Ligero: Lightweight Sublinear Arguments Without a Trusted Setup

Scott Ames
University of Rochester
sames@cs.rochester.edu

Carmit Hazay
Bar-Ilan University
carmit.hazay@cs.biu.ac.il

Yuval Ishai
Technion and UCLA
yuvali@cs.technion.ac.il

Muthuramakrishnan Venkitasubramaniam
University of Rochester
muthuv@cs.rochester.edu

ABSTRACT

We design and implement a simple zero-knowledge argument protocol for NP whose communication complexity is proportional to the square-root of the verification circuit size. The protocol can be based on any collision-resistant hash function. Alternatively, it can be made non-interactive in the random oracle model, yielding concretely efficient zk-SNARKs that do not require a trusted setup or public-key cryptography.

Our protocol is attractive not only for very large verification circuits but also for moderately large circuits that arise in applications. For instance, for verifying a SHA-256 preimage in zero-knowledge with 2^{-40} soundness error, the communication complexity is roughly 44KB (or less than 34KB under a plausible conjecture), the prover running time is 140 ms, and the verifier running time is 62 ms. This proof is roughly 4 times shorter than a similar proof of ZKB++ (Chase et al., CCS 2017), an optimized variant of ZKBoo (Giacomelli et al., USENIX 2016).

The communication complexity of our protocol is independent of the circuit structure and depends only on the number of gates. For 2^{-40} soundness error, the communication becomes smaller than the circuit size for circuits containing roughly 3 million gates or more. Our efficiency advantages become even bigger in an amortized setting, where several instances need to be proven simultaneously.

Our zero-knowledge protocol is obtained by applying an optimized version of the general transformation of Ishai et al. (STOC 2007) to a variant of the protocol for secure multiparty computation of Damgård and Ishai (Crypto 2006). It can be viewed as a simple zero-knowledge interactive PCP based on "interleaved" Reed-Solomon codes.

1 INTRODUCTION

Verifying outsourced computations is important for tasks and scenarios when there is an incentive for the party performing the computation to report incorrect answers. In this work, we present a concretely efficient argument protocol for NP whose communication complexity is proportional to the square root of the size of a circuit verifying the NP witness. Our argument system is in fact a zero-knowledge argument of knowledge, and it only requires the verifier to send public coins to the prover. The latter feature implies that it can be made non-interactive via the Fiat-Shamir transform [19], yielding an efficient implementation of zero-knowledge succinct non-interactive arguments of knowledge (zk-SNARKs [11]) without a trusted setup.

To put our work in the proper context, we give some relevant background. The last half decade has seen tremendous progress in designing and implementing efficient systems for verifiable computation (see [4, 47] for recent surveys). These efforts can be divided into three broad categories according to the underlying combinatorial machinery.

Doubly efficient interactive proofs: This line of work, initiated by Goldwasser, Kalai, and Rothblum [23] (following a rich line of work on interactive proofs with computationally unbounded provers [24, 37, 44]), provides sublinear communication, efficiently verifiable proofs for low-depth polynomial-time computations.[1] See [15, 41, 45, 46] and references therein for a survey of works along this line.

Probabilistically checkable proofs (PCPs) and their interactive variants: Originating from the works of Kilian [36] and Micali [39], recent works [4, 6, 8] have shown how to obtain efficient sublinear arguments for NP from PCPs [1–3]. Classical PCPs have been extended to allow additional interaction with the prover, first in the model of interactive PCP (IPCP) [35] and then in the more general setting of interactive oracle proofs (IOP) [9], also known as probabilistically checkable interactive proofs (PCIP) [41]. Arguments obtained via PCPs and IOPs have the advantages of not relying on public-key cryptography, not requiring a trusted setup, and offering conjectured security against quantum attacks. However, current implementations along this line are still quite far from having good concrete efficiency.

Linear PCPs: This line of work, initiated by Ishai, Kushilevitz, and Ostrovsky [28] (in the interactive or designated verifier setting) and by Groth [27] (in the non-interactive, public verification setting of SNARKs) obtains sublinear arguments for NP *with preprocessing* by

[1] The GKR technique has been recently extended to the case of NP statements by Zhang et al. [48]. However, the communication complexity of the resulting arguments still grows with the verification circuit depth, and moreover their efficient instantiation requires the use of public-key cryptography.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
CCS'17, Oct. 30–Nov. 3, 2017, Dallas, TX, USA.
© 2017 ACM. ISBN 978-1-4503-4946-8/17/10…$15.00
DOI: http://dx.doi.org/10.1145/3133956.3134104

combining *linear PCPs* with homomorphic public-key cryptography. In a linear PCP the verifier can obtain a small number of linear combinations of a proof vector. Linear PCPs are simpler to construct than classical PCPs and have served as the basis for some of the first implementations of verifiable computation protocols [43]. A very efficient construction of linear PCPs for NP that serves as the basis for most current SNARK implementations, including the ones used in zerocash [7], was given by Gennaro, Gentry, Parno, and Raykova in [21]. (The view of these SNARKs as being based on linear PCPs is due to Bitansky et al. [12] and Setty et al. [42].) Two practical disadvantages of the protocols along this line are that they are quite slow on the prover side (due to a heavy use of public-key cryptography), and their soundness in the non-interactive setting crucially relies on the existence of a long and "structured" common reference string that needs to either be generated by a trusted party or by an expensive distributed protocol.

Our goal in this work is to combine the best features of previous approaches to the extent possible:

> *Obtain a simple, concretely efficient, sublinear communication zero-knowledge argument system for NP, without any setup, complex PCP machinery, or expensive public-key operations.*

As discussed above, all prior works fall short of meeting the above goal on one or more counts.

1.1 Our Results

The main result of this work is a zero-knowledge argument protocol for NP with the following features.

- It is *sublinear*, in the sense that the asymptotic communication complexity is roughly the square root of the verification circuit size.
- It is simple to describe and analyze in a self-contained way.
- It only employs symmetric-key primitives (collision-resistant hash-functions) in a black-box way. Moreover, the protocol can be made non-interactive in the random oracle model by using the Fiat-Shamir transform [19], thus providing a light-weight implementation of (publicly verifiable) zero-knowledge SNARKs.
- It does not require any trusted setup, even in the non-interactive case.
- It is concretely efficient. We demonstrate its concrete efficiency via an implementation.
- In the multi-instance setting where many instances for the same NP verification circuit are required, we obtain improved amortized communication complexity with sublinear verification time.

Our protocol can be seen as a light-weight instance of the second category of protocols discussed above. However, instead of directly applying techniques from the PCP literature, we combine efficient protocols for secure multiparty computation (MPC) with a variant of the general transformation of Ishai, Kushilevitz, Ostrovsky, and Sahai (IKOS) [29] that transforms such MPC protocols to zero-knowledge interactive PCPs (ZKIPCP).

More concretely, we instantiate the MPC component with an optimized variant of the protocol of Damgård and Ishai [16] (similar to the one described in Appendix C of [33]) and transform it into a ZKIPCP by applying a more efficient variant of the IKOS transformation in the spirit of the IPS compiler [32]. In a nutshell, the main difference with respect to the original IKOS transformation is that we restrict the topology of the MPC network in a way that leads to a better trade-off between soundness error and communication complexity.

A key feature of the underlying MPC protocol is that its *total* communication complexity is roughly equal to the size of the circuit being evaluated, independently of the number of parties. Letting the number of parties be the square root of the circuit size, the communication per party is also roughly the square root of the circuit size. This translates into a ZKIPCP with similar parameters. See Section 4 for a self-contained presentation of the ZKIPCP obtained via the above approach.

The recent work of Giacomelli, Madsen and Orlandi [22] and its improvement due to Chase et al. [13] already demonstrated that the IKOS transformation can lead to concretely efficient zero-knowledge arguments, but where the communication is bigger than the verification circuit size. In the present work, we obtain a sublinear variant of this result by modifying both the IKOS transformation and the underlying MPC machinery.

To summarize, using the above approach we obtain a simple proof of the following theorem with good concrete efficiency:

Theorem 1.1 (Informal). *Assume the existence of collision-resistant hash-functions. Then there is a public-coin zero-knowledge argument for proving the satisfiability of a circuit* C *with communication complexity* $\tilde{O}(\sqrt{|C|})$.

Concrete efficiency. We now give more detailed information about the concrete efficiency of our implementation. The following numbers apply either to interactive zero-knowledge protocols based on collision-resistant hash functions or to non-interactive zk-SNARKs in the random oracle model obtained via the Fiat-Shamir transform. We refer the reader to Section 6 for more details and give only a few representative figures below.

The communication complexity of proving the satisfiability of an arithmetic circuit with $s > 30000$ gates over a finite field $\mathbb{F}$ of size $|\mathbb{F}| \geq 2^{128}$ with soundness error 2^{-40} consists of roughly $95\sqrt{s}$ field elements (or $70\sqrt{s}$ elements under Conjecture 4.1). For the case of 2^{-80} error, the communication is roughly $140\sqrt{s}$ (or $120\sqrt{s}$ under Conjecture 4.1).

In the case of Boolean circuits, the communication complexity becomes smaller than the circuit size for circuits with more than roughly 3 million gates. One concrete benchmark that has been used in prior works is verifying a SHA-256 preimage in zero-knowledge. For this benchmark, the communication complexity of our protocol with 2^{-40} soundness error is roughly 44KB (or less than 34KB under a Conjecture 4.1), the prover running time is 140 ms, and the verifier running time is 62 ms. This is roughly 4 times less communication than a similar proof of ZKB++ [13], an optimized variant of ZKBoo [22]. Requiring 2^{-80} soundness error doubles the communication (as in [13, 22]).

Our protocol easily extends to a multi-instance setting to provide additional benefits. In this setting, we can handle N instances of a circuit of size s with soundness error $2^{-\kappa}$ at an amortized communication cost per instance smaller than s when $N = \Omega(\kappa^2)$.

Moreover, the amortized verification time in the multi-instance setting is sublinear, involving a total of $O(s \log s + N \log N)$ field operations. Finally, the prover's running time grows linearly with the number of instances but still remains practically feasible for reasonable number of instances. For the SHA-256 circuit, we show that the amortized communication over 4096 instances is 2KB with amortized prover time of 151 ms and verification time of 500 μs. This amortization is relevant to natural applications, e.g., in the context of cryptocurrencies [7, 18].

Related work. In a concurrent and independent work [5], Ben-Sasson et al. use different techniques to construct concretely efficient IOPs that imply "transparent" proof systems, referred to as zk-STARKs, of the same type we obtain here. These zk-STARK constructions significantly improve over the previous ones from [4]. A preliminary comparison with the concrete efficiency of our construction suggests that our construction is generally more attractive in terms of prover computation time and also in terms of proof size for smaller circuits (say, of size comparable to a few SHA-256 circuits), whereas the construction from [5] is more attractive in terms of verifier computation time and proof size for larger circuits. We leave a more thorough comparison between the two approaches for future work.

2 PRELIMINARIES

Basic notations. We denote the security parameter by κ. We say that a function $\mu : \mathbb{N} \to \mathbb{N}$ is *negligible* if for every positive polynomial $p(\cdot)$ and all sufficiently large κ's it holds that $\mu(\kappa) < \frac{1}{p(\kappa)}$. We use the abbreviation PPT to denote probabilistic polynomial-time and denote by $[n]$ the set of elements $\{1, \ldots, n\}$ for some $n \in \mathbb{N}$. For an NP relation $\mathcal{R}$, we denote by $\mathcal{R}_x$ the set of witnesses of x and by $\mathcal{L}_{\mathcal{R}}$ its associated language. That is, $\mathcal{R}_x = \{w \mid (x, w) \in \mathcal{R}\}$ and $\mathcal{L}_{\mathcal{R}} = \{x \mid \exists\, w\ s.t.\ (x, w) \in \mathcal{R}\}$.

2.1 Collision-Resistant Hashing and Merkle Trees

Let $\{\mathcal{H}_\kappa\}_{\kappa\in\mathbb{N}} = \{H : \{0,1\}^{p(\kappa)} \to \{0,1\}^{p'(\kappa)}\}_\kappa$ be a family of hash functions, where $p(\cdot)$ and $p'(\cdot)$ are polynomials so that $p'(\kappa) \le p(\kappa)$ for sufficiently large $\kappa \in \mathbb{N}$. For a hash function $H \leftarrow \mathcal{H}_\kappa$ a Merkle hash tree [38] is a data structure that allows to commit to $\ell = 2^d$ messages by a single hash value h such that revealing any message requires only to reveal $O(d)$ hash values.

A Merkle hash tree is represented by a binary tree of depth d where the ℓ messages $m_1, \ldots, m_\ell$ are assigned to the leaves of the tree; the values assigned to the internal nodes are computed using the underlying hash function H that is applied on the values assigned to the children, whereas the value h that commits to $m_1, \ldots, m_\ell$ is assigned to the root of the tree. To open the commitment to a message m_i, one reveals m_i together with all the values assigned to nodes on the path from the root to m_i, and the values assigned to the siblings of these nodes. We denote the algorithm of committing to ℓ messages $m_1, \ldots, m_\ell$ by $h := \mathsf{Commit}_{\mathsf{M}}(m_1, \ldots, m_\ell)$ and the opening of m_i by $(m_i, \mathrm{path}(i)) := \mathsf{Open}_{\mathsf{M}}(h, i)$. Verifying the opening of m_i is carried out by essentially recomputing the entire path bottom-up and comparing the final outcome (i.e., the root) to the value given at the commitment phase.

The binding property of a Merkle hash tree is due to collision-resistance. Intuitively, this says that it is infeasible to efficiently find a pair (x, x') so that $H(x) = H(x')$, where $H \leftarrow \mathcal{H}_\kappa$ for sufficiently large κ. In fact, one can show that collision-resistance of $\{\mathcal{H}_\kappa\}_{\kappa\in\mathbb{N}}$ carries over to the Merkle hashing. Formally, we say that a family of hash functions $\{\mathcal{H}_\kappa\}_\kappa$ is collision-resistant if for any PPT adversary $\mathcal{A}$ the following experiment outputs 1 with probability $\mathrm{negl}(\kappa)$: (i) A hash function H is sampled from $\mathcal{H}_\kappa$; (ii) The adversary $\mathcal{A}$ is given H and outputs x, x'; (iii) The experiment outputs 1 if and only if $x \neq x'$ and $H(x) = H(x')$.

In the random oracle model, Merkle tree can be computed by replacing the function H with a random oracle ρ where statistical binding follows due to the hardness of finding a collision in this model. We denote this algorithm by $\mathsf{Commit}^{\mathrm{RO}}_{\mathsf{M}}$.

2.2 Zero-Knowledge Arguments

We denote by $\langle A(w), B(z)\rangle(x)$ the random variable representing the (local) output of machine B when interacting with machine A on common input x, when the random-input to each machine is uniformly and independently chosen, and A (resp., B) has auxiliary input w (resp., z).

Definition 2.1 (Interactive argument system). *A pair of* PPT *interactive machines $\langle \mathcal{P}, \mathcal{V}\rangle$ is called an* interactive proof system for a language $\mathcal{L}$ *if there exists a negligible function* negl *such that the following two conditions hold:*

(1) Completeness: *For every $x \in \mathcal{L}$ there exists a string w such that for every $z \in \{0,1\}^*$,*

$$\Pr[\langle \mathcal{P}(w), \mathcal{V}(z)\rangle(x) = 1] \ge 1 - \mathrm{negl}(|x|).$$

(2) Soundness: *For every $x \notin \mathcal{L}$, every interactive* PPT *machine $\mathcal{P}^*$, and every $w, z \in \{0,1\}^*$*

$$\Pr[\langle \mathcal{P}^*(w), \mathcal{V}(z)\rangle(x) = 1] \le \mathrm{negl}(|x|).$$

Definition 2.2 (Zero-knowledge). *Let $\langle \mathcal{P}, \mathcal{V}\rangle$ be an interactive proof system for some language $\mathcal{L}$. We say that $\langle \mathcal{P}, \mathcal{V}\rangle$ is* computational zero-knowledge with respect to an auxiliary input *if for every* PPT *interactive machine $\mathcal{V}^*$ there exists a* PPT *algorithm $\mathcal{S}$, running in time polynomial in the length of its first input, such that*

$$\{\langle \mathcal{P}(w), \mathcal{V}^*(z)\rangle(x)\}_{x\in\mathcal{L}, w\in\mathcal{R}_x, z\in\{0,1\}^*} \overset{\mathrm{c}}{\approx} \{\langle \mathcal{S}\rangle(x, z)\}_{x\in\mathcal{L}, z\in\{0,1\}^*}$$

(when the distinguishing gap is considered as a function of $|x|$). Specifically, the left term denote the output of $\mathcal{V}^*$ after it interacts with $\mathcal{P}$ on common input x whereas, the right term denote the output of $\mathcal{S}$ on x.

