# Algebraic Reductions of Knowledge

Abhiram Kothapalli[1] and Bryan Parno[2]

[1]Carnegie Mellon University, akothapalli@cmu.edu
[2]Carnegie Mellon University, parno@cmu.edu

**Abstract**

Arguments of knowledge are powerful cryptographic primitives that allow a prover to demonstrate that it *knows* a satisfying witness to a prescribed statement. Tremendous progress has been made in developing efficient argument systems by leveraging homomorphic structure in an increasingly composable and recursive manner. However, the extent to which homomorphisms can be composed and manipulated in the service of efficient argument systems is still not well understood. To this end, we introduce *reductions of knowledge*, a generalization of arguments of knowledge, which reduce checking a statement in one relation to checking a derived statement in another, and better capture the composable and recursive nature of arguments over homomorphisms. We construct and study the tensor reduction, which is capable of reducing any homomorphic statement composed via the tensor product, and provides knowledge soundness unconditionally when working over vector spaces. We show that tensor reductions generalize a large class of prior recursive techniques including the ubiquitous sumcheck protocol. We additionally show that tensor reductions can be employed to construct reductions of knowledge with logarithmic communication for familiar linear algebraic statements, and in turn, these can be composed to recover a reduction of knowledge for $\mathsf{NP}$ with logarithmic communication.

## Contents

# 1 Introduction

Arguments of knowledge [GMR89] are powerful cryptographic primitives that allow a prover to demonstrate (in zero-knowledge) to a verifier that it *knows* a satisfying witness for a prescribed statement. Such a pattern of interaction has been shown to provide strong integrity and privacy guarantees that enable a large class of cryptographic applications [DLFKP16, SCG$^+$14, KMS$^+$16, ZKP15]. Since the foundational works in arguments of knowledge [GMR89, GKR15, Kil92, BOGG$^+$88], consistent progress has been made in optimizing both asymptotics [GGPR13, GKR15, KMP20, Set20, CHM$^+$20, MBKM19, GWC19], and practical performance [WTS$^+$18, PHGR13, BBB$^+$18], under various cryptographic assumptions.

Modern arguments are designed to leverage homomorphic structure to achieve better asymptotics and concrete efficiency. For example, a variety of arguments [CHM$^+$20, Set20, KMP20, RZ21] leverage matrix representations of $\mathsf{NP}$ to compress constraints via random linear combinations. Batching techniques leverage the homomorphic properties of commitments to compress checks over polynomials [CHM$^+$20, KMP20], vectors [BBB$^+$18], and even $\mathsf{NP}$ instances [KST21, BCL$^+$21, BGH19, BDFG20]. A central body of work [BCC$^+$16, Lee20, BMMV19, AC20, BCS21] studies the consequences of arguments over structurally nested homomorphic objects such as hypercubes and matrices. A key insight is that such objects present sufficient algebraic structure for *recursive* arguments in which larger composed statements can be reduced to smaller constituent statements of the same form. Homomorphic structures that enable recursive techniques have become a staple in constructing efficient argument systems [Set20, WTS$^+$18, BBB$^+$18, KMP20, SL20]. However, while homomorphic structures have become an essential tool in practice, the full extent to which homomorphisms in general can be composed and manipulated to create efficient arguments of knowledge is not yet well understood.

Hence, we formalize and study *reductions of knowledge*, a generalization of arguments of knowledge, which better capture the compositional and recursive nature of modern argument systems. A central observation in this work is that a large class of existing techniques can be viewed as a sequence of reductions (of knowledge) over homomorphic statements. We design the tensor reduction as a generalization of such techniques. Specifically, any homomorphic statement equipped with an appropriate decomposition rule can be reduced via the tensor reduction. When working over vector spaces in particular, tensor reductions provide knowledge soundness *unconditionally*. We show that instantiating the tensor reduction over (linearized) multivariate polynomials decomposed as univariate polynomials recovers the sumcheck protocol [LFKN92], a cornerstone technique in the literature. We further employ the tensor reduction to construct reductions of knowledge with sublinear communication for several linear algebraic statements by simply providing appropriate tensor representations and decomposition rules. We compose such statements to create a reduction of knowledge for $\mathsf{NP}$ with sublinear communication.

## 1.1 Reductions of Knowledge

Recall that arguments of knowledge are defined over a relation $\mathcal{R}$ and allow a prover to show for some statement $u$ that it knows witness $w$ such that $(u, w) \in \mathcal{R}$. In contrast, a reduction of knowledge is defined over a pair of relations $(\mathcal{R}_1, \mathcal{R}_2)$, and allows a prover to show for some $u_1$ that it knows $w_1$ such that $(u_1, w_1) \in \mathcal{R}_1$ contingent on the fact that for some derived $u_2$ it knows $w_2$ such that $(u_2, w_2) \in \mathcal{R}_2$. In effect, this allows a prover to reduce the task of proving knowledge of a satisfying witness for a statement in $\mathcal{R}_1$ to the task of proving knowledge of a satisfying witness for new a statement in $\mathcal{R}_2$.

Reductions of knowledge provide a common language to reason about a large class of techniques: Arguments of knowledge can be naturally viewed as reductions of knowledge where $\mathcal{R}_2$ is set to be a canonical empty relation. More interestingly, reductions of knowledge can capture individual steps in a larger argument system: For instance, argument systems such as Spartan [Set20] and Marlin [CHM$^+$20] reduce the task of checking matrix-vector products to inner-products.

Reductions of knowledge are particularly useful for reasoning about recursive techniques. Reductions of knowledge can be seen as generalizing a large class of recursive split-and-fold techniques [BCC$^+$16, BBB$^+$18, BMMV19, Lee20], which iteratively reduce a statement over size $n$ vectors to a statement over size $n/2$ vectors. Reductions of knowledge also readily capture more general recursive techniques such as accumulation schemes [BCL$^+$21], folding schemes [KST21], and aggregation schemes [BDFG20], all of which iteratively reduce the task of checking an arbitrary number of structurally similar $\mathsf{NP}$ instances into checking a single $\mathsf{NP}$ instance of the same size. Reductions of knowledge allow for independent reasoning about knowledge soundness for each recursive step. We show that such local reasoning is sufficient to prove global soundness properties by utilizing a general composition theorem.

**Theorem 1 (Composition, Informal).** Consider relations $\mathcal{R}_1$, $\mathcal{R}_2$, and $\mathcal{R}_3$. Let $\Pi_{(1,2)}$ be a reduction of knowledge for $(\mathcal{R}_1, \mathcal{R}_2)$ and let $\Pi_{(2,3)}$ be a reduction of knowledge for $(\mathcal{R}_2, \mathcal{R}_3)$. Then $\Pi_{(2,3)} \circ \Pi_{(1,2)}$ is a reduction of knowledge for $(\mathcal{R}_1, \mathcal{R}_3)$.

## 1.2 The Tensor Reduction

In this work, we focus our attention on generalizing techniques which can be cast as reductions (of knowledge) over homomorphic statements. We start by defining a general tensor-based language to capture homomorphic statements and then discuss how to compose such statements via the tensor product. We then introduce the the tensor reduction as an *abstract* protocol to reduce composed homomorphic statements into constituent statements.

**Tensor-Based Statements** We study algebraic statements for which arguments of knowledge can be decomposed into a sequence of reductions of knowledge. Existing arguments that fit such a pattern are built around statements over linear algebraic objects such as matrices, vectors, polynomials, and homomorphisms. We generalize such objects by turning towards a central object in module theory: tensors. Tensors provide a unifying algebraic framework for describing both functions (when viewed as homomorphisms) and objects (when viewed as elements of a vector space). In light of this, we study statements of the form $t(u) \cong v$ for tensors $t$, $u$, and $v$. Because such statements are purely algebraic, they can be readily manipulated using operations such as linear combinations and scalar multiplication. More interestingly, they can be composed and decomposed using two powerful operators, the tensor product and direct sums, which provide a notion of multiplication and addition over entire spaces.

**The Tensor Product** The tensor product in particular is an abstract operation characterized by its universality property: The tensor product of any two vector spaces $U$ and $V$ must result in a new vector space, denoted $U \otimes V$, such that any bilinear mapping $\varphi \in U \times V \to W$ induces a unique homomorphism $\widetilde{\varphi} \in U \otimes V \to W$ such that $\widetilde{\varphi}(u \otimes v) = \varphi(u, v)$. This property gives us a natural way to compose two linear statements (which form a bilinear statement) into a new linear statement.

In practice, much like how abstract groups and rings must be instantiated with concrete objects such as elliptic curves and polynomials, the tensor product must be instantiated with a concrete operation that respects the prescribed universality property. For instance, the outer-product would be an appropriate notion of multiplication that respects the universality property with respect to vectors. Similarly, the Kronecker product would be an appropriate notion of multiplication that respects the universality property with respect to matrices. The tensor product corresponds with the standard notion of multiplication for objects with multiplication built in, such as rings and fields.

**Reducing Tensor-Based Statements** We develop a protocol to reduce a statement composed via the tensor product into statements over constituent spaces which we refer to as a tensor reduction. At a high level, the tensor reduction is an interactive protocol which reduces the task of checking arbitrary rank tensors (which cannot be decomposed) into checking rank-1 tensors (which can be decomposed) by taking linear combinations of elements in the constituent spaces based on the verifier's random challenges.

**Theorem 2 (Tensor Reduction).** For tensors $t = \sum_i r_i \otimes s_i \in (U \otimes V) \to (X \otimes Y)$ of rank $I$, $w = \sum_j u_j \otimes v_j : U \otimes V$ of rank $J$, and $z : X \otimes Y$ over ring $\mathsf{R}$, the tensor reduction reduces the task of checking

$$t(w) \cong z \tag{1}$$

to the task of checking

$$\Big(\sum_i \alpha^i r_i\Big)\Big(\sum_j \beta^j u_j\Big) \cong x \quad \text{and} \quad \Big(\sum_i \alpha^i s_i\Big)\Big(\sum_j \beta^j v_j\Big) \cong y \tag{2}$$

for verifier output $(\alpha, \beta, x, y)$. Formally, if (1) is true, then (2) is true with probability 1, and if (1) is false, then (2) is false with probability at least $1 - \frac{IJ}{|Q|}$. The prover complexity, verifier complexity, and communication complexity are all proportional to $IJ$.

Under the interpretation that the tensor product is an abstract operation that remains to be instantiated, the tensor reduction can be viewed as an abstract protocol in which both the underlying modules *and* the composition operation between them remain to be instantiated. Such an abstraction allows us to specify reductions over algebraic objects equipped only with an abstract decomposition rule, without having to reason with respect the particular operation needed to perform this decomposition. For example, existing split-and-fold techniques reduce a statement over a vector $v \in \mathbb{F}^n$ by splitting the vector into two equal halves $v_1 \in \mathbb{F}^{n/2}$ and $v_2 \in \mathbb{F}^{n/2}$ and then reasoning about how to combine checks over these two halves into a smaller statement over a vector $v' \in \mathbb{F}^{n/2}$. In contrast, we can observe that $\mathbb{F}^n \cong \mathbb{F}^{n/2} \otimes \mathbb{F}^2$ and apply the tensor reduction in a black-box manner with respect to this decomposition to get a statement over $\mathbb{F}^{n/2}$ and a (trivial) statement over $\mathbb{F}^2$. Later we can instantiate $\otimes$ with the outer-product.

## 1.3 Representing New and Existing Techniques as Tensor Reductions

Given a general language to compose statements expressed as tensors, and a general reduction for such statements, we turn our attention towards expressing existing techniques as tensor reductions.

### 1.3.1 Universality of Linearization

We first leverage the universality property of the tensor product to show that any arbitrary polynomial over modules can be expressed as a tensor composed with a *universal* non-linear map. While this does not guarantee tensors that can be reduced via the tensor reduction, it does guarantee linear statements which can be composed into larger statements that can be reduced by the tensor reduction.

**Lemma 1 (Universality of Linearization, Informal).** Let $\mathsf{R}$ be a commutative ring, and let $V$ be an $\mathsf{R}$-module. There exists a universal non-linear map $\iota$ such that for an arbitrary polynomial $\varphi \in \mathsf{R}^n \to V$, there exists a unique corresponding tensor $\widetilde{\varphi}$ such that $\widetilde{\varphi} \circ \iota = \varphi$.

### 1.3.2 Recovering the Sumcheck Protocol

We leverage the universality of linearization to show that the sumcheck protocol [LFKN92] can be expressed as a sequence of tensor reductions. Consider ring $\mathsf{R}$, arbitrary $\mathsf{R}$-module $V$, and subset $H \subseteq \mathsf{R}$. Recall that for some $P \in \mathsf{R}^n \to V$, and claimed sum $\sigma \in V$, the sumcheck protocol (generalized over modules [BCS21]) allows a verifier to recursively reduce the task of checking

$$\sum_{x_1,\ldots,x_n \in H} P(x_1,\ldots,x_n) = \sigma$$

to the task of checking for some $P' \in R^{n-1} \to V$ and $\sigma' \in V$

$$\sum_{x_1,\ldots,x_{n-1} \in H} P'(x_1,\ldots,x_{n-1}) = \sigma'$$

Starting with [GKR15], sumcheck protocols have proven to be a powerful building block for constructing argument systems [Set20, WTS+18, CMT12, ZXZS20, ZGK+17]. We show that the tensor reduction recovers the sumcheck protocol when instantiated over multivariate polynomials decomposed as univariate polynomials. We make this notion precise by showing that each step of the sumcheck protocol can be simulated by first linearizing the polynomial (via the universality of linearization), running the tensor reduction, and then mapping the resulting statement back into the original space.

**Theorem 3 (Structural Equivalence, Informal).** Consider ring $\mathsf{R}$, $\mathsf{R}$-module $V$, subset $H \subseteq \mathsf{R}$, and degree bound $K-1$. Let $\Pi_{\mathsf{SC}}$ represent the sumcheck protocol and $\Pi_{\mathsf{LSC}}$ represent a linearized variant of the sumcheck protocol afforded by the tensor reduction. Additionally let $\Phi$ denote the isomorphism that linearizes its input via the tensor product. Then, for an arbitrary sumcheck statement $(P,\sigma)$, the following diagram commutes.

$$\begin{tikzcd}
(P,\sigma) \arrow[r, "\Pi_{\mathsf{SC}}"] \arrow[d, "\Phi"] & (P',\sigma') \\
(\mathbf{P},\boldsymbol{\sigma}) \arrow[r, "\Pi_{\mathsf{LSC}}"] & (\mathbf{P}',\boldsymbol{\sigma}') \arrow[u, "\Phi^{-1}"]
\end{tikzcd}$$

When formally stating the above theorem (Theorem 8), we are careful to also show that the transcript generated by the linearized sumcheck protocol can be used to recover the transcript produced by the sumcheck protocol. This result is particularly useful because it shows that the tensor reduction, when instantiated with isomorphic algebraic objects, is structurally equivalent to the sumcheck protocol. Structural equivalence is a stronger notion than functional equivalence because it allows us to show that the tensor reduction can be used in protocols which leverage both the functional *and* structural properties of the sumcheck protocol.

[BCS21] leverage the structural properties of the sumcheck protocol to show that a large class of arguments with sublinear communication, including proofs of knowledge over vectors, inner-products, folding techniques, delegating polynomial evaluation, batching, and arguments for $\mathsf{NP}$, built using a variety of cryptographic tooling (e.g., discrete logarithms, pairings, groups of unknown order, and lattices), can be encoded as sumcheck protocols. Our equivalence theorem shows that all of the above results extend to tensor reductions.

