

Neo and SuperNeo: Post-quantum folding with pay-per-bit costs over small fields

Wilson Nguyen
Microsoft Research

Srinath Setty
Microsoft Research

Abstract.

We construct the first folding scheme that simultaneously achieves six desirable properties: plausible post-quantum security, pay-per-bit commitment costs, field-native arithmetic (the sum-check and norm checks run purely over a small field), support for general (non-SIMD) constraint systems, small-field support (e.g., Goldilocks), and low recursion overheads. No existing scheme satisfies all six: group-based schemes (e.g., HyperNova) lack post-quantum security and are tied to large elliptic-curve fields; lattice-based schemes (e.g., LatticeFold) require expensive ring arithmetic, lose pay-per-bit costs, and impose SIMD constraints; and hash-based schemes (e.g., Arc) incur large verifier circuits.

We present two lattice-based folding schemes for CCS—an NP-complete relation generalizing R1CS, Plonkish, and AIR—called Neo and SuperNeo. Neo satisfies five of the six properties but requires SIMD constraint systems; SuperNeo removes this restriction and satisfies all six. SuperNeo also natively supports CCS relations over arbitrary extension fields of the field underlying the Ajtai commitment, without relying on an NTT embedding. Both run a single invocation of the sum-check protocol over a small field extension and achieve pay-per-bit costs via new folding-friendly instantiations of Ajtai commitments under the Module-SIS assumption. At the core of our constructions are two new norm-preserving embeddings of field vectors into ring vectors that respect an evaluation homomorphism required for folding. We also introduce *interactive reductions*, a framework that generalizes reductions of knowledge and enables modular security proofs for composed lattice-based protocols.

Table of Contents

1	Introduction	3
1.1	Six desiderata for a practical folding scheme.....	3
1.2	Our work: Neo and SuperNeo	5
1.2.1	Challenges and prior solutions	6
1.2.2	Contributions of our work.....	8
1.3	Related works	12
2	Technical overview	13
2.1	Breaking down HyperNova	13
2.2	The Neo embedding	14
2.3	The SuperNeo embedding	14
2.4	Proving the security with interactive reductions	16
3	Overview of the following sections	17
4	Preliminaries	17
5	Embeddings and Evaluation Homomorphism	22
6	Strong and weak interactive reductions	27
7	SuperNeo’s folding scheme for CCS	28
7.1	Relations	28
7.2	A folding scheme for CCS via interactive reductions	29
7.3	Interactive reduction for CCS – Π_{CCS}	30
7.4	Random linear combination reduction – Π_{RLC}	32
7.5	Decomposition reduction – Π_{DEC}	33
8	Concrete parameters	34
8.1	Almost Goldilocks: $(2^{64} - 2^{32} + 1) - 32$	34
8.2	Goldilocks: $(2^{64} - 2^{32} + 1)$	34
8.3	Mersenne 61: $2^{61} - 1$	34
	References	35
A	AI Disclaimer	40
B	Deferred theorems and proofs	40
B.1	Proof of Composition Theorem (Theorem 12)	40
B.2	Proofs for Π_{CCS}	42
B.3	Proofs for Π_{RLC}	48
B.4	Π_{DEC} is a Reduction of Knowledge (Theorem 13).....	54
B.5	Hardness and Inversion Bound calculation scripts	56
B.6	Lattice Estimator Script	58

1 Introduction

A folding scheme [58] is a cryptographic primitive that reduces the task of checking that two instance-witness pairs are in some NP relation to the task of checking that a single instance-witness pair is in the same relation. As an example, for a circuit C and two public inputs (i.e., instances) x_1 and x_2 , a folding scheme reduces the task of checking that there exist witnesses w_1 and w_2 such that $C(w_1, x_1) = 1$ and $C(w_2, x_2) = 1$ to the task of checking that there exists a single witness w for a specific public input x such that $C(w, x) = 1$. Furthermore, the verifier’s work in a folding scheme is limited to roughly taking the weighted sum of the commitments to the underlying witnesses. By using a folding scheme in a recursive manner, one can continually fold many instance-witness pairs into a single instance-witness pair, providing powerful recursive succinct argument primitives such as incrementally verifiable computation (IVC) [83] and proof-carrying data (PCD) [13].

Importance of folding schemes: prover efficiency and efficient recursion. A modern approach to construct SNARKs [48, 67] is to combine a polynomial interactive oracle proof (PIOP) [23, 30, 79] with a polynomial commitment scheme (PCS) [47]. However, this yields a “monolithic” SNARK whose prover must prove a fixed-sized computation at once. To scale to larger computations, one typically breaks the computation into smaller pieces and uses SNARK recursion (a la IVC or PCD) to produce a succinct argument [11]. Folding schemes provide a more direct and efficient alternative: they allow recursion to operate at the “statement” level (i.e., prior to producing a PIOP or a PCS evaluation argument), yielding two concrete benefits. First, recursion overheads are far lower: with Nova [58], folding a proof takes only $\approx 10,000$ R1CS constraints, whereas traditional SNARK recursion takes millions [29, 31]. Second, the prover incurs far less work by avoiding a full PIOP and PCS evaluation argument: in monolithic SNARKs such as Marlin [30], the prover performs at least $20\times$ higher work over simply committing to a witness, whereas with state-of-the-art folding schemes [20, 35, 56, 57], the prover’s work is dominated by the cost to commit to a witness. This results in at least an order of magnitude speedup over monolithic SNARKs, and up to two orders of magnitude when the witness contains values from a small subset of the entire field.¹

A motivating application: post-quantum signature aggregation. Ethereum’s consensus layer relies on BLS signatures [18], which offer efficient aggregation: a single pairing check suffices for hundreds of thousands of validator attestations. However, BLS signatures are not post-quantum secure. Ethereum’s planned transition to a post-quantum scheme (e.g., the hash-based XMSS [45]) reintroduces the scalability challenge, since such schemes lack the algebraic structure of BLS and their signatures are large and expensive to verify individually. A natural solution is to use recursive SNARKs to aggregate these signatures [34] in a distributed manner (a la proof-carrying data). Folding schemes are well suited here: each signature can be folded into the aggregate proof, yielding significantly lower latency and prover cost at each recursive step than traditional SNARK recursion—a critical advantage when up to a million signatures must be aggregated within time-sensitive slots of a consensus protocol.

Realizing this approach requires a folding scheme that is both efficient *and* plausibly post-quantum secure. More broadly, what properties should a practical folding scheme satisfy?

1.1 Six desiderata for a practical folding scheme

State-of-the-art folding schemes [20, 35, 54–58] have converged on an efficient “recipe”: the prover commits to a witness with a linearly homomorphic scheme (Pedersen or KZG) and employs sum-check-type techniques [63]. For instance, the prover in HyperNova [56] performs a *single* multi-scalar multiplication (MSM) to commit to its witness, with costs that scale with the bit-width of witness values (“pay-per-bit”). However, these schemes are not post-quantum secure and are tied to ≈ 256 -bit elliptic-curve fields. Below, we distill six properties that a practical folding scheme should satisfy.

¹ If the witness contains “small” field elements (e.g., from the set $\{0, 1, \dots, 2^{32} - 1\}$), state-of-the-art folding schemes perform more than two orders of magnitude less work than a monolithic SNARK prover such as Marlin [30]. Proof systems such as Spartan and variants [79, 80] incur lower overheads than Marlin, but they must still produce a PIOP and PCS evaluation argument.

PQ: Post-quantum security. The scheme should be plausibly secure against quantum adversaries. Group-based folding schemes [20, 35, 56–58] rely on the hardness of the discrete logarithm problem, which Shor’s algorithm [81] can efficiently solve on a quantum computer.

PB: Pay-per-bit commitment costs. The cost to commit to a witness should scale with the bit-width of the witness values: for example, committing to a vector of b -bit values should be roughly $\log |\mathbb{F}|/b$ times cheaper than committing to values that span the full field $\mathbb{F}$. Group-based schemes achieve this via Pedersen or KZG commitments. LatticeFold, however, relies on the NTT embedding to map field vectors into ring vectors. Because the NTT is not a norm-preserving map, the resulting ring vectors have arbitrary norm regardless of the bit-width of the original witness elements. Since the Ajtai commitment scheme requires decomposing these arbitrary-norm ring elements for binding, the commitment cost is the same whether the original values are 1-bit or 64-bit. Hash-based schemes (e.g., Arc [24]) also lack this property, since their commitment costs are determined by the code rate and security parameter, not by witness bit-widths.

FN: Field-native arithmetic. Besides committing to the witness, the prover’s and the verifier’s work—in particular, the norm check and the sum-check protocol—should operate natively over a prime field (or extension field), without performing expensive polynomial ring arithmetic. Group-based and hash-based schemes satisfy this property. LatticeFold [14], however, runs the sum-check protocol over a cyclotomic polynomial ring rather than over a prime field (or extension field); ring operations are 10–100 $\times$ more expensive than field operations.²

CS: General constraint systems. The scheme should support general NP-complete constraint systems such as CCS [80] over a single witness vector, without requiring the constraint system to be “data parallel” (SIMD). Group-based and hash-based schemes satisfy this. LatticeFold, by contrast, packs a batch of independent constraints defined over a small prime field into a single constraint over a cyclotomic polynomial ring [14, Remark 4.1], imposing a SIMD requirement. Lova [37] avoids this issue but at the cost of only supporting the subset sum relation, not CCS.

SF: Small-field support. The scheme should work natively over small prime fields, including popular SNARK-friendly fields such as Goldilocks.³ By “small” we mean fields whose modulus q fits within a machine register—for example, M61 ($q = 2^{61} - 1$) or Goldilocks ($q = 2^{64} - 2^{32} + 1$). Small-field support is important for two reasons: such fields offer arithmetic that is an order of magnitude faster than 256-bit arithmetic, and SNARK-friendly fields enable efficient proof compression using existing SNARKs (e.g., Spartan with a FRI-based polynomial commitment scheme). Group-based schemes are tied to the scalar fields of elliptic curves, which are ≈ 256 bits for security. LatticeFold’s cyclotomic rings of the form $X^d + 1$ (with d a power of 2) cause popular fields like Goldilocks to fully split the ring, ruining security [14, §3.3]; furthermore, supporting small fields requires a non-trivial NTT factor degree $\tau_{\text{NTT}} \mid d$, introducing a $\tau_{\text{NTT}} \times$ multiplicative overhead in the protocols due to the need for τ_{NTT} -parallel repetition [14, §3.3, §5].

LR: Low recursion overheads. The recursive verifier circuit should be small enough that the per-step prover cost of IVC remains practical. Group-based schemes achieve this: Nova’s and NeutronNova’s verifier circuits are constant-sized: $\approx 10,000$ R1CS constraints. Hash-based schemes suffer from large verifier circuits; for example, Arc [24] requires $2 \cdot \lambda / \log(1/\rho)$ Merkle tree openings, translating to $\approx 1,600,000$ R1CS constraints at $\lambda = 128$ and $\rho = 1/2$ with Poseidon [44]. Prior to this, Bünz et al. [25] provide a different hash-based scheme with even worse verifier circuit overheads [24, Table 1], and only provides “bounded depth” IVC. Lova also incurs extreme overheads, reporting a prover time of $\approx 3,000$ seconds for a subset sum instance of length 2^{19} [37, Table 2], compared to 500 ms for Nova on an R1CS instance of the same size. More broadly, folding schemes that rely on ring sum-check techniques—such as LatticeFold [14] and SALSAA [60]—inherit high recursion overheads because the recursive verifier must hash ring elements rather than field elements. For

² Benchmarks report that a polynomial ring multiplication costs ≈ 213 ns [68], whereas a field multiplication with M61 costs a fraction of a nanosecond.

³ Another desirable property, not captured in Figure 1, is native support for CCS over an arbitrary extension field of the commitment field. The commitment scheme should not fix the choice of relation field (for example, through the NTT splitting of its cyclotomic ring). One of our constructions achieves this property.

	PQ	PB	FN	CS	SF	LR
<i>Group-based</i>						
HyperNova [56]	✗	✓	✓	✓	✗	✓
NeutronNova [57]	✗	✓	✓	✓	✗	✓
<i>Hash-based</i>						
Arc [24]	✓	✗	✓	✓	✓	✗
<i>Lattice-based</i>						
LatticeFold [14]	✓	✗	✗	✗	✗	✗
Lova [37]	✓	✗	✓	✗	✓	✗
Neo (this work)	✓	✓	✓	✗ [†]	✓	✓
LatticeFold+ [16]	✓	✗	✗	✗	✗	✗
SALSAA [60]	✓	✗	✗	✗	✓	✗
SuperNeo (this work)	✓	✓	✓	✓	✓	✓

[†]Neo requires a SIMD constraint system (CS) and is subsumed by SuperNeo, which removes this requirement. The table lists schemes in the order they appeared; we present Neo and SuperNeo separately because Neo was independently preprinted and several subsequent works build on it, so separating the two simplifies attributing techniques to the correct scheme. The subsequent work Cyclo [42] is equivalent to Neo and as such is omitted in the table; see Section 1.3.

Fig. 1: Comparison of folding schemes against the six desiderata. ✓ = satisfied, ✗ = not satisfied. See Section 1.1 for definitions.

example, a single ring element in LatticeFold occupies 64×64 bits = 4,096 bits, compared to 256 bits for a 256-bit field element in HyperNova, resulting in $16\times$ more data for the verifier circuit to hash. Achieving constant verifier circuit size like Nova or NeutronNova in the lattice setting is difficult and remains an open problem. Our goal is to achieve logarithmic recursion overhead (with similar constants) analogous to HyperNova.

Research question. *Can we build a folding scheme that satisfies all six desiderata—in particular, one that is post-quantum secure, works natively over small prime fields, and matches the efficiency of state-of-the-art group-based schemes?*

Figure 1 summarizes the landscape. No existing folding scheme satisfies all six desiderata. Group-based schemes meet PB, FN, CS, and LR but fail PQ and SF. Hash-based schemes achieve PQ but sacrifice PB and LR. LatticeFold, LatticeFold+, and SALSAA each achieve PQ but fail PB, FN, CS, and LR; LatticeFold and LatticeFold+ additionally fail SF. Neo satisfies PQ, PB, FN, SF, and LR, but requires SIMD constraint systems (CS). SuperNeo is the first scheme to satisfy all six.

1.2 Our work: Neo and SuperNeo

We present Neo, which satisfies five of the six desiderata, and SuperNeo, which removes Neo’s requirement for SIMD constraint systems and is the first folding scheme to satisfy all six (Figure 1). SuperNeo also natively supports CCS relations over an arbitrary extension field $\mathbb{L}$ of the field $\mathbb{F}$ underlying the Ajtai commitment, without tying $\mathbb{L}$ to the NTT splitting of the cyclotomic ring. The relation and sum-check challenge fields may differ and are embedded into a common ambient field $\mathbb{A}$. Our constructions are lattice analogs of HyperNova [56]: the prover commits to a CCS [80] witness using a lattice-based commitment scheme with pay-per-bit costs, runs a single invocation of the sum-check protocol over an extension of a small prime field⁴, and achieves plausible post-quantum security under a standard structured lattice assumption (Module-SIS). Both schemes also provide multi-folding, which folds multiple CCS instances at once, amortizing the decomposition costs required to manage lattice norm growth. By applying standard

⁴ When using a 64-bit field, a degree-2 extension is sufficient for 128 bits of security.

compilers from folding schemes to IVC [56, 58] and PCD [86], we obtain a plausibly post-quantum IVC/PCD scheme. Since our constructions natively support SNARK-friendly fields like Goldilocks, they enable efficient proof compression using Spartan [79, 80] with a FRI-based polynomial commitment scheme [10]—without requiring any non-native arithmetic or field emulation.