Our zero-knowledge protocols in fact satisfy the additional *proof of knowledge* property, which is important for some applications. See full version for more details.

2.3 Interactive PCPs

An interactive PCP [35] (IPCP) is a combination of a traditional PCP with an interactive proof. An IPCP is a special case of interactive oracle proofs (IOP) [9] (also known as probabilistically checkable interactive proofs [41]). We will be interested in zero-knowledge interactive PCPs [25] in which the verifier reads a small number of

bits from the PCP and exchanges a small number of bits with the prover $\mathcal{P}$. We formalize this notion below.

Definition 2.3 (Interactive PCP). *Let $\mathcal{R}(x, w)$ be an NP relation corresponding to an NP language $\mathcal{L}$. An* interactive PCP (IPCP) *system for $\mathcal{R}$ with parameters (q, l, ϵ) is a pair of* PPT *interactive machines $\langle \mathcal{P}, \mathcal{V} \rangle$ with the following properties.*

1. *Syntax: On common input x and prover input w, the prover $\mathcal{P}$ computes in time* poly$(|x|)$ *a bit string π (referred to as the PCP). The prover $\mathcal{P}$ and verifier $\mathcal{V}$ then interact, where the verifier has oracle access to π.*
2. Completeness: *If $(x, w) \in \mathcal{R}$ then*
$$\Pr[(\mathcal{P}(x, w), \mathcal{V}^{\pi}(x)) = 1] = 1.$$
3. Soundness: *For every $x \notin \mathcal{L}$, every (unbounded) interactive machine $\mathcal{P}^*$ and every $\tilde{\pi} \in \{0,1\}^*$,*
$$Pr[(\mathcal{P}^*, \mathcal{V}^{\tilde{\pi}}(x)) = 1] \leq \epsilon(|x|).$$
4. Complexity: *In the interaction $(\mathcal{P}(x, w), \mathcal{V}^{\pi}(x))$ at most $l(|x|)$ bits are communicated and $\mathcal{V}$ reads at most $q(|x|)$ bits of π.*

A public-coin *IPCP is one where every message sent by the verifier simply consists of fresh random bits.*

The notion of IPCP can be extended to additionally guarantee zero-knowledge. Our zero-knowledge variants of IPCP will achieve perfect zero-knowledge against honest verifiers. We present the definition of zero-knowledge IPCP next.

Definition 2.4 (Zero-knowledge IPCP). *Let $\langle \mathcal{P}, \mathcal{V} \rangle$ be an interactive PCP for $\mathcal{R}$. We say that $\langle \mathcal{P}, \mathcal{V} \rangle$ is an* (honest verifier, perfect) zero-knowledge IPCP *(or ZKIPCP for short) if there exists an expected polynomial time algorithm $\mathcal{S}$, such that for any $(x, w) \in \mathcal{R}$, the output of $S(x)$ is distributed identically to the view of $\mathcal{V}$ in the interaction $(\mathcal{P}(x, w), \mathcal{V}^{\pi}(x))$.*

3 FROM MPC TO ZKIPCP

3.1 Our MPC Model

As mentioned in the introduction, the efficiency of our constructions can be distilled to identifying the right MPC model and designing an efficient protocol in this model. In this regards we deviate from the original work of [29] which provided a general transformation from any honest majority MPC protocol that can compute arbitrary functionalities. In particular, our model is more in line with the watchlist mechanism (a-la [32]). We begin with the description of the MPC model and the protocol specifications that we will need to design our zero-knowledge protocol. In Section 4, we use such MPC protocols based on the works [14, 16, 32, 33].

In our model, we consider a *sender client* S, n servers $P_1, \ldots, P_n$ and a *receiver client* R. The sender has input x and a witness w with respect to some NP relation $\mathcal{R}$. The receiver and the servers do not receive any input, where the servers obtain random shares from the sender and evaluate the computed circuit. Upon receiving (x, w) from the sender, the functionality computes $\mathcal{R}(x, w)$ and forwards the result to the receiver R. We consider the specific network where we restrict the communication to a single message between S and the servers at the beginning of the protocol and a single message from the servers to the receiver R at the end of the protocol. The only way the servers may communicate with each other is via a broadcast. In our actual MPC protocol, the servers will never utilize such a broadcast. Nevertheless, our transformation from MPC to ZK can be easily extended to allow for the servers to invoke a broadcast. For simplicity, in our actual transformation, we will restrict the servers to not communicate with each other at all.

We consider the security of protocols in both the honest-but-curious (passive) and the malicious (active) models. In the former model, one may break the security requirements into the following correctness and privacy requirements.

Definition 3.1 (Correctness). *We say that Π realizes a deterministic $n+1$-party functionality $(x, r_1, \ldots, r_n)$ with perfect (resp., statistical) correctness if for all inputs $(x, r_1, \ldots, r_n)$, the probability that the output of some player is different from the output of f is 0 (resp., negligible in κ), where the probability is over the independent choices of the random inputs $r_1, \ldots, r_n$.*

Definition 3.2 (t_p-Privacy). *Let $1 \leq t_p < n$. We say that Π realizes f with perfect t_p-privacy if there is a* PPT *simulator $\mathcal{S}$ such that for any inputs $(x, r_1, \ldots, r_n)$ and every set of corrupted players $T \subset [n]$, where $|T| \leq t_p$, the joint view $\mathbf{View}_T(x, r_1, \ldots, r_n)$ of players in T is distributed identically to $\mathcal{S}(T, x, \{r_i\}_{i \in T}, f_T(x, r_1, \ldots, r_n))$.*

With respect to our MPC model defined above, we consider privacy in the presence of a static passive adversary that corrupts the receiver R and at most t_p servers.

In the malicious model, in which corrupted players may behave arbitrarily, security cannot be generally broken into correctness and privacy as above. However, for our purposes we only need the protocols to satisfy a weaker notion of security in the malicious model that is implied by the standard general definition. Specifically, it suffices that Π be t_p-private as above, and moreover it should satisfy the following notion of correctness in the malicious model.

Definition 3.3 (Statistical t_r-Robustness). *We say that Π realizes f with statistical t_r-robustness if it is perfectly correct in the presence of a honest-but-curious adversary as in Definition 3.1, and furthermore for any (unbounded) active adversary that* adaptively *corrupts a set T of at most t_r players, and for any inputs $(x, r_1, \ldots, r_n)$, the following robustness property holds. If there is no $(r_1, \ldots, r_n)$ such that $f(x, r_1, \ldots, r_n) = 1$, then the probability that R outputs 1 in an execution of Π in which the inputs of the honest players are consistent with $(x, r_1, \ldots, r_n)$ is negligible in κ where κ is a statistical parameter that the protocol Π receives as input.*

Our main theorems about our two-party ZK protocol are proven in the presence of a static active adversary, that corrupts the prover at the onset of the execution. Nevertheless, our proof relies on the security of the underlying MPC protocol (utilized in the MPC-in-the-head paradigm) being robust against an active adversary that *adaptively* corrupts a subset of the servers in the underlying MPC protocol. Concretely, with respect to our MPC model defined above, we consider robustness in the presence of an adaptive active adversary that corrupts the sender S and at most t_r servers.

Finally, when used in the MPC-in-the-head paradigm, we need the notion of consistent views between servers and the receiver that we define below.

Definition 3.4 (Consistent views). *We say that a pair of views V_i, V_j are consistent (with respect to the protocol Π and some public input x) if the outgoing messages implicit in V_i are identical to the incoming messages reported in V_j and vice versa.*

3.2 ZKIPCP for NP - The General Case

Next, we provide our compilation from an MPC protocol satisfying the requirements specified in Section 3.1 to an interactive PCP. We note that while the transformation presented in this section works for any MPC in the model as described in the previous section, we will simplify our MPC model as follows:

Two-phase: The protocol we consider will proceed in two phases: In Phase 1, the servers receive inputs from the sender and only perform local computation. After Phase 1, the servers obtain a public random string r of length l sampled via a coin-flipping oracle and broadcast to all servers. The servers use this in Phase 2 for their local computation at the end of which each server sends a single output message to the receiver R.

No broadcast: The servers never communicate with each other. Each server simply receives inputs from the sender at the beginning of Phase 1, then receives a public random string in Phase 2, and finally delivers a message to R.

Formally, let $\mathcal{L}$ be an NP-language with NP relation $\mathcal{R}$. Let x an NP statement that is the common input and let w be the private input of the prover. Our construction Π_{ZKIPCP} proceeds as follows:

Let Π be any MPC protocol in our model. We will now design a ZKIPCP protocol Π_{ZKIPCP} that meets Definition 2.3.

Protocol Π_{ZKIPCP}.

- **Input:** The prover $\mathcal{P}$ and the verifier $\mathcal{V}$ share a common input statement x and a circuit description C that realizes $\mathcal{R}$. $\mathcal{P}$ additionally has input w such that $\mathcal{R}(x, w) = 1$.
- **Oracle π:** The prover runs the MPC protocol Π "in-its-head" as follows. It picks a random input r_S and invokes S on $(x, w; r_S)$ and a random input r_i for every server P_i. The prover computes the views of the servers up to the end of Phase 1 in Π, denoted by $(V_1, \ldots, V_n)$, and sets the oracle as the n symbols $(V_1, \ldots, V_n)$.
- **The interactive protocol.**
  (1) $\mathcal{V}$ picks a random challenge r of length l and sends it to the sender.
  (2) Upon receiving the challenge r, prover $\mathcal{P}$ sends the view V of R.[2]
  (3) $\mathcal{V}$ computes the output of R from the view and checks if R does not abort. It then picks a random subset Q of $[n]$ of size t_p uniformly at random (with repetitions) from $[n]$, and queries the oracle on Q.
  (4) $\mathcal{V}$ obtains from the oracle the views of the servers in Q.
  (5) $\mathcal{V}$ aborts if the views of the servers are *inconsistent* with the view of R. Otherwise, it accepts and halts.

We are now ready to prove the following theorem.

Theorem 3.5. *Let f be the following functionality for a sender S and n servers $P_1, \ldots, P_n$ and receiver R. Given a public statement x and an additional input w received from S, the functionality delivers $\mathcal{R}(x, w)$ to R. Suppose that Π is a two-phase protocol in the MPC model specified in Section 3.1 that realizes f with statistical t_r-robustness (in the malicious model) and perfect t_p-privacy (in the honest-but-curious model), where $t_r < \lceil \frac{n}{2} \rceil - 1$. Then protocol Π_{ZKIPCP} described above is a ZKIPCP for NP relation $\mathcal{R}$, with soundness error $\left(1 - \frac{t_r}{n}\right)^{t_p} + \delta(\kappa)$ where $\delta(\kappa)$ is the robustness error of Π.*

Proof: Our proof follows by establishing completeness, soundness and zero-knowledge as required in Definitions 2.3-2.4.

Completeness: Completeness follows directly from the correctness of the underlying MPC protocol.

Soundness: Consider a statement $x \notin \mathcal{L}_{\mathcal{R}}$. We will show that no prover $\mathcal{P}^*$ can convince $\mathcal{V}$ beyond a negligible probability to accept a false statement. We will argue soundness by following an approach similar to [29] where we first identify an inconsistency graph and then invoke the properties of the underlying MPC. More precisely, we consider an inconsistency graph G based on the n views $V_1, \ldots, V_n$ and the view of the receiver R which contains the messages from servers $P_1, \ldots, P_n$ to R. Here, the servers and the receiver correspond to nodes in G and inconsistency between every pair of nodes is defined as in Definition 3.4. Then there are two cases depending on the graph G:

Case 1: There are more than t_r edges in G. In this case, we will argue that with high probability the set of servers opened by the verifier will hit one of these edges. Recall that the view of R is provided to the verifier. Therefore, for any edge in G between R and V_i, if the corresponding server P_i falls in Q, then the verifier rejects. The probability that all t_p servers chosen by the verifier misses all inconsistent edges is at most $\left(1 - \frac{t_r}{n}\right)^{t_p}$.

Case 2: There are fewer than t_r edges in G. In this case, we will argue that by the statistical t_r-robustness of the underlying MPC protocol Π, the verifier will reject except with probability $\delta(\kappa)$. More precisely, for every cheating strategy $\mathcal{P}^*$ in the ZK proof we will demonstrate an adversarial strategy $\mathcal{A}$ attacking the underlying MPC protocol such that the probability with which $\mathcal{V}$ accepts a false statement when interacting with $\mathcal{P}^*$ on a false statement will be bounded by the probability that R outputs 1 in an execution of the underlying MPC protocol with adversary $\mathcal{A}$.

More precisely, consider an adversary $\mathcal{A}$ that is participating in the MPC protocol with n servers, a sender and a receiver. Internally, $\mathcal{A}$ incorporates the code of $\mathcal{P}^*$ while emulating the roles of the oracle and $\mathcal{V}$. When the protocol begins, $\mathcal{P}^*$ set the oracle with the views of the servers as in Phase 1 of Π. These views simply contain the inputs sent to the servers (as all computations are local). Upon obtaining the views of the servers $\mathcal{A}$ will corrupt the sender in the external MPC execution, and acting as the sender, it will send as input to server P_i the value that was internally generated by $\mathcal{P}^*$ as the view of that server, namely V_i. Next, recall that in the MPC protocol the servers receive a random string from the coin-flipping oracle (in our protocol the verifier picks r as the challenge in Step 1). $\mathcal{A}$ will internally forward this string r to $\mathcal{P}^*$ as the message provided by the verifier. Next internally, $\mathcal{P}^*$ will generate the view of the receiver R from which $\mathcal{A}$ computes the inconsistency graph

[2] As the prover possesses all information about the servers, and the verifier always receives the broadcast message from each server, these broadcast messages can be sent directly from the prover to the verifier.

G. Recall that an edge is present between a server P_i and the receiver R in this graph if the view of P_i is inconsistent with the view of R and randomness r. Let T_1 be the set of the servers of size t^* that are connected to an edge in G. If $t^* > t_r$, then $\mathcal{A}$ aborts.

Otherwise, $\mathcal{A}$ proceeds with the internal execution by selecting t_p indices for the verifier's challenge, for which the oracle will reveal the views of these corresponding servers. If $\mathcal{V}$ rejects in the internal execution because any of these views are inconsistent, then $\mathcal{A}$ aborts. Otherwise, $\mathcal{A}$ continues with the external execution. Recall that in Phase 2, each server sends a single message to R. Then just before the servers send these messages, $\mathcal{A}$ (adaptively) corrupts the servers in T_1 and replaces their (honestly generated) messages sent to R by what was internally reported in the view of R, namely the messages sent by $\mathcal{P}^*$ to the verifier in the proof. Upon sending the message in Phase 2, let T_2 of size t'^* be the set of the servers that are inconsistent with the view o R. If $|T_1 \cup T_2| > t_r$, then $\mathcal{A}$ aborts. Otherwise, $\mathcal{A}$ (adaptively) corrupts the servers in t'^* and continues as above.

It follows from this description that the acceptance condition of the verifier in the internal emulation with $\mathcal{A}$ is identical to the output of R in the external MPC execution. Since the underlying MPC protocol is t_r-robust and the number of parties corrupted by $\mathcal{A}$ is t^* such that $|t^*| \leq t_r$, we have that R outputs 0 except with probability at most $\delta(\kappa)$.

We conclude that the verifier in the internal emulation by $\mathcal{P}^*$ accepts the proof of a false statement except with probability at most $\delta(\kappa)$. Next, we observe that the view of the verifier emulated by $\mathcal{A}$ in the internal emulation is identically distributed to the view of an honest verifier in an interactive with $\mathcal{P}^*$. Therefore, we can conclude that an honest verifier accepts a false statement in this case with probability at most $\delta(\kappa)$.

Applying a union bound, we conclude that the verifier accepts a false statement with probability at most $\left(1 - \frac{t_r}{n}\right)^{t_p} + \delta(\kappa)$.

Zero-knowledge: The zero-knowledge property follows from the t_p-privacy of the underlying MPC protocol Π. Namely, we construct a simulator $\mathcal{S}$ that invokes the simulator for the MPC protocol, denoted by $\mathcal{S}_\Pi$. In the simulation $\mathcal{S}_\Pi$ is required to produce the view of R upon receiving a challenge r and then, upon obtaining the t_p indexes that the verifier requests to open, outputs the views of these t_p servers, where $\mathcal{S}_\Pi$ simulates an adversary $\mathcal{A}$ that adaptively corrupts these servers at the end of the computation. □

Communication complexity: The main source of complexity is in revealing the view of R in the third message and revealing the view of the t_p servers in the last message. If the maximum size of the view of each server P_i for $i \in t_p$, is v_{size}, and the size of the view of R is v_R, then the total communication complexity from the prover is $t_p \cdot v_{\text{size}} + v_R$.