### 1.3.3 A Canonical Reduction of Knowledge over Tensors

Much like how the sumcheck protocol may or may not be used in the context of arguments *of knowledge*, tensor reductions may or may not be used in the context of reductions *of knowledge*. To add a knowledge component to the statement, there needs to be a meaningful notion of a witness. Given statements of the form $u(w) \cong v$ for tensors $u$, $w$, and $v$, we define the canonical tensor relation to be one in which $(u, v)$ is treated as the public statement and $w$ is treated as the witness. Because $w$ is now witnessed, we must also specify a rank bound.

**Definition 1 (Canonical Tensor Relation).** Consider modules $W$, $V$, and $U \cong (W \to V)$, i.e., the module of homomorphisms from $W$ to $V$. Let the corresponding canonical tensor relation be defined as

$$\mathcal{R}((U, V), W) = \left\{ ((u, v), w) \;\middle|\; \begin{array}{l} u \in U, v \in V, w \in W, \\ u(w) = v, \\ \mathsf{rank}(w) \leq J \end{array} \right\}$$

for some implicit rank bound $J$.

When considering vector spaces specifically (as opposed to modules), we can leverage tensor reductions to construct a reduction of knowledge for the canonical relation over homomorphisms composed by the tensor product.

**Theorem 4 (A Canonical Reduction of Knowledge over Tensors, Informal).** Consider field $\mathbb{F}$, length parameter $n$, $\mathbb{F}$-modules $W \cong W' \otimes \mathbb{F}^n$, $V \cong V' \otimes \mathbb{F}$, and $U \cong U' \otimes (\mathbb{F}^*)^n$ where $U' \cong (W' \to V')$. There exists a reduction of knowledge for

$$\Big(\mathcal{R}((U, V), W),\ \mathcal{R}((U', V'), W')\Big).$$

A notable implication of the above theorem is that knowledge soundness comes unconditionally when applying the tensor reduction to canonical relations over vector spaces. In other words, there exists a reduction *of knowledge* for any statement that can be expressed in the appropriate form. We also note that the above theorem holds orthogonally to any computational hardness assumptions. In particular, we could trivially prove that knowledge soundness holds for relations where the witness can be efficiently computed given only the statement. Thus, knowledge soundness is only a meaningful notion when it is assumed to be computationally difficult to compute a satisfying witness given only the statement.

### 1.3.4 A Reduction of Knowledge for $\mathsf{NP}$

The canonical reduction of knowledge over tensors abstracts both protocol design and corresponding knowledge-soundness reasoning. To produce a reduction of knowledge for statements of interest, we are largely tasked with providing a corresponding tensor-based representation equipped with an appropriate decomposition rule. We show that both linear-forms relations and bilinear-forms relations over fields can be naturally expressed in such a form. We compose the corresponding tensor reductions of knowledge to build a modular reduction of knowledge for $\mathsf{NP}$ with sublinear communication.

**A Reduction of Knowledge for Linear Forms** We start by constructing an argument of knowledge for linear forms with sublinear communication that resembles a similar protocol derived by [AC20]. Consider group $\mathbb{G}$ where the discrete logarithm problem is hard and corresponding field $\mathbb{F}$. For public key $G \in \mathbb{G}^n$, public vector $B \in \mathbb{F}^n$, commitment $\overline{A} \in \mathbb{G}$, and scalar $\sigma \in \mathbb{F}$, a linear-forms argument allows a prover shows that it knows $A \in \mathbb{F}^n$ such that $\langle B, A\rangle = \sigma$ and $\langle G, A\rangle = \overline{A}$. We observe that this statement can be expressed as the corresponding tensor statement

$$(G \oplus B)(A) = \overline{A} \oplus \sigma$$

where $\oplus$ denotes the direct sum (i.e., a Cartesian product). We reduce this statement via the canonical reduction to an identical statement over size $n/2$ vectors under the decomposition rule $\mathbb{G}^n \cong \mathbb{G}^{n/2} \otimes \mathbb{F}^2$, $\mathbb{F}^n \cong \mathbb{F}^{n/2} \otimes \mathbb{F}^2$. This corresponds to existing reduction techniques which decompose vectors into their first and second half [BCC+16, BBB+18, Lee20, BMMV19].

**A Reduction of Knowledge for Bilinear Forms** Next, we construct a reduction of knowledge from bilinear forms to linear forms with sublinear communication. For public keys $(G, H) \in (\mathbb{G}^m, \mathbb{G}^n)$, public matrix $M \in \mathbb{F}^{m\times n}$, commitments $\overline{A}, \overline{B} \in \mathbb{G}$, and scalar $\sigma \in \mathbb{F}$, a bilinear-forms argument allows a prover to show that it knows $A, B \in \mathbb{F}^n$ such that $A^\top M B = \sigma$, $\langle G, A\rangle = \overline{A}$, and $\langle H, B\rangle = \overline{B}$. Under the hardness of the discrete logarithm assumption and double pairing assumption [AFG+10], we observe that this statement can be expressed as the corresponding tensor statement

$$(G \otimes H \oplus M)(A \otimes B) = (\overline{A} \otimes \overline{B} \oplus \sigma) \tag{3}$$

We reduce this statement to an identical statement over a size $m \times n/2$ matrix under the observation that $M \in (\mathbb{F}^m \otimes \mathbb{F}^n) \cong (\mathbb{F}^m \otimes \mathbb{F}^{n/2}) \otimes \mathbb{F}^2$, $G \otimes H \in (\mathbb{G}^m \otimes \mathbb{G}^n) \cong (\mathbb{G}^m \otimes \mathbb{G}^{n/2}) \otimes \mathbb{F}^2$, and $A \otimes B \in (\mathbb{F}^m \otimes \mathbb{F}^n) \cong (\mathbb{F}^m \otimes \mathbb{F}^{n/2}) \otimes \mathbb{F}^2$. In the base case we have a statement over a size $m \times 1$ matrix, which we show can be reduced to checking a linear-forms statement. The computational hardness assumptions are a critical detail for arguing that checking Equation (3) is sufficient to check the original relation: The knowledge soundness property of the canonical reduction only guarantees that the prover knows *some* satisfying witness in $\mathbb{F}^m \otimes \mathbb{F}^n$ which may be of the form $\sum_i A_i \otimes B_i$. While this is a valid witness for the corresponding tensor statement, it is *not* a valid witness for the original statement. Here, we must invoke the appropriate assumptions to show that it is infeasible for the extractor to retrieve a witness that is not of the form $A \otimes B$.

**A Reduction of Knowledge for NP** [KMP20] observe that an instance in NP can be checked using a sequence of (sparse) bilinear forms over the same two vectors (each of which represents a constraint). We leverage this characterization of NP to represent an instance in NP as a sequence of tensor statements representing bilinear forms. We then take advantage of the inherent linearity of these statements to compress all of the checks into a single bilinear-forms check via a random linear combination. Putting everything together we recover the following theorem.

**Theorem 5 (A Reduction of Knowledge for NP, Informal).** For NP statements expressed over $n$ variables, $m = O(n)$ bilinear constraints, and $\ell < n$ public input values, there exists a reduction of knowledge from the task of checking $m$ bilinear forms over vectors of size $n$, and $\ell$ linear forms over vectors of size $n$, to the task of checking the trivial relationship. The prover and verifier time complexity is $O(n)$. The communication complexity is $O(\log n)$.

We additionally expect that our reductions of knowledge are compatible with standard optimization techniques in the literature. For instance, techniques such as checkable subspace sampling [RZ21] and polynomial commitments [KZG10, WTS+18, ZGK+17] can be used to achieve a sublinear verifier. We do not discuss these optimizations because they are unrelated to our primary goal of studying reductions of knowledge and the corresponding tensor reduction.

## 1.4 Related Work

**Split-and-Fold Techniques** Initiated by [BCC+16], a line of work [BBB+18, KMP20, WTS+18] focuses on developing inner-product arguments and corresponding applications based on Pedersen commitments under the discrete logarithm assumption. In each round a size $n$ vector is split into two size $n/2$ vectors and folded onto itself using a random linear combination to create a new statement over size $n/2$ vectors. Such techniques are generalized to matrices using two-tiered commitments [BMMV19, Lee20]. Reductions of knowledge can be seen as a definitional generalization of these styles of protocols, and tensor reductions can be seen as a generalization of the underlying split-and-fold technique. All split-and-fold techniques must work over objects with a natural notion of a first and second half such as vectors and matrices. Such a restriction is lifted for tensor reductions, which can be viewed as splitting one tensor with respect to the structure of another.

**Sumcheck Arguments** [BCS21] show that a large class of split-and-fold techniques can be viewed as a special case of sumcheck protocols over commitments, which they call sumcheck arguments. We show the converse of this result: That is, we show that tensor reductions, which can be interpreted as an abstracted folding technique, generalize sumcheck protocols. [BCS21] further show that sumcheck arguments can be instantiated with any commitment scheme which satisfies a certain structural decomposability property, and thus show that sumcheck arguments generalize folding techniques over prime-order groups, bilinear groups, unknown-order groups and ideal lattices. These results translate to our setting via our generalization result.

**Proof Batching Schemes** Several works focus on batching proofs for arguments of knowledge over NP [HN10, Dru15, BHK17]. This approach has been recently popularized by [BGH19] and generalized by [BDFG20], who show that statements can be efficiently compressed by performing a small local check at each step and aggregating the remaining expensive computation into a polynomial evaluation statement, which can be checked in batch. This approach can be viewed as a reduction of knowledge from checking an $\mathsf{NP}$ statement and a polynomial-evaluation statement to checking a polynomial-evaluation statement. [BCL+21] generalize the above approach for any relation (in particular $\mathsf{NP}$) for which there exists a succinct statement accumulator (and corresponding witness accumulator). This approach can be viewed as a reduction of knowledge from checking a statement and an accumulator to checking an accumulator. [KST21] present folding schemes, which fold two $\mathsf{NP}$ instance-witness pairs into a single $\mathsf{NP}$ instance-witness pair. Folding schemes can be viewed as a reduction of knowledge from checking the satisfiability of two statements to checking the satisfiability of a single statement. An open question left by our work is whether the underlying techniques for proof batching schemes can be efficiently expressed as tensor reductions.

**Compressed Sigma Protocol Theory** [AC20] generalize (and simplify) split-and-fold style arguments as a recursive Sigma protocol over linear forms that compresses the statement in each round. They additionally provide a technique to linearize bilinear relations over Pedersen commitments (and over bilinear groups [ACR20]). We recover similar protocols for linear forms and bilinear forms via the tensor reduction.

**Other Tensor-Based Protocols** [Mei13] designs a sumcheck protocol over tensor codes to recover $\mathsf{IP} = \mathsf{PSPACE}$. [BCG20] leverage this protocol to efficiently query tensor codes at structured random points. [RZ21] suggest that checkable subspace sampling can be suitably generalized to efficiently and randomly compress tensors along a dimension. We view these techniques as complementary to ours, as they could be composed with tensor reductions to achieve a sublinear verifier.

### 1.5 Open Questions

Many of the open questions left by our work revolve around fully characterizing the structural properties of tensor reductions. While we show that tensor reductions generalize sumcheck protocols, it would be interesting to characterize the class of reductions generalized by tensor reductions using a similar proof technique (thereby providing a universality property for tensor reductions).

Conversely, it is unclear whether tensor reductions can be expressed as sumcheck protocols. Such an equivalence would provide structural insights about both techniques. It is also unclear whether tensor reductions can be expressed under other general frameworks such as compressed sigma protocols.

While reductions of knowledge definitionally capture techniques that batch the satisfiability of entire $\mathsf{NP}$ instances, it is unclear whether such techniques can be viewed as efficient tensor reductions. Finally, while we provide a reduction of knowledge with sublinear communication for $\mathsf{NP}$ via tensor reductions, further study is needed in composing tensor reductions with existing techniques (e.g., checkable subspace sampling [RZ21], Fiat-Shamir transform [FS86], randomization) to achieve properties such as verifier succinctness, non-interactivity, and zero-knowledge.

## 2 Preliminaries

### 2.1 Module Theory

**Notation 1 ($\mathsf{R}$-Modules).** We assume finite, unital, commutative rings and finite modules (i.e., have a finite basis) throughout. For ring $\mathsf{R}$ and $\mathsf{R}$-modules $U$, $V$, let $(U \to V)$ denote the module of homomorphisms from $U$ to $V$. We let $U^*$ denote the dual-space of $U$, that is $U^* \cong (U \to \mathsf{R})$. We use $\{b_i\}$ to denote a module basis and use $\{\delta_i\}$ to denote a dual basis.

The definitions and corresponding lemmas below are adapted from Dummit and Foote [DF04]. We start by defining the tensor product and direct sum operations for modules over rings, which will be used throughout our development.[1]

**Definition 2 (Direct Sum).** The **direct sum** of $\mathsf{R}$-modules $U$ and $V$ represents a new module denoted $U \oplus V$ consisting of elements of the form $u \oplus v$ for $u \in U$ and $v \in V$, such that

$$\begin{aligned} r \cdot (u \oplus v) &= (ru \oplus rv) \\ u_1 \oplus v_1 + u_2 \oplus v_2 &= (u_1 + u_2) \oplus (v_1 + v_2) \end{aligned}$$

for elements $u \oplus v, u_1 \oplus v_1, u_2 \oplus v_2 \in U \oplus V$ and $r \in \mathsf{R}$.

The tensor product can be viewed as an abstract notion of multiplication that is characterized by the only required property, bilinearity.

**Definition 3 (Tensor Product).** The **tensor product** of $\mathsf{R}$-modules $U$ and $V$ represents a new module denoted $U \otimes V$ consisting of finite (formal) sums of elements of the form $u \otimes v$ for $u \in U$ and $v \in V$, such that

$$\begin{aligned} u_1 \otimes v + u_2 \otimes v &= (u_1 + u_2) \otimes v \\ u \otimes v_1 + u \otimes v_2 &= u \otimes (v_1 + v_2) \\ (r \cdot u) \otimes v &= u \otimes (r \cdot v) \end{aligned}$$

for $u, u_1, u_2 \in U$, $v, v_1, v_2 \in V$, and $r \in \mathsf{R}$.

An immediate consequence of the above definition is that any bilinear operation can be expressed as a homomorphism composed with the tensor product. This property is formally known as universality and can be naturally extended to multilinear maps.

**Lemma 2 (Universality of the Tensor Product).** Let $\mathsf{R}$ be a commutative ring, and let $U_1$, $U_2$, and $V$ be $\mathsf{R}$-modules. Define the map $\iota(U_1, U_2) \mapsto U_1 \otimes U_2$. Then, an arbitrary bilinear map $\varphi \in U_1 \times U_2 \to V$ induces a homomorphism $\widetilde{\varphi} \in (U_1 \otimes U_2 \to V)$ such that $\widetilde{\varphi} \circ \iota = \varphi$. In other words, the following diagram commutes[2].

$$\begin{tikzcd} U_1 \times U_2 \arrow[r, "\varphi"] \arrow[d, "\iota"] & V \\ U_1 \otimes U_2 \arrow[ur, "\widetilde{\varphi}"] & \end{tikzcd}$$

Direct sums and tensor products afford a notion of addition and multiplication over modules (and thus give us a notion of addition and multiplication over homomorphisms).

[1]See Appendix A for definitions of rings and modules.

[2]A diagram is said to *commute* if all paths along the arrows lead to the same result.