The key technical challenge is *embedding* field vectors (CCS witnesses) into the ring vectors that Ajtai commitments operate over, while preserving the algebraic structure—norm bounds, evaluation homomorphisms—that a sum-check-based folding scheme like HyperNova requires. We introduce two new norm-preserving embeddings (the **Neo** and **SuperNeo embeddings**) that resolve this challenge, and a new security framework (**interactive reductions**) that enables modular proofs of knowledge soundness for lattice-based protocols. We detail the problems with prior approaches and our solutions in the following sections.

1.2.1 Challenges and prior solutions

Achieving desiderata PB, FN, CS, and LR for a lattice-based folding scheme requires solving a common problem: how to embed field vectors into the ring vectors that Ajtai commitments [2] operate over. Any such embedding must support protocols that check both the norm of the committed ring vectors and CCS constraints on the underlying field vectors. Below, we describe the Ajtai commitment scheme and the challenges that prior solutions leave open. In the introduction and technical overview, we first restrict to CCS relations over the base field $\mathbb{F} := \mathbb{F}_q$ and let $\mathbb{K} := \mathbb{F}_{q^\nu}$ be the sum-check challenge field, chosen such that $1/|\mathbb{K}| = \text{negl}(\lambda)$. We later lift this restriction to support CCS relations over extension fields.

Ajtai Commitment Scheme [2, 76] (Informal) ⁵

- **Setup**($1^\kappa, n \in \mathbb{N}$) $\rightarrow A \in \mathbb{R}_{\mathbb{F}}^{\kappa \times n}$, where A is a random matrix over the polynomial ring $\mathbb{R}_{\mathbb{F}} := \mathbb{F}[X]/(\phi(X))$ with modulus $\phi(X)$ being a cyclotomic polynomial of degree d .
- **Commit**($A, \mathbf{z} \in \mathbb{R}_{\mathbb{F}}^n$) $\rightarrow c$, where $\mathbf{z}$ is an arbitrary ring vector message and $c := A \cdot \mathbf{z} \in \mathbb{R}_{\mathbb{F}}^\kappa$ is a binding commitment to $\mathbf{z}$ if the norm $\|\mathbf{z}\|_\infty < b$ is small enough.

Problem 1: Inefficient constraint checking & lack of algebraically friendly embeddings Prior work on lattice-based folding schemes [15] relied on the Number Theoretic Transform (NTT) [5, 62, 76, 78] to embed field vectors into ring vectors. The Number Theoretic Transform is a ring isomorphism between the ring $\mathbb{R}_{\mathbb{F}}$ and a product space of extension fields $\mathbb{F}_{q^{\tau_{\text{NTT}}}}^{d/\tau_{\text{NTT}}}$, so each ring operation naturally simulates a Single Instruction, Multiple Data (SIMD) [40, 41] operation over the underlying (d/τ_{NTT}) -tuple of field elements. For $\cdot \in \{+, \times\}$,

$$\begin{array}{ccc}
 (a_1, \dots, a_{d/\tau_{\text{NTT}}}) \in \mathbb{F}_{q^{\tau_{\text{NTT}}}}^{d/\tau_{\text{NTT}}} & \xleftrightarrow[\text{iNTT}]{\text{NTT}} & \mathbf{a} \in \mathbb{R}_{\mathbb{F}} \\
 \downarrow \cdot (b_1, \dots, b_{d/\tau_{\text{NTT}}}) & & \downarrow \cdot \mathbf{b} \\
 (a_1 \cdot b_1, \dots, a_{d/\tau_{\text{NTT}}} \cdot b_{d/\tau_{\text{NTT}}}) \in \mathbb{F}_{q^{\tau_{\text{NTT}}}}^{d/\tau_{\text{NTT}}} & \xleftrightarrow[\text{iNTT}]{\text{NTT}} & \mathbf{a} \cdot \mathbf{b} \in \mathbb{R}_{\mathbb{F}}
 \end{array}$$

Hence, given field vectors $z^{(1)}, \dots, z^{(d/\tau_{\text{NTT}})} \in \mathbb{F}^n$, define $\mathbf{z}_i := \text{iNTT}(z_i^{(1)}, \dots, z_i^{(d/\tau_{\text{NTT}})}) \in \mathbb{R}_{\mathbb{F}}$ for every $i \in [n]$. This yields the ring vector $\mathbf{z} := (\mathbf{z}_1, \dots, \mathbf{z}_n) \in \mathbb{R}_{\mathbb{F}}^n$. By adapting a technique from [19], prior work LatticeFold [15, Sec 3.3] showed how to express the norm constraint on a ring vector $\mathbf{z}$ as Hadamard product constraints $\prod_{a=0}^{b-1} (\hat{\mathbf{z}}_j - a) = 0$ (with a viewed as a constant ring element) over related ring vectors $\hat{\mathbf{z}}_1, \dots, \hat{\mathbf{z}}_{\tau_{\text{NTT}}}$ whose underlying embedded field vectors are the coefficient vectors of the committed ring vector $\mathbf{z}$. Since the NTT transformation is a ring isomorphism, they also showed that field constraints (like CCS) over the embedded field vectors could be checked as ring constraints over the committed ring vector $\mathbf{z}$ itself. To check both the Hadamard and CCS constraints on these ring vectors, LatticeFold relies on the celebrated sum-check protocol [63] over the ring to reduce checking these constraints to checking random multilinear

⁵ To streamline our exposition, we directly discuss the more efficient variant of Ajtai commitments based on Module Short-Integer-Solution (M-SIS), because they serve as the basis of most lattice-based proof systems [12, 15, 16, 19, 70, 76].

evaluations of the ring vectors themselves. The main downside of this approach is that the prover and verifier have to execute the sum-check protocol over the ring itself rather than just the underlying field for which the original CCS constraints are defined over. As such both parties must perform ring operations, which are significantly more expensive than field operations as they must either directly compute or simulate (via NTT) the degree- d polynomial arithmetic. Furthermore, the NTT transformation itself (required for these checks) adds a significant overhead to the prover runtime; as such, lattice-based proof systems attempt to minimize the number of NTT transformations required [12, 70].

Achieving FN (field-native arithmetic) and LR (low recursion overheads) requires checking the norm and CCS constraints by **only performing a sum-check over the field**, avoiding all ring operations during the sum-check protocol. Moreover, since our goal is to construct a folding scheme, the embedding must respect an **evaluation homomorphism** so that the resulting field multilinear polynomial evaluations can be folded together.

Problem 2: Lack of packing efficiency & Pay-per-bit cost & SIMD Lattice-based proof systems targeting field constraints (such as CCS) must rely on some embedding of field vectors into ring vectors to be able to commit to these field vectors with Ajtai commitments. Hence, packing efficiency (the density of field elements embedded into ring vectors) is a crucial metric for the efficiency of these proof systems. In particular, the highest packing efficiency possible (information-theoretically) for a ring vector of length n is $d \cdot n$ field elements. Unfortunately, the most algebraically friendly embedding, the NTT transform, has only a packing efficiency of $(d/\tau_{\text{NTT}}) \cdot n$ field elements for a ring vector $\mathbf{z}$ of length n . For secure small-field parameterizations [15], τ_{NTT} may exceed the minimum extension degree required for sum-check soundness. Over a 64-bit field with $d = 64$, degree 2 provides approximately 128-bit sum-check soundness (Definition 11) but yields a trivial low-norm invertibility bound (Theorem 4). Degree 4 keeps this bound non-trivial, allowing for non-trivial challenge sets (Definition 6). Moreover, regardless of the norm of the original field vectors, the NTT embedding results in a ring vector $\mathbf{z}$ which has arbitrary norm. Hence, to utilize the Ajtai commitment scheme, the vector $\mathbf{z} \in \mathbb{R}_{\mathbb{F}}^n$ must be expanded into a larger ring vector $\mathbf{z}' \in \mathbb{R}_{\mathbb{F}}^{\log_b(|\mathbb{F}|) \cdot n}$; this further reduces the packing efficiency by a factor of $\log_b(|\mathbb{F}|)$, regardless of the original norm of the embedded vectors. Furthermore, when using the NTT embedding, the cost to commitment to these field elements does not scale with their bit size; Pedersen-style commitments [47, 73, 85] for field vectors have this pay-per-bit cost. Finally, the NTT embedding requires that d/τ_{NTT} separate field vectors be embedded into a single ring vector to reach its peak packing efficiency; however, this inherently requires that the underlying proof system to use a SIMD constraint system (such as CCS over multiple field vectors simultaneously). SIMD constraint systems are not well-suited for applications where the constraints cannot be neatly split into independent, smaller systems. In particular, the ideal constraint system would be over a single field vector of length $d \cdot n$; rather than d/τ_{NTT} independent constraint systems over the d/τ_{NTT} separate field vectors of length n (since the prior can immediately simulate the later, and use a larger constraint system).

Achieving PB (pay-per-bit) and CS (general constraints) thus requires an *algebraically friendly* embedding with **optimal packing efficiency** of $d \cdot n$ field elements for a ring vector of length n , with a **pay-per-bit** commitment cost, and without the requirement for a SIMD constraint system.

Problem 3: Complex security proofs lacking modular analysis In the literature of lattice-based proof systems [12, 15, 16, 26, 33, 43, 65], there are often protocols Π which have a particular form and whose security proofs are complex and non-modular. In particular, the prover and verifier take in as input some commitments to witnesses along with some property that needs to be checked on the underlying committed witnesses (such as norm or CCS constraints). We identify that these protocols Π can be broken down into two stages Π_{property} and Π_{special} . The first stage Π_{property} is often some sound testing protocol (such as sum-check or random projections) run on the underlying committed witnesses, which produce additional algebraic claims (such as multilinear evaluations or random inner products) on the witnesses. The second stage Π_{special} is often a special-sound protocol [1, 7, 8, 33, 39] which takes (informally) a random linear combination of the original commitments and of the algebraic claims produced by Π_{property} ; this results in a new commitment and a new algebraic claim on the underlying witness. These stages seem to mirror the structure of a reduction of knowledge [52], where proving the composed protocol $\Pi := \Pi_{\text{special}} \circ \Pi_{\text{property}}$ is secure requires proving that both Π_{property} and Π_{special} are knowledge-sound. Unfortunately, this is not the case; in particular, the

individual stages Π_{property} and Π_{special} do not meet the strict definition of a reduction of knowledge. As such, the security analysis of these protocols Π is often done in an ad-hoc manner, where the security proof directly analyzes the composed protocol Π as a whole (often leading to complex, non-blackbox security arguments).

A separate challenge is **generalizing the framework of reductions of knowledge** to capture these individual stages Π_{property} and Π_{special} —in particular, identifying what core properties each stage must satisfy—such that **their composition Π is provably secure** (knowledge-sound).

1.2.2 Contributions of our work

Since our work subsumes and extends the prior work Neo, we will discuss both the contributions of the original Neo work and our extension SuperNeo together.

Contribution 1: Neo and SuperNeo Embeddings

Can we check the norm and CCS constraints by only performing a sum-check over the field?

To answer this question, we can ask why the prior NTT-based approach required ring operations in the sum-check protocol in the first place. The main reason is that the NTT transform simply does not preserve enough structure between the underlying field vectors and the committed ring vector to meaningfully check both the norm and CCS constraints by only relying on field constraints (which can be checked solely with a sum-check over the field). Since the NTT transform is not a norm-preserving map, the norm of the underlying field vectors does not directly correspond to the norm of the associated ring vector in which they are embedded. Hence, checking the norm of the ring vector cannot be done by simply applying constraints over the underlying field vectors. More broadly, checking any non-trivial constraints purely over the underlying field vector do not correspond with constraints over the coefficient vectors of the ring vector itself, and vice versa. Hence, the technique from [19] is required to reduce the norm constraints to ring constraints and the isomorphic property of the NTT map is required to reduce the underlying field CCS constraints to ring constraints. Now that both the norm and CCS constraints are reduced to ring constraints, the sum-check protocol has to be executed over the ring itself. We resolve this issue by introducing the **Neo embedding**, a map which directly embeds field vectors $z^{(1)}, \dots, z^{(d)} \in \mathbb{F}^n$ along the coefficient slots of a committed ring vector $\mathbf{z} \in \mathbb{R}_{\mathbb{F}}^n$. More formally, define $\mathbf{z} := \sum_{j=1}^d z^{(j)} \cdot X^{j-1}$, where the monomial X^{j-1} scales the field vector $z^{(j)}$. For every $i \in [n]$, $\mathbf{z}_i = \text{Emb}_{\mathbb{F}}(z_i^{(1)}, \dots, z_i^{(d)})$. Define $\text{Emb}_{\mathbb{F}} : \mathbb{F}^d \rightarrow \mathbb{R}_{\mathbb{F}}$ on a coefficient vector $\mathbf{a} = (a_1, \dots, a_d) \in \mathbb{F}^d$ by $\text{Emb}_{\mathbb{F}}(\mathbf{a}) := \sum_{j=1}^d a_j X^{j-1}$. Conversely, for a ring element $\mathbf{a} = \sum_{j=1}^d a_j X^{j-1} \in \mathbb{R}_{\mathbb{F}}$, define $\text{cf}(\mathbf{a}) := (a_1, \dots, a_d) \in \mathbb{F}^d$. Naturally, we extend $\text{cf}(\cdot)$ to ring vectors by applying it coordinatewise. This gives that $\text{cf}(\mathbf{z}) \in \mathbb{F}^{d \times n}$ has rows $z^{(1)}, \dots, z^{(d)}$.

$$\text{cf}(\mathbf{z}) = \begin{bmatrix} \hline z^{(1)} \\ \hline z^{(2)} \\ \hline \vdots \\ \hline z^{(d)} \end{bmatrix} = \left[\begin{array}{c|c|c|c} z_1 & z_2 & \cdots & z_n \end{array} \right]$$

Since the coefficient vectors coincide exactly with the underlying field vectors, the Neo embedding is a norm-preserving map; checking the norm of the ring vector can be done purely with field constraints over the underlying vectors $z^{(1)}, \dots, z^{(d)} \in \mathbb{F}^n$. Moreover, the field CCS constraints can immediately be checked over the underlying field vectors themselves. As a result, the sum-check protocol can be executed purely over the field, and no ring operations are required. The sum-check protocol reduces these norm and CCS constraints down to checking random multilinear evaluations of the underlying field vectors; $y = \widetilde{M}z(r)$ for some CCS matrix $M \in \mathbb{F}^{m \times n}$ and $z \in \{z^{(1)}, \dots, z^{(d)}\}$. Explicitly, an evaluation claim has the following form: Consider a committed ring vector $\mathbf{z}$ (with commitment $c = A\mathbf{z}$), do the multilinear extensions of the underlying field vectors $z^{(1)}, \dots, z^{(d)}$ evaluate to some claimed values $y^{(1)}, \dots, y^{(d)}$ at a random extension field point r ? Since we are constructing a folding scheme, our goal is to fold these evaluation claims together into a single evaluation claim by taking a random linear combination of the commitments and of the evaluation claims.

Does the Neo embedding respect an **evaluation homomorphism** such that we can fold these evaluation claims together?