We can adjust the parameters of the protocol we use in Section 4 subject to the constraint that $v_{\text{size}} v_R = O(|\mathrm{C}|)$. To minimize the communication complexity, if we set $t_p \cdot v_{\text{size}}$ and v_R to be roughly equal then we obtain the optimum complexity of our approach.

4 A DIRECT ZKIPCP CONSTRUCTION

In this section we give a self-contained description of our zero-knowledge interactive PCP protocol. This protocol is a slightly optimized version of the protocol obtained by applying our variant of the general "MPC to ZK" transformation from [30] (see Section 3) to the honest-majority MPC protocol from [16].

Coding notation. For a code $C \subseteq \Sigma^n$ and vector $v \in \Sigma^n$, denote by $d(v, C)$ the minimal distance of v from C, namely the number of positions in which v differs from the closest codeword in C, and by $\Delta(v, C)$ the set of positions in which v differs from such a closest codeword (in case of ties, take the lexicographically first closest codeword). We further denote by $d(V, C)$ the minimal distance between a vector set V and a code C, namely $d(V, C) = \min_{v \in V}\{d(v, C)\}$. Our ZKIPCP protocol uses Reed-Solomon (RS) codes, defined next.

Definition 4.1 (Reed-Solomon Code). *For positive integers n, k, finite field $\mathbb{F}$, and a vector $\eta = (\eta_1, \ldots, \eta_n) \in \mathbb{F}^n$ of distinct field elements, the code $\mathrm{RS}_{\mathbb{F},n,k,\eta}$ is the $[n, k, n-k+1]$ linear code over $\mathbb{F}$ that consists of all n-tuples $(p(\eta_1), \ldots, p(\eta_n))$ where p is a polynomial of degree $< k$ over $\mathbb{F}$.*

At a very high level, our ZKIPCP protocol proves the satisfiability of an arithmetic circuit C of size s in the following way. The prover arranges (a slightly redundant representation of) the s wire values of C on a satisfying assignment in an $O(\sqrt{s}) \times O(\sqrt{s})$ matrix, and encodes each row of this matrix using an RS code. The verifier challenges the prover to reveal $O(\sqrt{s})$ linear combinations of the entries of the codeword matrix, and checks their consistency with t randomly selected columns of this matrix, where t is a security parameter. In the following we describe the ZKIPCP construction in a bottom-up fashion, first addressing the case of IPCP (with no zero-knowledge) and then introduce the modifications required for making it zero-knowledge.

4.1 Testing Interleaved Linear Codes

We start by describing and analyzing a simple interactive prover-assisted protocol for simultaneously testing the membership of multiple vectors in a given linear code L. It will be convenient to view m-tuples of codewords in L as codewords in an interleaved code L^m. We formally define this notion below.

Definition 4.2 (Interleaved code). *Let $L \subset \mathbb{F}^n$ be an $[n, k, d]$ linear code over $\mathbb{F}$. We let L^m denote the $[n, mk, d]$ (interleaved) code over $\mathbb{F}^m$ whose codewords are all $m \times n$ matrices U such that every row U_i of U satisfies $U_i \in L$. For $U \in L^m$ and $j \in [n]$, we denote by $U[j]$ the jth symbol (column) of U.*

To test the membership of U in L^m, $\mathcal{V}$ challenges $\mathcal{P}$ to reveal a random linear combination of the rows U_i, and then checks that the revealed codeword is consistent with a randomly selected set of t columns of U.[3]

Test-Interleaved$(\mathbb{F}, L[n, k, d], m, t; U)$

[3]This test is implicitly used in the verifiable secret sharing sub-protocol of efficient MPC protocols from the literature, and in particular in the protocols from [16, 33] on which we build. Its soundness requires the MPC protocol to be *adaptively secure* to accommodate $\mathcal{P}$'s ability to make the locations of inconsistencies depend on $\mathcal{V}$'s random challenge; when the MPC adversary is adaptive, it can potentially corrupt all parties observing such inconsistencies. Indeed, the compiler from statistically secure MPC to ZK proofs from [30] relies on the adaptive security of the underlying MPC protocol.

- **Oracle:** A purported L^m-codeword U. Depending on the context, we may view U either as a matrix in $\mathbb{F}^{m\times n}$ in which each row U_i is a purported L-codeword, or as a sequence of n symbols $(U[1],\ldots,U[n])$, $U[j]\in\mathbb{F}^m$.
- **Interactive testing:**
  (1) $\mathcal{V}$ picks a random linear combinations $r\in\mathbb{F}^m$ and sends r to $\mathcal{P}$.
  (2) $\mathcal{P}$ responds with $w = r^T U \in \mathbb{F}^n$.
  (3) $\mathcal{V}$ queries a set $Q\subset[n]$ of t random symbols $U[j]$, $j\in Q$.
  (4) $\mathcal{V}$ accepts iff $w\in L$ and w is consistent with U_Q and r. That is, for every $j\in Q$ we have $\sum_{i=1}^m r_j\cdot U_{i,j} = w_j$.

  The following lemma follows directly from the linearity of L.

Lemma 4.1. *If $U\in L^m$ and $\mathcal{P}$ is honest, then $\mathcal{V}$ always accepts.*

Our soundness analysis will rely on the following lemma.

Lemma 4.2. *Let e be a positive integer such that $e < d/4$. Suppose $d(U^*, L^m) > e$. Then, for a random w^* in the row-span of U^*, we have*

$$\Pr[d(w^*, L)\le e]\le (e+1)/|\mathbb{F}|.$$

Proof: Let L^* be the row-span of U^*. We consider two cases.

Case 1: There exists $v^*\in L^*$ such that $d(v^*, L) > 2e$. In this case, we show that

$$\Pr_{w^*\in_R L^*}[d(w^*, L)\le e]\le 1/|\mathbb{F}|. \tag{1}$$

Indeed, using a basis for L^* that includes v^*, a random $w^*\in L^*$ can be written as $\alpha v^* + x$, where $\alpha\in_R\mathbb{F}$ and x is distributed independently of α. We argue that conditioned on any choice of x, there can be at most one choice of α such that $d(\alpha v^*+x, L)\le e$, which implies (1). This follows by observing that if $d(\alpha v^*+x_0, L)\le e$ and $d(\alpha' v^*+x_0, L)\le e$ for $\alpha\neq\alpha'$, then by the triangle inequality we have $d((\alpha-\alpha')v^*, L)\le 2e$, contradicting the assumption that $d(v^*, L) > 2e$.

Case 2: For every $v^*\in L^*, d(v^*, L)\le 2e$. We show that in this case $\Pr_{w^*\in_R L^*}[d(w^*, L)\le e]\le (e+1)/|\mathbb{F}|$. Let U_i^* be the i-th row of U^* and let $E_i = \Delta(U_i^*, L)$. Note that, since $2e < d/2$, each U_i^* can be written uniquely as $U_i^* = u_i + \chi_i$ where $u_i\in L$ and χ_i is nonzero exactly in its E_i entries. Let $E = \cup_{i=1}^m E_i$. Since $d(U^*, L^m) > e$, we have $|E| > e$. We show that for each $j\in E$, except with $1/|\mathbb{F}|$ probability over a random choice of w^* from L^*, either $j\in\Delta(w^*, L)$ or $d(w^*, L) > e$, from which the claim will follow.

Suppose $j\in E_i$. As before, we write $w^* = \alpha U_i^* + x$ for $\alpha\in_R\mathbb{F}$ and x distributed independently of α. Condition on any possible choice x_0 of x. Define a bad set

$$B_j = \{\alpha : j\notin\Delta(\alpha U_i^* + x_0, L)\ \wedge\ d(\alpha U_i^* + x_0, L)\le e\}.$$

We show that $|B_j|\le 1$. Suppose towards contradiction that there are two distinct $\alpha,\alpha'\in\mathbb{F}$ such that for $z = \alpha U_i^* + x_0$ and $z' = \alpha' U_i^* + x_0$ we have $d(z, L)\le e$, $d(z', L)\le e$, $j\notin\Delta(z, L)$, and $j\notin\Delta(z', L)$. Since $d > 4e$, for any z^* in the linear span of z and z' we have $j\notin\Delta(z^*, L)$. Since U_i^* is in this linear span, we have $j\notin\Delta(U_i^*, L)$, in contradiction to the assumption that $j\in E_i$.

We have shown that for each $j\in E$, conditioned on every possible choice of x, either $j\in\Delta(w^*, L)$ or $d(w^*, L) > e$ except with $1/|\mathbb{F}|$ probability over the choice of α. It follows that the same holds for a random choice of x. Taking a union bound over the first $e+1$ elements of E we get that $\Pr_{w^*\in_R L^*}[d(w^*, L)\le e]\le (e+1)/|\mathbb{F}|$ as required. □

We now prove the soundness of the testing procedure when the given oracle is far from L^m.

Theorem 4.3. *Let e be a positive integer such that $e < d/4$. Suppose $d(U^*, L^m) > e$. Then, for any malicious $\mathcal{P}$ strategy, the oracle U^* is rejected by $\mathcal{V}$ except with $\le (1-e/n)^t + (e+1)/|\mathbb{F}|$ probability.*

Proof: Letting $w^* = r^T U^*$, it follows from Lemma 4.2 that

$$\begin{aligned}\Pr[\mathcal{V}\text{ accepts }U^*] &\le \Pr[\mathcal{V}\text{ accepts}\mid d(w^*, L) > e]\\ &\quad + \Pr[d(w^*, L)\le e]\\ &\le \frac{\binom{n-e-1}{t}}{\binom{n}{t}} + (e+1)/|\mathbb{F}|\\ &\le (1-e/n)^t + (e+1)/|\mathbb{F}|\end{aligned}$$

as required. □

In Appendix A we present a simple generalization of the testing algorithm that uses σ linear combinations to amplify soundness.

4.1.1 A Tighter Analysis? We conjecture that the requirement $e < d/4$ in Theorem 4.3 can be relaxed to $e < d/3$ or possibly even $e < d/2$ with essentially the same soundness error bound. In fact, it suffices for this to hold for RS codes. Such a stronger version of Theorem 4.3 would yield roughly up to 25% improvement in the size of our ZK arguments. We leave the confirmation (or refutation) of this conjecture to future work. Below we reduce such a stronger version of Theorem 4.3 to a simple conjecture about the distance of points on an affine line from an RS code.

We start by showing that for any linear code over a sufficiently large field, when $e < d/3$ we can restrict the attention to Case 1 from the proof of Lemma 4.2.

Lemma 4.3. *Let L be an $[n,k,d]$ linear code over $\mathbb{F}$. Let e be a positive integer such that $e < d/3$ and $|\mathbb{F}|\ge e$. Suppose $d(U^*, L^m) > e$. Then, there exists $v^*\in L^*$ such that $d(v^*, L) > e$, where L^* is the row-span of U^*.*

Proof: Assume towards a contradiction that $d(v^*, L)\le e$ for all $v^*\in L^*$. Suppose $v_0^*\in L^*$ maximizes the distance from L. Since $d(U^*, L^m) > e$, there must be a row U_i^* such that $\Delta(U_i^*, L)\backslash\Delta(v_0^*, L)\neq\emptyset$. Let $v_0^* = u_0 + \chi_0$ and $U_i^* = u_i + \chi_i$ for $u_0, u_i\in L$ and χ_0, χ_i of weight $\le e$. We argue that there exists $\alpha\in\mathbb{F}$ such that for $\hat{v} = v_0^* + \alpha U_i^*$ we have $d(\hat{v}, L) > d(v_0^*, L)$, contradicting the choice of v_0^*. This follows by a union bound, noting that for any $j\in\Delta(v_0^*, L)\cup\Delta(U_i^*, L)$ there is at most one choice of α such that $\hat{v}_j = 0$. □

Given Lemma 4.3 it suffices to show that in any affine subspace of $\mathbb{F}^n$, either all points are e-close to L or almost all are not. This reduces to showing the same for 1-dimensional spaces. We state an explicit version of the conjecture for the case of RS codes.

Conjecture 4.1. *Let $L = \mathrm{RS}_{\mathbb{F},n,k,\eta}$ be a Reed-Solomon code with minimal distance $d = n-k+1$. Let e be a positive integer such that $e < d/3$. Then for every $u, v\in\mathbb{F}^n$, defining an affine line $\ell_{u,v} = \{u+\alpha v : \alpha\in\mathbb{F}\}$, either (1) for every $x\in\ell_{u,v}$ we have $d(x, L)\le e$, or (2) for at most d points $x\in\ell_{u,v}$ we have $d(x, L)\le e$.*

We do not have a counterexample to Conjecture 4.1 even when $e < d/2$ and even when L is a general linear code. Moreover, even

if the conjecture holds with a relaxed version of condition (2) (say, where d is replaced by n^2), this relaxed version is still almost as good in the context of the efficiency of our ZKIPCP.

The following conjectured stronger version of Lemma 4.2 follows from Lemma 4.3 by extending Conjecture 4.1 from lines to general affine subspaces. This extension follows from the fact that if a subspace has a point that is far from L, then we can partition the subspace (minus the point) into lines containing this point. Assuming the conjecture for lines, each line should be almost entirely far from the code, so the subspace should be almost entirely far from the code.

Lemma 4.4. *Suppose Conjecture 4.1 holds. Let $L = \mathsf{RS}_{\mathbb{F},n,k,\eta}$ be a Reed-Solomon code with minimal distance $d = n - k + 1$ and e a positive integer such that $e < d/3$. Suppose $d(U^*, L^m) > e$. Then, for a random w^* in the row-span of U^*, we have*

$$\Pr[d(w^*, L) \le e] \le d/|\mathbb{F}|.$$

Lemma 4.4 implies the following conjectured stronger version of Theorem 4.3.

Theorem 4.4. *Suppose Conjecture 4.1 holds. Let e be a positive integer such that $e < d/3$. Suppose $d(U^*, L^m) > e$. Then, for any malicious $\mathcal{P}$ strategy, the oracle U^* is rejected by $\mathcal{V}$ except with $\le (1 - e/n)^t + d/|\mathbb{F}|$ probability.*

4.2 Testing Linear Constraints over Interleaved Reed-Solomon Codes

In this section we describe an efficient procedure for testing that a message encoded by an interleaved RS code satisfies a given set of linear constraints. This generalizes a procedure from [26, 33] for testing that such an encoded message satisfies a given set of replication constraints.

In the following we assign a message in $\mathbb{F}^\ell$ to each codeword in $\mathbb{F}^n$ by considering a fixed set of ℓ evaluation points of the polynomial defined by the codeword. Note that while each codeword has a unique message assigned to it, several different codewords can be "decoded" into the same message. However, if the degree of the polynomial corresponding to the codeword is restricted to be smaller than ℓ, the encoding becomes unique.

Definition 4.5 (Encoded message). *Let $L = \mathsf{RS}_{\mathbb{F},n,k,\eta}$ be an RS code and $\zeta = (\zeta_1, \ldots, \zeta_\ell)$ be a sequence of distinct elements of $\mathbb{F}$ for $\ell \le k$. For $u \in L$ we define the message $\mathsf{Dec}_\zeta(u)$ to be $(p_u(\zeta_1), \ldots, p_u(\zeta_\ell))$, where p_u is the polynomial (of degree $< k$) corresponding to u. For $U \in L^m$ with rows $u^1, \ldots, u^m \in L$, we let $\mathsf{Dec}_\zeta(U)$ be the length-$m\ell$ vector $x = (x_{11}, \ldots, x_{1\ell}, \ldots, x_{m1}, \ldots, x_{m\ell})$ such that $(x_{i1}, \ldots, x_{i\ell}) = \mathsf{Dec}_\zeta(u^i)$ for $i \in [m]$. Finally, when ζ is clear from the context, we say that U* encodes *x if $x = \mathsf{Dec}_\zeta(U)$.*

We now describe a simple testing algorithm for checking that the message x encoded by U satisfies a given system of linear equations $Ax = b$, for $A \in \mathbb{F}^{m\ell \times m\ell}$ and $b \in \mathbb{F}^{m\ell}$. (We will always apply this test with a sparse matrix A containing $O(m\ell)$ nonzero entries.) The test simply picks a random linear combination $r \in \mathbb{F}^{m\ell}$ and checks that $(r^T A)x = r^T b$. Note that if $Ax \ne b$, the test will only pass with $1/|\mathbb{F}|$ probability. To make the test sublinear, we let the prover provide a polynomial encoding $(r^T A)x$ and check its consistency with $r^T b$ and with U on t randomly chosen symbols.

To further simplify the description and analysis of the testing algorithm, we assume that U is promised to be e-close to L^m. Our final IPCP we will run **Test-Interleaved** to ensure that if the promise is violated, this is caught with high probability.