**Definition 4 (Direct Sum of Homomorphisms).** The direct sum of two homomorphisms $r \in (U_1 \to V)$, $s \in (U_2 \to V)$ over $\mathsf{R}$-modules (where $\mathsf{R}$ is a commutative ring) produces a new homomorphism denoted $r \oplus s \in (U_1 \oplus U_2) \to V$, and defined as

$$(r \oplus s)(u_1 \oplus u_2) \mapsto r(u_1) + s(u_2).$$

Symmetrically, the direct sum of two homomorphisms $r \in (U \to V_1)$, $s \in (U \to V_2)$ over $\mathsf{R}$-modules produces a new homomorphism denoted $r \oplus s \in U \to (V_1 \oplus V_2)$, and defined as

$$(r \oplus s)(u) \mapsto r(u) \oplus s(u).$$

**Definition 5 (Tensor Product of Homomorphisms).** The tensor product of two homomorphisms $r \in (U \to X)$, $s \in (V \to Y)$ over $\mathsf{R}$-modules (where $\mathsf{R}$ is a commutative ring) produces a new homomorphism, denoted $r \otimes s \in (U \otimes V) \to (X \otimes Y)$, and defined as

$$(r \otimes s)(u \otimes v) \mapsto r(u) \otimes s(v).$$

We refer to homomorphisms over $\mathsf{R}$-modules in this context as **tensors**. $r \otimes s$ is itself an $\mathsf{R}$-module, and thus, tensors can be viewed as homomorphisms from tensors to tensors.

We additionally recall several useful identities which will help us identify necessary isomorphisms throughout our development.

**Lemma 3 (Commutativity and Associativity).** For $\mathsf{R}$-modules $U, V, W$ (where $\mathsf{R}$ is a commutative ring), the tensor product is commutative and associative up to isomorphism. That is $(U \otimes V) \otimes W \cong U \otimes (V \otimes W)$ and $U \otimes V \cong V \otimes U$.

**Lemma 4 (Distributivity).** For ring $\mathsf{R}$ and $\mathsf{R}$-modules $U_1, U_2, V_1, V_2$

$$(U_1 \oplus U_2) \otimes V \cong (U_1 \otimes V) \oplus (U_2 \otimes V)$$
$$U \otimes (V_1 \oplus V_2) \cong (U \otimes V_1) \oplus (U \otimes V_2).$$

**Lemma 5 (Scalar Multiplication).** For ring $\mathsf{R}$ and $\mathsf{R}$-module $U$, $\mathsf{R} \otimes U \cong U \otimes \mathsf{R} \cong U$.

## 2.2 Cryptographic Preliminaries

**Notation 2 (Cryptographic Variables).** For $n \in \mathbb{N}$, let $[n]$ denote $\{1, \dots, n\}$. When summing over a variable, we will omit the bounds when clear from context. We use $\lambda$ globally to denote the security parameter, and $\mathsf{negl}(\lambda)$ to denote negligible functions.

Typically, for soundness to hold when randomly sampling over rings, the set of admissible values must be constrained. We start by defining a valid sampling set over rings.

**Definition 6 (Sampling Set [BCS21]).** For ring $\mathsf{R}$ and $\mathsf{R}$-module $M$, subset $Q \subseteq \mathsf{R}$ is a **sampling set** for $M$ if for every $q_1, q_2 \in Q$, the map $\varphi(m) \mapsto (q_1 - q_2)m$ for $m \in M$ is injective.

To have a meaningful notion of knowledge soundness, we will need to rely on computational hardness assumptions. We adapt the bilinear relation assumption [BCS21], which can be viewed as a generalization of the discrete logarithm assumption, and the double pairing assumption [AFG$^+$10].

**Definition 7 (Bilinear Relation Assumption).** For ring $\mathsf{R}$, length parameter $n$, and security parameter $\lambda$, consider $\mathsf{R}$-modules $U$ and $V$ such that $|U| = O(2^\lambda)$ and $|V| = O(2^\lambda)$. The **bilinear relation assumption** holds for $U$ and $V$ if given randomly sampled $u_1, \ldots, u_n \in U$, there exists no polynomial-time algorithm to find non-trivial $v_1, \ldots, v_n \in V$ such that

$$\sum_{i \in [n]} u_i \otimes v_i \cong 0.$$

We say the reverse bilinear relation assumption holds for $U$ and $V$ if given randomly sampled $v_1, \ldots, v_n \in V$ there exists no polynomial-time algorithm to find non-trivial $u_1, \ldots, u_n \in U$ such that the above equation holds.

Dually, we can consider composite spaces such that given elements from both of the constituent spaces, it is *easy* to check that they satisfy the above relation. This ensures that that the verifier in the tensor reduction is able to perform its requisite checks efficiently.

**Definition 8 (Coset Equality Assumption).** For ring $\mathsf{R}$, length parameter $n$, and security parameter $\lambda$, consider $\mathsf{R}$-modules $U$ and $V$ such that $|U| = O(2^\lambda)$ and $|V| = O(2^\lambda)$. The **coset equality assumption** holds for $U$ and $V$ if for any $u_1, \ldots, u_n \in U$ and $v_1, \ldots, v_n \in V$, there exists a polynomial-time algorithm to check

$$\sum_{i \in [n]} u_i \otimes v_i \cong 0.$$

The above assumptions can be used to construct a generalized version of the Pedersen vector commitment scheme [Ped91] which we use to develop reductions of knowledge for linear-algebraic relations.

**Definition 9 (Commitment Scheme).** For ring $\mathsf{R}$, $\mathsf{R}$-modules $\mathbb{G}$ and $\mathbb{H}$, length parameter $n$, and security parameter $\lambda$, a **commitment scheme** over key-space $\mathbb{G}^n$ and message-space $\mathbb{H}^n$ is defined over polynomial-time algorithms $\mathsf{Gen}$, $\mathsf{Com}$, and $\mathsf{Open}$ where

- $\mathsf{pp} \leftarrow \mathsf{Gen}(\lambda, n)$: Takes as input security parameter $\lambda$ and length parameter $n$. Produces public parameters $\mathsf{pp}$
- $C \leftarrow \mathsf{Com}(\mathsf{pp}, \mathbf{v})$: Takes as input public parameters $\mathsf{pp}$ and vector $\mathbf{v} \in \mathbb{H}^n$. Outputs commitment $C$.
- $\{0,1\} \leftarrow \mathsf{Open}(\mathsf{pp}, C, \mathbf{v})$: Takes as input public parameters $\mathsf{pp}$, commitment $C$ and vector $\mathbf{v}$. Outputs 1 if $C = \mathsf{Com}(\mathsf{pp}, \mathbf{v})$.

A commitment scheme is said to be **binding** if it is hard for an adversary to open a commitment to two distinct vectors. That is, for any PPT adversary $\mathcal{A}$,

$$\Pr\left[\begin{array}{l|l} \mathsf{Open}(\mathsf{pp}, C, \mathbf{v_1}) = 1, & \mathsf{pp} \leftarrow \mathsf{Gen}(\lambda, n), \\ \mathsf{Open}(\mathsf{pp}, C, \mathbf{v_2}) = 1, & (C, \mathbf{v_1}, \mathbf{v_2}) \leftarrow \mathcal{A}(\mathsf{pp}) \\ \mathbf{v_1} \neq \mathbf{v_2} & \end{array}\right] = \mathsf{negl}(\lambda).$$

**Construction 1 (Generalized Pedersen Vector Commitments).** For ring $\mathsf{R}$, length parameter $n$, and security parameter $\lambda$, let $\mathbb{G}$ and $\mathbb{H}$ be $\mathsf{R}$-modules such that the bilinear relation assumption and the coset equality assumption hold for $(\mathbb{G}, \mathbb{H})$. Then, the generalized Pedersen vector commitment for key-space $\mathbb{G}^n$ and message-space $\mathbb{H}^n$ is defined as follows:

$$\begin{aligned}\mathsf{Gen}(\lambda, n) &\mapsto \mathbf{g} \overset{\$}{\leftarrow} \mathbb{G}^n \\ \mathsf{Com}(\mathbf{g}, \mathbf{v}) &\mapsto \sum_i \mathbf{g}_i \otimes \mathbf{v}_i\end{aligned}$$

and $\mathsf{Open}$ is defined canonically.

**Lemma 6 (Generalized Pedersen Vector Commitment Scheme).** Construction 1 is a binding commitment scheme.

*Proof.* Binding holds due to the bilinear relation assumption. Efficiency of $\mathsf{Open}$ holds due to the coset equality assumption. □

## 3 Reductions of Knowledge

Recall that in contrast to arguments of knowledge, reductions of knowledge are defined over a pair of relations $(\mathcal{R}_1, \mathcal{R}_2)$. A prover can use a reduction of knowledge to show for some $u_1$ that it knows $w_1$ such that $(u_1, w_1) \in \mathcal{R}_1$ contingent on the fact that it knows $w_2$ for some statement $u_2$ (derived from its interaction with the verifier) such that $(u_2, w_2) \in \mathcal{R}_2$. Correctness and soundness definitions follow naturally.

An appropriate extractor definition that captures knowledge-soundness requires more consideration. Intuitively, we would like that if a prover is able to convince a verifier on input $u_1$ to output some derived statement $u_2$ such that it knows a corresponding satisfying witness $w_2$, then it must have known a corresponding satisfying witness $w_1$ for $u_1$. Unfortunately, this intuitive definition is circular: We would like to define what it means to know $w_1$ with respect to the definition of knowing $w_2$. Prior approaches [Lee20, AC20] require that if the prover can output some auxiliary information $\mathsf{aux}$ such that a subsequent malicious prover $\mathcal{P}^*_{\mathsf{ARG}}(\mathsf{aux})$ can convince a verifier that it knows a valid witness to statement $u_2$ using a standard *argument of knowledge*, then there must exist an extractor that can produce $w_1$ given oracle access to $\mathcal{P}^*_{\mathsf{ARG}}(\mathsf{aux})$ and the next message function of $\mathcal{P}^*$. However, this requires holding on to two definitions of extractability when proving the soundness of a reduction of knowledge.

Alternatively, we can simply say that knowledge soundness holds if a malicious prover $\mathcal{P}^*$ can output a satisfying witness $w_2$ then there exists an extractor that can output $w_1$ provided oracle access to $\mathcal{P}^*$ (which includes its output $w_2$). At first glance, this seems like a weaker notion of soundness; however, we can show that this definition implies the prior definition. In particular, we can show that if there exists an extractor $\mathcal{E}$ that can retrieve $w_1$ given oracle access to $\mathcal{P}^*$ which outputs $w_2$, then there must exist an extractor $\mathcal{E}'$ that can retrieve $w_1$ given oracle access to $\mathsf{P}^*_{\mathsf{ARG}}(\mathsf{aux})$ where $\mathsf{aux}$ is output by $\mathcal{P}^*$. Indeed, $\mathcal{E}'$ can run $\mathcal{E}$ and provide it oracle access to $\mathcal{P}^*$. When $\mathcal{E}$ eventually requests the output $w_2$, $\mathcal{E}'$ runs the extractor for the argument system $\mathcal{E}_{\mathsf{ARG}}$ (with oracle access to $\mathsf{P}^*_{\mathsf{ARG}}(\mathsf{aux})$) to retrieve $w_2$, providing $\mathcal{E}$ all the information it needs to extract $w_1$.

Therefore, it is sufficient to say that knowledge soundness holds if there exists an extractor that can extract a satisfying $w_1$ provided the malicious prover manages to output a satisfying $w_2$. We formally define reductions of knowledge in the common reference string model.

**Definition 10 (Reduction of Knowledge).** Consider binary relations $\mathcal{R}_1$ and $\mathcal{R}_2$ over tuples consisting of statement-witness tuples. A **reduction** for $(\mathcal{R}_1, \mathcal{R}_2)$ consists of PPT setup algorithm $\mathcal{G}$, called the generator, and PPT interactive algorithms $\mathcal{P}$ and $\mathcal{V}$, called the prover and verifier respectively, with the following structure

- $\mathcal{G}(\lambda) \to \mathsf{pp}$: Takes as input security parameter $\lambda$; outputs public parameters $\mathsf{pp}$
- $\mathcal{P}(\mathsf{pp}, u_1, w_1) \to (u_2, w_2)$: Takes as input public parameters $\mathsf{pp}$, and statement-witness pair $(u_1, w_1) \in \mathcal{R}_1$. Interactively reduces the statement $(u_1, w_1) \in \mathcal{R}_1$ to a new statement $(u_2, w_2) \in \mathcal{R}_2$.
- $\mathcal{V}(\mathsf{pp}, u_1) \to u_2$: Takes as input public parameters $\mathsf{pp}$, and statement $u_1$ associated with $\mathcal{R}_1$. Interactively reduces the task of checking $u_1$ to the task of checking a new statement $u_2$ associated with $\mathcal{R}_2$.

Let $(u_2, w_2) \leftarrow \langle \mathcal{P}(w_1), \mathcal{V}\rangle(\mathsf{pp}, u_1)$ denote the verifier's output statement $u_2$ and the prover's output witness $w_2$ after interacting on common inputs $\mathsf{pp}$, $u_1$, and additional prover input $w_1$. Likewise, let $\mathsf{tr} \leftarrow \langle \mathcal{P}(w_1), \mathcal{V}\rangle(\mathsf{pp}, u_1)$ denote the interaction transcript. A reduction satisfies **completeness** if for all PPT adversaries $\mathcal{A}$

$$\Pr\left[\ (u_2, w_2) \in \mathcal{R}_2 \ \middle|\ \begin{array}{l} \mathsf{pp} \leftarrow \mathcal{G}(\lambda), \\ (u_1, w_1) \in \mathcal{R}_1 \leftarrow \mathcal{A}(\mathsf{pp}), \\ (u_2, w_2) \leftarrow \langle \mathcal{P}(w_1), \mathcal{V}\rangle(\mathsf{pp}, u_1) \end{array}\right] = 1.$$

A reduction satisfies **soundness** if for all PPT adversaries $\mathcal{P}^*$

$$\Pr\left[\ \begin{array}{c} \nexists\ w_1 \text{ s.t. } (u_1, w_1) \in \mathcal{R}_1 \wedge \\ \exists\ w_2 \text{ s.t. } (u_2, w_2) \in \mathcal{R}_2 \end{array} \ \middle|\ \begin{array}{l} \mathsf{pp} \leftarrow \mathcal{G}(\lambda), \\ u_1 \leftarrow \mathcal{P}^*(\mathsf{pp}), \\ u_2 \leftarrow \langle \mathcal{P}^*, \mathcal{V}\rangle(\mathsf{pp}, u_1) \end{array}\right] = \mathsf{negl}(\lambda).$$

A reduction satisfies **knowledge soundness** if there exists an expected polynomial-time extractor $\mathcal{E}$ such that for any polynomial-time adversary $\mathcal{A}$

$$\left| \begin{array}{l} \Pr\left[\ (u_2, w_2) \in \mathcal{R}_2 \ \middle|\ \begin{array}{l} \mathsf{pp} \leftarrow \mathcal{G}(\lambda), \\ (u_1, \mathsf{st}) \leftarrow \mathcal{A}(\mathsf{pp}), \\ (u_2, w_2) \leftarrow \langle \mathcal{A}(\mathsf{st}), \mathcal{V}(\mathsf{pp}, u_1)\rangle \end{array}\right] - \\ \Pr\left[\ (u_1, w_1) \in \mathcal{R}_1 \ \middle|\ \begin{array}{l} \mathsf{pp} \leftarrow \mathcal{G}(\lambda), \\ (u_1, \mathsf{st}) \leftarrow \mathcal{A}(\mathsf{pp}), \\ w_1 \leftarrow \mathcal{E}^{\langle \mathcal{A}(\mathsf{st}), \mathcal{V}(\mathsf{pp}, u_1)\rangle}(\mathsf{pp}) \end{array}\right] \end{array} \right| = \mathsf{negl}(\lambda).$$

where $\mathcal{E}$ has oracle access to the interaction $\langle \mathcal{A}(\mathsf{st}), \mathcal{V}(u_1)\rangle$ which permits rewinding to any point with refreshed verifier randomness. At the end of interaction, $\mathcal{E}$ is provided the witness $w_2$ output by $\mathcal{A}$. We call a reduction that satisfies knowledge soundness a **reduction of knowledge**. A reduction is **public-coin** if all messages sent from $\mathcal{V}$ to $\mathcal{P}$ are chosen uniformly at random.