We prove that the Neo embedding does indeed exactly respect the type of evaluation homomorphism required to fold these evaluation claims together. Consider two committed ring vectors $\mathbf{z}$ and $\mathbf{z}'$ (with commitments $c = A\mathbf{z}$ and $c' = A\mathbf{z}'$) and their underlying field vectors $z^{(1)}, \dots, z^{(d)}$ and $z'^{(1)}, \dots, z'^{(d)}$ with evaluations $y^{(1)}, \dots, y^{(d)}$ and $y'^{(1)}, \dots, y'^{(d)}$ at the same random extension field point r . We use the natural lift $\text{Emb}_{\mathbb{K}} : \mathbb{K}^d \rightarrow R_{\mathbb{K}}$ of the coefficient embedding, defined by $\text{Emb}_{\mathbb{K}}(a_1, \dots, a_d) := \sum_{j=1}^d a_j X^{j-1}$. We embed the evaluations into ring elements $\mathbf{y} := \text{Emb}_{\mathbb{K}}(y^{(1)}, \dots, y^{(d)})$ and $\mathbf{y}' := \text{Emb}_{\mathbb{K}}(y'^{(1)}, \dots, y'^{(d)})$. We show that taking the linear combination $\mathbf{z}'' := \mathbf{z} + \delta \cdot \mathbf{z}'$ ($c'' = c + \delta \cdot c'$) for any $\delta \in R_{\mathbb{F}}$ results in a ring vector $\mathbf{z}''$ (c'') whose underlying field vectors $z''^{(1)}, \dots, z''^{(d)}$ evaluate to the coefficients of the linear combination $\mathbf{y}'' := \mathbf{y} + \delta \cdot \mathbf{y}'$. If the challenge δ is a ring element with low-norm [3, 28], then the resulting vector $\mathbf{z}''$ will also have low norm and the commitment c'' will be a valid commitment to $\mathbf{z}''$.

While this embedding is natural, it is non-trivial to show that it also respects an **evaluation homomorphism**, which is required to batch multilinear evaluations of the underlying field vectors together (a requirement for folding schemes based on sum-check). In fact, it is surprising that this type of embedding would respect any sort of evaluation homomorphism for multilinear evaluations over the underlying field vectors, given that the commitment scheme and linear combination are defined over the ring. Given two ring vectors and multilinear evaluations of their underlying field vectors, why does taking a random linear combination of these ring vectors over the ring result in a ring vector whose underlying field vectors evaluate to the same random linear combination of the original field evaluations? For the NTT embedding, this is trivial since the NTT is a ring isomorphism, but for the Neo embedding over coefficients, this is not at all clear. In a prior version of this work, we proved that the Neo embedding respected this evaluation homomorphism with the fact that a ring multiplication (and hence the random linear combination) is merely a linear map over the field; in particular, a ring multiplication can be simulated by multiplying the coefficient vector by the rotation matrix $\text{rot}(\delta)$ (or circulant matrix) associated with the ring element δ . Because of this linearity, the ring linear combination must respect the corresponding multilinear evaluations of the coefficients. However, we present a much simpler interpretation and proof of the evaluation homomorphism by identifying the base field $\mathbb{F}$ as merely the constant polynomials in the cyclotomic ring $R_{\mathbb{F}} := \mathbb{F}[X]/(\phi(X))$ and identifying the ring $R_{\mathbb{F}}$ itself as base field polynomials in a larger ring $R_{\mathbb{K}} := \mathbb{K}[X]/(\phi(X))$ consisting of polynomials whose coefficients belong to the extension field $\mathbb{K}$.

$$\begin{aligned} R_{\mathbb{F}} &\subseteq R_{\mathbb{K}} \\ \cup &\quad \cup \\ \mathbb{F} &\subseteq \mathbb{K} \end{aligned}$$

Hence, multilinear evaluations of the underlying field vectors and linear combinations of the committed ring vectors can all be expressed as linear maps over the same larger ring $R_{\mathbb{K}}$, and the evaluation homomorphism follows almost immediately from this linearity. This embedding and evaluation homomorphism has been used by several subsequent works⁶, such as [16, 26, 69], and referred to as a *tensor of rings* approach [26] or *ring switching* [69].

Theorem 1 (Neo Evaluation Homomorphism (Informal)). Consider an arbitrary matrix $M \in \mathbb{F}^{m \times n}$, ring vectors $\mathbf{z}_1, \dots, \mathbf{z}_{\ell} \in R_{\mathbb{F}}^n$ with underlying field vectors

$$z_i^{(j)} := \text{cf}(\mathbf{z}_i)_j \in \mathbb{F}^n \quad \text{for } i \in [\ell], j \in [d],$$

scalars $\rho_1, \dots, \rho_{\ell} \in R_{\mathbb{F}}$, and an evaluation point $r \in \mathbb{K}^{\log m}$. For all $i \in [\ell]$, define

$$c_i := \text{Commit}(A, \mathbf{z}_i) \in R_{\mathbb{F}}^{\kappa}, \quad u_i := (\widehat{M z_i^{(1)}}(r), \dots, \widehat{M z_i^{(d)}}(r)) \in \mathbb{K}^d, \quad \mathbf{y}_i := \text{Emb}_{\mathbb{K}}(u_i) \in R_{\mathbb{K}}.$$

Additionally, define

$$c := \sum_{i \in [\ell]} \rho_i c_i \in R_{\mathbb{F}}^{\kappa}, \quad \mathbf{z} := \sum_{i \in [\ell]} \rho_i \mathbf{z}_i \in R_{\mathbb{F}}^n, \quad \mathbf{y} := \sum_{i \in [\ell]} \rho_i \mathbf{y}_i \in R_{\mathbb{K}}$$

⁶ We provide a more detailed comparison with related work in the following section.

We must have that $c = \text{Commit}(A, \mathbf{z})$ and, for every $j \in [d]$, the coefficient of $\mathbf{y}$ indexed by j equals $\widetilde{Mz^{(j)}}(r)$, where $z^{(j)} := \text{cf}(\mathbf{z})_j \in \mathbb{F}^n$ is the corresponding underlying field vector of $\mathbf{z}$.

Is there an embedding that has optimal packing efficiency, a pay-per-bit commitment cost, and does not require a SIMD constraint system?

If our goal is to commit to d field vectors of length n (SIMD constraint system), then the Neo embedding has the optimal packing efficiency of $d \cdot n$ field elements for a ring vector of length n . Furthermore, since the embedded field vectors are directly placed in the coefficient slots, then the commitment costs scales with the number of bits (we explain this in more detail in the technical overview). Unfortunately, the Neo embedding requires a SIMD constraint system for both efficiency and the evaluation homomorphism to hold; the same constraint system must be applied to all d underlying field vectors.

Thus, in this work, we also introduce the **SuperNeo embedding**, which is a norm-preserving embedding for a **single** field vector of length $d \cdot n$ into a ring vector of length n (which is $d \times$ shorter!) that satisfies all the desired properties (optimal packing, pay-per-bit, not SIMD) and preserves the required evaluation homomorphism. Given a field vector $z \in \mathbb{F}^{d \cdot n}$, split the vector into n sub-vectors of length d each, i.e., $z = [z_1, z_2, \dots, z_n]$ where each $z_i := [z_{i,1}, \dots, z_{i,d}] \in \mathbb{F}^d$. We will embed each sub-vector z_i as the coefficients of a single ring element $\mathbf{z}_i := \text{Emb}_{\mathbb{F}}(z_i) = \sum_{j=1}^d z_{i,j} X^{j-1} \in R_{\mathbb{F}}$. Define the packed embedding $\text{Pack}_{\mathbb{F}} : \mathbb{F}^{d \cdot n} \rightarrow R_{\mathbb{F}}^n$ by $\text{Pack}_{\mathbb{F}}(z) := [\mathbf{z}_1, \dots, \mathbf{z}_n]$. The resulting vector of ring elements $\mathbf{z} := \text{Pack}_{\mathbb{F}}(z) \in R_{\mathbb{F}}^n$ is the SuperNeo embedding of z .

$$z = \left[\begin{array}{c|c|c|c} z_1 & z_2 & \dots & z_n \end{array} \right]$$

$$\text{cf}(\mathbf{z}) = \left[\begin{array}{c|c|c|c} z_1 & z_2 & \dots & z_n \end{array} \right]$$

The SuperNeo embedding is a norm-preserving map since the underlying field vector $z \in \mathbb{F}^{dn}$ is exactly the coefficients of the committed ring vector $\mathbf{z} \in R_{\mathbb{F}}^n$. Hence, checking both the norm and CCS constraints can be done purely with field constraints over the underlying vector $z \in \mathbb{F}^{dn}$. As such, the sum-check protocol can be executed purely over the field, resulting in evaluation claims of the form $y = \widetilde{Mz}(r)$ for some CCS matrix $M \in \mathbb{F}^{m \times dn}$. Now, it is unclear whether the SuperNeo embedding respects the required evaluation homomorphism. Unlike the Neo embedding evaluation claims, we only have single evaluations $y, y' \in \mathbb{K}$ (instead of d); how can we embed these evaluations into ring elements $\mathbf{y}, \mathbf{y}'$ such that taking a random linear combination of the committed ring vectors $\mathbf{z}'' := \mathbf{z} + \delta \cdot \mathbf{z}' \in R_{\mathbb{F}}^n$ results in a ring vector whose underlying field vector $z'' \in \mathbb{F}^{dn}$ that evaluates to the embedded evaluation in $\mathbf{y}'' := \mathbf{y} + \delta \cdot \mathbf{y}'$? As we will see, arguing the evaluation homomorphism for the Neo embedding relies on the fact that the underlying field vectors are directly placed in the coefficient slots of the ring vector, but in the SuperNeo embedding we do not have this nice parallel structure (taking a random linear combination over the ring permutes and mixes the coefficients in a non-trivial way).

However, we show that there is indeed a corresponding choice of evaluation embedding of $y \in \mathbb{K}$ into $\mathbf{y} \in R_{\mathbb{K}}$ such that the evaluation homomorphism holds. To do so, we extend a technique from the lattice literature [12, 36, 65] called the inner product trick for cyclotomic rings, explicitly defined in Theorem 8. The key idea is that, for any cyclotomic ring, there exists a linear transform $\text{Trans}_{\mathbb{F}} : \mathbb{F}^d \rightarrow \mathbb{F}^d$ such that, for all $a, b \in \mathbb{F}^d$, the constant term of a ring product recovers a coefficient inner product: $\text{ct}(\text{Emb}_{\mathbb{F}}(\text{Trans}_{\mathbb{F}}(a)) \cdot \text{Emb}_{\mathbb{F}}(b)) = \langle a, b \rangle$.

Prior works used this trick to check the norm of ring vectors by simulating field inner products with ring multiplication, for example, through random projections [12] or products [65]. We extend the trick to vectors and matrices. In particular, for a CCS matrix $M \in \mathbb{F}^{m \times dn}$, define $\mathbf{M} := \text{Emb}_{\mathbb{F}}(\text{Trans}_{\mathbb{F}}(M)) \in R_{\mathbb{F}}^{m \times n}$. For $\mathbf{z} := \text{Pack}_{\mathbb{F}}(z)$, the constant terms of $\mathbf{M}\mathbf{z}$ recover Mz . Given $r \in \mathbb{K}^{\log m}$, the natural inclusions $\mathbb{F} \subseteq \mathbb{K}$ and $R_{\mathbb{F}} \subseteq R_{\mathbb{K}}$ allow us to evaluate $\mathbf{M}\mathbf{z} \in R_{\mathbb{F}}^m$ over $R_{\mathbb{K}}$. The desired evaluation $y = \widetilde{Mz}(r) \in \mathbb{K}$ is then the constant term of $\mathbf{y} := \widetilde{\mathbf{M}\mathbf{z}}(r) \in R_{\mathbb{K}}$. The evaluation homomorphism follows from linearity, similarly to the Neo embedding.

Theorem 2 (SuperNeo Evaluation Homomorphism (Informal)). *Consider an arbitrary matrix $M \in \mathbb{F}^{m \times dn}$, vectors $z_1, \dots, z_\ell \in \mathbb{F}^{dn}$, scalars $\rho_1, \dots, \rho_\ell \in \mathbb{R}_\mathbb{F}$, and an evaluation point $r \in \mathbb{K}^{\log m}$. Define $\mathbf{M} := \text{Emb}_\mathbb{F}(\text{Trans}_\mathbb{F}(M)) \in \mathbb{R}_\mathbb{F}^{m \times n}$. For every $i \in [\ell]$, define*

$$\mathbf{z}_i := \text{Pack}_\mathbb{F}(z_i) \in \mathbb{R}_\mathbb{F}^n, \quad c_i := \text{Commit}(A, \mathbf{z}_i) \in \mathbb{R}_\mathbb{F}^\kappa, \quad \mathbf{y}_i := \widetilde{\mathbf{M}\mathbf{z}_i}(r) \in \mathbb{R}_\mathbb{K}.$$

Additionally, define

$$c := \sum_{i \in [\ell]} \rho_i c_i \in \mathbb{R}_\mathbb{F}^\kappa, \quad \mathbf{z} := \sum_{i \in [\ell]} \rho_i \mathbf{z}_i \in \mathbb{R}_\mathbb{F}^n, \quad \mathbf{y} := \sum_{i \in [\ell]} \rho_i \mathbf{y}_i \in \mathbb{R}_\mathbb{K}$$

We must have that $c = \text{Commit}(A, \mathbf{z})$ and $\mathbf{y} = \widetilde{\mathbf{M}\mathbf{z}}(r)$. Moreover, for every $i \in [\ell]$, $\text{ct}(\mathbf{y}_i) = \widetilde{\mathbf{M}\mathbf{z}_i}(r)$. And, $\text{ct}(\mathbf{y}) = \widetilde{\mathbf{M}\mathbf{z}}(r)$, where $\mathbf{z} \in \mathbb{F}^{dn}$ is the unique field vector satisfying $\mathbf{z} = \text{Pack}_\mathbb{F}(z)$. The formal theorem statement appears in Theorem 11.

In Section 5, we generalize the evaluation homomorphism in Theorem 2 from the base field $\mathbb{F}$ to extension fields of arbitrary degree. Specifically, for a degree δ extension $\mathbb{B}$ of $\mathbb{F}$ and vector length n satisfying $d \mid \delta n$, the coefficient embedding, packed embedding, and transform above extend naturally to $\text{Emb}_\mathbb{B}$, $\text{Pack}_\mathbb{B} : \mathbb{B}^n \rightarrow \mathbb{R}_\mathbb{F}^{\delta n/d}$, and $\text{Trans}_\mathbb{B}$. For $M \in \mathbb{B}^{m \times n}$, $z \in \mathbb{B}^n$, and $r \in \mathbb{K}^{\log m}$, let $\mathbb{A}$ be a common extension field containing $\mathbb{B}$ and $\mathbb{K}$. The natural inclusions $\mathbb{R}_\mathbb{F} \subseteq \mathbb{R}_\mathbb{B} \subseteq \mathbb{R}_\mathbb{A}$ and $\mathbb{R}_\mathbb{K} \subseteq \mathbb{R}_\mathbb{A}$ make the construction well defined, and these maps satisfy $\text{ct}(\widetilde{\text{Emb}_\mathbb{B}(\text{Trans}_\mathbb{B}(M))\text{Pack}_\mathbb{B}(z)}(r)) = \widetilde{\mathbf{M}\mathbf{z}}(r) \in \mathbb{A}$. The generalized evaluation homomorphism is formalized in Theorem 11. We instantiate $\mathbb{B} = \mathbb{L} := \mathbb{F}_{q^r}$ and $n = n_L$ to support CCS relations over $\mathbb{L}$. Concretely, a vector $z \in \mathbb{L}^{n_L}$ represents $\tau n_L = n_F$ base field elements, whereas each element of $\mathbb{R}_\mathbb{F}$ has d base field coefficients. Thus, $\text{Pack}_\mathbb{L}$ maps z to a ring vector $\mathbf{z} \in \mathbb{R}_\mathbb{F}^{n_R}$ with $\tau n_L/d = n_R$, achieving optimal packing efficiency.

Contribution 2: Strong and Weak Interactive Reductions

Is there a natural meta-framework which captures common lattice-protocols Π_{property} and Π_{special} ⁷ which independently are not secure, but prove their composition $\Pi := \Pi_{\text{special}} \circ \Pi_{\text{property}}$ is secure?