Test-Linear-Constraints-IRS$(\mathbb{F}, L = \mathsf{RS}_{\mathbb{F},n,k,\eta}, m, t, \zeta, A, b; U)$

- **Oracle:** A purported L^m-codeword U that should encode a message $x \in \mathbb{F}^{m\ell}$ satisfying $Ax = b$.
- **Interactive testing:**
  (1) $\mathcal{V}$ picks a random vector $r \in \mathbb{F}^{m\ell}$ and sends r to $\mathcal{P}$.
  (2) $\mathcal{V}$ and $\mathcal{P}$ compute
  $$r^T A = (r_{11}, \ldots, r_{1\ell}, \ldots, r_{m1}, \ldots, r_{m\ell})$$
  and, for $i \in [m]$, let $r_i(\cdot)$ be the unique polynomial of degree $< \ell$ such that $r_i(\zeta_c) = r_{ic}$ for every $c \in [\ell]$.
  (3) $\mathcal{P}$ sends the $k + \ell - 1$ coefficients of the polynomial defined by $q(\bullet) = \sum_{i=1}^m r_i(\bullet) \cdot p_i(\bullet)$, where p_i is the polynomial of degree $< k$ corresponding to row i of U.
  (4) $\mathcal{V}$ queries a set $Q \subset [n]$ of t random symbols $U[j]$, $j \in Q$.
  (5) $\mathcal{V}$ accepts if the following conditions hold:
    (a) $\sum_{c\in[\ell]} q(\zeta_c) = \sum_{i\in[m], c\in[\ell]} r_{ic} b_{ic}$.
    (b) For every $j \in Q$, $\sum_{i=1}^m r_i(\eta_j) \cdot U_{i,j} = q(\eta_j)$.

We will analyze the test under the promise that the (possibly badly formed) U is close to L^m.

The following lemma easily follows by inspection.

Lemma 4.5. *If $U \in L^m$, U encodes x such that $Ax = b$, and $\mathcal{P}$ is honest, $\mathcal{V}$ always accepts.*

Soundness is argued by the following lemma.

Lemma 4.6. *Let e be a positive integer such that $e < d/2$. Suppose that a (badly formed) oracle U^* is e-close to a codeword $U \in L^m$ encoding $x \in \mathbb{F}^{m\ell}$ such that $Ax \ne b$. Then, for any malicious $\mathcal{P}$ strategy, U^* is rejected by $\mathcal{V}$ except with at most $((e+k+\ell)/n)^t + 1/|\mathbb{F}|$ probability.*

Proof: Let q be the polynomial generated in Step 3 following the honest $\mathcal{P}$ strategy on input U. Since we assume $Ax \ne b$, we have $\Pr_r[r^T Ax = r^T b] = 1/|\mathbb{F}|$. It follows that except with probability $1/|\mathbb{F}|$ over the choice of r in Step 1, the polynomial q fails to satisfy the condition in Step 5(a). Indeed, we have $\sum_{c\in[\ell]} q(\zeta_c) = (r^T A)x$ and $\sum_{i\in[m], c\in[\ell]} r_{ic} b_{ic} = r^T b$.

Next, we analyze the probability that a malicious $\mathcal{P}$ strategy is rejected conditioned on q failing in this way. Let q' be the polynomial sent by the prover. If $q' = q$, the $\mathcal{V}$ rejects in Step 5(a). If $q' \ne q$, then using the fact that q and q' are of degree at most $k + \ell - 2$, we have that the number of indices $j \in [n]$ for which $q(\eta_j) = q'(\eta_j)$ is at most $k + \ell - 2$. Let Q' be the set of indices on which they agree. Then $\mathcal{V}$ rejects in Step 5(b) whenever Q selected in Step 4 contains an index $i \notin Q' \cup E$. This fails to happen with probability at most $\binom{e+k+\ell-2}{t} / \binom{n}{t} \le ((e + k + \ell)/n)^t$. The lemma now follows by a union bound. □

4.3 Testing Quadratic Constraints over Interleaved Reed-Solomon Codes

In this section we describe a simple test for verifying that vectors $x, y, z \in \mathbb{F}^{m\ell}$, encoded by $U^x, U^y, U^z \in L^m$ respectively, satisfy the

constraints $x \odot y + a \odot z = b$ for some known $a, b \in \mathbb{F}^{m\ell}$, where $\odot$ denotes pointwise product. Letting $L = \mathsf{RS}_{\mathbb{F},n,k,\eta}$, $U_a = \mathsf{Enc}(a)$ and $U_b = \mathsf{Enc}(b)$, this reduces to checking that $U^x \odot U^y + U^a \odot U^z - U^b$ encodes the all-0 message $0^{m\ell}$ in the (interleaved extension of) $\hat{L} = \mathsf{RS}_{\mathbb{F},n,2k-1,\eta}$. This could be done using the general membership test for interleaved linear codes (**Test-Interleaved** from Section 4.1), since the set of codewords in $\hat{L}$ that encode the all-0 message is a linear subcode of $\hat{L}$. In the following we present this test in a self-contained way, exploiting the promise that U^x, U^y, U^z are close to L^m for a tighter analysis.

Test-Quadratic-Constraints-IRS$(\mathbb{F}, L = \mathsf{RS}_{\mathbb{F},n,k,\eta}, m, t, \zeta, a, b; U^x, U^y, U^z)$

- **Oracle:** Purported L^m-codewords U^x, U^y, U^z that should encode messages $x, y, z \in \mathbb{F}^{m\ell}$ satisfying $x \odot y + a \odot z = b$.
- **Interactive testing:**
  (1) Let $U^a = \mathsf{Enc}_\zeta(a)$ and $U^b = \mathsf{Enc}_\zeta(b)$.
  (2) $\mathcal{V}$ picks a random linear combinations $r \in \mathbb{F}^m$ and sends r to $\mathcal{P}$.
  (3) $\mathcal{P}$ sends the $2k-1$ coefficients of the polynomial p_0 defined by $p_0(\bullet) = \sum_{i=1}^m r_i \cdot p_i(\bullet)$, where $p_i(\bullet) = p_i^x(\bullet) \cdot p_i^y(\bullet) + p_i^a(\bullet) \cdot p_i^z(\bullet) - p_i^b(\bullet)$, and where p_i^x, p_i^y, p_i^z are the polynomials of degree $< k$ corresponding to row i of U^x, U^y, U^z, and p_i^a, p_i^b are the polynomials of degree $< \ell$ corresponding to row i of U^a, U^b.
  (4) $\mathcal{V}$ picks a random index set $Q \subset [n]$ of size t, and queries $U^x[j], U^y[j], U^z[j]$, $j \in Q$.
  (5) $\mathcal{V}$ accepts if the following conditions hold:
    (a) $p_0(\zeta_c) = 0$ for every $c \in [\ell]$.
    (b) For every $j \in Q$, it holds that

$$\sum_{i=1}^m r_i \cdot \left[U_{i,j}^x \cdot U_{i,j}^y + U_{i,j}^a \cdot U_{i,j}^z - U_{i,j}^b \right] = p_0(\eta_j).$$

The following lemma follows again directly from the description.

Lemma 4.7. *If $U^x, U^y, U^z \in L^m$ encode vectors $x, y, z \in \mathbb{F}^{m\ell}$ satisfying $x \odot y + a \odot z = b$ and $\mathcal{P}$ is honest, $\mathcal{V}$ always accepts.*

Soundness is argued by the following lemma.

Lemma 4.8. *Let e be a positive integer such that $e < d/2$. Let U^{x*}, U^{y*}, U^{z*} be badly formed oracles and let $U^* \in \mathbb{F}^{3m \times n}$ be the matrix obtained by vertically juxtaposing the corresponding $m \times n$ matrices. Suppose $d(U^*, L^{3m}) \le e$, and let U^x, U^y, U^z, respectively, be the (unique) codewords in L^m that are closest to U^{x*}, U^{y*}, U^{z*}. Suppose U^x, U^y, U^z encode x, y, z such that $x \odot y + a \odot z \neq b$. Then, for any malicious $\mathcal{P}$ strategy, (U^{x*}, U^{y*}, U^{z*}) is rejected by $\mathcal{V}$ except with at most $1/|\mathbb{F}| + ((e + 2k)/n)^t$ probability.*

Proof: Let p_0 be the polynomial generated in Step 3 following the honest $\mathcal{P}$ strategy on U^x, U^y, U^z. Since x, y, z do not satisfy the constraints, except with probability $1/|\mathbb{F}|$ over the choice of r in Step 2, the polynomial p_0 fails to satisfy the condition in Step 5(a). Indeed, we have $p_0(\bullet) = \sum_{i=1}^m r_i \cdot p_i(\bullet)$ and there must exists an i and ζ_c such that $p_i(\zeta_c) \neq 0$. Next, we analyze the probability that a malicious $\mathcal{P}$ strategy is rejected conditioned on p_0 failing in this way.

Let p_0' be the polynomial sent by the prover. If $p_0' = p_0$, $\mathcal{V}$ rejects in Step 5(a). If $p_0' \neq p_0$, then using the fact that p_0 and p_0' are of degree at most $2k-2$, we have that the number of indices $j \in [n]$ for which $p_0(\eta_j) = p_0'(\eta_j)$ is at most $2k-2$. Let Q' be the set of indices on which p_0 and p_0' agree. Then $\mathcal{V}$ rejects in Step 5(b) whenever Q selected in Step 4 contains an index $i \notin Q' \cup E$, where $E = \Delta(U^*, L^{3m})$. This fails to happen with probability at most $\binom{e+2k-2}{t} / \binom{n}{t} \le ((e + 2k)/n)^t$. The lemma now follows by a union bound. □

4.4 IPCP for Arithmetic Circuits

In this section, we provide our IPCP for arithmetic circuits. Fix a large finite field $\mathbb{F}$. Let $\mathrm{C} : \mathbb{F}^{n_i} \to \mathbb{F}$ be an arithmetic circuit. Without loss of generality, we will assume that the circuit contains only ADD and MULTIPLY gates with fan-in two. We show how a prover can convince a verifier that $\mathrm{C}(w) = 1$.

Protocol IPCP$(\mathrm{C}, \mathbb{F})$**.**

- **Input:** The prover $\mathcal{P}$ and the verifier $\mathcal{V}$ share a common input arithmetic circuit $\mathrm{C} : \mathbb{F}^{n_i} \to \mathbb{F}$ and input statement x. $\mathcal{P}$ additionally has input $\overline{\alpha} = (\alpha_1, \ldots, \alpha_{n_i})$ such that $\mathrm{C}(\overline{\alpha}) = 1$.
- **Oracle π:** Let m, ℓ be integers such that $m \cdot \ell > n_i + s$ where s is the number of gates in the circuit. Then $\mathcal{P}$ generates an extended witness $w \in \mathbb{F}^{m\ell}$ where the first $n_i + s$ entries of w are

$$(\alpha_1, \ldots, \alpha_{n_i}, \beta_1, \ldots, \beta_s)$$

  where β_i is the output of the i^{th} gate when evaluating $\mathrm{C}(\overline{\alpha})$. $\mathcal{P}$ defines a system of constraints that contains the following constraint for every multiplication gate g in the circuit C

$$\beta_a \cdot \beta_b - \beta_c = 0$$

  and for every addition gate, the constraint

$$\beta_a + \beta_b - \beta_c = 0$$

  where β_a and β_b are the input values to the gate g and β_c is the output value in the extended witness. For the output gate we include the constraint $\beta_a + \beta_b - 1 = 1$ if the final gate is an addition gate, and $\beta_a \cdot \beta_b - 1 = 0$ if it is a multiplication gate. $\mathcal{P}$ constructs vectors x, y and z in $\mathbb{F}^{m\ell}$ where the j^{th} entry of x, y and z contains the values β_a, β_b, and β_c corresponding to the j^{th} multiplication gate in w. $\mathcal{P}$ and $\mathcal{V}$ construct matrices P_x, P_y and P_z in $\mathbb{F}^{m\ell \times m\ell}$ such that

$$x = P_x w, y = P_y w, z = P_z w.$$

  Finally, it constructs matrix $P_{\mathbf{add}} \in \mathbb{F}^{m\ell \times m\ell}$ such that the j^{th} position of $P_{\mathbf{add}} w$ equals $\beta_a + \beta_b - \beta_c$ where β_a, β_b, and β_c correspond to the j^{th} addition gate of the circuit in w. Let $U^w, U^x, U^y, U^z \in L^m$ respectively encode w, x, y, z where $L = \mathsf{RS}_{\mathbb{F},n,k,\eta}$. $\mathcal{P}$ sets the oracle π as $U \in L^{4m}$ which is set as the vertical juxtaposition of the following four matrices $U^w, U^x, U^y, U^z \in L^m$.
- **Notation:** We denote by $(-\mathbf{1})^{m\ell}$ (respectively, $\mathbf{0}^{m\ell}$) the vector in $\mathbb{F}^{m\ell}$ whose entries are all equal to -1 (respectively, equal to 0). We further denote by I_n, the $n \times n$ identity matrix.
- **The interactive protocol:**
  $\mathcal{V}$ and $\mathcal{P}$ run the following tests.
  (1) // `Test if` U `is` e`-close to a code in` L^{4m}
  Test-Interleaved$(\mathbb{F}, L, 4m, t; U)$

(2) // Test if addition gates are correct.
Test-Linear-Constraints-IRS
$(\mathbb{F}, L, m, t, \zeta, P_{\mathbf{add}}, \mathbf{0}^{m\ell}; U^w)$
(3) // Test if multiplication gates are correct.
- **Test-Linear-Constraints-IRS**
$\left(\mathbb{F}, L, 2m, t, \zeta, \left[I_{m\ell} | -P_x\right], \mathbf{0}^{2m\ell}; \begin{bmatrix} U^x \\ U^w \end{bmatrix}\right)$
- **Test-Linear-Constraints-IRS**
$\left(\mathbb{F}, L, 2m, t, \zeta, \left[I_{m\ell} | -P_y\right], \mathbf{0}^{2m\ell}; \begin{bmatrix} U^y \\ U^w \end{bmatrix}\right)$
- **Test-Linear-Constraints-IRS**
$\left(\mathbb{F}, L, 2m, t, \zeta, \left[I_{m\ell} | -P_z\right], \mathbf{0}^{2m\ell}; \begin{bmatrix} U^a \\ U^w \end{bmatrix}\right)$
- **Test-Quadratic-Constraints-IRS**
$(\mathbb{F}, L, m, t, \zeta, (-\mathbf{1})^{m\ell}, \mathbf{0}^{m\ell}; U^x, U^y, U^z)$

Since all the tests open the same number of columns t in U_w, U_x, U_y, U_z, the $\mathcal{V}$ will simply open t columns of U. $\mathcal{V}$ rejects if it rejects in any of the tests above.

The completeness of our IPCP follows from the following lemma.

Lemma 4.9. *If $U^w, U^x, U^y, U^z \in L^m$ encode vectors $w, x, y, z \in \mathbb{F}^{m\ell}$ satisfying*

$x = P_x w,\ y = P_y w,\ z = P_z w,\ x \odot y + (-\mathbf{1})^{m\ell} \odot z = \mathbf{0}^{m\ell},\ P_{\mathbf{add}} w = \mathbf{0}^{m\ell}$ *and $\mathcal{P}$ is honest, $\mathcal{V}$ always accepts.*

The proof follows directly from Lemmas 4.1, 4.5 and 4.7. Next, soundness is argued by the following lemma.

Lemma 4.10. *Let e be a positive integer such that $e < d/4$ and suppose that there exists no $\overline{\alpha}$ such that $C(\overline{\alpha}) = 1$. Then, for any maliciously formed oracle $\widehat{U}$ and any malicious prover strategy, the verifier rejects except with at most $(e+6)/|\mathbb{F}| + (1-e/n)^t + 5((e+2k)/n)^t$ probability.*

Proof: On a high-level, soundness will essentially follow by the soundness of the individual tests and the overall soundness error follows by a direct application of a union bound over the soundness of these tests. In more detail, let U be the vertical juxtaposition of $U^{w*}, U^{x*}, U^{y*}, U^{z*}$. Then we argue soundness by considering the following cases and applying a union bound:

Case $d(U, L^{4m}) > e$: Since $e < d/4$, we can conclude from Lemma 4.2 that the verifier rejects in **Test-Interleaved** executed in Step 1 except with probability $(1-e/n)^t + (e+1)/|\mathbb{F}|$.