With a definition in hand, we can now prove a general composition theorem for reductions of knowledge. This greatly simplifies soundness proofs for protocols that compose (or recursively employ) reductions of knowledge. Like other recursive techniques [BCCT13, Val08, KST21, BCL+21], each recursive step induces a polynomial blowup in the corresponding extractor, and thus composition cannot be used more than a logarithmic number of times without additional computational assumptions. We recommend [BCCT13] for a detailed discussion on such assumptions.

**Theorem 6 (Composition).** Consider binary relations $\mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3$. Let $(\mathcal{G}_1, \mathcal{P}_1, \mathcal{V}_1)$ be a reduction of knowledge for $(\mathcal{R}_1, \mathcal{R}_2)$. Let $(\mathcal{G}_2, \mathcal{P}_2, \mathcal{V}_2)$ be a reduction of knowledge for $(\mathcal{R}_2, \mathcal{R}_3)$. Then $(\mathcal{G}, \mathcal{P}, \mathcal{V})$ is a reduction of knowledge for $(\mathcal{R}_1, \mathcal{R}_3)$ where

$$\begin{aligned}\mathcal{G}(\lambda) &:= (\mathcal{G}_1(\lambda), \mathcal{G}_2(\lambda))\\ \mathcal{P}(\mathsf{pp}, u_1, w_1) &:= \mathcal{P}_2(\mathsf{pp}_2, \mathcal{P}_1(\mathsf{pp}_1, u_1, w_1))\\ \mathcal{V}(\mathsf{pp}, u_1) &:= \mathcal{V}_2(\mathsf{pp}_2, \mathcal{V}_1(\mathsf{pp}_1, u_1)).\end{aligned}$$

*Proof.* Correctness follows by observation. As for knowledge soundness, consider arbitrary adversary $\mathcal{A}$ attacking the knowledge soundness of $(\mathcal{G}, \mathcal{P}, \mathcal{V})$. Let $\mathsf{pp} \leftarrow \mathcal{G}(\lambda)$ and let $(u_1, \mathsf{st}) \leftarrow \mathcal{A}(\mathsf{pp})$. We must construct an extractor $\mathcal{E}$ such that on input $\mathsf{pp}$ and with oracle access to $\langle \mathcal{A}(\mathsf{st}), \mathcal{V}(u_1) \rangle$ produces $w_1$ such that $(u_1, w_1) \in \mathcal{R}_1$ with nearly the same success probability as $\mathcal{A}$.

Our high level approach is as follows: We initially construct $\mathcal{A}_2$ which attacks the knowledge soundness of $\Pi_2 = (\mathcal{G}_2, \mathcal{P}_2, \mathcal{V}_2)$ using $\mathcal{A}$. The knowledge soundness of $\Pi_2$ guarantees a corresponding extractor $\mathcal{E}_2$. We then construct adversary $\mathcal{A}_1$ which uses $\mathcal{E}_2$ to attack the knowledge soundness of $\Pi_1 = (\mathcal{G}_1, \mathcal{P}_1, \mathcal{V}_1)$. Once again, the knowledge soundness of $\Pi_1$ guarantees a corresponding extractor $\mathcal{E}_1$. The desired extractor $\mathcal{E}$ then uses $\mathcal{E}_1$ to produce $w_1$.

We first construct polynomial-sized adversary $\mathcal{A}_2$ attacking $\Pi_2$ as follows

**Adversary $\mathcal{A}_2$ :**

1. Assuming that $\mathcal{A}$ has already interacted with $\mathcal{V}_1$ continue the interaction between $\mathcal{A}$ and $\mathcal{V}_2$: On a query from $\mathcal{V}_2$, query $\mathcal{A}$ and respond with the response of $\mathcal{A}$.
2. Output the witness $w_3$ output by $\mathcal{A}$.

Let $\epsilon_2$ be the success probability of $\mathcal{A}_2$. By the knowledge soundness of $\Pi_2$, there exists a corresponding expected polynomial-time extractor $\mathcal{E}_2$ such that on input $\mathsf{pp}_2$ and oracle access to $\langle \mathcal{A}_2, \mathcal{V}_2(u_2) \rangle$ produces $w_2$ such that $(u_2, w_2) \in \mathcal{R}_2$ with probability $\epsilon_2 - \mathsf{negl}(\lambda)$.

We now construct a polynomial-sized adversary $\mathcal{A}_1$ attacking $\Pi_1$ as follows

**Adversary $\mathcal{A}_1(\mathsf{pp}_2)$ :**

1. On a query from $\mathcal{V}_1$, query $\mathcal{A}$ and respond with the response of $\mathcal{A}$
2. Once $\mathcal{V}_1$ finishes querying, run $\mathcal{E}_2^{\langle \mathcal{A}_2, \mathcal{V} \rangle}(\mathsf{pp}_2)$ to retrieve and output $w_2$.

Let $\epsilon_1$ be the success probability of $\mathcal{A}_1$. By the knowledge soundness of $\Pi_1$, there exists a corresponding expected polynomial-time extractor $\mathcal{E}_1$ such that on input $\mathsf{pp}_1$ and oracle access to $\langle \mathcal{A}_1, \mathcal{V}_1 \rangle$ produces $w_1$ such that $(u_1, w_1) \in \mathcal{R}_1$ with probability $\epsilon_1 - \mathsf{negl}(\lambda)$.

Given extractor $\mathcal{E}_1$ we can construct $\mathcal{E}$ as follows

**Extractor** $\mathcal{E}((\mathsf{pp}_1, \mathsf{pp}_2))$ :

1. Run $\mathcal{E}_1$ on input $\mathsf{pp}_1$ giving it oracle access to $\langle \mathcal{A}_1(\mathsf{pp}_2), \mathcal{V} \rangle$ to retrieve and output $w_1$.

Let $\epsilon$ be the success probability of $\mathcal{A}$ in outputting $w_3$ that satisfies $u_3$ output by $\mathcal{V}$. Because $\mathcal{A}_2$ is always run after the first half of $\mathcal{A}$ has interacted with the first half of $\mathcal{V}$, by construction, we have that $\mathcal{A}_2$ succeeds with the same probability as $\mathcal{A}$. This means that $\mathcal{E}_2$ succeeds with probability $\epsilon - \mathsf{negl}(\lambda)$. However, by construction, this means that $\mathcal{A}_1$ succeeds with probability $\epsilon - \mathsf{negl}(\lambda)$. In turn, this means that $\mathcal{E}$ succeeds in outputting $w_1$ that satisfies $u_1$ input to $\mathcal{V}$ with probability $\epsilon - \mathsf{negl}(\lambda)$.

□

When proving constructions secure, reasoning about knowledge-soundness directly is typically cumbersome. To alleviate this issue, prior work [BCC+16] observes that most protocols are algebraic: The corresponding extractor typically runs the malicious prover multiple times with refreshed verifier randomness to retrieve accepting transcripts, which can be interpolated to retrieve the witness. Leveraging this insight, [BCC+16] provide a general extraction lemma, which states that to prove knowledge-soundness for algebraic protocols, it is sufficient to show that there exists an extractor that can produce a satisfying witness when provided a *tree of accepting transcripts* with refreshed verifier randomness at each layer. This proof technique has been adapted to various settings [BCL+21, KST21, BBB+18, BCS21], and we similarly provide the corresponding lemma for reductions of knowledge.

**Definition 11 (Tree of Transcripts).** Consider a $m$-round reduction $(\mathcal{G}, \mathcal{P}, \mathcal{V})$. For $n_1, \ldots, n_m \in \mathbb{N}$, an $(n_1, \ldots, n_m)$-tree accepting of transcripts for fixed statement $u_1$ comprises of $n_1$ partial transcripts with fresh randomness in the verifier's first message; and for each such transcript, $n_2$ partial transcripts with fresh randomness in the verifier's second message; and so on, for a total of $\prod_i n_i$ leaves comprising of accepting transcripts and corresponding witnesses $w_2$ output by the prover.

**Lemma 7 (Tree Extraction [BCS21]).** An $m$-round reduction $(\mathcal{G}, \mathcal{P}, \mathcal{V})$ satisfies knowledge soundness if there exists an PPT extractor $\chi$ such that for all instances $u_1$, outputs a satisfying witness $w_1$ with probability $1 - \mathsf{negl}(\lambda)$, given an $(n_1, \ldots, n_m)$-tree of accepting transcripts where $\prod_i n_i = \mathsf{poly}(\lambda)$.

*Proof (Sketch).* The proof of Lemma 7 is similar to that of [KST21], which considers the setting in which an extractor repeatedly runs the malicious prover with refreshed verifier randomness to produce both a tree of accepting transcripts and a tree of corresponding prover outputs. □

## 4 Tensor Reductions

Recall that the tensor reduction reduces the task of checking the evaluation of a tensor to checking the evaluation of tensors in constituent spaces via random linear combinations. Such a pattern can be viewed as generalization of recursive split-and-fold techniques introduced by [BCC+16], and expanded upon in various settings [BBB+18, BMMV19, BCS21].

**Construction 2 (Tensor Reduction).** Suppose for tensors $t \in (U \to X) \otimes (V \to Y)$ of rank $I$, $w \in U \otimes V$ of rank $J$, and $z \in X \otimes Y$ over ring $\mathsf{R}$, a verifier would like to check

$$t(w) \cong z \tag{4}$$

where $t$, $w$ can be decomposed as follows:

$$t = \sum_{i \in [I]} r_i \otimes s_i$$
$$w = \sum_{j \in [J]} u_j \otimes v_j.$$

By definition, the verifier can check (4) by checking $\sum_{i,j} r_i(u_j) \otimes s_i(v_j) \cong z$. Therefore, the prover begins by computing and sending $x_{ij} \leftarrow r_i(u_j)$ and $y_{ij} \leftarrow s_i(v_j)$ for all $i \in [I], j \in [J]$. The verifier checks

$$\sum_{i \in [I], j \in [J]} x_{ij} \otimes y_{ij} \cong z.$$

The verifier must still check that $x_{ij} = r_i(u_j)$ and $y_{ij} = s_i(v_j)$ for all $i, j$. To do so, the verifier takes a random linear combination of these checks by sending random $\alpha, \beta$ from sampling set $Q \subseteq \mathsf{R}$, and computing $x = \sum_{i,j} \alpha^i \beta^j x_{ij}, y = \sum_{i,j} \alpha^i \beta^j y_{ij}$. The verifier then outputs $(\alpha, \beta, x, y)$, reducing the original check to the task of (recursively) checking

$$\Big(\sum_i \alpha^i r_i\Big)\Big(\sum_j \beta^j u_j\Big) \cong x \quad \text{and} \quad \Big(\sum_i \alpha^i s_i\Big)\Big(\sum_j \beta^j v_j\Big) \cong y.$$

**Theorem 7 (Tensor Reduction, Theorem 2 Restated).** For tensors $t = \sum_i r_i \otimes s_i \in (U \otimes V) \to (X \otimes Y)$ of rank $I$, $w = \sum_j u_j \otimes v_j : U \otimes V$ of rank $J$, and $z : X \otimes Y$ over ring $\mathsf{R}$, the tensor reduction reduces the task of checking

$$t(w) \cong z \tag{5}$$

to the task of checking

$$\Big(\sum_i \alpha^i r_i\Big)\Big(\sum_j \beta^j u_j\Big) \cong x \quad \text{and} \quad \Big(\sum_i \alpha^i s_i\Big)\Big(\sum_j \beta^j v_j\Big) \cong y \tag{6}$$

for verifier output $(\alpha, \beta, x, y)$. Formally, if (5) is true, then (6) is true with probability 1, and if (5) is false, then (6) is false with probability at least $1 - \frac{IJ}{|Q|}$. The prover complexity, verifier complexity, and communication complexity are all proportional to $IJ$.

*Proof.* This follows from the properties of a sampling set and the Schwartz-Zippel Lemma [Sch80] extended to modules. □

**Example 1 (Knowledge Argument).** In order to build intuition for the mechanics of the tensor reduction, we detail how an argument of knowledge[3] for Pedersen commitments [Ped91] with

[3]See Appendix A, Definition 15

sublinear communication can be expressed as a tensor reduction. The resulting protocol is nearly identical to the recursive knowledge argument presented by [BCC$^+$16]. Consider group $\mathbb{G}$ of prime order $p$ where the discrete logarithm problem is hard and field $\mathbb{F} = \mathbb{Z}_p$, and treat both as $\mathbb{F}$-modules. Consider some public key $G \in \mathbb{G}^n$. We can interpret elements $g \in \mathbb{G}$ as a function mapping from $\mathbb{F}$ to $\mathbb{G}$ given by $g(a) \mapsto ga$. Suppose a verifier would like to check for some commitment $C \in \mathbb{G}$, that the prover knows vector $A \in \mathbb{F}^n$ such that

$$G(A) = C. \tag{7}$$

We show how to recursively reduce via the tensor reduction the task of checking Equation (7) to the task of checking a single scalar multiplication.

In the base case, when $n = 2^0$, the prover simply sends $A$ and the verifier checks $G(A) = C$.

Suppose now that $n = 2^i$. We have that

$$\mathbb{G}^n \cong \mathbb{G}^{n/2} \otimes (\mathbb{F}^*)^2$$

and

$$\mathbb{F}^n \cong \mathbb{F}^{n/2} \otimes (\mathbb{F})^2.$$

Let $\{\delta_0, \delta_1\}$ and $\{b_0, b_1\}$ be bases for $(\mathbb{F}^*)^2$ and $(\mathbb{F})^2$ respectively. Then, we have that

$$\begin{aligned} G &= G_0 \otimes \delta_0 + G_1 \otimes \delta_1 \\ A &= A_0 \otimes b_0 + A_1 \otimes b_1 \end{aligned}$$

for some $G_0, G_1 \in \mathbb{G}^{n/2}$ and $A_0, A_1 \in \mathbb{F}^{n/2}$. These terms could be interpreted as the first and second half of vectors $G$ and $A$. Therefore, checking Equation (7) is equivalent to checking

$$\Big(\sum_i G_i \otimes \delta_i\Big)\Big(\sum_j A_j \otimes b_j\Big) = C \otimes 1.$$

Stepping through the tensor reduction with respect to this decomposition, we have that the prover sends to the verifier $(G_i)(A_j), \delta_i(b_j)$ for $i, j \in \{0,1\}$. Explicitly, the prover sends the terms $(G_0(A_0), 1)$, $(G_0(A_1), 0)$, $(G_1(A_0), 0)$, and $(G_1(A_1), 1)$. We recognize that the first and last terms correspond with the first and second half of commitment $C$, and the middle two terms are cross terms. This pattern is standard among protocols that employ recursive "folding" [KST21, BBB$^+$18, BCS21].