We introduce the notion of *interactive reductions*, which are a generalization of reductions of knowledge introduced in [52]. In particular, an interactive reduction shares the exact same API (structure) as a reduction of knowledge, but is not required to be knowledge sound. In this way, we can view a reduction of knowledge merely as a knowledge-sound interactive reduction. We introduce two types of interactive reductions, which we call *strong* Π_{strong} and *weak* Π_{weak} interactive reductions, and prove a new composition theorem that the composition $\Pi := \Pi_{\text{weak}} \circ \Pi_{\text{strong}}$ of a strong interactive reduction with a weak interactive reduction results in a reduction of knowledge Π . These *strong* and *weak* properties are quite natural and easy to show for many protocols in the lattice-based proof system literature [12, 15, 33, 43, 65], such as sum-check and random projections for the strong property, and special-sound protocols for the weak property. As such, this composition theorem provides a powerful tool to prove the security of composed protocols Π in a simple and modular manner. To our knowledge, this is the first work to show, quite unintuitively, that insecure protocols can be composed together to yield an overall secure protocol. As we will see soon, these properties are not restricted to lattice-based protocols, but are quite general and capture protocols in group-based proof systems [27, 56, 57] and may capture recent work in the hash-based proof systems [6, 21, 22, 84].

On the topic of security proofs, we also extend the analysis of lattice-based folding schemes to more general cyclotomic polynomials (rather than just those of the form $\phi(X) = X^d + 1$ for d a power of two); this enables a much wider choice of fields, as we will explain later. Namely, we are able to support the use of the Goldilocks field [75], a field of order $p = 2^{64} - 2^{32} + 1$ and very efficient arithmetic. Because Goldilocks has high 2-adicity ($2^{32} \mid (p - 1)$), it is particularly SNARK-friendly; a SNARK instantiated over this field can succinctly prove knowledge of lattice-based accumulators and thus enable efficient verification for SuperNeo.

⁷ Recall: Π_{property} is a sound testing protocol (e.g., sum-check) that produces algebraic claims on committed witnesses, and Π_{special} is a special-sound protocol that takes a random linear combination of the commitments and claims.

This mirrors MicroNova’s use of HyperKZG to compress its accumulators [85]. The only requirements on the cyclotomic polynomial are that d and q are large enough for M-SIS hardness and that the polynomial admits a sufficiently large challenge set; see Section 8 for concrete parameter choices.

1.3 Related works

Lattice-based folding schemes are a recent area of research. We compare with folding schemes [15, 16, 38, 60], a lattice-based SNARK [26], and a lattice-based polynomial commitment scheme [69].

Lattice-based folding schemes. Lova [38] is the only lattice-based folding scheme based solely on the unstructured Short-Integer-Solution (SIS) problem rather than Module-SIS, making it arguably the safest in terms of assumptions, but at the cost of efficiency (several orders of magnitude slower) and generality (it targets subset-sum constraints rather than R1CS/CCS). LatticeFold+ [16, 17], the followup to LatticeFold [15], introduces an algebraic range-proof technique that encodes lookup tables into ring elements and checks norm constraints via index proofs and sum-checks, enabling significantly higher norm bounds than Hadamard-product-based approaches [19]. Despite claiming small-field support, their protocols and parameterization are ring protocols over a large prime field; the authors acknowledge [17] that their technique for field-native sum-check and the evaluation homomorphism for folding are adapted from Neo. LatticeFold+ re-interprets our technique as a tensor-of-rings approach; we provide a simpler interpretation by identifying the various fields and rings as subrings of $R_{\mathbb{K}} := \mathbb{K}[X]/(\phi(X))$. SALSAA [60] is the first lattice-based folding scheme that natively relies on ℓ_2 -norm constraints rather than ℓ_∞ -norm constraints. SALSAA improves and extends the lattice-based proof system framework from RoK, paper, SISsors [49] and RoK and Roll [50], achieving linear prover runtime (down from quasilinear) via ring sum-check techniques. However, unlike other lattice-based folding schemes, SALSAA relies on a relatively new assumption called the *vanishing Short-Integer-Solution* (vSIS) assumption [32, 46].

The folding scheme Cyclo [42] is a reinterpretation of the original Neo folding scheme (not SuperNeo). As noted by the authors, *Our methods are ultimately equivalent to those of Neo* (Sec 7, pg 24). Cyclo’s contributions are to (1) provide an alternative presentation of Neo, (2) observe that if soundness is only required for an *explicit bounded number* of steps, then the norm check on the accumulator (or running instance) can be removed for efficiency, and (3) provide an initial benchmark. Unlike Neo or Cyclo, SuperNeo’s embedding has optimal packing efficiency. Additionally, SuperNeo is compatible with Cyclo’s contribution (2).

Cyclo uses the Neo embedding but restricts the logical witness to a single vector $z := \sum_{i=1}^d b^{i-1} \cdot z^{(i)} \in \mathbb{F}^n$, where the $z^{(i)} \in \mathbb{F}^n$ are base- b decomposition vectors ($\|z^{(i)}\|_\infty < b$) committed via the SIMD/coefficient slots (rather than using the $z^{(1)}, \dots, z^{(d)}$ independently). This is the same approach taken in the original Neo paper; evaluations can be recovered as $\tilde{z}(r) = \sum_{i=1}^d b^{i-1} \cdot \tilde{z}^{(i)}(r)$. Packing is optimal only when the base- b decomposition exactly fills the d coefficient slots; a mismatch wastes slots, which does not occur in SuperNeo.

To be concrete, consider $b = 2$, the Goldilocks field ($\log_2 |\mathbb{F}| = 64$), and $\phi(X) = X^{54} + X^{27} + 1$ ($d = 54$). A full field vector requires 64 decomposition slots but only 54 are available, so a second ring vector must absorb the remaining 10 slots (wasting 44 slots). A power-of-two cyclotomic with $d = 64$ would match the bit-width but yields a trivial challenge set for this field. In practice, witnesses are often low-norm (< 32 bits), so a single commitment suffices, but $54 - 32 = 22$ slots are still wasted; both d and q must be large enough for M-SIS hardness.

More broadly, to our knowledge, no prior lattice based folding scheme [15, 16, 38, 42, 60] natively folds relations over an arbitrary extension field $\mathbb{L}$ of the commitment field $\mathbb{F}$ without tying $\mathbb{L}$ to the NTT splitting of the cyclotomic ring. SuperNeo instead uses coefficient packing and natural inclusions into a common ambient field $\mathbb{A}$, so its embedding and evaluation homomorphism require no NTT transform.

Lattice-based SNARKs. Symphony [26] is a lattice-based SNARK that uses high-arity folding to prove repetitive NP claims without heuristically instantiating the random oracle (as is typical of IVC/PCD-based recursion). Symphony also relies on the tensor-of-rings approach, which as noted in LatticeFold+ [17] is a re-interpretation of Neo’s techniques. Additionally, Symphony relies on space-efficient sum-check, which imposes asymptotic and concrete overheads compared to the sum-check in Neo and SuperNeo.

Lattice-based polynomial commitments. Hachi [69] is the first lattice-based polynomial commitment scheme supporting extension-field evaluation of field polynomials, adapting Greyhound [70] by replacing random projections with a sum-check over the extension field for norm constraints and providing a technique to reduce extension-field evaluation proofs to ring statements via Greyhound/Labrador [12, 70]. Neo (prior to Hachi) is the first work to check norms of committed ring vectors via a sum-check purely over the extension field; despite the use of circulant matrices in the evaluation-homomorphism proof, the sum-check reduction itself requires no ring operations. SuperNeo (concurrent with Hachi) uses a related but distinct application of the inner product trick: in Hachi it proves multilinear evaluations for a polynomial commitment, while in SuperNeo it folds CCS matrix evaluations for a folding scheme. Both achieve the same packing efficiency: committing to a vector of length $d \cdot n$ requires only a ring vector of length n .

2 Technical overview

We recall HyperNova and the challenges of moving to lattice-based commitments (§2.1), then present the Neo (§2.2) and SuperNeo (§2.3) embeddings, and finally our interactive reductions framework (§2.4).

2.1 Breaking down HyperNova

HyperNova [56] *reduces* CCS claims into random evaluation claims. These claims are instances in a *CCS relation*, CCS, and a *CCS evaluation relation*, CE. For this discussion, let $\mathbf{s} := (\{M_j \in \mathbb{F}^{m \times n}\}_{j \in [t]}, f \in \mathbb{F}[X_1, \dots, X_t])$ be the CCS structure and let $\text{Commit} : \mathbb{F}^n \rightarrow \mathbb{G}$ be a linearly homomorphic commitment map. For a public input x and witness w , set $z := [x, w] \in \mathbb{F}^n$. The evaluation point r and claimed evaluations y_j belong to $\mathbb{F}$.

$$\begin{aligned} \text{CCS} &:= \left\{ (\mathbf{s}; (c, x); w) : \begin{array}{l} c = \text{Commit}(z) \wedge \\ f(\widetilde{M_1 z}, \dots, \widetilde{M_t z}) \text{ vanishes on } \{0, 1\}^{\log m} \end{array} \right\} \\ \text{CE} &:= \left\{ (\mathbf{s}; (c, x, r, \{y_j\}_{j \in [t]}); w) : \begin{array}{l} c = \text{Commit}(z) \wedge \\ \forall j \in [t], y_j = \widetilde{M_j z}(r) \end{array} \right\} \end{aligned}$$

To be considered a folding scheme, we must be able to describe HyperNova as an interactive reduction $\Pi : \text{CCS} \times \text{CE} \rightarrow \text{CE}$, which takes as input a CCS relation claim and a CCS evaluation relation claim and outputs a single CCS evaluation relation claim.⁸ Unlike the original HyperNova paper (for reasons that will become clear later), we decide to break up HyperNova into two interactive reductions which will be composed to form the overall reduction:

$$(\text{Informal}) \quad \Pi_{\text{CCS}} : \text{CCS} \times \text{CE} \rightarrow \text{CE}^2 \quad \text{and} \quad \Pi_{\text{RLC}} : \text{CE}^2 \rightarrow \text{CE}.$$

The first reduction Π_{CCS} utilizes the sum-check protocol [63, 79] to reduce a CCS claim and a CCS evaluation claim over a point r' into a pair of CCS evaluation claims over the same point r .

$$\begin{aligned} (\mathbf{s}; (c, x); w) \in \text{CCS} \quad \wedge \quad (\mathbf{s}; (c', x', r'; \{y'_j\}_{j \in [t]}); w') \in \text{CE} \\ \downarrow \Pi_{\text{CCS}} \\ (\mathbf{s}; (c, x, r, \{y_j\}_{j \in [t]}); w) \in \text{CE} \quad \wedge \quad (\mathbf{s}; (c', x', r, \{y'_j\}_{j \in [t]}); w') \in \text{CE} \end{aligned}$$

The second reduction Π_{RLC} combines two CCS evaluation claims over the same point r into a single CCS evaluation claim over point r . In particular, the reduction Π_{RLC} is trivial if $\text{Commit} : \mathbb{F}^n \rightarrow \mathbb{G}$ is itself a linear map from field vectors to a group $\mathbb{G}$ (i.e. linearly homomorphic). To combine both of these claims, the verifier just samples a random challenge $\delta \in \mathbb{F}$ and checks the following combined claim: $(\mathbf{s}; (c^*, x^*, r, \{y_j^*\}_{j \in [t]}); w^*) \in \text{CE}$, where

$$c^* := c + \delta c', \quad x^* := x + \delta x', \quad \forall j \in [t], y_j^* := y_j + \delta y'_j, \quad w^* := w + \delta w'.$$

⁸ In the technical overview, we consider only folding single instances of each relation for simplicity, but our actual protocol works for folding multiple instances at once.

Because Commit is a linear map over field vectors, we have that the new commitment $c^* = \text{Commit}(z^*)$ for $z^* := [x^*, w^*]$. Similarly, for each $j \in [t]$, we have that $z \mapsto \widetilde{M_j z}(r)$ is also a linear map over field vectors, so $y_j^* = \widetilde{M_j z^*}(r)$. Thus, the combined claim is indeed a valid CCS evaluation claim as desired.

In order to construct the Neo and SuperNeo folding scheme, we will need to construct a commitment scheme for field vectors (using Ajtai commitments) that has a similar *evaluation homomorphism* property, which allows us to combine CCS evaluation claims by taking random linear combinations over the ring rather than the field. In doing so, we have the essential components to adapt the HyperNova folding scheme to the lattice setting. In addition, we will have to add norm-checks into the sum-check protocol and add a decomposition reduction [12, 15, 72] to reduce norm growth. Later, we also discuss the technical difficulties when trying to use prior extraction techniques to prove the security of our reductions in the lattice setting, which is not apparent in the original HyperNova setting.

2.2 The Neo embedding

The Neo embedding is incredibly simple to describe, and what is surprising is that it also preserves both norm and the required form of evaluation homomorphism. To embed field vectors $z^{(1)}, \dots, z^{(d)} \in \mathbb{F}^n$ into a ring vector $\mathbf{z} \in \mathbb{R}_{\mathbb{F}}^n$, the Neo embedding simply embeds the field vectors along the d coefficient slots.

$$\mathbf{z} = \sum_{j=1}^d z^{(j)} \cdot X^{j-1} \iff \forall i \in [n], z_i = \text{Emb}_{\mathbb{F}}(z_i^{(1)}, \dots, z_i^{(d)})$$

Here $\text{Emb}_{\mathbb{F}} : \mathbb{F}^d \rightarrow \mathbb{R}_{\mathbb{F}}$ is the coefficient embedding defined by $\text{Emb}_{\mathbb{F}}(a_1, \dots, a_d) := \sum_{j=1}^d a_j X^{j-1}$. In the expression above, $z^{(j)} \cdot X^{j-1}$ denotes scaling the field vector $z^{(j)}$ by the monomial X^{j-1} , so every element in the scaled vector has the form $c \cdot X^{j-1}$ for some constant $c \in \mathbb{F}$.

Preserving evaluation homomorphism The first observation is that any collection of elements in the field $\mathbb{F}$ can be naturally embedded into the ring $\mathbb{R}_{\mathbb{F}} \subseteq \mathbb{R}_{\mathbb{K}}$ by interpreting them as constant polynomials. The second observation is that $\mathbb{K}$ also contains an isomorphic copy of the base field $\mathbb{F}$. Thus, a polynomial with coefficients in $\mathbb{F}$ can also be interpreted as a polynomial with coefficients in $\mathbb{K}$. Therefore, CCS matrices $M_j \in \mathbb{F}^{m \times n}$ and an evaluation point $r \in \mathbb{K}^{\log m}$ can both be interpreted over $\mathbb{R}_{\mathbb{K}}$. What happens when we take the multilinear extension of $\mathbf{z}$ and evaluate it at r over the ring? Multilinear evaluation is equivalent to taking an inner product $\widetilde{\mathbf{z}}(r) = \langle \mathbf{z}, \hat{r} \rangle$, where $\hat{r} \in \mathbb{K}^n$ is derived from r . Moreover, multiplying a ring element $a(X) = \sum_{i=0}^{d-1} a_i X^i \in \mathbb{R}_{\mathbb{F}}$ by a constant $c \in \mathbb{K}$ simply scales each coefficient, giving $\sum_{i=0}^{d-1} (ca_i) X^i \in \mathbb{R}_{\mathbb{K}}$. Combining these observations, evaluating $\mathbf{z}$ at r is equivalent to evaluating each underlying field vector at r and stacking the results as coefficients in a ring element.

$$\mathbf{y} := \widetilde{\mathbf{z}}(r) = \langle \mathbf{z}, \hat{r} \rangle = \left\langle \sum_{j=1}^d z^{(j)} X^{j-1}, \hat{r} \right\rangle = \sum_{j=1}^d \widetilde{z^{(j)}}(r) X^{j-1} \in \mathbb{R}_{\mathbb{K}}$$

Moreover, this property extends to matrix-vector multiplications. For $r \in \mathbb{K}^{\log m}$, the ring evaluation $\mathbf{y}_j := \widetilde{M_j \mathbf{z}}(r) \in \mathbb{R}_{\mathbb{K}}$ has coefficients $\widetilde{M_j z^{(1)}}(r), \dots, \widetilde{M_j z^{(d)}}(r)$.