Case $d(U, L^{4m}) \leq e$: Next, let $U^w, U^x, U^y, U^z \in L^m$ be the codes that are respectively close to $U^{w*}, U^{x*}, U^{y*}, U^{z*}$ and encode the messages w, x, y, z. Recall that there exists no w, x, y, z that satisfy all the following constraints:
$x = P_x w,\ y = P_y w,\ z = P_z w,$
$x \odot y + (\mathbf{-1})^{m\ell} \odot z = \mathbf{0}^{m\ell}$ and $P_{\mathbf{add}} w = \mathbf{0}^{m\ell}$.
Then we can conclude from Lemmas 4.6 and 4.8 by applying a union bound on the corresponding tests that the verifier rejects except with probability:

$$5/|\mathbb{F}| + 4 \cdot ((e+k+\ell)/n)^t + ((e+2k)/n)^t < 5 \cdot \left(1/|\mathbb{F}| + ((e+2k)/n)^t\right).$$

□

The following theorem follows from the construction described above and the preceding Lemmas.

Theorem 4.6. *Fix parameters n, m, ℓ, k, t, e such that $e < (n-k)/4$. Let $\mathrm{C} : \mathbb{F}^{n_i} \to \mathbb{F}$ be an arithmetic circuit of size s, where $|\mathbb{F}| \geq n$ and $m \cdot \ell > n_i + s$. Then protocol $\mathrm{IPCP}(\mathrm{C}, \mathbb{F})$ satisfies the following:*

- Completeness: *If $\overline{\alpha}$ is such that $C(\overline{\alpha}) = 1$ and oracle π is generated honestly as described in the protocol, then $\Pr[(\mathcal{P}(\mathrm{C}, w), \mathcal{V}^{\pi}(\mathrm{C})) = 1] = 1$.*
- Soundness: *If there is no $\overline{\alpha}$ is such that $C(\overline{\alpha}) = 1$, then for every (unbounded) prover strategy $\mathcal{P}^*$ and every $\tilde{\pi} \in \mathbb{F}^{4mn}$, $Pr[(\mathcal{P}^*, \mathcal{V}^{\tilde{\pi}}(\mathrm{C})) = 1] \leq (e+6)/|\mathbb{F}| + (1-e/n)^t + 5((e+2k)/n)^t$.*
- Complexity: *The number of field operations performed is $\mathrm{poly}(|\mathrm{C}|, n)$. The number of field elements communicated by $\mathcal{P}$ to $\mathcal{V}$ is $n + 4 \cdot (k+\ell-1) + 2 \cdot k - 1$ whereas $\mathcal{V}$ reads t symbols from $\mathbb{F}^{4m}$.*

The first term in the communication cost is the communication incurred by the test-interleaved protocol, the second term is due to the four linear-constraints tests and the final term results from our quadratic-constraint test.

We remark here that we can improve the communication of the protocol by letting s only count the number of multiplication gates by incorporating the linear constraints resulting from the addition gates into the linear constraints used to define the vectors x, y and z.

4.5 IPCP for Boolean Circuits

In order to obtain the benefits in soundness from running our IPCP over a large field $\mathbb{F}$, we show how we can prove the validity of a Boolean circuit $\mathrm{C} : \{0,1\}^{n_i} \to \{0,1\}$ by encoding the witness in any larger field $\mathbb{F}$. First, in the witness, the prover will map the Boolean 0 to the additive identity ϵ_0 in $\mathbb{F}$ and 1 to the multiplicative identity ϵ_1 in $\mathbb{F}$. Now, we can enforce that each element in the witness is a 0 or 1, by introducing a quadratic constraint $\beta^2 - \beta = 0$.

Given that binary constraints are already enforced, next, we proceed to demonstrating how we incorporate the constraints from XOR and ADD gates.

In fact, we will show all gate constraints can be expressed as a linear relation on the witness bits. Let x be a column vector consisting of the witness string to the evaluation of the circuit. We will construct a matrix A and a column vector w such that if w is binary and is a valid witness then the elements of Aw will all be 0, and if w is binary and is not a valid witness then at least one element of Aw will be nonzero. For each XOR and AND gate in the circuit we will create a row of the matrix corresponding to the enforcement of that relation in the witness. The vector w besides including the input bits x, will include one additional bit for each XOR and AND gate in the circuit. We explain the purpose of these extra bits next.

Given integers b_1 and b_2 consider the arithmetic constraint $b_1 + b_2 = r_0 + 2 \cdot r_1$ over integers. In this constraint, if we enforce that all values are bits then r_0 is the XOR of b_1 and b_2 and r_1 is the AND of b_1 and b_2. If in our witness w we want to make sure that b_1 XOR b_2 is b_3 then we include an auxiliary bit d and enforce the linear constraint $b_1 + b_2 = b_3 + 2 \cdot d$. A similar constraint can be established for an AND relation analogously. To conclude, we observe that if the values have been enforced to be a binary constraint then checking

the arithmetic constraints over integers can be done by checking the equation modulo a sufficiently large prime ($p \geq 3$).

We can also extend this idea to consider more complex gates such as addition modulo 2^{32} over 32-bit inputs and outputs. This can be expressed as a linear constraint over the bits. Suppose $a = (a_0, \ldots, a_{31})$, $b = (b_0, \ldots, b_{31})$ and $c = (c_0, \ldots, c_{31})$ are the input and output bits, the constraint $a + b = c \bmod 2^{32}$ can be expressed as

$$\sum_{i=0}^{31} 2^i \cdot a_i + \sum_{i=0}^{31} 2^i \cdot b_i = 2^{32} \cdot d + \sum_{i=0}^{31} 2^i \cdot c_i$$

where d is an auxiliary input bit and all values are enforced to be bits. However, this will require $\mathbb{F}$ with characteristic $p > 2^{33}$.

4.6 Achieving Zero-Knowledge

Note first that the verifier obtains two types of information in two different building blocks of the IPCP. First, it obtains linear combinations of codewords in a linear code L. Second, it probes a small number of symbols from each codeword. Since codewords are used to encode the NP witness, both types of information give the verifier partial information about the NP witness, and thus the basic IPCP we described is not zero-knowledge.

Fortunately, making the IPCP zero-knowledge only requires introducing small modifications to the construction and analysis. The second type of "local" information about codewords is made harmless by making the encoding randomized, so that probing just a few symbols in each codeword reveals no information about the encoded message. The high level idea for making the first type of information harmless is to use an additional random codeword for blinding the linear combination of codewords revealed to the verifier. However, this needs to be done in a way that does not compromise soundness. Below we describe the modifications required for each of the IPCP ingredients.

4.6.1 ZK Testing of Interleaved Linear Codes. Recall that in the verification algorithm **Test-Interleaved** from Section 4.1, $\mathcal{V}$ obtains a linear combination of the form $w = r^T U$, where $U \in \mathbb{F}^{m \times n}$ is a matrix whose rows should be codewords in L. A natural approach for making this linear combination hide U is by allowing the prover to add to the rows of U an additional random codeword u' that is used for blinding.

A simple implementation of this idea that provides a slightly inferior soundness guarantee is the following. Apply the algorithm **Test-Interleaved** to L^{m+1}, with an extended oracle U' whose first m rows contain U and whose last row is u'. Letting $w' = r^T U + r' u'$ be the random linear combination obtained by $\mathcal{V}$, the test fails to be zero-knowledge when $r' = 0$, which occurs with $1/|\mathbb{F}|$ probability. Alternatively, settling for a slightly worse soundness guarantee (where $e/|\mathbb{F}|$ is replaced by $e/(|\mathbb{F}| - 1)$), one could just let r' be a random *nonzero* field element, and get perfect zero-knowledge.

It turns out, however, that one could fix r' to 1 and still get the same soundness guarantee about U as in Lemma 4.2 since we can apply the same the decomposition argument. This "affine" variant of **Test-Interleaved** is described and analyzed in Appendix C.

4.6.2 ZK Testing of Linear Constraints over Interleaved Reed-Solomon Codes. The verification algorithm for the linear constraints $Ax = b$ samples a random vector r, obtains $r^T Ax$, and compares it with $r^T b$. Looking more carefully at our actual protocol, the verifier obtains a polynomial $q(\bullet)$ and checks whether the equality $\sum_{c \in [\ell]} q(\zeta_c) = \sum_{i \in [m], c \in [\ell]} r_{ic} b_{ic}$ holds. While the sum itself does not reveal any additional information beyond what is already known, namely $r^T b$, the individual evaluations of q, i.e. $q(\zeta_c)$ themselves may reveal information about the inputs.

To hide this, a simple idea is for $\mathcal{P}$ to provide an additional vector u' along with U that encodes a message $(\gamma_1, \ldots, \gamma_\ell)$ such that $\sum_{c \in [\ell]} \gamma_c = 0$, and append to A constraints that sum the entries in the message encoded in u' and check if it is equal to 0.

However, as before, this will yield less than optimal soundness guarantee. Instead we take the following approach that provides the same soundness guarantee as the original (non-affine version of the) test. In We apply the algorithm Test-Linear-Constraints-IRS to L^{m+1} where $L = \mathsf{RS}_{\mathbb{F},n,k,\eta}$, with an extended oracle U' whose first m rows contain U and whose last row is u' where additionally u' encodes a message $(\gamma_1, \ldots, \gamma_\ell)$ such that $\sum_{c \in [\ell]} \gamma_c = 0$. Letting $q(\bullet) = \sum_{i=1}^{m} r_i(\bullet) \cdot p_i(\bullet) + r_{\mathsf{blind}}(\bullet)$ be the polynomial obtained by $\mathcal{V}$ where $r_{\mathsf{blind}}(\bullet)$ is a polynomial (of degree $< k + \ell - 1$) corresponding to u', we can show that the soundness of the resulting scheme will be the same as for Lemma 4.6. This "affine" variant of **Test-Linear-Constraints** is described and analyzed in the full version.

4.6.3 ZK Testing of Quadratic Constraints over Interleaved Reed-Solomon Codes. We modify the quadratic constraint testing procedure in the same way as we modified the linear constraint testing. Concretely, we apply the algorithm **Test-Quadratic-Constraint** to L^{3m+1} where $L = \mathsf{RS}_{\mathbb{F},n,k,\eta}$, with an extended oracle U' whose first $3m$ rows contain U^x, U^y, U^z and whose last row is u' where additionally u' encodes a message 0^ℓ. Letting $p_0(\bullet) = \sum_{i=1}^{m} r_i \cdot p_i(\bullet) + r_{\mathsf{blind}}(\bullet)$ be the polynomial obtained by $\mathcal{V}$ where $r_{\mathsf{blind}}(\bullet)$ is a polynomial (of degree $< 2k - 1$) corresponding to u', we can show that the soundness of the resulting scheme will be the same as for Lemma 4.8. This "affine" variant of **Test-Quadratic-Constraints** is described and analyzed in the full version.

4.7 The Final ZKIPCP

In this section provide a self contained description of the final ZKIPCP protocol, combining all of the previous sub-protocols. In this section, we provide our ZKIPCP for arithmetic circuits over a large field $\mathbb{F}$. On a high-level, the protocol is essentially the IPCP construction from Section 4.4 with the exception that we replace all the tests with the generalized affine version (with repetitions).

Protocol ZKIPCP$(\mathrm{C}, \mathbb{F})$**.**

- **Input:** The prover $\mathcal{P}$ and the verifier $\mathcal{V}$ share a common input arithmetic circuit $\mathrm{C} : \mathbb{F}^{n_i} \to \mathbb{F}$ and input statement x. $\mathcal{P}$ additionally has input $\overline{\alpha} = (\alpha_1, \ldots, \alpha_{n_i})$ such that $\mathrm{C}(\overline{\alpha}) = 1$.
- **Oracle π:** Let m, ℓ be integers such that $m \cdot \ell > n_i + s$ where s is the number of gates in the circuit. Then $\mathcal{P}$ generates an extended witness $w \in \mathbb{F}^{m\ell}$ where the first $n_i + s$ entries of w are $(\alpha_1, \ldots, \alpha_{n_i}, \beta_1, \ldots, \beta_s)$ where β_i is the output of the i^{th} gate when evaluating $\mathrm{C}(\overline{\alpha})$. $\mathcal{P}$ constructs vectors x, y and z in $\mathbb{F}^{m\ell}$ where the j^{th} entry of x, y and z contains the values β_a, β_b, and β_c corresponding to the j^{th} multiplication gate in w. $\mathcal{P}$ and $\mathcal{V}$ construct matrices P_x, P_y and P_z in $\mathbb{F}^{m\ell \times m\ell}$ such that

$$x = P_x w, y = P_y w, z = P_z w.$$

Finally, it constructs matrix $P_{\mathbf{add}} \in \mathbb{F}^{m\ell \times m\ell}$ such that the j^{th} row of $P_{\mathbf{add}} w$ equals $\beta_a + \beta_b - \beta_c$ where β_a, β_b, and β_c correspond to the j^{th} addition gate of the circuit in w. The prover samples random codewords $U^w, U^x, U^y, U^z \in L^m$ where $L = \mathsf{RS}_{\mathbb{F},n,k,\eta}$ subject to $w = \mathsf{Dec}_\zeta(U^w), x = \mathsf{Dec}_\zeta(U^x), y = \mathsf{Dec}_\zeta(U^y), z = \mathsf{Dec}_\zeta(U^z)$ where $\zeta = (\zeta_1, \ldots, \zeta_\ell)$ is a sequence of distinct elements disjoint from $(\eta_1, \ldots, \eta_n)$. Let $u'_h, u^x_h, u^y_h, u^z_h, u^0_h, u^{\mathbf{add}}_h$ be auxiliary rows sampled randomly from L for every $h \in [\sigma]$ where each of $u^x_h, u^y_h, u^z_h, u^{\mathbf{add}}_h$ encodes an independently sampled random ℓ messages $(\gamma_1, \ldots, \gamma_\ell)$ subject to $\sum_{c\in[\ell]} \gamma_c = 0$ and u^0_h encodes 0^ℓ. $\mathcal{P}$ sets the oracle as $U \in L^{4m}$ which is set as the vertical juxtaposition of the matrices $U^w, U^x, U^y, U^z \in L^m$.

- **The interactive protocol:**
  (1) For every $h \in [\sigma]$, $\mathcal{V}$ picks the following random elements and sends them to $\mathcal{P}$:
  – $r_h \in \mathbb{F}^{4m}$,
  // Testing Interleaved Reed-Solomon Codes
  – $r^{\mathbf{add}}_h \in \mathbb{F}^{m\ell}$,
  // Testing Addition Gates
  – $r^x_h, r^y_h, r^z_h, \in \mathbb{F}^{m\ell}, r^q_h \in \mathbb{F}^m$.
  // Testing Multiplication Gates
  (2) For every $h \in [\sigma]$, $\mathcal{P}$ responds with
  – $v_h = (r_h)^T U + u'_h \in \mathbb{F}^n$,
  // Testing Interleaved Reed-Solomon Codes
  – Polynomial $q^{\mathbf{add}}_h(\bullet)$ of degree $< k + \ell - 1$ where $q^{\mathbf{add}}_h(\bullet) = r^{\mathbf{add}}_{\mathsf{blind},h}(\bullet) + \sum_{i=1}^m r^{\mathbf{add}}_{h,i}(\bullet) \cdot p_i(\bullet)$, such that p_i is the polynomial of degree $< k$ corresponding to row i of U^w, $r^{\mathbf{add}}_{h,i}(\cdot)$ is the unique polynomial of degree $< \ell$ such that $r^{\mathbf{add}}_{h,i}(\zeta_c) = ((r^{\mathbf{add}}_h)^T P_{\mathbf{add}})_{ic}$ for every $c \in [\ell]$, and $r^{\mathbf{add}}_{\mathsf{blind},h}(\bullet)$ is the polynomial of degree $< k + \ell - 1$ corresponding to $u^{\mathbf{add}}_h$.
  // Testing Addition Gates
  – Polynomials $q^x_h(\bullet), q^y_h(\bullet), q^z_h(\bullet)$ of degree $< k + \ell - 1$, and $p_{0,h}(\bullet)$ of degree $< 2k - 1$ where
  $q^x_h(\bullet) = r^x_{\mathsf{blind},h}(\bullet) + \sum_{i=1}^m r^x_{h,i}(\bullet) \cdot p^x_i(\bullet) + \sum_{i=m+1}^{2m} r^x_{h,i}(\bullet) \cdot p_{i-m}(\bullet))$,
  $q^y_h(\bullet) = r^y_{\mathsf{blind},h}(\bullet) + \sum_{i=1}^m r^y_{h,i}(\bullet) \cdot p^x_i(\bullet) + \sum_{i=m+1}^{2m} r^y_{h,i}(\bullet) \cdot p_{i-m}(\bullet))$,
  $q^z_h(\bullet) = r^z_{\mathsf{blind},h}(\bullet) + \sum_{i=1}^m r^z_{h,i}(\bullet) \cdot p^x_i(\bullet) + \sum_{i=m+1}^{2m} r^z_{h,i}(\bullet) \cdot p_{i-m}(\bullet))$,
  $p_{0,h}(\bullet) = r^0_{\mathsf{blind},h}(\bullet) + \sum_{i=1}^m \left(r^q_h\right)_i \cdot \left(p^x_i(\bullet) \cdot p^y_i(\bullet) - p^z_i(\bullet)\right)$
  where for $a \in \{x, y, z\}$, p^a_i is the polynomial of degree $< k$ corresponding to row i of U^a, $r^a_{h,i}(\cdot)$ be the unique polynomial of degree $< \ell$ such that $r^a_{h,i}(\zeta_c) = ((r^a_h)^T [I_{m\ell} \mid -P_a])_{ic}$ for every $c \in [\ell]$, $r^a_{\mathsf{blind},h}(\bullet)$ is the polynomial of degree $< k + \ell - 1$ corresponding to u^a_h and $r^0_{\mathsf{blind},h}$ is the polynomial of degree $< 2k - 1$ corresponding u^0_h.
  // Testing Multiplication Gates
  (3) $\mathcal{V}$ picks a random index set $Q \subset [n]$ of size t, and queries $U[j]$ that is the vertical juxtaposition of $U^x_h[j], U^y_h[j], U^z_h[j], U^w_h[j], u^x_h[j], u^y_h[j], u^z_h[j], u^{\mathbf{add}}_h[j], u'_h[j], j \in Q$ and accepts if the following conditions hold for every $h \in [\sigma]$:
  – For every $j \in Q$ we have $\sum_{i=1}^{4m} r_h[j] \cdot U_{i,j} + u'_h[j] = v_h[j]$.
  // Testing Interleaved Reed-Solomon Codes
  $\sum_{c\in[\ell]} q^{\mathbf{add}}_h(\zeta_c) = 0$ and for every $j \in Q$ we have $u^{\mathbf{add}}_h[j] + \sum_{i=1}^m r^{\mathbf{add}}_{h,i}(\eta_j) \cdot U^w_{i,j} = q^{\mathbf{add}}_h(\eta_j)$.
  // Testing Addition Gates
  – For every $a \in \{x, y, z\}$,
  $\sum_{c\in[\ell]} q^a_h(\zeta_c) = 0$ and for every $j \in Q$ we have $u^a_h[j] + \sum_{i=1}^m r^a_{h,i}(\eta_j) \cdot U^a_{i,j} + \sum_{i=m+1}^{2m} r^a_{h,i}(\eta_j) \cdot U^w_{i-m,j} = q^a_h(\eta_j)$.
  $p_{0,h}(\zeta_c) = 0$ for every $c \in [\ell]$ and for every $j \in Q$,
  $u^0_h[j] + \sum_{i=1}^m (r^q_h)_i \cdot \left[U^x_{i,j} \cdot U^y_{i,j} - U^z_{i,j}\right] = p_{0,h}(\eta_j)$.
  // Testing Multiplication Gates