Upon receiving these terms, the verifier checks that

$$\sum_{i,j \in \{0,1\}} G_i(A_j) \otimes \delta_i(b_j) \cong C,$$

which holds because

$$\begin{aligned} G_0(A_1) \otimes 1 + G_0(A_1) \otimes 0 + G_1(A_0) \otimes 0 + G_1(A_1) \otimes 1 &= (G_0(A_0) + G_1(A_1)) \otimes 1 \\ &\cong C. \end{aligned}$$

The verifier then samples and sends random $\alpha, \beta \leftarrow \mathbb{F}$, and sets the new statements to be checked to be

$$(G_0 + \alpha G_1)(A_0 + \beta A_1) = G_0(A_0) + \beta G_0(A_1) + \alpha G_1(A_0) + \alpha\beta G_1(A_1)$$

and

$$(\delta_0 + \alpha\delta_1)(b_0 + \beta b_1) = 1 + \beta \cdot 0 + \alpha \cdot 0 + \alpha\beta \cdot 1$$

The latter check is trivial and can be done immediately by the verifier. As for the former check, because $(G_0 + \alpha G_1) \in \mathbb{G}^{n/2}$ and $(A_0 + \beta A_1) \in \mathbb{F}^{n/2}$, it can be recursively reduced by invoking a knowledge argument over vectors of size $n = 2^{i-1}$.

## 5 Representing New and Existing Techniques as Tensor Reductions

### 5.1 Universality of Linearization

We start by showing that any non-linear computation can be expressed as a linear computation (i.e., a tensor) composed with a universal non-linear transformation. This result becomes pivotal later when showing that tensor reductions generalize sumcheck protocols. To build intuition, we start by considering the concrete setting of univariate polynomials over a field $\mathbb{F}$. Consider degree $k$ polynomial $P$. Then for some coefficients $c_0, \ldots, c_k \in \mathbb{F}$, $P$ can be expressed as

$$P(x) = c_0 + c_1 x + c_2 x^2 + \ldots + c_k x^k$$

$P$ is clearly non-linear in $x$; however, we observe that the non-linear transformation on $x$ does not depend on the coefficients of $P$. In particular we can define non-linear map $\iota : \mathbb{F} \to \mathbb{F}^k$ as follows:

$$\iota(x) \mapsto (1, x, x^2, \ldots, x^k)$$

and then define a *linear* map $\widetilde{P}$ that takes as input $\iota(x)$ and computes an inner-product with the vector of coefficients $(c_0, \ldots, c_k)$. That is,

$$\widetilde{P}((1, x, \ldots, x^k)) \mapsto \langle (c_0, \ldots, c_k), (1, x, \ldots, x^k) \rangle.$$

and thus we have that $P = \widetilde{P} \circ \iota$ where $\iota$ is a *universal* non-linear map, and $\widetilde{P}$ is a homomorphism. We generalize this approach to construct homomorphisms from multilinear polynomials over modules (Theorem 8).

**Lemma 8 (Universality of Linearization).** Let $\mathsf{R}$ be a commutative ring, and let $V$ be an $\mathsf{R}$-module. For degree bounds $K_1, \ldots, K_n$, let the map $\iota \in \mathsf{R}^n \to \otimes_{j \in [n]} \mathsf{R}^{K_j}$ be defined as follows:

$$\iota(u_1, \ldots, u_n) \mapsto \bigotimes_{j \in [n]} \bigoplus_{k \in [K_j]} u_j^k.$$

Then, an arbitrary polynomial $\varphi \in \mathsf{R}^n \to V$ induces a homomorphism $\widetilde{\varphi}$ mapping from $\otimes_{i \in [n]} \mathsf{R}^{K_i}$ to $V$ such that $\widetilde{\varphi} \circ \iota = \varphi$. In other words, the following diagram commutes.

$$\begin{tikzcd}
\mathsf{R}^n \arrow[r, "\varphi"] \arrow[d, "\iota"] & V \\
\otimes_j \mathsf{R}^{K_j} \arrow[ur, "\widetilde{\varphi}"] &
\end{tikzcd}$$

*Proof.* Because $\varphi$ is a polynomial, we have that

$$\varphi(u_1, \ldots, u_n) = \sum_i v_i \otimes \varphi_{i,1}(u_1) \otimes \ldots \otimes \varphi_{i,n}(u_n)$$

for polynomials $\varphi_{1,j}, \ldots, \varphi_{n,j}$ where $\varphi_{i,j}$ maps from $U_j$ to $\mathsf{R}$, and $v_i \in V$. Then for some degree bounds $K_1, \ldots, K_n$, we have

$$\varphi(u_1, \ldots, u_n) = \sum_i v_i \otimes \bigotimes_j \sum_{k \in \{0,\ldots,K_j\}} r_{i,j,k}(u_j^k)$$

for $r_{i,j,k} \in \mathsf{R}$. Aggregating the inputs, we have

$$\varphi(u_1, \ldots, u_n) = \left(\sum_i v_i \otimes \bigotimes_j \bigoplus_{k \in \{0,\ldots,K_j\}} r_{i,j,k}\right)\left(\bigotimes_j \bigoplus_{k \in \{0,\ldots,K_j\}} u_j^k\right)$$

where homomorphism $\oplus_{k \in \{0,\ldots K_j\}} r_{i,j,k}$ mapping from $\mathsf{R}^{K_j}$ to $\mathsf{R}$ is defined as

$$\Big(\bigoplus_{k \in \{0,\ldots,K_j\}} r_{i,j,k}\Big)\Big(\bigoplus_{k \in \{0,\ldots,K_j\}} u_j^k\Big) \mapsto \sum_{k \in \{0,\ldots,K_j\}} r_{i,j,k}(u_j^k)$$

Let

$$\widetilde{\varphi} \coloneqq \left(\sum_i v_i \otimes \bigotimes_j \bigoplus_{k \in [K_j]} r_{i,j,k}\right).$$

By observation, homomorphisms mapping to $\mathsf{R}$ are closed under direct sums and tensor products, and therefore $\widetilde{\varphi}$ is a homomorphism. Additionally, we have by definition

$$\left(\bigotimes_j \bigoplus_{k \in [K_j]} u_j^k\right) = \iota(u_1, \ldots, u_n).$$

And therefore, we have that

$$\varphi(u_1, \ldots, u_n) = (\widetilde{\varphi} \circ \iota)(u_1, \ldots, u_n).$$

$\square$

## 5.2 Recovering the Sumcheck Protocol

The previous sections give us a general methodology to linearize any polynomial over modules. We take advantage of this result to express the sumcheck protocol as a tensor reduction over (linearized) polynomials. As a result, we have that a large class of split-and-fold techniques [BCC+16, AC20, BBB+18] instantiated under a variety of cryptographic assumptions (e.g., discrete log, groups of unknown order, lattices, pairings) can be expressed as tensor reductions via the results of Bootle et al. [BCS21].

Our high level approach is as follows: First, we recall a simplified definition of the sumcheck protocol. Next, we define a linearized sumcheck protocol which represents running the tensor reduction on linearized multivariate polynomials decomposed as univariate polynomials. This effectively fixes the modules and decomposition rules necessary to fully specify the tensor reduction. Finally, we prove that a single step of the sumcheck protocol is structurally equivalent to a single step of the linearized sumcheck protocol.

We begin by recalling the sumcheck protocol generalized to modules [BCS21]. We make several simplifications for the sake of a more lucid presentation: First, we only define and consider a single recursive step of the sumcheck and prove that it is structurally equivalent to a single recursive step of the tensor reduction instantiated over multivariate polynomials. Equivalence between the full sumcheck protocol and the full recursive tensor reduction follows by induction. Second, we have the verifier immediately compute the statement polynomial in each recursive step, as opposed to deferring this computation until the end. The purpose of this modification is to avoid having to carry the randomness generated by both the tensor reduction and sumcheck protocol throughout all the rounds in a global statement. Finally, we assume that both protocols use the standard monomial basis. Similar results hold for an arbitrary basis.

**Relation 1 (Sumcheck Relation).** Consider ring $\mathsf{R}$, $\mathsf{R}$-module $V$, and subset $H \subseteq \mathsf{R}$. The sumcheck relation $\mathsf{R}_{\mathsf{SC}}$, characterized by the number of variables $n$, is defined to be

$$\mathcal{R}_{\mathsf{SC}}(n) = \left\{ \ ((P, \sigma), \perp) \ \middle| \ \begin{array}{l} P \in \mathsf{R}^n \to V, \sigma \in V, \\ \sum_{x_1,\ldots,x_n \in H} P(x_1, \ldots, x_n) = \sigma. \end{array} \right\}$$

For notational simplicity, we omit $\perp$.

**Construction 3 (Sumcheck Protocol [LFKN92, BCS21]).** Consider ring $\mathsf{R}$, $\mathsf{R}$-module $V$, and subset $H \subseteq \mathsf{R}$. Suppose for some polynomial $P \in \mathsf{R}^n \to V$ with degree $K - 1$ in each variable, and claimed sum $\sigma \in V$, the verifier would like to check

$$(P, \sigma) \in \mathcal{R}_{\mathsf{SC}}(n)$$

The prover sends to the verifier *evaluations* of degree $K - 1$ polynomial

$$p(X) = \sum_{x_1,\ldots,x_{n-1} \in H} P(x_1, \ldots, x_{n-1}, X).$$

The verifier checks

$$\sum_{x_n \in H} p(x_n) = \sigma.$$

The verifier then samples and sends $\alpha$ from a sampling set $Q$ in $\mathsf{R}$ and computes

$$\sigma' \leftarrow p(\alpha)$$
$$P'(X_1, \ldots, X_{n-1}) \leftarrow P(X_1, \ldots, X_{n-1}, \alpha),$$

reducing the original task to the task of checking

$$(P', \sigma') \in \mathcal{R}_{\mathsf{SC}}(n - 1).$$

**Lemma 9 (Sumcheck Protocol [LFKN92, BCS21]).** The sumcheck protocol is a reduction for $(\mathcal{R}_{\mathsf{SC}}(n), \mathcal{R}_{\mathsf{SC}}(n - 1))$.

Next we describe a linearized sumcheck statement and a corresponding linearized sumcheck protocol which leverages the tensor reduction.

**Relation 2 (Linearized Sumcheck Relation).** Consider ring $\mathsf{R}$, $\mathsf{R}$-module $V$, and subset $H \subseteq \mathsf{R}$. Suppose for some polynomial map $P \in \mathsf{R}^n \to V$ with degree $K-1$ in each variable, and claimed sum $\boldsymbol{\sigma} \in V$, the verifier would like to check

$$\sum_{u_1,\ldots,u_n \in H} P(u_1, \ldots, u_n) = \boldsymbol{\sigma}. \tag{8}$$

By the universality of linearization (Theorem 8), there exists tensor $\mathbf{P} \in V \otimes \bigotimes_{i\in[n]} \mathsf{R}^K$ such that $P = \mathbf{P} \circ \iota$. Because $\mathbf{P}$ is linear in its inputs, by letting

$$\mathbf{U} = \sum_{u_1,\ldots,u_n \in H} \iota(u_1, \ldots, u_n),$$

the verifier can check equation (8) by checking

$$\mathbf{P}(\mathbf{U}) \cong \boldsymbol{\sigma}.$$

This motivates defining the corresponding *linearized* sumcheck relation

$$\mathcal{R}_{\mathsf{LSC}}(n) = \left\{ \ ((\mathbf{P}, \boldsymbol{\sigma}), \perp) \ \middle| \ \begin{array}{l} \mathbf{P} \in V \otimes \bigotimes_{i\in[n]} \mathsf{R}^K, \boldsymbol{\sigma} \in V, \\ \mathbf{U} = \sum_{u_1,\ldots,u_n \in H} \iota(u_1, \ldots, u_n), \\ \mathbf{P}(\mathbf{U}) = \boldsymbol{\sigma} \end{array} \right\}$$

For notational simplicity, we omit $\perp$.

**Construction 4 (Linearized Sumcheck Protocol).** For arbitrary ring $\mathsf{R}$, $\mathsf{R}$-module $V$, subset $H \subseteq \mathsf{R}$, and degree bound $K$, we build a reduction for $(\mathcal{R}_{\mathsf{LSC}}(n), \mathcal{R}_{\mathsf{LSC}}(n-1))$. Let $\{\delta_1, \ldots, \delta_K\}$ represent a canonical basis for $\mathsf{R}^K$ and let $H = \{h_1, \ldots, h_m\}$. For $(\mathbf{P}, \boldsymbol{\sigma}) \in \mathsf{R}_{\mathsf{LSC}}(n)$, and $\mathbf{U} = \sum_{u_1\ldots,u_n \in H} \iota(u_1, \ldots, u_n)$ we have that

$$\mathbf{P} = \sum_{i\in[K]} \mathbf{P}_i \otimes \delta_i \in \Big(V \otimes \big( \bigotimes_{\ell\in[n-1]} \mathsf{R}^K \big)\Big) \otimes \mathsf{R}^K$$

for some $\mathbf{P}_i$, and

$$\mathbf{U} = \sum_{j\in[m]} \mathbf{U}' \otimes \mathbf{h}_j \in \Big( \bigotimes_{\ell\in[n-1]} \mathsf{R}^K \Big) \otimes \mathsf{R}^K.$$

where $\mathbf{U}' = \sum_{u_1,\ldots,u_{n-1} \in H} \iota(u_1, \ldots, u_{n-1})$, and $\mathbf{h}_j = (h_j^0, \ldots, h_j^{K-1})$. Applying the tensor reduction with respect to this decomposition reduces the verifier's task of checking the original check to the task of checking

$$\Big(\sum_i \alpha^i \delta_i\Big)\Big(\sum_j \beta^j \mathbf{h}_j\Big) = x$$

which the verifier checks immediately and

$$\Big(\sum_i \alpha^i \mathbf{P}_i\Big)\Big((\sum_j \beta^j)\mathbf{U}'\Big) = y$$

for $\alpha, \beta, x, y \in \mathsf{R}$ generated during the reduction. Thus, the verifier computes $\mathbf{P}' = (\sum_i \alpha^i \mathbf{P}_i)$ and $\boldsymbol{\sigma}' = y/(\sum_j \beta^j)$ and reduces the original check to the task of checking $(\mathbf{P}', \boldsymbol{\sigma}') \in \mathcal{R}_{\mathsf{LSC}}(n-1)$.

Given constructions for both the sumcheck protocol and the linearized sumcheck protocol, we can now prove that the two are structurally equivalent. To do so we will show that a single iteration of the sumcheck protocol is equivalent to first linearizing the statement polynomials, running the linearized sumcheck protocol and mapping the resulting statement back into the original space, and additionally show that the generated transcript from the linearized sumcheck protocol can be used to recover the transcript produced by the standard sumcheck protocol. It is important to note that we *cannot* show that the linearized sumcheck protocol transcript is equivalent to the sumcheck protocol transcript. This is because the tensor reduction transcript inherently contains more structural information, which must be thrown out to recover the sumcheck protocol transcript.