We can now see that the evaluation homomorphism trivially holds for the Neo embedding. Consider two Neo embedded ring vectors $\mathbf{z}, \mathbf{z}' \in \mathbb{R}_{\mathbb{F}}^n$ with commitments $c = A\mathbf{z}$ and $c' = A\mathbf{z}'$. Also consider their evaluations $\mathbf{y} := \widetilde{M_j \mathbf{z}}(r)$ and $\mathbf{y}' := \widetilde{M_j \mathbf{z}'}(r)$ in $\mathbb{R}_{\mathbb{K}}$ at the same point $r \in \mathbb{K}^{\log m}$. For an arbitrary ring scalar $\delta \in \mathbb{R}_{\mathbb{F}}$, define $\mathbf{z}^* := \mathbf{z} + \delta \mathbf{z}' \in \mathbb{R}_{\mathbb{F}}^n$, $c^* := c + \delta c'$, and $\mathbf{y}^* := \mathbf{y} + \delta \mathbf{y}'$. The underlying field vectors of $\mathbf{z}^*$ must evaluate to the coefficients of $\mathbf{y}^*$ at r . This follows because multilinear evaluation is linear over $\mathbb{R}_{\mathbb{K}}$, so $\mathbf{y}^* = \widetilde{M_j \mathbf{z}^*}(r)$, whose coefficients are exactly the evaluations of the underlying field vectors of $\mathbf{z}^*$. The vector $\mathbf{z}^*$ remains in $\mathbb{R}_{\mathbb{F}}^n$, and its underlying field vectors remain over $\mathbb{F}$. Thus, as long as $\mathbf{z}^*$ has sufficiently small norm, which holds when δ is sampled from a strong sampling set $\mathcal{C}$, the commitment c^* is a valid commitment to $\mathbf{z}^*$ under the Ajtai commitment scheme.

2.3 The SuperNeo embedding

The SuperNeo embedding Given a field vector $z \in \mathbb{F}^{d \cdot n}$, split the vector into n subvectors of length d each, i.e., $z = [z_1, z_2, \dots, z_n]$ where each $z_i := [z_{i,1}, \dots, z_{i,d}] \in \mathbb{F}^d$. We will embed each subvector z_i as the

coefficients of a single ring element $z_i := \text{Emb}_{\mathbb{F}}(z_i) = \sum_{j=1}^d z_{i,j} X^{j-1} \in R_{\mathbb{F}}$. Define the packed embedding $\text{Pack}_{\mathbb{F}} : \mathbb{F}^{d \cdot n} \rightarrow R_{\mathbb{F}}^n$ by $\text{Pack}_{\mathbb{F}}(z) := [z_1, \dots, z_n]$. The resulting vector $\mathbf{z} := \text{Pack}_{\mathbb{F}}(z) \in R_{\mathbb{F}}^n$ is the SuperNeo embedding of z .

Evaluation Homomorphism For cyclotomic rings, there exists a linear transform $\text{Trans}_{\mathbb{F}} : \mathbb{F}^d \rightarrow \mathbb{F}^d$ such that for all $a, b \in \mathbb{F}^d$, we have the constant term $\text{ct}(\text{Emb}_{\mathbb{F}}(\text{Trans}_{\mathbb{F}}(a)) \cdot \text{Emb}_{\mathbb{F}}(b)) = \langle a, b \rangle$, where $\text{Emb}_{\mathbb{F}}$ is the coefficient embedding defined above. In layman's terms, a product over the ring simulates an inner product over the field. We extend this transform blockwise to vectors and rowwise to matrices. For a vector $u \in \mathbb{F}^{d \cdot n}$, applying $\text{Trans}_{\mathbb{F}}$ to each subvector of length d gives $\text{ct}(\langle \text{Emb}_{\mathbb{F}}(\text{Trans}_{\mathbb{F}}(u)), \mathbf{z} \rangle) = \langle u, z \rangle$. For a matrix $M_j \in \mathbb{F}^{m \times d \cdot n}$, applying $\text{Trans}_{\mathbb{F}}$ to each row gives $\mathbf{M}_j := \text{Emb}_{\mathbb{F}}(\text{Trans}_{\mathbb{F}}(M_j)) \in R_{\mathbb{F}}^{m \times n}$. Thus, the constant term vector satisfies $\text{ct}(\mathbf{M}_j \mathbf{z}) = M_j z$, where we extract the constant terms of each of the m ring elements in $\mathbf{M}_j \mathbf{z} \in R_{\mathbb{F}}^m$.

Given a point $r \in \mathbb{K}^{\log m}$, the natural inclusions $\mathbb{F} \subseteq \mathbb{K}$ and $R_{\mathbb{F}} \subseteq R_{\mathbb{K}}$ allow us to evaluate $\mathbf{M}_j \mathbf{z} \in R_{\mathbb{F}}^m$ over $R_{\mathbb{K}}$. Define the ring evaluation $\mathbf{y}_j := \widetilde{\mathbf{M}_j \mathbf{z}}(r) \in R_{\mathbb{K}}$. As argued before, evaluating at r evaluates each of the underlying coefficient vectors of $\mathbf{M}_j \mathbf{z}$ in parallel. Hence, the constant coefficient of $\mathbf{y}_j$ is exactly the desired field evaluation $y_j := \widetilde{M_j z}(r)$. Moreover, since $z \mapsto \widetilde{\mathbf{M}_j \mathbf{z}}(r)$ is a linear map over $R_{\mathbb{K}}$, similar to the case in Neo, the evaluation homomorphism property is identically preserved. In particular, given field vectors $z, z' \in \mathbb{F}^{d \cdot n}$, define their embeddings $\mathbf{z} := \text{Pack}_{\mathbb{F}}(z)$ and $\mathbf{z}' := \text{Pack}_{\mathbb{F}}(z')$, and their ring evaluations $\mathbf{y}_j := \widetilde{\mathbf{M}_j \mathbf{z}}(r)$ and $\mathbf{y}'_j := \widetilde{\mathbf{M}_j \mathbf{z}'}(r)$. For an arbitrary scalar $\delta \in R_{\mathbb{F}}$, let $\mathbf{z}^* := \mathbf{z} + \delta \mathbf{z}'$ and $\mathbf{y}^*_j := \mathbf{y}_j + \delta \mathbf{y}'_j$. Then, the constant coefficient of $\mathbf{y}^*_j$ is exactly the field evaluation $\widetilde{M_j z^*}(r)$, where $z^* := \text{Pack}_{\mathbb{F}}^{-1}(\mathbf{z}^*)$ is the underlying field vector of $\mathbf{z}^*$.

For simplicity, this technical overview discusses CCS relations over the base field $\mathbb{F}$. In Section 5, we further generalize the inner product trick, packed embedding, and transform to directly support folding CCS relations over arbitrary extension fields of $\mathbb{F}$. For a degree δ extension $\mathbb{B}$ and vector length n satisfying $d \mid \delta n$, Definitions 13, 14, and 15 define the compatible maps

$$\text{Pack}_{\mathbb{B}} : \mathbb{B}^n \rightarrow R_{\mathbb{F}}^{\delta n/d}, \quad \text{Trans}_{\mathbb{B}} : \mathbb{B}^{m \times n} \rightarrow \mathbb{B}^{m \times \delta n}, \quad \text{Emb}_{\mathbb{B}} : \mathbb{B}^{m \times \delta n} \rightarrow R_{\mathbb{B}}^{m \times \delta n/d}.$$

The packed embedding places the δn base field coordinates of $z \in \mathbb{B}^n$ into $\delta n/d$ ring elements, retaining optimal packing efficiency. For $M \in \mathbb{B}^{m \times n}$, Theorem 9 gives

$$\text{ct}(\text{Emb}_{\mathbb{B}}(\text{Trans}_{\mathbb{B}}(M))\text{Pack}_{\mathbb{B}}(z)) = Mz.$$

For $r \in \mathbb{K}^{\log m}$ and a common extension field $\mathbb{A}$ containing $\mathbb{B}$ and $\mathbb{K}$, the natural inclusions $R_{\mathbb{F}} \subseteq R_{\mathbb{B}} \subseteq R_{\mathbb{A}}$ and $R_{\mathbb{K}} \subseteq R_{\mathbb{A}}$ make the evaluation well defined. Theorem 10 gives

$$\text{ct}(\widetilde{\text{Emb}_{\mathbb{B}}(\text{Trans}_{\mathbb{B}}(M))\text{Pack}_{\mathbb{B}}(z)}(r)) = \widetilde{Mz}(r) \in \mathbb{A}.$$

Finally, Theorem 11 shows that folding the packed vectors and their evaluation claims with scalars in $R_{\mathbb{F}}$ preserves these identities.

Pay-per-bit costs Neo and SuperNeo are both norm preserving embeddings. Computing the Ajtai commitment $c \leftarrow Az \in R_{\mathbb{F}}^{\kappa}$, for $A \in R_{\mathbb{F}}^{\kappa \times n}$ and $\mathbf{z} \in R_{\mathbb{F}}^n$, requires a ring matrix-vector product. For $a \in R_{\mathbb{F}}$, the rotation matrix $\text{rot}(a) \in \mathbb{F}^{d \times d}$ is a matrix which simulates ring multiplication. Its i th column is $\text{rot}(a)_i := \text{cf}(aX^{i-1} \bmod \phi(X)) \in \mathbb{F}^d$, so its columns are essentially rotations of the coefficients of a . When both the degree $d \leq 64$ and field bit size $\lceil \log_2 |\mathbb{F}| \rceil \leq 64$ are small, a ring operation $a \cdot b \in R_{\mathbb{F}}$ is most efficiently implemented (using AVX-512) with a rotation matrix product $\text{cf}(a \cdot b) = \text{rot}(a) \cdot \text{cf}(b) = \sum_{i=1}^d b_i \cdot \text{rot}(a)_i$ over the field $\mathbb{F}$, where b_i denotes the i th coefficient of b . The dominating costs are scaling $b_i \cdot \text{rot}(a)_i \in \mathbb{F}^d$, which scales linearly with the norm of b_i . When the b_i 's are small (such as bits), the cost to compute the ring operation is essentially adding the rotations $\text{rot}(a)_i \in \mathbb{F}^d$ for which b_i is non-zero. For security reasons [15], the cyclotomic ring cannot be perfectly splitting (i.e., $R_{\mathbb{F}} \not\cong \mathbb{F}^d$), so using the NTT transform is a more inefficient method to compute ring operations than simply adding scaled rotations. In particular, the NTT transform (in the non-full splitting setting) is more memory intensive, is less cache friendly, and computing high degree

extension field multiplications (e.g., $16 \mathbb{F}_{q^4}$ multiplications, where degree 4 is needed for a non-trivial low-norm invertibility bound) is concretely more expensive than computing the corresponding base field scaling and additions.

2.4 Proving the security with interactive reductions

To understand why interactive reductions are necessary, we first need to understand why the original proof of security for the HyperNova folding scheme [59] does not carry over to SuperNeo. In particular, when we move to using lattice-based commitments (such as Ajtai commitments), we will have to extract candidate openings z that not only satisfy the linear relationship $c = Az$, but are also sufficiently low norm (as Ajtai commitments are only binding for low norm openings). However, in the process of solving for these candidate openings, the extractor will end up obtaining vectors that potentially may have arbitrary norm. This is often the case with lattice-based proof systems, which often must combine some information-theoretic sound method to check the norm of committed vectors (such as random projections or sum-check) with a method to batch prove knowledge of linear openings (special-sound protocols). Recall (unlike in the original work) that we broke the HyperNova folding scheme down into two reductions: $\Pi_{\text{CCS}} : \text{CCS} \times \text{CE} \rightarrow \text{CE}^2$ and $\Pi_{\text{RLC}} : \text{CE}^2 \rightarrow \text{CE}$. The first reduction Π_{CCS} reduced the CCS and CCS evaluation claim into two CCS evaluations claims by using the sum-check protocol and the second reduction Π_{RLC} combined the two CCS evaluation claims into a single CCS evaluation claim by taking a random linear combination. The first issue arises when we try to prove Π_{RLC} is knowledge sound in the lattice setting. If we follow the same extraction strategy in HyperNova, we would produce two candidate commitment openings z_1 and z'_1 by solving the linear system $z_1^* = z_1 + \delta_1 z'_1$ and $z_2^* = z_1 + \delta_2 z'_1$ for two different random challenges δ_1 and δ_2 . In particular, this will require scaling by the inverse of the difference of the two challenges $(\delta_1 - \delta_2)^{-1}$. However, in the ring setting, even if δ_1 and δ_2 are low norm, the inverse $\Delta := (\delta_1 - \delta_2)^{-1}$ may have arbitrarily high norm, which would lead to candidate openings z_1 and z'_1 that also have arbitrarily high norm. Thus, we cannot guarantee that the extracted openings are valid openings for the commitments, but that they do indeed satisfy the linear relations $c = Az$ and $z \mapsto \widetilde{M}z(r)$. This means that Π_{RLC} is not knowledge sound for the input relation, but instead a relaxed relation where we drop the low norm requirement on the openings. However, there is something special about these candidate openings. In particular, in the process of extracting these openings, we produced what are referred to as **relaxed** openings $\Delta \cdot c = Az$ for the original commitments where z is a low norm ring vector and Δ is low norm ring element. Under the hardness of MSIS, producing multiple relaxed openings for the same commitment is hard. Thus, even if we run the extractor multiple times, it will ultimately only be able to output a single pair of candidate openings (z_1, z'_1) ; otherwise, it would be able to produce multiple relaxed openings for the same commitment. In summary, while Π_{RLC} by itself is not knowledge sound for the original relation, we can construct an extractor which extracts candidate witnesses for a relaxed relation, and is restricted to only being able to output a unique candidate witness (with high probability). Informally, we call an interactive reduction that satisfies these properties as a **weak interactive reduction**.

The second issue arises when we try to prove that Π_{CCS} is knowledge sound. In a folklore security argument (for sum-check protocols), the extractor for Π_{CCS} essentially rewinds the potentially malicious prover multiple times to obtain multiple candidate openings (the output witnesses) for the same commitment. These candidate openings must also satisfy the evaluations outputted by the sum-check protocol (assuming the prover is successful in both executions). However, since the commitment is binding, all these candidate openings must be the same. Hence, we are able to argue that the candidate opening (witness) from the first execution must have evaluated to the correct evaluations required by the sum-check protocol in the second execution. Observe, however, that the second execution used verifier challenges that were independent of the first execution. Hence, with high probability, the extracted candidate opening must satisfy the required CCS claims. This argument crucially relies on the fact that the commitment scheme is binding to the candidate openings outputted by the prover. This is trivial in the original HyperNova setting since the commitments are binding to the whole message space. However, in the lattice setting, the commitments are only binding to low norm openings.

To see where this goes wrong, consider that we need to prove that the overall composition $\Pi := \Pi_{\text{RLC}} \circ \Pi_{\text{CCS}}$ is knowledge sound. Π_{RLC} is only a weak interactive reduction. Hence, it can only extract candidate openings for a relaxed relation, where the low norm requirement is dropped, but all linear relations are still satisfied.

Thus, these candidate openings may not be bound by the original input commitments. However, in the case of *weak* interactive reductions, we know that the extractor can only output a unique candidate opening (with high probability). Hence, all candidate openings extracted from multiple executions of the protocol must be the same. Therefore, by the same logic above, the candidate openings must satisfy the required CCS claims with high probability. Additionally, if we include sum-check constraints which ensure that the extracted openings are low norm, then we can ensure that the extracted openings are indeed valid openings for the original commitments. Generally, what do we require from the first interactive reduction Π_{CCS} to ensure that the overall composition is knowledge sound? Essentially, if the malicious prover must always output the same output *relaxed* witness (with high probability), then there exist an extractor which can extract a valid witness for the original relation. We call an interactive reduction that satisfies this property a **strong interactive reduction**.

