note on efficient zero-knowledge proofs and arguments (extended abstract)," in *Proceedings of the 24th Annual ACM Symposium on Theory of Computing, May 4-6, 1992, Victoria, British Columbia, Canada*, 1992, pp. 723–732. [Online]. Available: https://doi.org/10.1145/129712.129782

[10] S. Micali, "Computationally sound proofs," *SIAM J. Comput.*, vol. 30, no. 4, pp. 1253–1298, 2000. [Online]. Available: https://doi.org/10.1137/S0097539795284959

[11] E. Ben-Sasson, A. Chiesa, and N. Spooner, "Interactive oracle proofs," in *Theory of Cryptography - 14th International Conference, TCC 2016-B, Beijing, China, October 31 - November 3, 2016, Proceedings, Part II*, 2016, pp. 31–60. [Online]. Available: https://doi.org/10.1007/978-3-662-53644-5_2

[12] I. Dinur, "Mildly exponential reduction from gap 3sat to polynomial-gap label-cover," *Electronic Colloquium on Computational Complexity (ECCC)*, vol. 23, p. 128, 2016. [Online]. Available: http://eccc.hpi-web.de/report/2016/128

[13] P. Manurangsi and P. Raghavendra, "A birthday repetition theorem and complexity of approximating dense csps," in *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, 2017, pp. 78:1–78:15. [Online]. Available: https://doi.org/10.4230/LIPIcs.ICALP.2017.78

[14] B. Applebaum, "Exponentially-hard gap-csp and local PRG via local hardcore functions," in *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, 2017, pp. 836–847. [Online]. Available: https://doi.org/10.1109/FOCS.2017.82

[15] E. Ben-Sasson, Y. Kaplan, S. Kopparty, O. Meir, and H. Stichtenoth, "Constant rate PCPs for circuit-sat with sublinear query complexity," *J. ACM*, vol. 63, no. 4, pp. 32:1–32:57, 2016. [Online]. Available: https://doi.org/10.1145/2901294

[16] O. Reingold, G. N. Rothblum, and R. 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*, 2016, pp. 49–62. [Online]. Available: https://doi.org/10.1145/2897518.2897652

[17] E. Ben-Sasson, A. Chiesa, A. Gabizon, M. Riabzev, and N. Spooner, "Interactive oracle proofs with constant rate and query complexity," in *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, 2017, pp. 40:1–40:15. [Online]. Available: https://doi.org/10.4230/LIPIcs.ICALP.2017.40

[18] E. Ben-Sasson, I. Bentov, Y. Horesh, and M. Riabzev, "Fast reed-solomon interactive oracle proofs of proximity," in *45th International Colloquium on Automata, Languages, and Programming, ICALP 2018, July 9-13, 2018, Prague, Czech Republic*, 2018, pp. 14:1–14:17. [Online]. Available: https://doi.org/10.4230/LIPIcs.ICALP.2018.14

[19] E. Ben-Sasson, A. Chiesa, M. Riabzev, N. Spooner, M. Virza, and N. P. Ward, "Aurora: Transparent succinct arguments for R1CS," in *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part I*, 2019, pp. 103–128. [Online]. Available: https://doi.org/10.1007/978-3-030-17653-2_4

[20] E. Ben-Sasson, I. Bentov, Y. Horesh, and M. Riabzev, "Scalable zero knowledge with no trusted setup," in *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part III*, 2019, pp. 701–732. [Online]. Available: https://doi.org/10.1007/978-3-030-26954-8_23

[21] E. Ben-Sasson, L. Goldberg, S. Kopparty, and S. Saraf, "DEEP-FRI: sampling outside the box improves soundness," *Electronic Colloquium on Computational Complexity (ECCC)*, vol. 26, p. 44, 2019. [Online]. Available: https://eccc.weizmann.ac.il/report/2019/044

[22] L. Fortnow and R. Santhanam, "Infeasibility of instance compression and succinct PCPs for NP," *J. Comput. Syst. Sci.*, vol. 77, no. 1, pp. 91–106, 2011. [Online]. Available: https://doi.org/10.1016/j.jcss.2010.06.007

[23] Y. T. Kalai and R. Raz, "Interactive PCP," in *Automata, Languages and Programming, 35th International Colloquium, ICALP 2008, Reykjavik, Iceland, July 7-11, 2008, Proceedings, Part II - Track B: Logic, Semantics, and Theory of Programming & Track C: Security and Cryptography Foundations*, 2008, pp. 536–547. [Online]. Available: https://doi.org/10.1007/978-3-540-70583-3_44