**Theorem 8 (Structural Equivalence).** Consider ring $\mathsf{R}$, $\mathsf{R}$-module $V$, subset $H = \{h_1, \ldots, h_m\} \subseteq \mathsf{R}$, and degree bound $K-1$. Define the isomorphism $\Phi$ from a statement in $\mathcal{R}_{\mathsf{SC}}$ to a statement in $\mathcal{R}_{\mathsf{LSC}}$ as follows

$$\Phi((P, \sigma)) \mapsto (\mathbf{P}, \sigma)$$

where $\mathbf{P}$ is the tensor guaranteed by the universality of linearization such that $P = \mathbf{P} \circ \iota$. Additionally, define the map $\tau$ from a transcript of $\langle \mathcal{P}_{\mathsf{SC}}, \mathcal{V}_{\mathsf{SC}}(\rho)\rangle$ to a transcript of $\langle \mathcal{P}_{\mathsf{LSC}}, \mathcal{V}_{\mathsf{LSC}}(\rho)\rangle$ as follows

$$\tau\Big(\Big\{r_{ij}, s_{ij} \big| i \in [K], j \in [m]\Big\}, \alpha, \beta\Big) \mapsto \Big\{\sum_i r_{ij} \cdot s_{ij} \big| j \in [m]\Big\}, \alpha.$$

Then, the following diagram commutes.

$$\begin{tikzcd}
(P, \sigma) \in \mathcal{R}_{\mathsf{SC}}(n) \arrow[r, "\langle \mathcal{P}_{\mathsf{SC}}, \mathcal{V}_{\mathsf{SC}}(\rho)\rangle"] \arrow[d, "\Phi"] & (P', \sigma') \in \mathcal{R}_{\mathsf{SC}}(n-1), \mathsf{tr}_{\mathsf{SC}} \\
(\mathbf{P}, \boldsymbol{\sigma}) \in \mathcal{R}_{\mathsf{LSC}}(n) \arrow[r, "\langle \mathcal{P}_{\mathsf{LSC}}, \mathcal{V}_{\mathsf{LSC}}(\rho)\rangle"] & (\mathbf{P}', \boldsymbol{\sigma}') \in \mathcal{R}_{\mathsf{LSC}}(n-1), \mathsf{tr}_{\mathsf{LSC}} \arrow[u, "{\Phi^{-1}, \tau}"']
\end{tikzcd}$$

for identical verifier randomness $\rho$.

*Proof.* Consider arbitrary $(P, \sigma) \in \mathcal{R}_{\mathsf{SC}}(n)$. Let

$$P(X_1, \ldots, X_n) = P_1(X_1, \ldots, X_{n-1}) \cdot X_n^0 + \ldots + P_K(X_1, \ldots, X_{n-1}) \cdot X_n^{K-1}$$

Then, by linearity of $\Phi$, we have

$$\mathbf{P} = \mathbf{P}_1 \otimes \delta_1 + \mathbf{P}_2 \otimes \delta_2 + \ldots + \mathbf{P}_K \otimes \delta_K$$

where $\mathbf{P} = \Phi(P)$ and $\mathbf{P}_i = \Phi(P_i)$. Thus, for $\mathbf{h}_j = (h_j^0, \ldots, h_j^K)$, $\mathcal{P}_{\mathsf{LSC}}$ sends to $\mathcal{V}_{\mathsf{LSC}}$

$$\{\mathbf{P}_i(\mathbf{U}'), \delta_i(\mathbf{h}_j) | i \in [K], j \in [m]\} = \{\mathbf{P}_i(\mathbf{U}'), h_j^i | i \in [K], j \in [m]\}$$

However this means

$$\tau(\{\mathbf{P}_i(\mathbf{U}'), h_j^i | i \in [K], j \in [m]\}) = \Big\{\sum_i \mathbf{P}_i(\mathbf{U}') \cdot h_j^i | j \in [m]\Big\}$$

$$= \Big\{ \sum_{x_1,\ldots,x_n} P(x_1, \ldots, x_{n-1}, h_j) | j \in [m] \Big\}$$

which is precisely equal to the set of evaluations of $p(X)$ that $\mathcal{P}_{\mathsf{SC}}$ sends to $\mathcal{V}_{\mathsf{SC}}$. Additionally, by assumption both $\mathcal{V}_{\mathsf{LSC}}$, and $\mathcal{V}_{\mathsf{SC}}$ are initialized with the same randomness. This means that $\mathsf{tr}_{\mathsf{LSC}}$ and $\mathsf{tr}_{\mathsf{SC}}$ contain the same challenge $\alpha$ sent by the verifier. Therefore we have that $\tau(\mathsf{tr}_{\mathsf{LSC}}) = \mathsf{tr}_{\mathsf{SC}}$.

Next, we observe that

$$\Phi^{-1}(\mathbf{P}') = \Phi^{-1}\Big(\sum_i \mathbf{P}_i \cdot \alpha^i\Big) = \sum_i \Phi^{-1}(\mathbf{P}_i) \cdot \alpha^i = \sum_i P_i \cdot \alpha^i = P'.$$

Additionally, we observe that

$$\boldsymbol{\sigma}' = \sum_{i,j} \alpha^i \beta^j \mathbf{P}_i(\mathbf{U}')/\Big(\sum_j \beta^j\Big) = \sum_i \alpha^i \mathbf{P}_i(\mathbf{U}') = \sum_{x_1,\ldots,x_n} P(x_1, \ldots, x_{n-1}, \alpha) = \sigma'$$

Therefore, we have also have that $\Phi^{-1}(\mathbf{P}', \boldsymbol{\sigma}') = (P', \sigma')$. □

**Corollary 1 (Linearized Sumcheck Protocol).** The linearized sumcheck protocol is a reduction for $(\mathcal{R}_{\mathsf{LSC}}(n), \mathcal{R}_{\mathsf{LSC}}(n-1))$.

*Proof.* Completeness follows from Theorem 8. Soundness follows from the soundness of the tensor reduction. □

## 5.3 A Canonical Reduction of Knowledge over Tensors

Recall that the canonical relation for a statement of the form $u(w) \cong v$ treats $(u, v)$ as the statement and $w$ as the witness. In a corresponding reduction of knowledge, a prover argues that it knows $w$ such that $u(w) \cong v$. We first recall the formal definition for the canonical relation over tensor statements. We then construct the corresponding canonical reduction of knowledge (when working over vector spaces in particular), which becomes a central building block in reductions of knowledge for $\mathsf{NP}$.

**Definition 12 (Canonical Tensor Relation, Definition 1 Restated).** For modules $W$, $V$, and $U \cong (W \to V)$, let the corresponding canonical tensor relation be defined as

$$\mathcal{R}((U, V), W) = \left\{ ((u, v), w) \;\middle|\; \begin{array}{l} u \in U, v \in V, w \in W, \\ u(w) = v, \\ \mathsf{rank}(w) \le J \end{array} \right\}$$

for some implicit rank bound $J$.

**Construction 5 (Canonical Reduction of Knowledge over Tensors).** Consider field $\mathbb{F}$, length parameter $n$, $\mathbb{F}$-modules $W \cong W' \otimes \mathbb{F}^n$, $V \cong V' \otimes \mathbb{F}$, and $U \cong U' \otimes (\mathbb{F}^*)^n$ where $U' \cong (W' \to V')$. We construct a reduction of knowledge for $(\mathcal{R}((U, V), W), \mathcal{R}((U', V'), W'))$. Let $\{b_i\}$ be a basis for $\mathbb{F}^n$ and let $\{\delta_i\}$ be the corresponding dual basis. Suppose the prover and verifier are provided statement $u = \sum_i u_i \otimes \delta_i \in U$, and $v \in V$. Additionally, suppose the prover is provided satisfying witness $w = \sum_j w_j \otimes b_i \in W$. The prover and verifier run the tensor reduction on the statement

$$u(w) \cong v.$$

At the end of interaction, the verifier outputs $(\alpha, \beta, v', c)$ where $c \in \mathbb{F}$. The prover and verifier compute $u' = \sum_i \alpha^i u_i$ and the prover additionally computes $w' = \sum_j \beta^j w_j$ as dictated by the tensor reduction. The prover and verifier set the new check to be $((u', v'), w') \in \mathcal{R}((U', V'), W')$.

**Theorem 9 (Canonical Reduction of Knowledge over Tensors).** Consider field $\mathbb{F}$, length parameter $n$, $\mathbb{F}$-modules $W \cong W' \otimes \mathbb{F}^n$, $V \cong V' \otimes \mathbb{F}$, and $U \cong U' \otimes (\mathbb{F}^*)^n$ where $U' \cong (W' \to V')$. Construction 5 is a reduction of knowledge for

$$\Big(\mathcal{R}((U, V), W),\ \mathcal{R}((U', V'), W')\Big).$$

*Proof.* Consider instance $u = \sum_{i \in [I]} u_i \otimes \delta_i$ and $v$. We prove knowledge soundness via tree extraction (Lemma 7). That is, we construct extractor $\chi$ that outputs $w$ such that $u(w) \cong v$ given a tree of accepting transcripts and corresponding output prover witnesses.

Indeed, let $I$ be the rank of the statement, and $J$ be the implicit rank bound specified by the relation. Suppose the extractor $\chi$ is provided with $IJ$ accepting transcripts $\tau_{mk}$ with the same prover's first message

$$\{(v_{1,ij}, v_{2,ij}) | i \in [I], j \in [J]\}$$

and with randomness $(\alpha_m, \beta_{mk})$ for $m \in [I], k \in [J]$. Let $w'_{mk} \in W'$ for $k \in [IJ]$ denote the corresponding satisfying witnesses. For $m \in [I]$, the extractor solves for $w_{mj} \in W'$ for $j \in [J]$ such that

$$\sum_{j \in [J]} \beta_{mk}^j w_{mj} \cong w'_{mk} \tag{9}$$

for $k \in [J]$ using an inverse Vandermonde matrix (where invertibility is afforded by working over a field). Because $w'_{mk}$ is a satisfying witness, for all $m \in [I], k \in [J]$ we have that

$$\Big(\sum_i \alpha_m^i u_i\Big)\Big(w'_{mk}\Big) \cong \Big(\sum_{i,j} \alpha_m^i \beta_{mk}^j v_{1,ij}\Big).$$

Then, by Equation (9) we have for all $m \in [I], k \in [J]$

$$\Big(\sum_i \alpha_m^i u_i\Big)\Big(\sum_j \beta_{mk}^j w_{mj}\Big) \cong \Big(\sum_{i,j} \alpha_m^i \beta_{mk}^j v_{1,ij}\Big).$$

This in turn implies that for all $m \in [I]$ and $j \in [J]$

$$\Big(\sum_i \alpha_m^i u_i\Big)\Big(w_{mj}\Big) \cong \Big(\sum_i \alpha_m^i v_{1,ij}\Big). \tag{10}$$

To compute a satisfying witness, the extractor first computes $a_{nj}$ for all $m \in [I], j \in [J]$ such that for all $i \in \{0, \dots, I-1\}$

$$\sum_{m \in [I]} \alpha_m^i a_{mj} \cong v_{2,ij}. \tag{11}$$

Next, the extractor computes

$$w \leftarrow \sum_{l\in[I]} \sum_{m\in[I]} \sum_{j\in[J]} a_{mj} \cdot \alpha_m^l \cdot w_{mj} \otimes b_l \tag{12}$$

We will now show that $w$ is a satisfying witness. Indeed, we have

$$\begin{aligned}
u(w) &\cong \Big(\sum_i u_i \otimes \delta_i\Big)\Big(\sum_{l,m,j} a_{mj} \cdot \alpha_m^l \cdot w_{mj} \otimes b_l\Big) && \text{By (12).}\\
&\cong \sum_{i,l,m,j} a_{mj} \cdot \alpha_m^l \cdot u_i(w_{mj}) \otimes \delta_i(b_l) \\
&\cong \sum_{i,m,j} a_{mj} \cdot \alpha_m^i \cdot u_i(w_{mj}) && \text{By } \delta_i(b_l) \cong 0 \text{ for } i \neq l.\\
&\cong \sum_{m,j} a_{mj} \cdot \sum_i \alpha_m^i \cdot u_i(w_{mj}) \\
&\cong \sum_{m,j} a_{mj} \cdot \sum_i \alpha_m^i \cdot v_{1,ij} && \text{By (10).}\\
&\cong \sum_{i,j} v_{1,ij} \cdot \sum_m \alpha_m^i a_{mj} \\
&\cong \sum_{i,j} v_{1,ij} \cdot v_{2,ij} && \text{By (11).}\\
&\cong v. && \text{By the verifier's check.}
\end{aligned}$$

□

## 5.4 A Reduction of Knowledge for NP

We demonstrate that the canonical tensor reduction of knowledge can be used to construct reductions of knowledge for various linear-algebraic relations. We first construct a reduction of knowledge for linear-forms that is reminiscent of a similar protocol derived by [AC20]. We then construct a reduction of knowledge from bilinear-forms to linear-forms, which in turn enables a reduction of knowledge for NP. We start by defining a trivial relation to represent arguments of knowledge as reductions of knowledge.

**Relation 3 (Boolean Relation).** Let $\mathcal{R}_{\mathsf{bool}} = \{(\mathsf{true}, \bot)\}$. A reduction of knowledge that reduces to $\mathcal{R}_{\mathsf{bool}}$ can be seen as reducing a statement to either true or false. This corresponds to the standard notion of an argument of knowledge.

**Relation 4 (Linear Forms).** Consider group $\mathbb{G}$ of prime order $p$ and field $\mathbb{F} = \mathbb{Z}_p$ such that the bilinear relation and the coset equality assumptions hold for $(\mathbb{G}, \mathbb{F})$, and treat both as $\mathbb{F}$-modules. Consider public key $G \in \mathbb{G}^n$. Suppose the verifier would like to check for some commitment $\overline{A} \in \mathbb{G}$, public vector $B \in (\mathbb{F}^*)^n$, and scalar $\sigma \in \mathbb{F}$ that the prover knows vector $A \in \mathbb{F}^n$ such that

$$G(A) = \overline{A} \quad \text{and} \quad B(A) = \sigma. \tag{13}$$

We define homomorphism $G \oplus B \in \mathbb{G}^n \oplus (\mathbb{F}^*)^n$ as follows

$$(G \oplus B)(A) \mapsto G(A) \oplus B(A).$$

Then, checking Equation (13) is equivalent to checking

$$(G \oplus B)(A) \cong \overline{A} \oplus \sigma.$$

Therefore, by defining

$$\begin{aligned}\mathsf{LF}_n &= ((\mathsf{LF}_n^*, \mathsf{LF}_n^\mathsf{T}), \mathsf{LF}_n) \\ &= ((\mathbb{G}^n \oplus (\mathbb{F}^*)^n, \mathbb{G} \oplus \mathbb{F}), \mathbb{F}^n),\end{aligned}$$

a linear form can be checked by checking membership in the corresponding canonical relation $\mathcal{R}(\mathsf{LF}_n)$.

**Construction 6 (Linear Forms Knowledge Reduction).** For $n = 2^i$ where $i \geq 0$, we construct a reduction of knowledge for $(\mathcal{R}(\mathsf{LF}_n), \mathcal{R}_{\mathsf{bool}})$ by recursively composing reductions of knowledge for $(\mathcal{R}(\mathsf{LF}_n), \mathcal{R}(\mathsf{LF}_{n/2}))$ with a base case reduction of knowledge for $(\mathcal{R}(\mathsf{LF}_1), \mathcal{R}_{\mathsf{bool}})$.

In the base case, when $n = 2^0$, the prover simply sends witness $A \in \mathbb{F}$. The verifier then outputs $\mathsf{true}$ if $A$ satisfies the relation.