In our work, we use this framework to prove the security of our SuperNeo folding scheme $\Pi_{\text{SuperNeo}} := \Pi_{\text{DEC}} \circ \Pi_{\text{RLC}} \circ \Pi_{\text{CCS}}$ by decomposing it into a strong interactive reduction Π_{CCS} , a weak interactive reduction Π_{RLC} , and a final reduction of knowledge Π_{DEC} [12, 15, 72].

Lemma 1. *The sequential composition $\Pi_{\text{RLC}} \circ \Pi_{\text{CCS}}$ is a **reduction of knowledge** (Definition 9) from $\text{CCS}(b, \mathcal{L})^K \times \text{CE}(b, \mathcal{L})^k$ to $\text{CE}(B, \mathcal{L})$.*

Proof. Follows directly from Π_{CCS} being a strong interactive reduction (Lemma 7), Π_{RLC} being a weak interactive reduction (Lemma 8), and the strong-weak composition theorem (Theorem 12). $\square$

Theorem 3. *The sequential composition $\Pi_{\text{DEC}} \circ \Pi_{\text{RLC}} \circ \Pi_{\text{CCS}}$ is a **reduction of knowledge** (Definition 9) from $\text{CCS}(b, \mathcal{L})^K \times \text{CE}(b, \mathcal{L})^k$ to $\text{CE}(b, \mathcal{L})^k$.*

Proof. Follows directly from $\Pi_{\text{RLC}} \circ \Pi_{\text{CCS}}$ being a reduction of knowledge (Lemma 1), Π_{DEC} being a reduction of knowledge (Theorem 13), and sequential composition of reductions of knowledge being reductions of knowledge (Lemma 4). $\square$

3 Overview of the following sections

Since the SuperNeo embedding lends itself to a more efficient and natural description of our folding scheme (since it does not require a SIMD constraint system), the rest of the paper will be focused on the adaption of HyperNova-like interactive reductions with the SuperNeo embedding (of course, with the appropriate lifting and analysis in the lattice setting). We defer the original Neo work to the current eprint [71].

The rest of the paper is organized as follows. In Section 4, we provide the necessary preliminaries for our work, including the syntax and security definitions for interactive reductions. In Section 5, we define the evaluation homomorphism embedding and generalize it to CCS relations over arbitrary extension fields of $\mathbb{F}$; this is a key technical tool used in our SuperNeo folding scheme. In Section 6, we give the formal definitions of strong and weak interactive reductions, and provide the exact composition theorem needed. In Section 7.1, we define the CCS relation and the CCS evaluation relation, which are the main relations used in our folding scheme. In Section 7.3, we describe the strong interactive reduction Π_{CCS} that reduces the CCS and CCS evaluation claims into CCS evaluation claims. In Section 7.4, we describe the weak interactive reduction Π_{RLC} that combines multiple CCS evaluation claims into a single CCS evaluation claim. In Section 7.5, we describe the final reduction of knowledge Π_{DEC} that reduces the norm of the evaluation claims from $B = b^k$ to b .

4 Preliminaries

Notation We let λ denote the security parameter and $\text{negl}(\lambda)$ denote a negligible function in λ . Throughout the paper, the depicted asymptotics depend on λ , but we elide this for brevity. We let PPT denote probabilistic polynomial time and EPT denote expected probabilistic polynomial time. We let $[n]$ denote the set $\{1, \dots, n\}$, and $\{u_i\}_{i \in [n]}$ denote the set $\{u_1, \dots, u_n\}$.

Polynomials Let $\mathbb{B}$ be a field. We write $\mathbb{B}^d[X_1, \dots, X_n]$ to denote multivariate polynomials over $\mathbb{B}$ in the variables $(X_1, \dots, X_n)$ with degree bound $\leq d$ for each variable. We omit the superscript if there is no degree bound. For $\ell \in \mathbb{N}_{\geq 1}$, define

$$\text{ZS}_\ell(\mathbb{B}) := \{P \in \mathbb{B}[X_1, \dots, X_\ell] : P(x) = 0 \text{ for every } x \in \{0, 1\}^\ell\}.$$

When the coefficient field is clear from context, we write ZS_ℓ . We denote the polynomial $\text{eq}(x, y) = \prod_{i=1}^\ell (x_i \cdot y_i + (1 - x_i) \cdot (1 - y_i))$, which outputs 1 if $x = y$ and 0 otherwise for $x, y \in \{0, 1\}^\ell$. For a vector $v \in \mathbb{B}^n$, where n is a power of two, we let $\tilde{v} \in \mathbb{B}^1[X_1, \dots, X_{\log n}]$ denote the multilinear polynomial extension of v : $\tilde{v} = \sum_{b \in \{0, 1\}^{\log n}} \text{eq}(X_1, \dots, X_{\log n}, b) \cdot v_b$. Let $\mathbb{B}$ and $\mathbb{K}$ be subfields of a common field $\mathbb{A}$. For $v \in \mathbb{B}^n$ and $r \in \mathbb{K}^{\log n}$, we evaluate $\tilde{v}$ at r through the natural inclusions into $\mathbb{A}$; explicitly, $\tilde{v}(r) := \sum_{b \in \{0, 1\}^{\log n}} \text{eq}(r, b) \cdot v_b \in \mathbb{A}$. More generally, let $R_\mathbb{B}$ and $R_\mathbb{K}$ be subrings of a common commutative ring $R_\mathbb{A}$. For $h \in R_\mathbb{B}^n$ and $r \in R_\mathbb{K}^{\log n}$, define $\tilde{h}(r) := \sum_{b \in \{0, 1\}^{\log n}} \text{eq}(r, b) \cdot h_b \in R_\mathbb{A}$.

Lemma 2 (Schwartz-Zippel [77]). *let $g : \mathbb{F}^\ell \rightarrow \mathbb{F}$ be a nonzero ℓ -variate polynomial of total degree at most d . Then, on any finite set $S \subseteq \mathbb{F}$,*

$$\Pr_{x \leftarrow S^\ell} [g(x) = 0] \leq d/|S|.$$

Lemma 3. *Let $\mathbb{B}$ be a field and $Q \in \mathbb{B}[X_1, \dots, X_\ell]$ be an arbitrary multivariate polynomial. Define multivariate polynomial $Q'(\vec{X}, \vec{Z}) := \text{eq}(\vec{X}, \vec{Z}) \cdot Q(\vec{X})$.*

$$0 = \sum_{\vec{x} \in \{0, 1\}^\ell} Q'(\vec{x}, \vec{Z}) \quad \text{if and only if} \quad Q(\vec{X}) \in \text{ZS}_\ell$$

Definition 1 (Fields, Rings, and Dimensions).

Fields: We identify the following fields $\mathbb{F}$, $\mathbb{L}$, and $\mathbb{K}$ with their corresponding subfields of $\mathbb{A}$. As such, we treat each field as a subfield of the larger fields under the natural inclusions $\mathbb{F} \subseteq \mathbb{L} \subseteq \mathbb{A}$ and $\mathbb{F} \subseteq \mathbb{K} \subseteq \mathbb{A}$.

- **Base field:** Let $\mathbb{F} := \mathbb{F}_q$ be a finite field of prime order q .
- **Native relation field:** Let $\mathbb{L} := \mathbb{F}_{q^\tau}$ be a degree- τ extension of $\mathbb{F}$.
- **Sum-check challenge field:** Let $\mathbb{K} := \mathbb{F}_{q^\nu}$ be a degree- ν extension of $\mathbb{F}$ such that $1/|\mathbb{K}| = \text{negl}(\lambda)$.
- **Ambient field:** Let $\mathbb{A} := \mathbb{F}_{q^{\text{lcm}(\tau, \nu)}}$ be the degree- $\text{lcm}(\tau, \nu)$ extension of $\mathbb{F}$ containing both $\mathbb{L}$ and $\mathbb{K}$.

Rings: Let $\Phi(X) := X^d + \Phi_{d-1}X^{d-1} + \dots + \Phi_1X + \Phi_0 \in \mathbb{F}[X]$ be the η -th cyclotomic polynomial with degree d . For every field $\mathbb{B} \in \{\mathbb{F}, \mathbb{L}, \mathbb{K}, \mathbb{A}\}$, define the ring $R_\mathbb{B} := \mathbb{B}[X]/(\Phi(X))$. We treat each ring as a subring of the larger rings under the natural inclusions $R_\mathbb{F} \subseteq R_\mathbb{L} \subseteq R_\mathbb{A}$ and $R_\mathbb{F} \subseteq R_\mathbb{K} \subseteq R_\mathbb{A}$.

Dimensions: Let $m, n_L, n_{L, \text{in}}, n_\mathbb{F}, n_{\mathbb{F}, \text{in}}, n_\mathbb{R}, n_{\mathbb{R}, \text{in}}, u, t, k, K \in \mathbb{N}_{\geq 1}$.

- **CCS parameters:** m denotes the number of constraints and is a power of two, u denotes a strict upper bound on the total degree of the CCS polynomial, and t denotes the number of CCS matrices.
- **Vector lengths:** n_L denotes the length of a relation vector $z \in \mathbb{L}^{n_L}$, $n_\mathbb{F}$ denotes the length of the corresponding flattened base-field vector $z^\flat \in \mathbb{F}^{n_\mathbb{F}}$ (defined shortly), and $n_\mathbb{R}$ denotes the length of its packed ring-vector representation $\mathbf{z} \in R_\mathbb{F}^{n_\mathbb{R}}$. These dimensions satisfy $n_\mathbb{F} := \tau \cdot n_L = d \cdot n_\mathbb{R} \leq m$. Define the padding length $n_{\text{pad}} := m - n_\mathbb{F} \in \mathbb{N}_{\geq 0}$.
- **Input lengths:** $n_{L, \text{in}}, n_{\mathbb{F}, \text{in}}$, and $n_{\mathbb{R}, \text{in}}$ denote the corresponding input lengths and satisfy $n_{\mathbb{F}, \text{in}} := \tau \cdot n_{L, \text{in}} = d \cdot n_{\mathbb{R}, \text{in}}$ and $n_{L, \text{in}} \leq n_L$.
- **Instance counts:** k and K denote the numbers of instances.

Norm bounds: Let $b, B = b^k < q/2 \in \mathbb{N}_{\geq 2}$ be norm bounds.

Remark 1 (Compatible dimensions). For a target native vector length n , set

$$s := \frac{d}{\gcd(d, \tau)}, \quad n_L := s \left\lceil \frac{n}{s} \right\rceil, \quad n_F := \tau n_L, \quad n_R := \frac{n_F}{d}.$$

This chooses the smallest $n_L \geq n$ satisfying $d \mid \tau n_L$, with minimal overhead $0 \leq n_L - n \leq s - 1$.

Definition 2 (Coefficient maps). For every field $\mathbb{B} \in \{\mathbb{F}, \mathbb{L}, \mathbb{K}, \mathbb{A}\}$, consider the ring $R_{\mathbb{B}}$ defined in Definition 1. For an element $a = \sum_{\ell=1}^d a_{\ell} X^{\ell-1} \in R_{\mathbb{B}}$, define its **coefficient vector** and **constant term** as

$$\text{cf}(a) := (a_1, \dots, a_d) \in \mathbb{B}^d, \quad \text{ct}(a) := a_1 \in \mathbb{B}.$$

Both maps are trivially $\mathbb{B}$ -linear, and the coefficient map is bijective.

For a vector $z = (z_1, \dots, z_m) \in R_{\mathbb{B}}^m$, define its **coefficient matrix** and **constant-term vector** as

$$\text{cf}(z) := [\text{cf}(z_1) \cdots \text{cf}(z_m)] \in \mathbb{B}^{d \times m}, \quad \text{ct}(z) := (\text{ct}(z_1), \dots, \text{ct}(z_m)) \in \mathbb{B}^m.$$

Concretely, $\text{cf}(z)_{\ell, j} := \text{cf}(z_j)_{\ell}$ for $\ell \in [d]$ and $j \in [m]$. We denote the ℓ -th row of $\text{cf}(z)$ by $\text{cf}(z)_{\ell} \in \mathbb{B}^m$.

Definition 3 (Norm). For an element $a \in \mathbb{F}$, let $a' \in \{0, \dots, q-1\}$ denote its integer representative. Define $\|a\|_{\infty} := \min\{a', q - a'\}$. For a vector $z = (z_1, \dots, z_n) \in \mathbb{F}^n$, define its ℓ_{∞} -norm as $\|z\|_{\infty} := \max_{i \in [n]} \|z_i\|_{\infty}$. For an element $a \in R_{\mathbb{F}}$ and a vector $z = (z_1, \dots, z_m) \in R_{\mathbb{F}}^m$, define $\|a\|_{\infty} := \|\text{cf}(a)\|_{\infty}$ and $\|z\|_{\infty} := \max_{i \in [m]} \|z_i\|_{\infty}$.

Decomposition For b, k as in Definition 1 and every $s \in \mathbb{N}_{\geq 1}$, let $\text{split}_b : \{z \in \mathbb{F}^s : \|z\|_{\infty} < b^k\} \rightarrow (\mathbb{F}^s)^k$ be the coordinate-wise signed b -ary decomposition map such that, for $\text{split}_b(z) = (z_1, \dots, z_k)$,

$$z = \sum_{i=1}^k b^{i-1} \cdot z_i \quad \text{and} \quad \|z_i\|_{\infty} < b \quad \text{for all } i \in [k].$$

Definition 4 (Module Homomorphism). Modules are a generalization of vector spaces for which the field of scalars is replaced by a ring R . Suppose R is a commutative ring with identity 1 and G is an abelian (commutative) group. The group G is an R -module if there is an operation $\cdot : R \times G \rightarrow G$ such that for all $r, s \in R$ and $x, y \in G$, $r \cdot (x + y) = r \cdot x + r \cdot y$, $(r + s) \cdot x = r \cdot x + s \cdot x$, $(rs) \cdot x = r \cdot (s \cdot x)$, $1 \cdot x = x$. Suppose G_1 and G_2 are R -modules. Similarly, an R -module homomorphism is a map $\mathcal{L} : G_1 \rightarrow G_2$ that is a generalization of a linear map of vector spaces. $\mathcal{L}$ is an R -module homomorphism if for all $x, y \in G_1$ and $r \in R$, $\mathcal{L}(x + y) = \mathcal{L}(x) + \mathcal{L}(y)$ and $\mathcal{L}(r \cdot x) = r \cdot \mathcal{L}(x)$.

Definition 5 (Module short integer solution [61, 64, 74]). Define the ring $R_{\mathbb{Z}} := \mathbb{Z}[X]/(\Phi(X))$. The $\text{MSIS}_{m, B}^{\infty, \kappa, q}$ problem is defined as follows: Given a matrix $M \xleftarrow{\$} R_{\mathbb{F}}^{\kappa \times m}$ sampled uniformly at random, find a non-zero vector $z \in R_{\mathbb{Z}}^m$ such that $Mz = 0 \pmod{q}$ and $\|z\|_{\infty} < B$, where $\|z\|_{\infty} := \max_{i \in [m], \ell \in [d]} |z_{i, \ell}|$ for $z_i = \sum_{\ell=1}^d z_{i, \ell} X^{\ell-1}$.

Theorem 4 (Low norm invertibility [66, Theorem 1.1, Conjecture 2.6]). Let $z \in \mathbb{N}$ such that $z \mid \eta$, $q \equiv 1 \pmod{z}$, and $\text{ord}_{\eta}(q) = \eta/z$. Define $b_{\text{inv}} := 1/\sqrt{\tau(z)} \cdot q^{1/\phi(z)}$ where $\tau(z) := z$ if z is odd, otherwise $\tau(z) = z/2$. For an arbitrary $a \in R_{\mathbb{F}}$, if $0 < \|a\|_{\infty} < b_{\text{inv}}$, then a is invertible in $R_{\mathbb{F}}$.