The completeness of our ZKIPCP follows from the next lemma.

Lemma 4.11. *If $U^w, U^x, U^y, U^z \in L^m$ encode vectors $w, x, y, z \in \mathbb{F}^{m\ell}$ satisfying*

$$x = P_x w,\ y = P_y w,\ z = P_z w,\ x \odot y + (-\mathbf{1})^{m\ell} \odot z = \mathbf{0}^{m\ell},\ P_{\mathbf{add}} w = \mathbf{0}^{m\ell}$$

and $\mathcal{P}$ is honest, $\mathcal{V}$ always accepts.

Next, soundness is argued by the following lemma.

Lemma 4.12. *Let e be a positive integer such that $e < d/4$. Suppose that there exists no $\overline{\alpha}$ such that $C(\overline{\alpha}) = 1$. Then, for any maliciously formed oracle U^* and any malicious prover strategy, the verifier rejects except with at most $(e + 6)/|\mathbb{F}|^\sigma + (1 - e/n)^t + 5((e + 2k)/n)^t$ probability.*

The proofs of the preceding two lemmas follow analogously to the proofs of Lemma 4.9 and Lemma 4.10. The next lemma establishes the honest verifier zero-knowledge property.

Lemma 4.13. *If $k > \ell + t$, $\langle \mathcal{P}, \mathcal{V} \rangle$ is an* (honest verifier, perfect) zero-knowledge IPCP.

Proof: To demonstrate zero-knowledge against honest verifier, we need to provide a simulator S that can given the randomness provided by the honest verifier $\mathcal{V}$, be able to generate a transcript. For every $h \in [\sigma]$, the simulator first generates:

- random polynomial $q^{\mathbf{add}}_h$ of degree $< k + \ell - 1$ such that $\sum_{c\in[\ell]} q^{\mathbf{add}}_h(\zeta_c) = 0$.
- for $a \in \{x, y, z\}$, random polynomial q^a_h of degree $< k+\ell-1$ such that $\sum_{c\in[\ell]} q^a_h(\zeta_c) = 0$.
- random polynomial $p_{0,h}$ of degree $< 2k - 1$ such that $p_{0,h}(\zeta_c) = 0$ for every $c \in [\ell]$.
- random vector $v_h \in \mathbb{F}^n$.

Next, it samples random elements from $\mathbb{F}$ for $U^x_h[j], U^y_h[j], U^z_h[j], U^w_h[j]$, for every $j \in Q$. Finally, given the random challenges from $\mathcal{V}$, it sets $u'_h[j], u^{\mathbf{add}}_h[j], u^x_h[j], u^y_h[j], u^z_h[j], u^0_h[j]$ as follows:

- $u'_h[j] = \sum_{i=1}^{4m} r_h[j] \cdot U_{i,j} - v_h[j]$.
- $u^{\mathbf{add}}_h[j] = \sum_{i=1}^m r^{\mathbf{add}}_{h,i}(\eta_j) \cdot U^w_{i,j} - q^{\mathbf{add}}_h(\eta_j)$
- for $a \in \{x, y, z\}$, $u^a_h[j] = \sum_{i=1}^m r^a_{h,i}(\eta_j) \cdot U^a_{i,j} + \sum_{i=m+1}^{2m} r^a_{h,i}(\eta_j) \cdot U^w_{i-m,j} - q^a_h(\eta_j)$.
- $u^0_h[j] = \sum_{i=1}^m (r^q_h)_i \cdot \left[U^x_{i,j} \cdot U^y_{i,j} - U^z_{i,j}\right] - p_{0,h}(\eta_j)$.

Our simulation achieves perfect zero knowledge. This follows from the fact that in an honest execution with the prover $\mathcal{P}$, the distribution of $\{U_h^x[j], U_h^y[j], U_h^z[j], U_h^w[j]\}_{j\in Q}$ are uniformly distributed and given that $u'_h, u_h^{\mathbf{add}}, u_h^x, u_h^y, u_h^z, u_h^0$ are uniformly chosen, the polynomials $q_h^{\mathbf{add}}, q_h^x, q_h^y, q_h^z, p_{0,h}$ and the vector v_h are uniformly distributed in their respective spaces. □

The following theorem follows from the construction described above and the preceding Lemmas.

Theorem 4.7. *Fix parameters n, m, ℓ, k, t, e such that $e < (n-k)/4$. Let $\mathrm{C} : \mathbb{F}^{n_i} \to \mathbb{F}$ be an arithmetic circuit of size s, where $|\mathbb{F}| \geq \ell + n$, $m \cdot \ell > n_i + s$ and $k > \ell + t$. Then protocol* ZKIPCP$(\mathrm{C}, \mathbb{F})$ *satisfies the following:*

- Completeness: *If $\overline{\alpha}$ is such that $C(\overline{\alpha}) = 1$ and oracle π is generated honestly as described in the protocol, then* $\Pr[(\mathcal{P}(\mathrm{C}, w), \mathcal{V}^{\pi}(\mathrm{C})) = 1] = 1$.
- Soundness: *If there is no $\overline{\alpha}$ is such that $C(\overline{\alpha}) = 1$, then for every (unbounded) prover strategy $\mathcal{P}^*$ and every $\tilde{\pi} \in \mathbb{F}^{4mn}$, $Pr[(\mathcal{P}^*, \mathcal{V}^{\tilde{\pi}}(x)) = 1] \leq (e+6)/|\mathbb{F}|^{\sigma} + (1-e/n)^t + 5((e+2k)/n)^t$.*
- Zero Knowledge: *For every adversary verifier $\mathcal{V}^*$, there exists a simulator $\mathcal{S}$ such that the output of $S^{\mathcal{V}^*}(\mathrm{C})$ is distributed identically to the view of $\mathcal{V}$ in the $(\mathcal{P}(\mathrm{C}, w), \mathcal{V}^{\pi}(\mathrm{C}))$.*
- Complexity: *The number of field $\mathbb{F}$ operations performed is* poly$(|\mathrm{C}|, n)$. *The number of field elements communicated by $\mathcal{P}$ to $\mathcal{V}$ is $\sigma \cdot n + 4 \cdot \sigma \cdot (k + \ell - 1) + \sigma \cdot (2 \cdot k - 1)$ whereas $\mathcal{V}$ reads t symbols from $\mathbb{F}^{4m+5\sigma}$.*

5 FROM ZKIPCP TO ZK

In this section we describe variants of known transformations from (sublinear) zero-knowledge PCP to (sublinear) zero-knowledge argument. The latter can either be interactive using collision-resistant hash functions, or non-interactive in the random oracle model.

5.1 The Interactive Variant

General transformations from (non-interactive) ZKPCP to (interactive) ZK arguments that make a black-box use of collision-resistant hash functions were given in [31, 34]. Here we address the more general case of ZKIPCP, where in addition to the proof oracle there is additional interaction between the prover and the verifier.

Using the ZKIPCP, an honest-verifier ZK protocol proceeds as follows. The prover commits to each entry of the proof oracle using a statistically hiding commitment scheme and then compresses the commitment using a Merkle hash tree (cf. Section 2.1). Note that both steps can be realized by making a black-box use of any family $\mathcal{H}$ of collision-resistant hash functions. The rest of the ZK protocol mimics the ZKIPCP, where the prover opens the committed values that correspond to the verifier's queries. Malicious verifiers can be handled using standard techniques, as done in [31, 34]. See full version for more details.

The communication complexity of the ZK argument includes the communication complexity of the ZKIPCP protocol and communication resulting from committing the oracle Π and decommitting to the queries Q.

5.2 The Non-Interactive Variant

It is possible to directly compile our previous protocol into a non-interactive protocol using a random oracle, where the verifier's messages are emulated by applying the random oracle on the partial transcript in each round following the Fiat-Shamir transform [19]. A formal description and analysis of this transformation is presented in [9] for interactive oracle proofs (IOP) model which generalizes (public-coin) IPCP.

In slight more detail, in this transformation the prover uses the random oracle to generate the verifier's messages and complete the execution (computing its own messages) based on the emulated verifier's messages, where instead of using an oracle, the prover commits to its proof and messages using Merkle hash trees. Completeness follows directly. If we start with an IOP that additionally is zero-knowledge (ZKIPCP in our case), [9] show that this transformation preserves (statistical) zero-knowledge property. Namely, the resulting protocol can be proved to be zero-knowledge in the random-oracle model.

In [9], the soundness of the transformed protocol is shown to essentially match the soundness of the original protocol up to an additive term that roughly depends on the product of q^2 and $2^{-\lambda}$ where q is an upper bound on the number of queries made to the random oracle by a malicious prover and λ is the output length of the random oracle. More precisely, [9] relates the soundness of the transformed protocol to the state restoration soundness of the underlying IPCP and collision-probability of queries to the random oracle. State-restoration soundness refers to the soundness of the IOP protocol against cheating prover strategies that may rewind the verifier back to any previously seen state, where every new continuation from a state invokes the next-message function of the verifier with fresh randomness. In [9], they show that for any (IOP) the state-restoration soundness of an IOP protocol is bounded by $\binom{T}{k(x)} \cdot \epsilon(x)$ and the soundness of the transformed protocol is $\binom{T}{k(x)} \cdot \epsilon(x) + 3(T^2+1) \cdot 2^{-\lambda}$ where T bounds the number of queries made by cheating provers to the random oracle, $k(x)$ is the round complexity of the IOP and $\epsilon(x)$ is the (standard) soundness of the IOP.

In the full version, we tighten the analysis presented in [9] for the particular ZKIPCP constructed in Section 4.7 and show that the soundness of the transformed protocol is $T \cdot \epsilon(x) + 3(T^2+1) \cdot 2^{-\lambda}$ where $\epsilon(x)$ is the soundness of the ZKIPCP, T bounds the number of queries made by cheating provers to the random oracle and λ is the output length of the random oracle.

5.3 Sublinear Zero-Knowledge Argument

In this section, we describe how to set the parameters of our zero-knowledge argument protocol to obtain communication that is sublinear in the circuit size. We consider first an arithmetic circuit over a large field $\mathbb{F}$. Following our transformation, the communication complexity of the zero-knowledge protocol that is compiled based on our ZKIPCP is

$$[n \cdot \sigma + 4 \cdot \sigma \cdot (k + \ell - 1) + \sigma \cdot (2 \cdot k - 1) + t \cdot (4 \cdot m + 5 \cdot \sigma)] \cdot \lceil \log |\mathbb{F}| \rceil + t \cdot \lceil \log n \rceil \cdot h$$

where h is the output length of the hash-function. For security parameter κ, when $\mathbb{F}$ is large (i.e. $|\mathbb{F}| > O(2^{\kappa})$) we can set $\sigma = 1$

for $2^{-\kappa}$-security. The other terms in the soundness are $(1-e/n)^t$ and $((e+2k)/n)^t$ where $e < d/4 = (n-k)/4$ and $k = t + \ell$. We can optimize our parameters by setting $k = O(\kappa)$, $n = O(k)$ and $e = (n-k)/4$ to achieve $2^{-\kappa}$-security. The next constraint on the parameters requires that $m \cdot \ell$ is at least as large as the witness size, which is $O(s)$ where s is the number of (addition and multiplication) gates in the circuit. Optimal values for m and ℓ can be obtained by equating the dominating costs in the communication, namely, $O(\ell)$ and $O(t \cdot m)$ which implies $\ell = O(\sqrt{s\kappa})$ and $m = O(\sqrt{s/\kappa})$. Overall the communication complexity with these parameters will be $O(\sqrt{s\kappa}\lceil \log |\mathbb{F}| \rceil)$. In the full version we show that, the communication complexity of proving the satisfiability of an arithmetic circuit of size $s > 30000$ over a finite field $\mathbb{F}$, $|\mathbb{F}| \geq 2^{128}$ with soundness error 2^{-40} consists of roughly $95\sqrt{s}$ field elements (or $70\sqrt{s}$ elements under Conjecture 4.1). If we want security 2^{-80}, the communication is roughly $140\sqrt{s}$ (or $120\sqrt{s}$ under Conjecture 4.1).

Optimizing for Boolean circuits requires additional effort. In this case there is yet another degree of freedom, namely the field size. The only constraint on the field size is that there are sufficiently many evaluation points, namely, we need $|\mathbb{F}| \geq \ell + n$. For a given statistical security parameter κ, we can show that for sufficiently large circuits the optimal communication complexity is $O(k \cdot \sqrt{s \log s})$ where $|\mathbb{F}| = O(\sqrt{s})$. For smaller circuits, the optimal value requires a more careful analysis as the low order terms are significant and we present our results in the next section.

5.4 Multi-Instance Amortization

If we want to prove that $C(x_i, \cdot)$ is satisfiable for N public inputs x_i, we can simplify our ZKIPCP construction as follows. The prover first computes the combined witness $w = w_1, \ldots, w_N$ that is comprised of N witnesses, each is computed as in the single instance case. Next, it arranges the witnesses in blocks of size $\ell = N$, where block j contains the j^{th} bits of each of the N witnesses. The public inputs x_i define public blocks. The number of non-public blocks equals the size of the witness of a single instance, which is $m = |w_i| = O(s)$. The prover then encodes the blocks of messages into $U \in L^m$.

Even for moderately large N, the multi-instance variant provides significant savings in both computational and communication costs. This is because we do not need to rearrange the wire values as we do in the single instance case. The total communication complexity is

$$[n \cdot \sigma + 4 \cdot \sigma \cdot (2N + t) + 2 \cdot \sigma \cdot (N + t - 2) + t \cdot (4 \cdot s + 5 \cdot \sigma)] \cdot \lceil \log |\mathbb{F}| \rceil + t \cdot \lceil \log n \rceil \cdot h.$$

For sufficiently large fields, we can set $t = O(\kappa)$ where if $N > O(\kappa^2)$, then the proof length is shorter than sN bits. Note that this threshold is independent of the circuit size.