For $n = 2^i$ where $i \geq 1$, we leverage the tensor reduction to construct a knowledge reduction for $(\mathcal{R}(\mathsf{LF}_n), \mathcal{R}(\mathsf{LF}_{n/2}))$. Indeed, we have that

$$\mathsf{LF}_n^* \cong (\mathbb{G}^n \oplus (\mathbb{F}^*)^n) \cong (\mathbb{G}^{n/2} \oplus (\mathbb{F}^*)^{n/2}) \otimes (\mathbb{F}^*)^2 \cong \mathsf{LF}_{n/2}^* \otimes (\mathbb{F}^*)^2,$$

and

$$\mathsf{LF}_n \cong \mathbb{F}^n \cong \mathbb{F}^{n/2} \otimes \mathbb{F}^2 \cong \mathsf{LF}_{n/2} \otimes \mathbb{F}^2.$$

Additionally, we have that

$$\mathsf{LF}_n^\mathsf{T} \cong (\mathbb{G} \oplus \mathbb{F}) \cong (\mathbb{G} \oplus \mathbb{F}) \otimes \mathbb{F} \cong \mathsf{LF}_{n/2}^\mathsf{T} \otimes \mathbb{F}.$$

Therefore, applying the canonical reduction with respect to this decomposition reduces the task of checking membership in $\mathcal{R}(\mathsf{LF}_n)$ to the task of checking membership in $\mathcal{R}(\mathsf{LF}_{n/2})$.

**Lemma 10 (Linear Forms Knowledge Reduction).** Construction 6 is a reduction of knowledge for $(\mathcal{R}(\mathsf{LF}_n), \mathcal{R}_{\mathsf{bool}})$. The prover and verifier time complexity is $O(n)$. The communication complexity is $O(\log n)$.

*Proof.* Completeness follows by observation and knowledge soundness holds by the knowledge soundness of the underlying canonical reduction. In recursive call $i$, the verifier and prover take $O(2^i)$ time to compute the new statement. The prover additionally takes $O(2^i)$ time to compute the new witness and the values to send to the verifier. Because $(\mathbb{F}^*)^2$ and $\mathbb{F}^2$ each have a basis of size 2, the degree bound $J$ can be set to 2, and the verifier can have the prover only send 4 values. Because there are a total of $\log(n)$ recursive calls, the total prover and runtime is $O(n)$, and the total communication complexity is $O(\log n)$. □

**Relation 5 (Bilinear Forms).** Consider group $\mathbb{G}$ of prime order $p$ and corresponding field $\mathbb{F} = \mathbb{Z}_p$ such that the bilinear relation assumption holds for $(\mathbb{G}, \mathbb{F})$ and $(\mathbb{G}, \mathbb{G})$, and treat both as $\mathbb{F}$-modules. Consider public keys $G, H \in \mathbb{G}^m, \mathbb{G}^n$. Suppose the verifier would like to check for some public matrix

$M \in \mathbb{F}^{m\times n}$, commitments $\overline{A}, \overline{B} \in \mathbb{G}$, and scalar $\sigma \in \mathbb{F}$, that the prover knows vectors $A, B \in \mathbb{F}^m, \mathbb{F}^n$ such that

$$A^\top MB = \sigma \tag{14}$$
$$G(A) = \overline{A} \tag{15}$$
$$H(B) = \overline{B}. \tag{16}$$

By the universality of the tensor product (Lemma 2), there exists homomorphism $\mathbf{M} \in (\mathbb{F}^*)^m \otimes (\mathbb{F}^*)^n$ such that $\mathbf{M}(A \otimes B) = A^\top MB$. Therefore, to check Equation (14), it is sufficient to check $\mathbf{M}(A \otimes B) = \sigma$. To check Equations (15) and (16), due to the bilinear relation assumption, it is sufficient to check $(G \otimes H)(A \otimes B) = \overline{A} \otimes \overline{B}$. We define homomorphism $G \otimes H \oplus \mathbf{M} \in \mathbb{G}^m \otimes \mathbb{G}^n \oplus (\mathbb{F}^*)^m \otimes (\mathbb{F}^*)^n$ as follows

$$(G \otimes H \oplus \mathbf{M})(A \otimes B) \mapsto G(A) \otimes H(B) \oplus \mathbf{M}(A \otimes B).$$

By the previous assertions, checking Equations (14), (15), and (16) is equivalent to checking

$$(G \otimes H \oplus \mathbf{M})(A \otimes B) \cong \overline{A} \otimes \overline{B} \oplus \sigma.$$

Therefore, by defining

$$\begin{aligned}\mathsf{BF}_{m,n} &= ((\mathsf{BF}^*_{m,n}, \mathsf{BF}^\mathsf{T}_{m,n}), \mathsf{BF}_{m,n}) \\ &= ((\mathbb{G}^n \otimes \mathbb{G}^n \oplus (\mathbb{F}^*)^m \otimes (\mathbb{F}^*)^n, \mathbb{G} \otimes \mathbb{G} \oplus \mathbb{F}), \mathbb{F}^m \otimes \mathbb{F}^n)\end{aligned}$$

a bilinear form can be checked by checking membership in the corresponding canonical relation $\mathcal{R}(\mathsf{BF}_{m,n})$.[4]

**Construction 7 (Bilinear Forms Knowledge Reduction).** For $n = 2^i$ where $i \geq 0$, we construct a reduction of knowledge for $(\mathcal{R}(\mathsf{BF}_{m,n}), \mathcal{R}(\mathsf{LF}_m))$ by recursively composing reductions of knowledge for $(\mathcal{R}(\mathsf{BF}_{m,n}), \mathcal{R}(\mathsf{BF}_{m,n/2}))$ with a base case reduction of knowledge for $(\mathcal{R}(\mathsf{BF}_{m,1}), \mathcal{R}(\mathsf{LF}_m))$. Consider instance $G \otimes H \oplus \mathbf{M} \in \mathsf{BF}^*_{m,n}$ and $\overline{A} \otimes \overline{B} \oplus \sigma \in \mathsf{BF}^\mathsf{T}_{m,n}$.

In the base case, when $n = 2^0$ the prover simply sends $B \in \mathbb{F}$. The verifier is tasked with checking

$$(G \otimes H)(A \otimes B) \cong \overline{A} \otimes \overline{B} \quad \text{and} \quad \mathbf{M}(A \otimes B) \cong \sigma.$$

The verifier checks $H(B) \cong \overline{B}$ to reduce the former check to the task of checking $G(A) \cong \overline{A}$. The verifier then partially evaluates $\mathbf{M} \in (\mathbb{F}^*)^m \otimes \mathbb{F}^*$ on $B$ to retrieve $\mathbf{V} \in (\mathbb{F}^*)^m$ which reduces the latter check to the task of checking $\mathbf{V}(A) = \sigma$. Aggregating, the verifier is tasked with checking the following linear form

$$(G \oplus \mathbf{V})(A) \cong \overline{A} \oplus \sigma.$$

Suppose now that $n = 2^i$ for $i \geq 1$. We have that

$$\mathsf{BF}^*_{m,n} \cong (\mathbb{G}^m \otimes \mathbb{G}^n) \oplus ((\mathbb{F}^*)^m \otimes (\mathbb{F}^*)^n)$$

[4]Alternatively we can encode bilinear forms with $\mathsf{BF}_{m,n} = \mathsf{LF}_m \otimes \mathsf{LF}_n$ which is more structurally satisfying but leads to a reduction of knowledge that is both more complicated and concretely more expensive.

$$\cong \big((\mathbb{G}^m \otimes \mathbb{G}^{n/2}) \oplus ((\mathbb{F}^*)^m \otimes (\mathbb{F}^*)^{n/2})\big) \otimes (\mathbb{F}^*)^2$$
$$\cong \mathsf{BF}_{m,n/2} \otimes (\mathbb{F}^*)^2.$$

and

$$\mathsf{BF}_{m,n} \cong \mathbb{F}^m \otimes \mathbb{F}^n \cong (\mathbb{F}^m \otimes \mathbb{F}^{n/2}) \otimes \mathbb{F}^2 \cong \mathsf{BF}_{m,n/2} \otimes \mathbb{F}^2.$$

Additionally, we have that

$$\mathsf{BF}^{\mathsf{T}}_{m,n} \cong \mathbb{G} \otimes \mathbb{G} \oplus \mathbb{F} \cong (\mathbb{G} \otimes \mathbb{G} \oplus \mathbb{F}) \otimes \mathbb{F} \cong \mathsf{BF}^{\mathsf{T}}_{m,n/2} \otimes \mathbb{F}.$$

Therefore, the prover and verifier can recursively apply the canonical reduction with respect to this decomposition.

**Lemma 11 (Bilinear Forms Knowledge Reduction).** Construction 7 is a reduction of knowledge for $(\mathcal{R}(\mathsf{BF}_{m,n}), \mathcal{R}(\mathsf{LF}_m))$. The prover and verifier time complexity is $O(n)$. The communication complexity is $O(\log n)$.

*Proof.* Completeness follows by observation. The prover and verifier communication and computation complexity follow from a similar argument as Lemma 10.

As for knowledge soundness, consider instance $G \otimes H \oplus \mathbf{M} \in \mathsf{BF}^*_{m,n}$ and $\overline{A} \otimes \overline{B} \oplus \sigma \in \mathsf{BF}^{\mathsf{T}}_{m,n}$. We must construct an extractor that, when provided with oracle access to the interaction between the prover and verifier, outputs a witness $A \otimes B \in \mathbb{F}^n \otimes \mathbb{F}^m$ such that

$$(G \otimes H \oplus \mathbf{M})(A \otimes B) \cong \overline{A} \otimes \overline{B} \oplus \sigma.$$

Indeed, we have for each rewinded interaction the prover outputs witness $A$ such that

$$(G \oplus \mathbf{V})(A) \cong \overline{A} \oplus \sigma.$$

By the bilinear relation assumption over $(\mathbb{G}, \mathbb{F})$ we must have that all of these witnesses are equivalent with overwhelming probability. Let $A = (a_1, \ldots, a_n)$ denote this witness. Knowledge soundness of the base case follows by observation. Thus, by the knowledge soundness of the canonical tensor reduction and composability of knowledge soundness (Theorem 6), the extractor can retrieve witness $\sum_i A'_i \otimes B'_i \in \mathbb{F}^n \otimes \mathbb{F}^m$ such that

$$\Big(G \otimes H \oplus \mathbf{M}\Big)\Big(\sum_i A'_i \otimes B'_i\Big) \cong \overline{A} \otimes \overline{B} \oplus \sigma.$$

We will show that due to the bilinear relation assumption over $(\mathbb{G}, \mathbb{F})$ and $(\mathbb{G}, \mathbb{G})$, the witness must be of the form $A \otimes B$ for some efficiently computable $B$. Indeed, rearranging we have that

$$\sum_i A'_i \otimes B'_i \cong \sum_i \delta_i \otimes B_i$$

for canonical basis $\{\delta_i\}$ for $\mathbb{F}^n$, and some $B_i \in \mathbb{F}^m$. Then, we have

$$\Big(G \otimes H\Big)\Big(\sum_i \delta_i \otimes B_i\Big) \cong \sum_i G_i \otimes \overline{B}_i$$

where $\overline{B}_i = H(B_i)$. Additionally, we have

$$\overline{A} \otimes \overline{B} \cong \Big( \sum_i a_i \cdot G_i \Big) \otimes \overline{B} \cong \sum_i G_i \otimes (a_i \cdot \overline{B}).$$

By the bilinear relation assumption over $(\mathbb{G}, \mathbb{G})$ we have $\overline{B}_i \cong a_i \cdot \overline{B}$ for all $i \in [n]$ with overwhelming probability. This in turn implies $H(a_i^{-1} \cdot B_i) \cong \overline{B}$ for all $i \in [n]$. Then, by the bilinear relation assumption over $(\mathbb{G}, \mathbb{F})$, we have that

$$a_1^{-1} \cdot B_1 \cong \ldots \cong a_n^{-1} \cdot B_n$$

with overwhelming probability. Let $B = a_i^{-1} \cdot B_i$ denote the above value. Then we have that $A \otimes B$ is a satisfying witness because

$$A \otimes B \cong \sum_i a_i \cdot \delta_i \otimes B \cong \sum_i \delta_i \otimes B_i \cong \sum_i A'_i \otimes B'_i.$$

$\square$

**Relation 6 (NP Relation).** [KMP20] present an algebraic constraint system to encode arithmetic circuits where each constraint is a bilinear form. We can take advantage of this formulation to encode arguments for NP as linear statements. Consider group $\mathbb{G}$ of prime order and corresponding field $\mathbb{F}$ such that the bilinear relation assumption holds for $(\mathbb{G}, \mathbb{F})$.

For $n$ variables and $m = O(n)$ constraints, the constraint system consists of $m$ sparse matrices $M_1, \ldots, M_n \in \mathbb{F}^{n \times n}$ such that the total number of non-zero values in *all* matrices combined is $O(n)$, and public vector $X \in \mathbb{F}^{\ell}$. A vector $W \in \mathbb{F}^{n-\ell}$ is considered a satisfying witness if

$$Z^\top M_i Z = 0 \quad \forall i \in [m]$$

where $Z = (X, W)$. We formulate satisfiability of the above constraint system as an argument of knowledge as follows: Consider public key $G \in \mathbb{G}^n$ and a commitment $\overline{Z} \in \mathbb{G}$. Suppose a verifier would like to check that a prover knows $Z \in \mathbb{F}^n$ such that

$$\begin{aligned} (Z_1, \ldots, Z_\ell) &= X \\ Z^\top M_i Z &= 0 \quad \forall i \in [k] \\ G(Z) &= \overline{Z}. \end{aligned}$$

The first check is equivalent to checking $\ell$ linear forms over the same witness

$$(G \oplus \delta_i)(Z) \cong \overline{Z} \oplus X_i \quad \forall i \in [\ell]$$

As for the latter checks, recall for each $M_i$ there exists a corresponding tensor $\mathbf{M}_i \in \mathbb{F}^n \otimes \mathbb{F}^n$. Then, performing the latter two checks is equivalent to checking $m$ bilinear forms over the same witness and image

$$(G \otimes G \oplus \mathbf{M}_i)(Z \otimes Z) \cong \overline{Z} \otimes \overline{Z} \oplus 0 \quad \forall i \in [m]$$

**Construction 8 (A Reduction of Knowledge for $\mathsf{NP}$).** Suppose an $\mathsf{NP}$ instance with $n$ variables, $m$ constraints, and an $\ell$-sized public vector is expressed as $m$ bilinear forms in $\mathsf{BF}_{n,n}$ with the same witness and image, and $\ell$ linear forms in $\mathsf{LF}_n$ with the same witness. We construct a reduction of knowledge that reduces the task of checking such an instance into the task of checking membership in $\mathcal{R}_{\mathsf{bool}}$.