[24] S. Goldwasser, Y. T. Kalai, and G. N. Rothblum, "Delegating computation: Interactive proofs for muggles," *J. ACM*, vol. 62, no. 4, pp. 27:1–27:64, 2015. [Online]. Available: https://doi.org/10.1145/2699436

[25] O. Goldreich, *Computational complexity - a conceptual perspective*. Cambridge University Press, 2008. [Online]. Available: https://doi.org/10.1017/CBO9780511804106

[26] O. Reingold, G. N. Rothblum, and R. D. Rothblum, 2017, personal Communication.

[27] N. Immerman, "Nondeterministic space is closed under complementation," *SIAM J. Comput.*, vol. 17, no. 5, pp. 935–938, 1988. [Online]. Available: https://doi.org/10.1137/0217058

[28] R. Szelepcsényi, "The moethod of focing for nondeterministic automata," *Bulletin of the EATCS*, vol. 33, pp. 96–99, 1987.

[29] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of Computer Computations*, R. E. Miller and J. W. Thatcher, Eds. New York: Plenum Press, 1975, pp. 85–103.

[30] O. Goldreich and J. Håstad, "On the complexity of interactive proofs with bounded communication," *Inf. Process. Lett.*, vol. 67, no. 4, pp. 205–214, 1998. [Online]. Available: https://doi.org/10.1016/S0020-0190(98)00116-1

[31] E. Ben-Sasson, O. Goldreich, P. Harsha, M. Sudan, and S. P. Vadhan, "Robust PCPs of proximity, shorter PCPs, and applications to coding," *SIAM J. Comput*, vol. 36, no. 4, pp. 889–974, 2006.

[32] I. Dinur and O. Reingold, "Assignment testers: Towards a combinatorial proof of the PCP theorem," *SIAM J. Comput.*, vol. 36, no. 4, pp. 975–1024, 2006. [Online]. Available: https://doi.org/10.1137/S0097539705446962

[33] F. Ergün, R. Kumar, and R. Rubinfeld, "Fast approximate probabilistically checkable proofs," *Inf. Comput.*, vol. 189, no. 2, pp. 135–159, 2004. [Online]. Available: https://doi.org/10.1016/j.ic.2003.09.005

[34] G. N. Rothblum, S. P. Vadhan, and A. Wigderson, "Interactive proofs of proximity: delegating computation in sublinear time," in *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, 2013, pp. 793–802. [Online]. Available: https://doi.org/10.1145/2488608.2488709

[35] T. Gur and R. D. Rothblum, "Non-interactive proofs of proximity," *Computational Complexity*, vol. 27, no. 1, pp. 99–207, 2018. [Online]. Available: https://doi.org/10.1007/s00037-016-0136-9

[36] O. Goldreich, S. P. Vadhan, and A. Wigderson, "On interactive proofs with a laconic prover," *Computational Complexity*, vol. 11, no. 1-2, pp. 1–53, 2002. [Online]. Available: https://doi.org/10.1007/s00037-002-0169-0

[37] Y. T. Kalai and R. D. Rothblum, "Arguments of proximity - [extended abstract]," in *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part II*, 2015, pp. 422–442. [Online]. Available: https://doi.org/10.1007/978-3-662-48000-7_21

[38] O. Goldreich, "On doubly-efficient interactive proof systems," *Foundations and Trends in Theoretical Computer Science*, vol. 13, no. 3, pp. 158–246, 2018. [Online]. Available: https://doi.org/10.1561/0400000084

[39] O. Meir, "IP = PSPACE using error-correcting codes," *SIAM J. Comput.*, vol. 42, no. 1, pp. 380–403, 2013. [Online]. Available: https://doi.org/10.1137/110829660

[40] ——, "Combinatorial PCPs with efficient verifiers," *Computational Complexity*, vol. 23, no. 3, pp. 355–478, 2014. [Online]. Available: https://doi.org/10.1007/s00037-014-0080-5

[41] I. S. Reed and G. Solomon, "Polynomial codes over certain finite fields," *SIAM Journal of the Society for Industrial and Applied Mathematics*, vol. 8, no. 2, pp. 300–304, 1960.