6 IMPLEMENTATION AND RESULTS

We implemented our protocol in C++ using Shoup's NTL library for the finite field operations. We used `BOOST::MPI` for emulating a network. To pick the evaluation points, we chose a prime that had sufficiently large power of two roots of unity and set η_i and ζ_j values to be roots of unity. This enabled us to perform interpolation and evaluation using inverse FFT and FFT operations. We ran our experiments on Intel Core i7-4720HQ CPU 2.60 GHz, 4 cores, 8 GiB RAM. For our collision resistant hash function we used SHA-256.

We primarily compare our work with ZKB++/ZKBoo [13, 22] that qualitatively match our result in most aspects. The crucial advantage of our approach over ZKBoo/ZKB++ is that our communication is sublinear in the circuit size whereas ZKBoo/ZKB++ incurs communication that is proportional in the circuit size. Moreover, in the amortized setting, our approach provides significantly better communication cost and runtimes over ZKBoo/ZKB++.

The primary Boolean predicate we used to demonstrate our implementation was verifying a SHA-256 certificate as it is the common benchmark used in prior works. Namely, on a common input a 256-bit string y and a private input x of the prover, the prover convinces the verifier that $\text{SHA256}(x) = y$.

Optimizations. A standard computation of the SHA certificate, involves AND and XOR gates and addition modulo 2^{32} gates. One approach is to simply realize the addition modulo 2^{32} gates using Boolean AND and XOR gates. We follow a different approach, where we express the addition modulo 2^{32} gate consistency as a linear constraint over the bits of the inputs and output of the gate. Following Section 4.5, this can be efficiently realized if we rely on a prime field larger than 2^{33}. However, as mentioned in the previous section, to obtain optimal communication for a given witness size requires choosing a field of a specific size. To handle this, we incorporate these addition gates by considering a word size of $\lceil \log |\mathbb{F}| \rceil$ and performing 32-bit additions using arithmetic over the smaller word size. Following these optimizations, results in a witness size of 33928 bits for the SHA-256 certificate (for $|\mathbb{F}| \geq 2^{14}$). We ran our protocol for different circuit sizes and for each size, we ran ad-hoc optimizers to obtain optimal parameters for soundness 2^{-40}. We remark that a tighter soundness error for the tests described in Section 4 which in turn is used in our ZKIPCP can be obtained by discarding the last inequality in Lemma 4.2, 4.6 and 4.8. We relied on these better bounds in our optimizer (as opposed to the cleaner bounds that appear in the Lemma statements). For the case of 2^{-80} soundness error the communication and computation costs doubles (as in [13, 22]). For boolean circuits, the quadratic constraints only involves checking if each element of the witness is binary and we can simplify the test in Section 4.3 by eliminating x, y, z and having the prover compute $p_0(\bullet) = \sum_i r_i \cdot (p_i^w(\bullet) \cdot p_i^w(\bullet) - p_i^w(\bullet))$.

In Figure 1, we compare the prover and verifier running times for verifying circuits of sizes varying from 2048 gates to 400000 gates.[4] The computational complexity of both the prover and the verifier in the single instance setting are proportional to $O(s \log s)$ field operations, where s is the circuit size. The optimal field size can be asymptotically shown to be $O(\log s)$ resulting in a overall computational complexity of $O(s \log^2 s)$. We remark here that if we make uniformity assumptions on the circuit, then the verifier's computational complexity becomes sublinear in the circuit size. In fact, the multi-instance setting can be seen as a uniformity assumption and here the verifier's complexity is indeed smaller than the computational complexity.

[4]Note that our proof length and computation times are not influenced by circuit topology and only depend on the witness size which in turn depends only on the number of gates. In fact, in the case of Boolean circuits, they are independent of gate composition (assuming the circuit comprises of only XOR and AND gates).

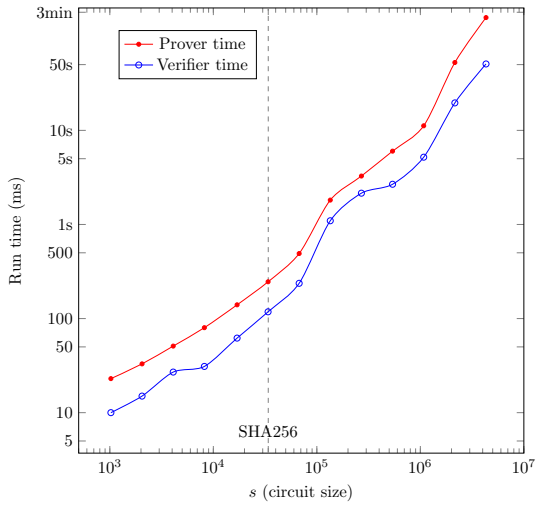

Figure 1: Prover and verifier running times for verifying a single instance of different circuit sizes.

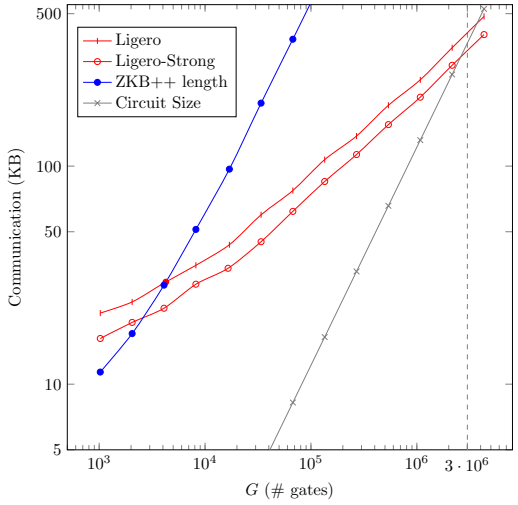

Figure 2: Proof lengths for proving a single instance of difference circuit sizes.

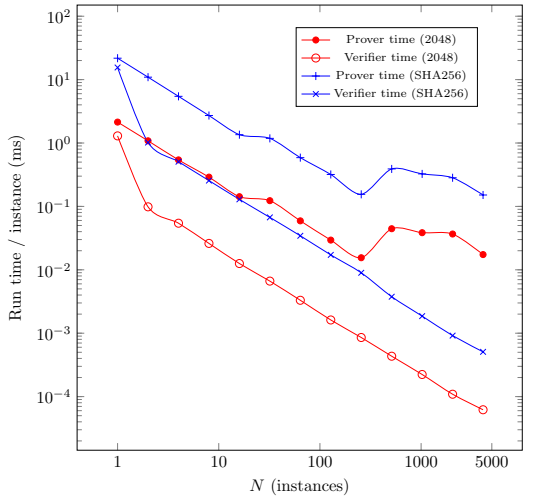

Figure 3: Amortized prover and verifier running times for verifying multiple instances of 2048 and gate circuit and SHA-256 circuit.

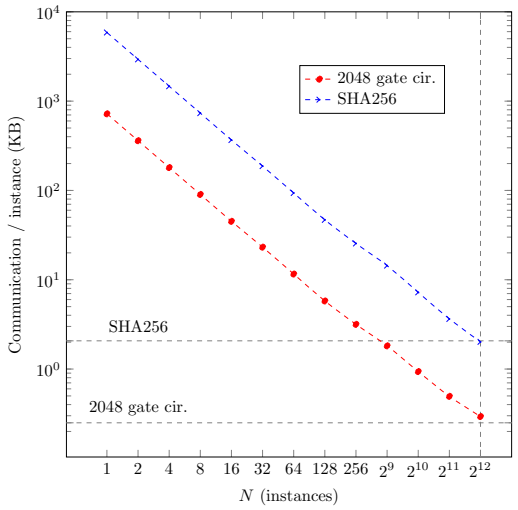

Figure 4: Amortized proof lengths for multiple instances of 2048 gate circuit and SHA-256 circuit.

In Figure 2, we provide the communication complexity in kilobytes (KB) of our zero-knowledge argument. We plot two instantiations of our protocol. We provide communication cost for our provable variant labelled Ligero and the variant that assumes Conjecture 4.1 labelled as Ligero-Strong. We observe that Ligero-Strong yields a 20% reduction in communication on the average. The communication for a SHA certificate is 44KB with Ligero and 34KB with Ligero-Strong. We can also see in Figure 2 that beyond 3 million gates our communication cost is smaller than the circuit size.

We compare our complexity with ZKB++ [13, 22]). We first note that the complexity of ZKB++ only depends on the number of Boolean AND gates (as XOR's are for free). In our implementation we relied on prime fields and our communication cost depends on the number of AND and XOR gates. However, had we relied on extension field of $\mathbb{GF}_2$, we can eliminate the dependence on XOR gates similar to ZKB++. In this variant of our protocol, each AND gate will incur 3 bits in the witness. In Figure 2, in order to make a fair comparison with ZKB++ for a circuit of size s, we plot the communication cost incurred by the ZKB++ protocol for a circuit of size $2s/3$. The idea is that for a circuit comprising of $2s/3$ AND gates the cost of ZKB++ is compared with the communication cost of our protocol assuming a characteristic-2 FFT implementation. The threshold for which our approach incurs lesser communication than ZKB++ is roughly 3000 (AND) gates.

In Figures 3 and 4 we provide our prover and verifier running times and communication for the multi-instance version of our protocol. We take a 2048 gate circuit and the SHA-256 circuit to illustrate our performance. The heaviest part of the verification involves performing FFTs over domains of sizes N and s where s is the circuit size. Since we considered 1 to 4096 instances, even for moderately sized circuits the FFT over domain of size s dominates the cost. The prover complexity, on the other hand, varies as $Ns \log s$. We see a reduction in the amortized prover's cost per instance with periodic jumps because we perform FFT over a larger domain, which is usually set to a power of 2.

The communication complexity varies additively in N and s. The amortized communication cost per instance decreases linearly because, similar to the verifier complexity, s dominates the complexity until N becomes significant compared to s.

7 CONCLUSIONS AND FUTURE WORK

We designed and implemented a zero-knowledge argument for NP that simultaneously offers good concrete efficiency and sublinear communication in the circuit size. As the computational complexity of our protocol is dominated by polynomial evaluations and interpolations, we can rely on efficient FFT implementations for minimizing its computational cost. In the following we mention some additional optimizations that we have not fully explored.

The current implementation relies on prime fields. This allows us to optimize arithmetics over integers by considering a sufficiently large prime. Moreover, the witness includes two bits per gate for both XOR and AND gates. If we instead rely on characteristic 2 fields, then the witness size will require three bits per AND gate and 0 bits for XOR gates. Hence there is a tradeoff in choosing between the two options. It is also unclear how the FFT algorithms compare for characteristic 2 and prime fields, though fast implementations for the characteristic 2 case are known [10, 20].

The verification of our zero-knowledge argument needs to evaluate a polynomial on a subset of the points in the domain. We currently implement this by having the verifier evaluate the polynomial on the entire domain via FFT and extract the points in this subset. Improving this will improve the verifier's efficiency. Relying on GPU for FFT computations can also bring significant savings. Finally, one can exploit a repetitive circuit structure ("uniformity") to reduce verification time. We currently only take advantage of this for reducing the amortized cost of verifying multiple evaluations of the same circuit. On the prover side, more than 50% of the total prover time for small circuits and 66% for large circuits is spent on computing the hashes of the leaves of the Merkle tree. This leaves room for improvement by relying on space- and cache-efficient hashing algorithms.

Finally, it would be interesting to explore the concrete efficiency of other approaches to lightweight sublinear zero-knowledge arguments. In particular, one could consider constructions of PCPs based on bivariate polynomials such as the one of Polishchuk and Spielman [40] (see [8] for work in this direction), or the zero-knowledge PCP obtained by applying our general transformation to the MPC protocol from [17]. This type of constructions can be further simplified by applying an interactive procedure for testing linear constraints as we do in Section 4.2.

Acknowledgments. We thank Eli Ben-Sasson, Swastik Kopparty, abhi shelat, Salil Vadhan, and Mike Walfish for useful discussions and pointers, the anonymous CCS reviewers for helpful comments, and Victor Shoup for his assistance with the NTL library.

The first and last authors were supported by Google Faculty Research Grant and NSF Awards CNS-1526377 and CNS-1618884. The second author was supported by the European Research Council under the ERC consolidators grant agreement n. 615172 (HIPS), and by the BIU Center for Research in Applied Cryptography and Cyber Security in conjunction with the Israel National Cyber Bureau in the Prime Minister's Office. The third author was supported by a DARPA/ARL SAFEWARE award, DARPA Brandeis program under Contract N66001-15-C-4065, NSF Frontier Award 1413955, NSF grants 1619348, 1228984, 1136174, and 1065276, ERC grant 742754, NSF-BSF grant 2015782, ISF grant 1709/14, BSF grant 2012378, a Xerox Faculty Research Award, a Google Faculty Research Award, an equipment grant from Intel, and an Okawa Foundation Research Grant. This material is based upon work supported by the Defense Advanced Research Projects Agency through the ARL under Contract W911NF-15-C-0205. The views expressed are those of the authors and do not reflect the official policy or position of Google, the Department of Defense, the National Science Foundation, or the U.S. Government.

REFERENCES

[1] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. 1998. Proof Verification and the Hardness of Approximation Problems. *J. ACM* 45, 3 (1998), 501–555.

[2] Sanjeev Arora and Shmuel Safra. 1998. Probabilistic Checking of Proofs: A New Characterization of NP. *J. ACM* 45, 1 (1998), 70–122.

[3] László Babai, Lance Fortnow, Leonid A. Levin, and Mario Szegedy. 1991. Checking Computations in Polylogarithmic Time. In *STOC*. 21–31.

[4] Eli Ben-Sasson, Iddo Bentov, Alessandro Chiesa, Ariel Gabizon, Daniel Genkin, Matan Hamilis, Evgenya Pergament, Michael Riabzev, Mark Silberstein, Eran Tromer, and Madars Virza. 2017. Computational Integrity with a Public Random String from Quasi-Linear PCPs. In *EUROCRYPT*. 551–579.

[5] Eli Ben-Sasson, Iddo Bentov, Ynon Horesh, and Michael Riabzev. 2017. Scalable, transparent, and post-quantum secure computational integrity. Manuscript. (2017). Slides at https://people.eecs.berkeley.edu/~alexch/docs/pcpip_bensasson.pdf.

[6] Eli Ben-Sasson, Alessandro Chiesa, Ariel Gabizon, Michael Riabzev, and Nicholas Spooner. 2016. Short Interactive Oracle Proofs with Constant Query Complexity, via Composition and Sumcheck. *IACR Cryptology ePrint Archive* 2016 (2016), 324.