Definition 6 (Strong sampling sets [3, 28]). Define $\mathcal{C} \subseteq R_{\mathbb{F}}$ to be any set of ring elements such that for any distinct elements $a, b \in \mathcal{C}$, $\|a - b\|_{\infty} < b_{\text{inv}}$ (Theorem 4). Furthermore, we define the

$$\text{expansion factor of } \mathcal{C} := \max_{\substack{v \in R_{\mathbb{F}} \setminus \{0\} \\ \rho \in \mathcal{C}}} \frac{\|\rho v\|_{\infty}}{\|v\|_{\infty}}$$

Theorem 5 (Expansion factors [3]). Let $\mathcal{C}$ be a strong sampling set over the cyclotomic ring $R_{\mathbb{F}}$ (Definition 6), We denote the Euler totient function as ϕ . We must have that the expansion factor of $\mathcal{C}$ is $\leq 2 \cdot \phi(\eta) \cdot \max_{\rho \in \mathcal{C}} \|\rho\|_{\infty}$. If $\Phi(X) = X^d + 1$ for a power-of-two d , then the factor of 2 can be removed.

Definition 7 (Ring Commitment Scheme). A ring commitment scheme $\text{com} := (\text{Setup}, \text{Commit})$ consists of two PPT algorithms:

- $\text{Setup}(1^\lambda, m) \rightarrow \text{pp}$: Takes as input a security parameter 1^λ and length $m \in \mathbb{N}_{\geq 1}$, outputs public parameters pp .
- $\text{Commit}(\text{pp}, z) \rightarrow c$: Takes as input public parameters pp and a vector $z \in R_{\mathbb{F}}^m$, outputs a commitment $c \in \mathbb{C}$.

A ring commitment scheme can satisfy the following properties:

B-binding: For every length $m = \text{poly}(\lambda)$ and every EPT adversary $\mathcal{A}$, a ring commitment scheme is B-binding (for $B \in \mathbb{N}$) if the following probability holds:

$$\Pr \left[\begin{array}{l} \text{Commit}(\text{pp}, z_1) = \text{Commit}(\text{pp}, z_2) \\ \wedge \|z_1\|_{\infty}, \|z_2\|_{\infty} < B, \\ \wedge z_1 \neq z_2 \end{array} \middle| \begin{array}{l} \text{pp} \leftarrow \text{Setup}(1^\lambda, m) \\ z_1, z_2 \in R_{\mathbb{F}}^m \leftarrow \mathcal{A}(\text{pp}) \end{array} \right] \leq \epsilon_{\text{bind}}(B)$$

for $\epsilon_{\text{bind}}(B) \leq \text{negl}(\lambda)$. We refer to a pair of vectors (z_1, z_2) which satisfies the conditions in the probability as a **B-binding collision**.

(B,C)-relaxed binding: For every length $m = \text{poly}(\lambda)$ and every EPT adversary $\mathcal{A}$, a ring commitment scheme is (B,C)-relaxed binding (for $B \in \mathbb{N}$ and set $\mathcal{C} \subseteq R_{\mathbb{F}}$) if the following probability holds:

$$\Pr \left[\begin{array}{l} \Delta_1 \cdot c = \text{Commit}(\text{pp}, z_1) \\ \wedge \Delta_2 \cdot c = \text{Commit}(\text{pp}, z_2) \\ \wedge \|z_1\|_{\infty}, \|z_2\|_{\infty} < B, \\ \wedge \Delta_1 z_2 \neq \Delta_2 z_1 \end{array} \middle| \begin{array}{l} \text{pp} \leftarrow \text{Setup}(1^\lambda, m) \\ \left(\begin{array}{l} c \in \mathbb{C}, \\ \Delta_1, \Delta_2 \in (\mathcal{C} - \mathcal{C}), \\ z_1, z_2 \in R_{\mathbb{F}}^m \end{array} \right) \leftarrow \mathcal{A}(\text{pp}) \end{array} \right] \leq \epsilon_{\text{rlx}}(B, \mathcal{C})$$

for $\epsilon_{\text{rlx}}(B, \mathcal{C}) \leq \text{negl}(\lambda)$. We refer to a tuple of elements $(c, \Delta_1, \Delta_2, z_1, z_2)$ which satisfies the conditions in the probability as a **(B,C)-relaxed binding collision**.

Homomorphic: For every $m \in \mathbb{N}$ and $\text{pp} \in \text{Setup}(1^\lambda, m)$, the commitment algorithm, $\text{Commit}(\text{pp}, \cdot) : R_{\mathbb{F}}^m \rightarrow \mathbb{C}$, is an $R_{\mathbb{F}}$ -module homomorphism.

Definition 8 (Ajtai commitment scheme [2]). Let message length $m \in \mathbb{N}$, and let λ be a security parameter with $\kappa := \kappa(\lambda)$ and $q := q(\lambda)$, where $|\mathbb{F}| = q$. Define $\mathbb{C} := R_{\mathbb{F}}^{\kappa}$. The Ajtai commitment scheme $\text{com} := (\text{Setup}, \text{Commit})$ consists of the following PPT algorithms:

- $\text{Setup}(1^\lambda, m) \rightarrow \text{pp}$: Sample a random matrix $M \xleftarrow{\$} R_{\mathbb{F}}^{\kappa \times m}$. Output $\text{pp} \leftarrow M$.
- $\text{Commit}(\text{pp}, z) \rightarrow c$: Given parameters $\text{pp} := M$ and vector $z \in R_{\mathbb{F}}^m$, output $c \leftarrow Mz \in \mathbb{C}$.

Theorem 6 (Properties [2, 7, 9, 14]). The Ajtai commitment scheme (Definition 8) is a ring commitment scheme (Definition 7) that is **homomorphic**, **B-binding** (assuming the $\text{MSIS}_{m, 2B}^{\infty, \kappa, q}$ problem (Definition 5) is hard), and **(B,C)-relaxed binding** (assuming the $\text{MSIS}_{m, 4TB}^{\infty, \kappa, q}$ problem is hard and $\mathcal{C}$ is a strong sampling set (Definition 6) with expansion factor T (Theorem 5)).

Relation Products For relations $\mathcal{R}_1$ and $\mathcal{R}_2$ over public parameter, structure, instance, and witness pairs we define the relation $\mathcal{R}_1 \times \mathcal{R}_2$ such that $(\text{pp}, \text{s}, (u_1, u_2), (w_1, w_2)) \in \mathcal{R}_1 \times \mathcal{R}_2$ if and only if $(\text{pp}, \text{s}, u_1, w_1) \in \mathcal{R}_1$, and $(\text{pp}, \text{s}, u_2, w_2) \in \mathcal{R}_2$. We let $\mathcal{R}^n$ denote $\mathcal{R} \times \dots \times \mathcal{R}$ for n times.

By a slight abuse of notation, whenever we write $\text{CE}(b, \mathcal{L})^n$ (Definition 21), we mean the subrelation of the ordinary n -fold product in which all n instances contain the same evaluation point $r \in \mathbb{K}^{\log m}$. All other relation products retain the ordinary product meaning defined above.

Definition 9 (Interactive Reductions [51,53]). Consider relations $\mathcal{R}_1$ and $\mathcal{R}_2$ over parameters, structure, instance, and witness tuples. An **interactive reduction** from $\mathcal{R}_1$ to $\mathcal{R}_2$ is defined by PPT algorithms $(\mathcal{G}, \mathcal{K}, \mathcal{P}, \mathcal{V})$ called the generator, encoder, prover, and verifier respectively with the following interface.

- $\mathcal{G}(1^\lambda, \text{sz}) \rightarrow \text{pp}$: Takes as input a security parameter 1^λ and size parameters sz . Outputs public parameters pp .
- $\mathcal{K}(\text{pp}, \text{s}) \rightarrow (\text{pk}, \text{vk})$: Takes as input public parameters pp and a structure s . Deterministically, outputs a prover key pk and a verifier key vk .
- $\mathcal{P}(\text{pk}, u_1, w_1) \rightarrow (u_2, w_2)$: Takes as input a proving key pk and an instance-witness pair (u_1, w_1) . Interactively reduces the task of checking $(\text{pp}, \text{s}, u_1, w_1) \in \mathcal{R}_1$ to the task of checking $(\text{pp}, \text{s}, u_2, w_2) \in \mathcal{R}_2$.
- $\mathcal{V}(\text{vk}, u_1) \rightarrow u_2$: Takes as input a verifier key vk and an instance u_1 in $\mathcal{R}_1$. Interactively reduces the task of checking the instance u_1 to the task of checking a new instance u_2 in $\mathcal{R}_2$.

Let $\langle \mathcal{P}, \mathcal{V} \rangle$ denote the interaction between $\mathcal{P}$ and $\mathcal{V}$. We treat $\langle \mathcal{P}, \mathcal{V} \rangle$ as a function that takes as input $((\text{pk}, \text{vk}), u_1, w_1)$ and runs the interaction on the prover's input (pk, u_1, w_1) and the verifier's input (vk, u_1) . At the end of the interaction, $\langle \mathcal{P}, \mathcal{V} \rangle$ outputs the verifier's instance u_2 and the prover's witness w_2 .

A **reduction of knowledge** [52] is an interactive reduction, $(\mathcal{G}, \mathcal{K}, \mathcal{P}, \mathcal{V})$, that satisfies the following properties:

- (i) **Completeness**: For any EPT adversary $\mathcal{A}$, given $\text{pp} \leftarrow \mathcal{G}(1^\lambda, \text{sz})$, $(\text{s}, u_1, w_1) \leftarrow \mathcal{A}(\text{pp})$ such that $(\text{pp}, \text{s}, u_1, w_1) \in \mathcal{R}_1$, we have that the prover's output instance is equal to the verifier's output instance u_2 , and that
$$(\text{pp}, \text{s}, \langle \mathcal{P}, \mathcal{V} \rangle((\text{pk}, \text{vk}), u_1, w_1)) \in \mathcal{R}_2.$$
- (ii) **Knowledge soundness**: For any EPT adversary $(\mathcal{A}, \mathcal{P}^*)$, there exists an EPT extractor $\mathcal{E}$ such that if the success probability of the adversary

$$\epsilon(\mathcal{A}, \mathcal{P}^*) := \Pr \left[(\text{pp}, \text{s}, \langle \mathcal{P}^*, \mathcal{V} \rangle((\text{pk}, \text{vk}), u_1, \text{st})) \in \mathcal{R}_2 \mid \begin{array}{l} \text{pp} \leftarrow \mathcal{G}(1^\lambda, \text{sz}) \\ (\text{s}, u_1, \text{st}) \leftarrow \mathcal{A}(\text{pp}) \\ (\text{pk}, \text{vk}) \leftarrow \mathcal{K}(\text{pp}, \text{s}) \end{array} \right]$$

$\geq 1/\text{poly}(\lambda)$, then we have that

$$\Pr \left[(\text{pp}, \text{s}, u_1, w_1) \in \mathcal{R}_1 \mid \begin{array}{l} \text{pp} \leftarrow \mathcal{G}(1^\lambda, \text{sz}) \\ (\text{s}, u_1, \text{st}) \leftarrow \mathcal{A}(\text{pp}) \\ (\text{pk}, \text{vk}) \leftarrow \mathcal{K}(\text{pp}, \text{s}) \\ w_1 \leftarrow \mathcal{E}(\text{pp}, \text{s}, u_1, \text{st}) \end{array} \right] \geq \epsilon(\mathcal{A}, \mathcal{P}^*) - \text{negl}(\lambda).$$

- (iii) **Public Coin**: All of the verifier's messages are uniformly random strings of some prescribed length. Furthermore, the verifier's messages contain all of the random coins (randomness) used by the verifier.⁹

In this work, we are primarily interested in building folding schemes, a particular type of reduction of knowledge that reduces the task of checking a fresh instance in a relation $\mathcal{R}$ together with a running instance in an accumulator relation $\mathcal{R}_{\text{ACC}}$ to checking a new running instance in $\mathcal{R}_{\text{ACC}}$.

Definition 10 (Folding scheme). A folding scheme for a relation $\mathcal{R}$ is a reduction of knowledge of type $\mathcal{R} \times \mathcal{R}_{\text{ACC}} \rightarrow \mathcal{R}_{\text{ACC}}$ for some accumulator relation $\mathcal{R}_{\text{ACC}}$.

⁹ If a reduction of knowledge is public-coin, then it trivially satisfies the property of **public reducibility** described in [53] as the execution of the verifier $\mathcal{V}$ can be emulated using the randomness from the transcript.

Lemma 4 (Sequential composition [51,53]). For reductions of knowledge $\Pi_1 = (\mathcal{G}, \mathcal{K}, \mathcal{P}_1, \mathcal{V}_1) : \mathcal{R}_1 \rightarrow \mathcal{R}_2$ and $\Pi_2 = (\mathcal{G}, \mathcal{K}, \mathcal{P}_2, \mathcal{V}_2) : \mathcal{R}_2 \rightarrow \mathcal{R}_3$, we have that $\Pi_2 \circ \Pi_1 = (\mathcal{G}, \mathcal{K}, \mathcal{P}, \mathcal{V}) : \mathcal{R}_1 \rightarrow \mathcal{R}_3$ is a reduction of knowledge where $\mathcal{K}(\text{pp}, \text{s})$ computes (pk, vk) and where

$$\begin{aligned}\mathcal{P}(\text{pk}, u_1, w_1) &= \mathcal{P}_2(\text{pk}, \mathcal{P}_1(\text{pk}, u_1, w_1)) \\ \mathcal{V}(\text{vk}, u_1) &= \mathcal{V}_2(\text{vk}, \mathcal{V}_1(\text{vk}, u_1, w_1))\end{aligned}$$

Definition 11 (The sum-check protocol [63]). For the challenge and ambient fields $\mathbb{K} \subseteq \mathbb{A}$ from Definition 1, the sum-check protocol $\text{SumCheck}(\mathbb{T}; Q)$ is a classic interactive proof protocol between two PPT algorithms $(\mathcal{P}, \mathcal{V})$. The protocol reduces a sum-check claim $\mathbb{T} = \sum_{x \in \{0,1\}^\ell} Q(x)$, where $\mathbb{T} \in \mathbb{A}$ is the claimed sum and $Q \in \mathbb{A}^{\leq d}[X_1, \dots, X_\ell]$ is an ℓ -variate polynomial of individual degree at most d , to an evaluation claim $v \stackrel{?}{=} Q(r)$, where $r \stackrel{\$}{\leftarrow} \mathbb{K}^\ell$ is a uniformly random point and $v \in \mathbb{A}$ is the claimed evaluation. The verifier can check this claim by querying Q at r . The protocol is public-coin, has a completeness error of 0, and has a soundness error of at most $\ell d / |\mathbb{K}|$. A self-contained description of the sum-check protocol can be found in this note [82].