We first construct a reduction of knowledge that reduces the task of checking $m$ bilinear forms and $\ell$ linear forms to the task of checking a single bilinear form and a single linear form. The verifier can encode $\ell$ linear forms as a single linear form by taking a random linear combination. In more detail, suppose the verifier is tasked with checking

$$(G \oplus \delta_i)(Z) \cong \overline{Z} \oplus X_i \quad \forall i \in [\ell], \tag{17}$$

where $(G \oplus \delta_i) \in \mathsf{LF}^*_n$, $Z \in \mathsf{LF}_n$, and $\overline{Z} \oplus X_i \in \mathsf{LF}^{\mathsf{T}}_n$. The verifier samples and sends $\alpha \xleftarrow{\$} \mathbb{F}$, and reduces the task of checking Equation (17) to the task of checking the following linear form.

$$\Big(\sum_i \alpha^i (G \oplus \delta_i)\Big)(Z) \cong \sum_i \alpha^i (\overline{Z} \oplus X_i). \tag{18}$$

Similarly, the verifier can encode $m$ bilinear forms as a single bilinear form by taking a random linear combination. In particular, suppose the verifier is tasked with checking

$$(G \otimes G \oplus \mathbf{M}_i)(Z \otimes Z) \cong \overline{Z} \otimes \overline{Z} \oplus 0 \quad \forall i \in [m] \tag{19}$$

where $(G \otimes G \oplus \mathbf{M}_i) \in \mathsf{BF}^*_{n,n}$, $Z \otimes Z \in \mathsf{BF}_{n,n}$, and $(\overline{Z} \otimes \overline{Z} \oplus 0) \in \mathsf{BF}^{\mathsf{T}}_{n,n}$. The verifier samples and sends $\beta \xleftarrow{\$} \mathbb{F}$, and reduces the task of checking Equation (19) to the task of checking the following bilinear form

$$\Big(\sum_i \beta^i (G \otimes G \oplus \mathbf{M}_i)\Big)(Z \otimes Z) \cong \Big(\sum_i \beta^i\Big)(\overline{Z} \otimes \overline{Z} \oplus 0) \tag{20}$$

To check Equation (18), the prover and verifier run a reduction of knowledge for linear forms. To check Equation (20), The prover and verifier run a reduction of knowledge from bilinear forms to linear forms and then a reduction of knowledge for linear forms.

**Theorem 10 (A Reduction of Knowledge for $\mathsf{NP}$).** For arbitrary $n = 2^i, m = 2^j, \ell \in \mathbb{N}$ for $i, j \in \mathbb{N}$, Construction 8 is a reduction of knowledge from the task of checking $m$ bilinear forms in $\mathsf{BF}_{n,n}$ and $\ell$ linear forms in $\mathsf{LF}_n$ representing an $\mathsf{NP}$ instance to the task of checking membership in $\mathcal{R}_{\mathsf{bool}}$. The prover and verifier time complexity is $O(n)$. The communication complexity is $O(\log n)$.

*Proof (Sketch).* Completeness and knowledge soundness follow from the completeness and knowledge soundness of the underlying reductions of knowledge. Communication and computational complexity similarly follow. □

## 6 Acknowledgments

We thank Zoe Wellner for feedback on the appropriate use of concepts in module theory, Jonathan Lee for insights on proving the knowledge-soundness of tensor reductions and equivalence to sumcheck protocols, and Vipul Goyal for comments on earlier versions of this draft.

## A Supplementary Definitions

**Definition 13 (Ring [DF04]).** A **ring** $\mathsf{R}$ is a set together with two binary operations $+$ and $\times$ satisfying the following axioms:

(i) $(\mathsf{R}, +)$ is a commutative group,

(ii) $\times$ is associative:

$$(a \times b) \times c = a \times (b \times c) \quad \text{for all } a, b, c \in \mathsf{R},$$

(iii) multiplication distributes over addition:

$$(a + b) \times c = (a \times c) + (b \times c) \quad \text{and} \quad a \times (b + c) = (a \times b) + (a \times c)$$

for all $a, b, c \in \mathsf{R}$.

The ring $\mathsf{R}$ is *commutative* if multiplication is commutative. The ring $\mathsf{R}$ is said to have an *identity* (denoted 1) if there is an element $1 \in \mathsf{R}$ such that

$$1 \times a = a \times 1 = a \quad \text{for all } a \in \mathsf{R}.$$

**Definition 14 (Module [DF04]).** Consider commutative ring $\mathsf{R}$. An **$\mathsf{R}$-module** is a set $M$ together with

(i) A binary operation $+$ such that $(M, +)$ is a commutative group,

(ii) A map $R \times M \to M$, denoted by $rm$, for all $r \in \mathsf{R}$, $m \in M$ such that

$$\begin{aligned}(r + s)m &= rm + sm\\(rs)m &= r(sm)\\r(m + n) &= rm + rn\end{aligned}$$

for all $r, s \in R$ and $m, n \in M$.

(iii) In the case that $\mathsf{R}$ has a 1,

$$1m = m$$

for all $m \in M$.

**Definition 15 (Argument of Knowledge).** Consider ternary relation $\mathcal{R}$ over tuples consisting of public parameters, statement, and witness. An argument for $\mathcal{R}$ consists of PPT setup algorithm $\mathcal{G}$ called the generator and PPT interactive algorithms $\mathcal{P}$ and $\mathcal{V}$, called the prover and verifier respectively, with the following structure

- $\mathcal{G}(\lambda) \to \mathsf{pp}$: Takes as input security parameter $\lambda$, outputs public parameters $\mathsf{pp}$
- $\mathcal{P}(\mathsf{pp}, u, w)$: Takes as input public pameters $\mathsf{pp}$, statement $u$, and witness $w$. Interactively proves that $(\mathsf{pp}, u, w) \in \mathcal{R}$.

- $\mathcal{V}(\mathsf{pp}, u) \to \{0,1\}$: Takes as input public parameters $\mathsf{pp}$, and statement $u$. Outputs 0 for reject and 1 for accept.

Let $\langle \mathcal{P}(w), \mathcal{V} \rangle(\mathsf{pp}, u) = b$ denote the verifier's output bit $b$ after interacting with prover on common inputs $\mathsf{pp}$, $u$, and additional prover input $w$.

An argument satisfies **perfect completeness** if

$$\Pr\left[\ \langle \mathcal{P}(w), \mathcal{V} \rangle(\mathsf{pp}, u) = 1 \ \middle|\ (\mathsf{pp}, u, w) \in \mathcal{R}\ \right] = 1.$$

An argument satisfies **knowledge soundness** if for any PPT adversary $\mathcal{P}^*$ there exists a PPT extractor $\mathcal{E}$ such that for all instances $u$ associated with $\mathcal{R}$

$$\begin{aligned}&\Pr\left[\ (\mathsf{pp}, u, w) \in \mathcal{R} \ \middle|\ \begin{array}{l}\mathsf{pp} \leftarrow \mathcal{G}(\lambda),\\ w \leftarrow \mathcal{E}(\mathsf{pp}, u, \rho)\end{array}\ \right] \geq \\ &\Pr\left[\ \langle \mathcal{P}^*(\rho), \mathcal{V} \rangle(\mathsf{pp}, u) = 1 \ \middle|\ \mathsf{pp} \leftarrow \mathcal{G}(\lambda)\ \right] - \mathsf{negl}(\lambda)\end{aligned}$$

where $\rho$ denotes the input randomness for $\mathcal{P}^*$.

## References

[AC20] Thomas Attema and Ronald Cramer. Compressed sigma-protocol theory and practical application to plug & play secure algorithmics. In *Annual International Cryptology Conference*, pages 513–543. Springer, 2020.

[ACR20] Thomas Attema, Ronald Cramer, and Matthieu Rambaud. Compressed sigma-protocols for bilinear group arithmetic circuits and applications. Technical report, Cryptology ePrint Archive, Report 2020/1447, 2020.

[AFG$^+$10] Masayuki Abe, Georg Fuchsbauer, Jens Groth, Kristiyan Haralambiev, and Miyako Ohkubo. Structure-preserving signatures and commitments to group elements. In *Annual Cryptology Conference*, pages 209–236. Springer, 2010.

[BBB$^+$18] Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Greg Maxwell. Bulletproofs: Short proofs for confidential transactions and more. In *IEEE S&P*, 2018.

[BCC$^+$16] Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Jens Groth, and Christophe Petit. Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting. In *EUROCRYPT*, 2016.

[BCCT13] Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. Recursive composition and bootstrapping for snarks and proof-carrying data. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 111–120, 2013.

[BCG20] Jonathan Bootle, Alessandro Chiesa, and Jens Groth. Linear-time arguments with sublinear verification from tensor codes. In *Theory of Cryptography Conference*, pages 19–46. Springer, 2020.

[BCL$^+$21] Benedikt Bünz, Alessandro Chiesa, William Lin, Pratyush Mishra, and Nicholas Spooner. Proof-carrying data without succinct arguments. In *Annual International Cryptology Conference*, pages 681–710. Springer, 2021.

[BCS21] Jonathan Bootle, Alessandro Chiesa, and Katerina Sotiraki. Sumcheck arguments and their applications. Cryptology ePrint Archive, Report 2021/333, 2021. https://eprint.iacr.org/2021/333.

[BDFG20] Dan Boneh, Justin Drake, Ben Fisch, and Ariel Gabizon. Halo infinite: Recursive zk-snarks from any additive polynomial commitment scheme. Cryptology ePrint Archive, Report 2020/1536, 2020. https://ia.cr/2020/1536.

[BGH19] Sean Bowe, Jack Grigg, and Daira Hopwood. Recursive proof composition without a trusted setup. *Cryptol. ePrint Arch., Tech. Rep*, 1021:2019, 2019.

[BHK17] Zvika Brakerski, Justin Holmgren, and Yael Kalai. Non-interactive delegation and batch np verification from standard computational assumptions. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 474–482, 2017.

[BMMV19] Benedikt Bünz, Mary Maller, Pratyush Mishra, and Noah Vesely. Proofs for inner pairing products and applications. Cryptology ePrint Archive, Report 2019/1177, 2019.

[BOGG$^+$88] Michael Ben-Or, Oded Goldreich, Shafi Goldwasser, Johan Håstad, Joe Kilian, Silvio Micali, and Phillip Rogaway. Everything provable is provable in zero-knowledge. In *CRYPTO*, 1988.

[CHM$^+$20] Alessandro Chiesa, Yuncong Hu, Mary Maller, Pratyush Mishra, Noah Vesely, and Nicholas Ward. Marlin: Preprocessing zkSNARKs with universal and updatable SRS. In *EUROCRYPT*, 2020.

[CMT12] Graham Cormode, Michael Mitzenmacher, and Justin Thaler. Practical verified computation with streaming interactive proofs. In *ITCS*, 2012.

[DF04] David S. Dummit and Richard M. Foote. *Abstract algebra*. Wiley, 3rd ed edition, 2004.

[DLFKP16] Antoine Delignat-Lavaud, Cédric Fournet, Markulf Kohlweiss, and Bryan Parno. Cinderella: Turning shabby X. 509 certificates into elegant anonymous credentials with the magic of verifiable computation. In *IEEE S&P*, 2016.

[Dru15] Andrew Drucker. New limits to classical and quantum instance compression. *SIAM Journal on Computing*, 44(5):1443–1479, 2015.

[FS86] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *EUROCRYPT*, 1986.

[GGPR13] Rosario Gennaro, Craig Gentry, Bryan Parno, and Mariana Raykova. Quadratic span programs and succinct NIZKs without PCPs. In *EUROCRYPT*, 2013.

[GKR15] Shafi Goldwasser, Yael Tauman Kalai, and Guy N Rothblum. Delegating computation: Interactive proofs for muggles. *JACM*, 62(4), 2015.

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SICOMP*, 18(1), 1989.

[GWC19] Ariel Gabizon, Zachary J. Williamson, and Oana Ciobotaru. Plonk: Permutations over lagrange-bases for oecumenical noninteractive arguments of knowledge. Cryptology ePrint Archive, Report 2019/953, 2019. https://eprint.iacr.org/2019/953.

[HN10] Danny Harnik and Moni Naor. On the compressibility of NP instances and cryptographic applications. *SIAM Journal on Computing*, 39(5):1667–1713, 2010.

[Kil92] Joe Kilian. A note on efficient zero-knowledge proofs and arguments. In *STOC*, 1992.

[KMP20] Abhiram Kothapalli, Elisaweta Masserova, and Bryan Parno. Poppins: A direct construction for asymptotically optimal zksnarks. Cryptology ePrint Archive, Report 2020/1318, 2020. https://eprint.iacr.org/2020/1318.

[KMS+16] Ahmed Kosba, Andrew Miller, Elaine Shi, Zikai Wen, and Charalampos Papamanthou. Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. In *IEEE S&P*, 2016.

[KST21] Abhiram Kothapalli, Srinath Setty, and Ioanna Tzialla. Nova: Recursive zero-knowledge arguments from folding schemes. Cryptology ePrint Archive, Report 2021/370, 2021. https://eprint.iacr.org/2021/370.

[KZG10] Aniket Kate, Gregory M Zaverucha, and Ian Goldberg. Constant-size commitments to polynomials and their applications. In *ASIACRYPT*, 2010.

[Lee20] Jonathan Lee. Dory: Efficient, transparent arguments for generalised inner products and polynomial commitments. *IACR Cryptol. ePrint Arch.*, 2020:1274, 2020.

[LFKN92] Carsten Lund, Lance Fortnow, Howard Karloff, and Noam Nisan. Algebraic methods for interactive proof systems. *JACM*, 39(4), 1992.

[MBKM19] Mary Maller, Sean Bowe, Markulf Kohlweiss, and Sarah Meiklejohn. Sonic: Zero-knowledge SNARKs from linear-size universal and updatable structured reference strings. In *CCS*, 2019.

[Mei13] Or Meir. Ip= pspace using error-correcting codes. *SIAM Journal on Computing*, 42(1):380–403, 2013.

[Ped91] Torben Pryds Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *Annual international cryptology conference*, 1991.

[PHGR13] Bryan Parno, Jon Howell, Craig Gentry, and Mariana Raykova. Pinocchio: Nearly practical verifiable computation. In *IEEE S&P*, 2013.

[RZ21] Carla Ràfols and Arantxa Zapico. An algebraic framework for universal and updatable snarks. Cryptology ePrint Archive, Report 2021/590, 2021. https://eprint.iacr.org/2021/590.

[SCG+14] Eli Ben Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer, and Madars Virza. Zerocash: Decentralized anonymous payments from Bitcoin. In *IEEE S&P*, 2014.

[Sch80] Jacob T Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *JACM*, 27(4), 1980.

[Set20] Srinath Setty. Spartan: Efficient and general-purpose zkSNARKs without trusted setup. In *CRYPTO*, 2020.

[SL20] Srinath Setty and Jonathan Lee. Quarks: Quadruple-efficient transparent zksnarks. Cryptology ePrint Archive, Report 2020/1275, 2020. https://ia.cr/2020/1275.

[Val08] Paul Valiant. Incrementally verifiable computation or proofs of knowledge imply time/space efficiency. In *TCC*, 2008.

[WTS$^+$18] Riad S Wahby, Ioanna Tzialla, Abhi Shelat, Justin Thaler, and Michael Walfish. Doubly-efficient zkSNARKs without trusted setup. In *IEEE S&P*, 2018.

[ZGK$^+$17] Yupeng Zhang, Daniel Genkin, Jonathan Katz, Dimitrios Papadopoulos, and Charalampos Papamanthou. vSQL: Verifying arbitrary SQL queries over dynamic outsourced databases. In *IEEE S&P*, 2017.

[ZKP15] Yupeng Zhang, Jonathan Katz, and Charalampos Papamanthou. IntegriDB: Verifiable SQL for outsourced databases. In *CCS*, 2015.

[ZXZS20] Jiaheng Zhang, Tiancheng Xie, Yupeng Zhang, and Dawn Song. Transparent polynomial delegation and its applications to zero knowledge proof. In *IEEE S&P*, 2020.