[42] D. E. Muller, “Application of boolean algebra to switching circuit design and to error detection,” *Trans. I.R.E. Prof. Group on Electronic Computers*, vol. 3, no. 3, pp. 6–12, 1954. [Online]. Available: https://doi.org/10.1109/IREPGELC.1954.6499441

[43] I. S. Reed, “A class of multiple-error-correcting codes and the decoding scheme,” *Trans. of the IRE Professional Group on Information Theory (TIT)*, vol. 4, pp. 38–49, 1954. [Online]. Available: https://doi.org/10.1109/TIT.1954.1057465

[44] H. Stichtenoth, “Transitive and self-dual codes attaining the tsfasman-vla/spl breve/dut$80-zink bound,” *IEEE Trans. Information Theory*, vol. 52, no. 5, pp. 2218–2224, 2006.

[45] H. Randriambololona, “An upper bound of singleton type for componentwise products of linear codes,” *IEEE Trans. Information Theory*, vol. 59, no. 12, pp. 7936–7939, 2013. [Online]. Available: https://doi.org/10.1109/TIT.2013.2281145

[46] E. Ben-Sasson and M. Sudan, “Robust locally testable codes and products of codes,” *Random Structures and Algorithms*, vol. 28, no. 4, pp. 387–402, 2006. [Online]. Available: http://dx.doi.org/10.1002/rsa.20120

[47] E. Ben-Sasson and M. Viderman, “Composition of semi-LTCs by two-wise tensor products,” *Computational Complexity*, vol. 24, no. 3, pp. 601–643, 2015. [Online]. Available: http://dx.doi.org/10.1007/s00037-013-0074-8

[48] M. Viderman, “A combination of testability and decodability by tensor products,” *Random Structures and Algorithms*, vol. 46, no. 3, pp. 572–598, 2015.

[49] S. Kopparty, O. Meir, N. Ron-Zewi, and S. Saraf, “High-rate locally correctable and locally testable codes with sub-polynomial query complexity,” *J. ACM*, vol. 64, no. 2, pp. 11:1–11:42, 2017. [Online]. Available: https://doi.org/10.1145/3051093

[50] T. Gur, G. Ramnarayan, and R. D. Rothblum, “Relaxed locally correctable codes,” in *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*, 2018, pp. 27:1–27:11. [Online]. Available: https://doi.org/10.4230/LIPIcs.ITCS.2018.27

[51] P. Valiant, “The tensor product of two codes is not necessarily robustly testable,” in *RANDOM*. Springer, 2005, pp. 472–481.

[52] O. Goldreich and O. Meir, “The tensor product of two good codes is not necessarily locally testable,” *Information Processing Letters*, vol. 112, no. 8-9, pp. 351–355, 2012.

[53] L. Babai, L. Fortnow, and C. Lund, “Non-deterministic exponential time has two-prover interactive protocols,” *Computational Complexity*, vol. 1, pp. 3–40, 1991.

[54] M. Sudan, “Probabilistically checkable proofs - lecture notes,” 2000. [Online]. Available: http://madhu.seas.harvard.edu/MIT/pcp/pcp.ps

[55] C. Lund, L. Fortnow, H. J. Karloff, and N. Nisan, “Algebraic methods for interactive proof systems,” *J. ACM*, vol. 39, no. 4, pp. 859–868, 1992.

[56] O. Ben-Eliezer, E. Fischer, A. Levi, and R. D. Rothblum, “Hard properties with (very) short pcpps and their applications,” *CoRR*, vol. abs/1909.03255, 2019. [Online]. Available: http://arxiv.org/abs/1909.03255

[57] J. Håstad, “Some optimal inapproximability results,” *J. ACM*, vol. 48, no. 4, pp. 798–859, 2001. [Online]. Available: https://doi.org/10.1145/502090.502098

[58] A. Abboud, A. Rubinstein, and R. R. Williams, “Distributed PCP theorems for hardness of approximation in P,” in *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, 2017, pp. 25–36. [Online]. Available: https://doi.org/10.1109/FOCS.2017.12

[59] A. Rubinstein, “Hardness of approximate nearest neighbor search,” in *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, 2018, pp. 1260–1268. [Online]. Available: https://doi.org/10.1145/3188745.3188916

[60] L. Chen, S. Goldwasser, K. Lyu, G. Rothblum, and A. Rubinstein, “Fine-grained complexity meets ip = pspace,” in *SODA*. SIAM, 2019, pp. 1–20.