[7] Eli Ben-Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. 2014. Zerocash: Decentralized Anonymous

Payments from Bitcoin. In *2014 IEEE Symposium on Security and Privacy, SP 2014, Berkeley, CA, USA, May 18-21, 2014*. 459–474.
[8] Eli Ben-Sasson, Alessandro Chiesa, Daniel Genkin, and Eran Tromer. 2013. On the concrete efficiency of probabilistically-checkable proofs. In *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*. 585–594.
[9] Eli Ben-Sasson, Alessandro Chiesa, and Nicholas Spooner. 2016. Interactive Oracle Proofs. In *TCC*. 31–60.
[10] Eli Ben-Sasson, Matan Hamilis, Mark Silberstein, and Eran Tromer. 2016. Fast Multiplication in Binary Fields on GPUs via Register Cache. In *Proceedings of the 2016 International Conference on Supercomputing, ICS 2016, Istanbul, Turkey, June 1-3, 2016*. 35:1–35:12.
[11] Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. 2013. Recursive composition and bootstrapping for SNARKS and proof-carrying data. In *STOC*. 111–120.
[12] Nir Bitansky, Alessandro Chiesa, Yuval Ishai, Rafail Ostrovsky, and Omer Paneth. 2013. Succinct Non-interactive Arguments via Linear Interactive Proofs. In *TCC*. 315–333.
[13] Melissa Chase, David Derler, Steven Goldfeder, Claudio Orlandi, Sebastian Ramacher, Christian Rechberger, Daniel Slamanig, and Greg Zaverucha. 2017. Post-Quantum Zero-Knowledge and Signatures from Symmetric-Key Primitives. *IACR Cryptology ePrint Archive* 2017 (2017), 279.
[14] Hao Chen and Ronald Cramer. 2006. Algebraic Geometric Secret Sharing Schemes and Secure Multi-Party Computations over Small Fields. In *CRYPTO*. 521–536.
[15] Graham Cormode, Michael Mitzenmacher, and Justin Thaler. 2012. Practical verified computation with streaming interactive proofs. In *ITCS*. 90–112.
[16] Ivan Damgård and Yuval Ishai. 2006. Scalable Secure Multiparty Computation. In *CRYPTO*. 501–520.
[17] Ivan Damgård, Yuval Ishai, and Mikkel Krøigaard. 2010. Perfectly Secure Multiparty Computation and the Computational Overhead of Cryptography. In *EUROCRYPT*. 445–465.
[18] George Danezis, Cédric Fournet, Markulf Kohlweiss, and Bryan Parno. 2013. Pinocchio coin: building zerocoin from a succinct pairing-based proof system. In *PETShop'13, Proceedings of the 2013 ACM Workshop on Language Support for Privacy-Enhancing Technologies, Co-located with CCS 2013, November 4, 2013, Berlin, Germany*. 27–30.
[19] Amos Fiat and Adi Shamir. 1986. How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In *CRYPTO*. 186–194.
[20] Shuhong Gao and Todd Mateer. 2010. Additive Fast Fourier Transforms over Finite Fields. *IEEE Trans. Inf. Theor.* 56, 12 (Dec. 2010), 6265–6272.
[21] Rosario Gennaro, Craig Gentry, Bryan Parno, and Mariana Raykova. 2013. Quadratic Span Programs and Succinct NIZKs without PCPs. In *EUROCRYPT*. 626–645.
[22] Irene Giacomelli, Jesper Madsen, and Claudio Orlandi. 2016. ZKBoo: Faster Zero-Knowledge for Boolean Circuits. In *USENIX*. 1069–1083.
[23] Shafi Goldwasser, Yael Tauman Kalai, and Guy N. Rothblum. 2015. Delegating Computation: Interactive Proofs for Muggles. *J. ACM* 62, 4 (2015), 27:1–27:64.
[24] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. 1985. The Knowledge Complexity of Interactive Proof-Systems (Extended Abstract). In *STOC*. 291–304.
[25] Vipul Goyal, Yuval Ishai, Mohammad Mahmoody, and Amit Sahai. 2010. Interactive Locking, Zero-Knowledge PCPs, and Unconditional Cryptography. In *CRYPTO*. 173–190.
[26] Jens Groth. 2009. Linear Algebra with Sub-linear Zero-Knowledge Arguments. In *CRYPTO*. 192–208.
[27] Jens Groth. 2010. Short Pairing-Based Non-interactive Zero-Knowledge Arguments. In *ASIACRYPT*. 321–340.
[28] Yuval Ishai, Eyal Kushilevitz, and Rafail Ostrovsky. 2007. Efficient Arguments without Short PCPs. In *CCC*. 278–291.
[29] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. 2007. Zero-knowledge from secure multiparty computation. In *STOC*. 21–30.
[30] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. 2009. Zero-Knowledge Proofs from Secure Multiparty Computation. *SIAM J. Comput.* 39, 3 (2009), 1121–1152.
[31] Yuval Ishai, Mohammad Mahmoody, and Amit Sahai. 2012. On Efficient Zero-Knowledge PCPs. In *TCC*. 151–168.
[32] Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. 2008. Founding Cryptography on Oblivious Transfer - Efficiently. In *CRYPTO*. 572–591.
[33] Yuval Ishai, Manoj Prabhakaran, and Amit Sahai. 2009. Secure Arithmetic Computation with No Honest Majority. In *TCC*. 294–314. Full version: IACR Cryptology ePrint Archive 2008: 465.
[34] Yuval Ishai and Mor Weiss. 2014. Probabilistically Checkable Proofs of Proximity with Zero-Knowledge. In *TCC*. 121–145.
[35] Yael Tauman Kalai and Ran Raz. 2008. Interactive PCP. In *ICALP*. 536–547.
[36] Joe Kilian. 1992. A Note on Efficient Zero-Knowledge Proofs and Arguments (Extended Abstract). In *STOC*. 723–732.
[37] Carsten Lund, Lance Fortnow, Howard J. Karloff, and Noam Nisan. 1990. Algebraic Methods for Interactive Proof Systems. 2–10.
[38] Ralph C. Merkle. 1989. A Certified Digital Signature. In *CRYPTO*. 218–238.
[39] Silvio Micali. 1994. CS Proofs (Extended Abstracts). In *FOCS*. 436–453.
[40] Alexander Polishchuk and Daniel A. Spielman. 1994. Nearly-linear size holographic proofs. In *Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing, 23-25 May 1994, Montréal, Québec, Canada*. 194–203.
[41] Omer Reingold, Guy N. Rothblum, and Ron D. Rothblum. 2016. Constant-round interactive proofs for delegating computation. In *STOC*. 49–62.
[42] Srinath T. V. Setty, Benjamin Braun, Victor Vu, Andrew J. Blumberg, Bryan Parno, and Michael Walfish. 2013. Resolving the conflict between generality and plausibility in verified computation. In *Eighth Eurosys Conference 2013, EuroSys '13, Prague, Czech Republic, April 14-17, 2013*. 71–84.
[43] Srinath T. V. Setty, Richard McPherson, Andrew J. Blumberg, and Michael Walfish. 2012. Making argument systems for outsourced computation practical (sometimes). In *19th Annual Network and Distributed System Security Symposium, NDSS 2012, San Diego, California, USA, February 5-8, 2012*.
[44] Adi Shamir. 1990. IP=PSPACE. 11–15.
[45] Justin Thaler. 2013. Time-Optimal Interactive Proofs for Circuit Evaluation. In *CRYPTO*. 71–89.
[46] Victor Vu, Srinath T. V. Setty, Andrew J. Blumberg, and Michael Walfish. 2013. A Hybrid Architecture for Interactive Verifiable Computation. In *SP*. 223–237.
[47] Michael Walfish and Andrew J. Blumberg. 2015. Verifying computations without reexecuting them. *Commun. ACM* 58, 2 (2015), 74–84.
[48] Yupeng Zhang, Daniel Genkin, Jonathan Katz, Dimitrios Papadopoulos, and Charalampos Papamanthou. 2017. vSQL: Verifying Arbitrary SQL Queries over Dynamic Outsourced Databases. In *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017*. 863–880.

A GENERALIZED INTERLEAVED CODE TESTING

In this section we present a generalized version of the testing algorithm that uses σ linear combinations to amplify soundness. This algorithm is useful for obtaining better soundness over small fields.

Generalized-Test-Interleaved$(\mathbb{F}, L[n,k,d], m, t, \sigma; U)$

- **Oracle:** A purported L^m-codeword U. Depending on the context, we may view U either as a matrix in $\mathbb{F}^{m\times n}$ in which each row is a purported L-codeword, or as a sequence of n symbols $(U_1,\ldots,U_n)$, $U_i \in \mathbb{F}^m$.
- **Parameters:**
  - Probing parameter $t < n$ (number of symbols U_j read by $\mathcal{V}$).
  - Repetition parameter σ (number of random linear combinations).
- **Interactive testing:**
  (1) $\mathcal{V}$ picks σ random linear combinations $r_1,\ldots,r_\sigma \in \mathbb{F}^m$ and sends them to $\mathcal{P}$.
  (2) $\mathcal{P}$ responds with $w_h = r_h^T U \in \mathbb{F}^n$, $h = 1,\ldots,\sigma$.
  (3) $\mathcal{V}$ queries a set $Q \subset [n]$ of t random symbols U_j, $j \in Q$.
  (4) $\mathcal{V}$ accepts iff all w_h are in L and are consistent with U_Q and r_h. That is, for every $j \in Q$ and $1 \le h \le \sigma$, we have $\sum_{i=1}^m (r_h)_j \cdot U_{i,j} = (w_h)_j$.

Lemma A.1. *If $U \in L^m$ and $\mathcal{P}$ is honest, then $\mathcal{V}$ always accepts.*

Lemma A.2. *Let e be a positive integer such that $e < d/4$. Suppose $d(U^*, L^m) > e$. Then, for a random w^* in the row-span of U^*, we have*

$$\Pr[d(w^*, L) \le e] \le (e+1)/|\mathbb{F}|^\sigma.$$

Theorem A.1. *Let e be a positive integer such that $e < d/4$. Suppose $d(U^*, L^m) \ge e$. Then, for any malicious $\mathcal{P}$ strategy, the oracle U^* is rejected by $\mathcal{V}$ except with $\le (1-e/n)^t + (e+1)/|\mathbb{F}|^\sigma$ probability.*

A proof of a generalization of this test appears in Section C.

B AFFINE INTERLEAVED CODE TESTING

Affine-Test-Interleaved$(\mathbb{F}, L[n,k,d], m, t; U, u')$

- **Oracle:** A purported L^m-codeword U and an additional auxiliary row vector $u' \in \mathbb{F}^n$.
- **Interactive testing:**
  (1) $\mathcal{V}$ picks random linear combinations $r \in \mathbb{F}^m$ and sends r to $\mathcal{P}$.
  (2) $\mathcal{P}$ responds with $w = r^T U + u' \in \mathbb{F}^n$.
  (3) $\mathcal{V}$ queries a set $Q \subset [n]$ of t random symbols $U_j, j \in Q$, as well as $u'_j, j \in Q$.
  (4) $\mathcal{V}$ accepts iff $w \in L$ and w is consistent with U_Q, u'_Q, and r. That is, for every $j \in Q$ we have $\sum_{i=1}^m r_j \cdot U_{i,j} + u'_j = w_j$.

Lemma B.1. *If $U \in L^m$, $u' \in L$, and $\mathcal{P}$ is honest, then $\mathcal{V}$ always accepts.*

Lemma B.2. *Let e be a positive integer such that $e < d/4$. Suppose $d(U^*, L^m) > e$. Then, for arbitrary $u' \in \mathbb{F}^n$ and a random w^* in the row-span of U^*, we have $\Pr[d(w^*, L) \le e] \le (e+1)/|\mathbb{F}|$.*

Theorem B.1. *Let e be a positive integer such that $e < d/4$. Suppose $d(U^*, L^m) \ge e$. Then, for an arbitrary $u' \in \mathbb{F}^n$ and any malicious $\mathcal{P}$ strategy, the oracle U^* is rejected by $\mathcal{V}$ except with $\le (1-e/n)^t + (e+1)/|\mathbb{F}|$ probability.*

A proof of a generalization of this test appears in the next section.

C GENERALIZED AFFINE INTERLEAVED CODE TESTING

For the purpose of obtaining a zero-knowledge IPCP, the following "affine" variant of **Test-Interleaved** is useful. Whenever $\mathcal{V}$ requests a random linear combination of the rows of U, this linear combination will be masked with an additional blinding vector $u' \in \mathbb{F}^n$. The vector u', which is also given as part of the proof oracle, will be picked by an honest $\mathcal{P}$ at random from L and will therefore hide all information about U whose rows are from L. The soundness of the test should hold even when u' is adversarially chosen and is not necessarily a codeword. We generalize it further following the previous section to achieve better soundness by repetition.

Generalized-Affine-Test-Interleaved$(\mathbb{F}, L[n,k,d], m, t, \sigma; U, u')$

- **Oracle:** A purported L^m-codeword U and additional auxiliary row vectors $u'_1, \ldots, u'_\sigma \in \mathbb{F}^n$.
- **Interactive testing:**
  (1) $\mathcal{V}$ picks a random linear combinations $r_1, \ldots, r_\sigma \in \mathbb{F}^m$ and sends r to $\mathcal{P}$.
  (2) $\mathcal{P}$ responds with $w_h = r_h^T U + u'_h \in \mathbb{F}^n$, $h = 1, \ldots, \sigma$.
  (3) $\mathcal{V}$ queries a set $Q \subset [n]$ of t random symbols $U_j, j \in Q$, as well as $(u_\sigma)_j, j \in Q$.
  (4) $\mathcal{V}$ accepts iff all $w_h \in L$ and are consistent with $w_h$and U_Q, u_h, and r_h. That is, for every $j \in Q$ we have $\sum_{i=1}^m (r_h)j \cdot U_{i,j} + (u'_h)_j = w_j$.

Completeness follows directly from the description. A formal proof of our soundness is presented below.

Lemma C.1. *If $U \in L^m$, $u'_1, \ldots, u'_\sigma \in L$, and $\mathcal{P}$ is honest, then $\mathcal{V}$ always accepts.*

Lemma C.2. *Let e be a positive integer such that $e < d/4$. Suppose $d(U^*, L^m) > e$. Then, for arbitrary $u'_1, \ldots, u'_\sigma \in \mathbb{F}^n$ and a random w^* in the row-span of U^*, we have $\Pr[\forall\ h \in [\sigma], d(w^* + u'_h, L) \le e] \le (e+1)/|\mathbb{F}|^\sigma$.*

Proof: The proof of Lemma C.2 is almost identical to the proof of Lemma 4.2. The high level reason why the same argument works is that the decomposition $w^* = \alpha v^* + x$ where x is independent of α still holds even for $w^* + u'_h$ (since u'_h is a fixed vector, and so $u'_h + x$ is independent of α). We provide the full proof below.

Let L^* be the row-span of U^*. We consider two cases similar to our proof of Lemma 4.2.

Case 1: There exists $v^* \in L^*$ such that $d(v^*, L) > 2e$. In this case, we show that

$$\Pr_{w^* \in_R L^*}[\forall\ h \in [\sigma], d(w^* + u'_h, L) \le e] \le 1/|\mathbb{F}|^\sigma. \tag{2}$$

Indeed, using a basis for L^* that includes v^*, a random $w^* \in L^*$ can be written as $\alpha v^* + x$, where $\alpha \in_R \mathbb{F}$ and x is distributed independently of α. We argue that conditioned on any choice of x, there can be at most one choice of α such that $d(\alpha v^* + x + u'_h, L) \le e$. We can conclude the case from this as the probability over $r_1, \ldots, r_h$ that $d((r_h)^T U + u'_h, L) \le e$ holds for every h is at most $1/|\mathbb{F}|^\sigma$. This follows by observing that if $d(\alpha v^* + x_0 + u'_h, L) \le e$ and $d(\alpha' v^* + x_0 + u'_h, L) \le e$ for $\alpha \ne \alpha'$, then by the triangle inequality we have $d((\alpha - \alpha')v^*, L) \le 2e$. Since, by assumption $d(v^*, L) > 2e$ and this implies $d(v^*, L) > 2e$, we arrive at a contradiction.

Case 2: For every $v^* \in L^*, d(v^*, L) \le 2e$. We show that in this case $\Pr_{w^* \in_R L^*}[\forall\ h \in [\sigma], d(w^* + u'_h, L) \le e] \le (e+1)/|\mathbb{F}|^\sigma$. Let U_i^* be the i^th row of U^* and let $E_i = \Delta(U_i^*, L)$. Note that, since $2e < d/2$, each U_i^* can be written uniquely as $U_i^* = u_i + \chi_i$ where $u_i \in L$ and χ_i is nonzero exactly in its E_i entries. Let $E = \cup_{i=1}^m E_i$. Since $d(U^*, L^m) > e$, we have $|E| > e$. We show below that for $j \in E$, except with $1/|\mathbb{F}|$ probability over a random choice of w^* from L^*, either $j \in \Delta(w^* + u'_h, L)$ or $d(w^* + u'_h, L) > e$. First, we conclude the case and the proof of Lemma assuming this holds. We observe that this implies that with probability at most $1/|F|^\sigma$ over the choice of $r_1, \ldots, r_\sigma$, it holds that, for all $h, j \notin \Delta((r_h)^T U + u'_h, L)$ and $d((r_h)^T U + u'_h, L) \le e$. Taking a union bound over the first $e+1$ elements of E the claim follows.

Suppose $j \in E_i$ and fix an arbitrary $h \in [\sigma]$. As before, we write $w^* = \alpha U_i^* + x$ for $\alpha \in_R \mathbb{F}$ and x distributed independently of α. Condition on any possible choice x_0 of x. Define a bad set

$$B_j = \{\alpha : j \notin \Delta(\alpha U_i^* + x_0 + u_h, L) \wedge d(\alpha U_i^* + x_0 + u_h, L) \le e\}.$$

We show that $|B_j| \le 1$. Suppose for contradiction that there are two distinct $\alpha, \alpha' \in \mathbb{F}$ such that for $z = \alpha U_i^* + x_0 + u_h$ and $z' = \alpha' U_i^* + x_0 + u_h$ we have $d(z, L) \le e$, $d(z', L) \le e$, $j \notin \Delta(z, L)$, and $j \notin \Delta(z', L)$. Since $d > 4e$, for any z^* in the linear span of z and z' we have $j \notin \Delta(z^*, L)$. Since $(\alpha - \alpha')U_i^* = z - z'$ is in this linear span, we have $j \notin \Delta(U_i^*, L)$, in contradiction to the assumption $j \in E_i$. □

Theorem C.1. *Let e be a positive integer such that $e < d/4$. Suppose $d(U^*, L^m) \ge e$. Then, for arbitrary $u'_1, \ldots, u'_\sigma \in \mathbb{F}$ and any malicious $\mathcal{P}$ strategy, the oracle U^* is rejected by $\mathcal{V}$ except with $\le (1-e/n)^t + (e+1)/|\mathbb{F}|^\sigma$ probability.*