Definition 12 (Special sets [39]). Let $\mathcal{C}$ be a set and $\ell \in \mathbb{N}$. Consider two vectors $x, y \in \mathcal{C}^\ell$. We define the relation $\equiv_i$ for $i \in [\ell]$ as follows:

$$x \equiv_i y \iff x_i \neq y_i \wedge x_j = y_j \text{ for all } j \in [\ell] \setminus \{i\}.$$

A special set $\text{SS}(\mathcal{C}, \ell)$ is as follows:

$$\text{SS}(\mathcal{C}, \ell) = \left\{ (\vec{c}, \vec{c}_1, \dots, \vec{c}_\ell) \in (\mathcal{C}^\ell)^{\ell+1} : \begin{array}{l} \forall i \in [\ell], \\ \vec{c} \equiv_i \vec{c}_i \end{array} \right\},$$

Theorem 7 (Coordinate-wise extraction [39, Lemma 7.1]). Let $\mathcal{C}$ be a finite set, $\ell \in \mathbb{N}$, and $\vec{\mathcal{C}} := \mathcal{C}^\ell$ be a challenge space. Let $A : \vec{\mathcal{C}} \rightarrow \{0,1\}^*$ be an arbitrary (probabilistic) expected polynomial-time algorithm (adversary), and $V : \vec{\mathcal{C}} \times \{0,1\}^* \rightarrow \{0,1\}$ be an arbitrary (probabilistic) expected polynomial-time function (verification). Define the success probability of adversary A as

$$\epsilon^V(A) := \Pr_{\vec{c} \stackrel{\$}{\leftarrow} \vec{\mathcal{C}}} [V(\vec{c}, A(\vec{c})) = 1]$$

Then, there exists an expected polynomial-time oracle algorithm E_A (extractor) that makes at most $\ell+1$ queries to A in expectation and with probability at least $\epsilon^V(A) - \frac{\ell}{|\mathcal{C}|}$ outputs $\ell+1$ pairs $(\vec{c}, w), (\vec{c}_1, w_1), \dots, (\vec{c}_\ell, w_\ell)$ such that

- $V(\vec{c}, w) = 1$,
- for all $i \in [\ell]$, $V(\vec{c}_i, w_i) = 1$,
- and $(\vec{c}, \vec{c}_1, \dots, \vec{c}_\ell) \in \text{SS}(\mathcal{C}, \ell)$.

5 Embeddings and Evaluation Homomorphism

Let $\mathbb{B}$ be a degree- δ extension of $\mathbb{F}$ with a fixed ordered $\mathbb{F}$ -basis $(\beta_1, \dots, \beta_\delta) \in \mathbb{B}^\delta$. Later, we concretely instantiate $\mathbb{B}$ as either $\mathbb{F}$ or $\mathbb{L}$ from Definition 1 for the norm and relation checks, respectively.

Definition 13 (Coefficient Embedding).

Element embedding: For a vector $v = (v_1, \dots, v_d) \in \mathbb{B}^d$, define $\text{Emb}_{\mathbb{B}}(v) := \sum_{i=1}^d v_i X^{i-1} \in \mathbb{R}_{\mathbb{B}}$. By Definition 2, $\text{Emb}_{\mathbb{B}}$ is the inverse of the coefficient map $\text{cf}(\cdot)$; in particular, $\text{cf}(\text{Emb}_{\mathbb{B}}(v)) = v$.

Vector embedding: For $s \in \mathbb{N}_{\geq 1}$ and $v \in \mathbb{B}^{ds}$, partition $v = (v_1, \dots, v_s)$ into blocks $v_j \in \mathbb{B}^d$ and define

$$\text{Emb}_{\mathbb{B}}(v) := (\text{Emb}_{\mathbb{B}}(v_1), \dots, \text{Emb}_{\mathbb{B}}(v_s)) \in \mathbb{R}_{\mathbb{B}}^s.$$

Matrix embedding: For $M \in \mathbb{B}^{m \times ds}$, let $M_i = (M_{i,1}, \dots, M_{i,s})$ denote the i -th row partitioned into blocks $M_{i,j} \in \mathbb{B}^d$. Define $\text{Emb}_{\mathbb{B}}(M) \in \mathbb{R}_{\mathbb{B}}^{m \times s}$ by

$$(\text{Emb}_{\mathbb{B}}(M))_{i,j} := \text{Emb}_{\mathbb{B}}(M_{i,j}), \quad \forall i \in [m], j \in [s].$$

Since the element embedding is bijective, its vector and matrix extensions are also bijective, with inverses applied blockwise.

Definition 14 (Flattening and Packed Embedding).

For $n \in \mathbb{N}_{\geq 1}$, consider an arbitrary vector $z = (z_1, \dots, z_n) \in \mathbb{B}^n$.

Flattening map: Recall that $(\beta_1, \dots, \beta_\delta) \in \mathbb{B}^\delta$ is the fixed ordered $\mathbb{F}$ -basis of $\mathbb{B}$. Uniquely write each $z_i = \sum_{j=1}^{\delta} z_{i,j} \beta_j$ for $z_{i,j} \in \mathbb{F}$.

$$\text{Flat}_{\mathbb{B}}(z) := (z_{1,1}, \dots, z_{1,\delta}, \dots, z_{n,1}, \dots, z_{n,\delta}) \in \mathbb{F}^{\delta n}$$

Thus, $\text{Flat}_{\mathbb{B}} : \mathbb{B}^n \rightarrow \mathbb{F}^{\delta n}$ is an $\mathbb{F}$ -linear bijection.

Packed embedding: If $d \mid \delta n$,

$$\text{Pack}_{\mathbb{B}} : \mathbb{B}^n \rightarrow \mathbb{R}_{\mathbb{F}}^{\delta n/d}, \quad \text{Pack}_{\mathbb{B}}(z) := \text{Emb}_{\mathbb{F}}(\text{Flat}_{\mathbb{B}}(z))$$

The packed embedding is an $\mathbb{F}$ -linear bijection because both $\text{Flat}_{\mathbb{B}}$ and $\text{Emb}_{\mathbb{F}}$ are $\mathbb{F}$ -linear bijections.

Remark 2 (Instantiations). For norm checks, we instantiate $\mathbb{B} = \mathbb{F}$ (i.e., $\delta = 1$) with the trivial basis (1). Then, the map $\text{Flat}_{\mathbb{F}}$ is the identity map.

$$\text{Flat}_{\mathbb{F}} : \mathbb{F}^{n_{\mathbb{F}}} \rightarrow \mathbb{F}^{n_{\mathbb{F}}}, \quad \text{Pack}_{\mathbb{F}} : \mathbb{F}^{n_{\mathbb{F}}} \rightarrow \mathbb{R}_{\mathbb{F}}^{n_{\mathbb{R}}}.$$

For relations, we instantiate $\mathbb{B} = \mathbb{L}$ (i.e., $\delta = \tau$) with any fixed ordered $\mathbb{F}$ -basis $(\beta_1, \dots, \beta_\tau) \in \mathbb{L}^\tau$. Then,

$$\text{Flat}_{\mathbb{L}} : \mathbb{L}^{n_{\mathbb{L}}} \rightarrow \mathbb{F}^{n_{\mathbb{F}}}, \quad \text{Pack}_{\mathbb{L}} : \mathbb{L}^{n_{\mathbb{L}}} \rightarrow \mathbb{R}_{\mathbb{F}}^{n_{\mathbb{R}}}.$$

Both packed embeddings have codomain $\mathbb{R}_{\mathbb{F}}^{n_{\mathbb{R}}}$ because Definition 1 gives $n_{\mathbb{F}} = \tau n_{\mathbb{L}} = d n_{\mathbb{R}}$.

For a relation vector $z \in \mathbb{L}^{n_{\mathbb{L}}}$, define

$$z^b := \text{Flat}_{\mathbb{L}}(z) \in \mathbb{F}^{n_{\mathbb{F}}}, \quad z := \text{Pack}_{\mathbb{L}}(z) = \text{Pack}_{\mathbb{F}}(z^b) \in \mathbb{R}_{\mathbb{F}}^{n_{\mathbb{R}}}.$$

Thus, the norm and relation checks use the same packed ring vector z , viewed as encoding z^b over $\mathbb{F}$ and z over $\mathbb{L}$, respectively.

Prior works [36, 65] use an *inner-product trick* based on a linear transform $\text{Trans}_{\mathbb{F}} : \mathbb{F}^d \rightarrow \mathbb{F}^d$, formally defined below, to simulate, for $a, b \in \mathbb{F}^d$, a field inner product $\langle a, b \rangle \in \mathbb{F}$ via a ring multiplication $\text{Emb}_{\mathbb{F}}(\text{Trans}_{\mathbb{F}}(a)) \cdot \text{Emb}_{\mathbb{F}}(b) \in \mathbb{R}_{\mathbb{F}}$ satisfying $\text{ct}(\text{Emb}_{\mathbb{F}}(\text{Trans}_{\mathbb{F}}(a)) \cdot \text{Emb}_{\mathbb{F}}(b)) = \langle a, b \rangle$. We establish new results that generalize this inner-product trick to arbitrary extension fields $\mathbb{B}$ of $\mathbb{F}$ and their corresponding lifted rings $\mathbb{R}_{\mathbb{B}}$, with self-contained proofs.

Lemma 5 (Constant-Term Matrix). Define the symmetric matrix $T \in \mathbb{F}^{d \times d}$ where $T_{i,j} := \text{ct}(X^{i+j})$ for $i, j \in \{0, \dots, d-1\}$. Then T is invertible.

Proof. Write $\mathbb{R}_{\mathbb{F}} = \mathbb{F}[X]/(X^d + f_{d-1}X^{d-1} + \dots + f_1X + f_0)$ with $f_0 \in \{+1, -1\}$. Since $X^d \equiv -(f_{d-1}X^{d-1} + \dots + f_1X + f_0)$, we have $\text{ct}(X^d) = -f_0$, while $\text{ct}(X^k) = 0$ for $1 \leq k < d$. Thus $T_{i,j} = 0$ for $0 < i+j < d$, $T_{i,j} = -f_0$ for $i+j = d$, and trivially $T_{0,0} = \text{ct}(X^0 = 1) = 1$. Let $N \in \mathbb{F}^{(d-1) \times (d-1)}$ be the submatrix of T obtained by deleting row 0 and column 0. Then $\det(T) = T_{0,0} \cdot \det(N) = \det(N)$. Observe that $N_{i,j} = 0$ for $i+j < d-2$ and $N_{i,j} = \pm 1$ for $i+j = d-2$. Recursively expanding along the top row yields $\det(N) = \pm 1$; hence $\det(T) = \pm 1$ and T is invertible. $\square$

Corollary 1. Let $T_{\mathbb{B}} \in \mathbb{B}^{d \times d}$ be the lift of $T \in \mathbb{F}^{d \times d}$ under the natural inclusion $\mathbb{F}^{d \times d} \subseteq \mathbb{B}^{d \times d}$. Then $T_{\mathbb{B}}$ is symmetric and invertible.

Proof. Symmetry is unchanged, and $\det(T_{\mathbb{B}}) = \det(T) = \pm 1$ remains nonzero in $\mathbb{B}$. $\square$

Definition 15 (Inner Product Transform). Recall that $\beta := (\beta_1, \dots, \beta_\delta) \in \mathbb{B}^\delta$ is the fixed ordered basis vector for $\mathbb{B}$ over $\mathbb{F}$. Let $n \in \mathbb{N}_{\geq 1}$ satisfy $d \mid \delta n$.

Vector transform: Define $\text{Trans}_{\mathbb{B}} : \mathbb{B}^n \rightarrow \mathbb{B}^{\delta n}$ as follows.

Given $a = (a_1, \dots, a_n) \in \mathbb{B}^n$, set $s := \delta n/d$ and partition

$$a \otimes \beta = (\beta_1 a_1, \dots, \beta_\delta a_1, \dots, \beta_1 a_n, \dots, \beta_\delta a_n) = (u_1, \dots, u_s), \quad u_j \in \mathbb{B}^d.$$

Then set $\text{Trans}_{\mathbb{B}}(a) := (T_{\mathbb{B}}^{-1} u_1, \dots, T_{\mathbb{B}}^{-1} u_s)$.

Matrix transform: For $M \in \mathbb{B}^{m \times n}$ with rows $M_1, \dots, M_m \in \mathbb{B}^n$, define the lifted map $\text{Trans}_{\mathbb{B}} : \mathbb{B}^{m \times n} \rightarrow \mathbb{B}^{m \times \delta n}$ by

$$\text{Trans}_{\mathbb{B}}(M) := \begin{bmatrix} \text{Trans}_{\mathbb{B}}(M_1) \\ \vdots \\ \text{Trans}_{\mathbb{B}}(M_m) \end{bmatrix} \in \mathbb{B}^{m \times \delta n}.$$

Equivalently, $(\text{Trans}_{\mathbb{B}}(M))_i := \text{Trans}_{\mathbb{B}}(M_i)$ for all $i \in [m]$.

Remark 3 (Efficiency and Sparsity Preservation). When $\phi(X)$ is a power-of-two or trinomial cyclotomic, applying $T_{\mathbb{B}}^{-1}$ to a vector in $\mathbb{B}^d$ requires only permutations, negations, and a constant number of additions per entry, and therefore costs $O(d)$ operations over $\mathbb{B}$. For $a \in \mathbb{B}^n$, the transform $\text{Trans}_{\mathbb{B}}(a)$ applies $T_{\mathbb{B}}^{-1}$ to $\delta n/d$ blocks, requiring $(\delta n/d) \cdot O(d) = O(\delta n)$ operations. Forming $a \otimes \beta \in \mathbb{B}^{\delta n}$ also requires $O(\delta n)$ operations, so the complete transform costs $O(\delta n)$ operations over $\mathbb{B}$.

If $M \in \mathbb{B}^{m \times n}$ is sparse, then its basis expansion $M \otimes \beta \in \mathbb{B}^{m \times \delta n}$ has the same density as M . For these cyclotomics, $T_{\mathbb{B}}^{-1}$ has constant column sparsity, so the block transform preserves sparsity up to a constant factor. Thus, $\text{Trans}_{\mathbb{B}}(M)$ is also sparse.

Lemma 6 (Block Inner Product Trick). Let $T_{\mathbb{B}}$ be the lifted constant-term matrix from Corollary 1. For all $x, y \in \mathbb{B}^d$,

$$\text{ct}(\text{Emb}_{\mathbb{B}}(T_{\mathbb{B}}^{-1}x) \cdot \text{Emb}_{\mathbb{B}}(y)) = \langle x, y \rangle \in \mathbb{B}.$$

Proof. Let $w := T_{\mathbb{B}}^{-1}x \in \mathbb{B}^d$. We index the coordinates of w and y , and the rows and columns of $T_{\mathbb{B}}$, by $\{0, \dots, d-1\}$. By Definition 13, $\text{Emb}_{\mathbb{B}}(w) \cdot \text{Emb}_{\mathbb{B}}(y) = \sum_{i,j=0}^{d-1} w_i y_j X^{i+j}$. By Lemma 5 and Corollary 1, $(T_{\mathbb{B}})_{i,j} = \text{ct}(X^{i+j})$ for all $i, j \in \{0, \dots, d-1\}$. Thus, by the linearity of the constant-term map,

$$\text{ct}(\text{Emb}_{\mathbb{B}}(w) \cdot \text{Emb}_{\mathbb{B}}(y)) = \sum_{i,j=0}^{d-1} w_i y_j \text{ct}(X^{i+j}) = w^{\top} T_{\mathbb{B}} y.$$

Since $T_{\mathbb{B}}$ is symmetric, $T_{\mathbb{B}}^{-1}$ is also symmetric, and hence

$$w^{\top} T_{\mathbb{B}} y = (T_{\mathbb{B}}^{-1}x)^{\top} T_{\mathbb{B}} y = x^{\top} T_{\mathbb{B}}^{-1} T_{\mathbb{B}} y = x^{\top} y = \langle x, y \rangle.$$

$\square$

Theorem 8 (Inner Product Trick). Let $n \in \mathbb{N}_{\geq 1}$ satisfy $d \mid \delta n$, and set $s := \delta n/d$. Recall from Definition 13, 14, and 15 the maps

$$\text{Trans}_{\mathbb{B}} : \mathbb{B}^n \rightarrow \mathbb{B}^{ds}, \quad \text{Emb}_{\mathbb{B}} : \mathbb{B}^{ds} \rightarrow \mathbb{R}_{\mathbb{B}}^s, \quad \text{Pack}_{\mathbb{B}} : \mathbb{B}^n \rightarrow \mathbb{R}_{\mathbb{F}}^s.$$

For all $a, b \in \mathbb{B}^n$,

$$\text{ct}(\langle \text{Emb}_{\mathbb{B}}(\text{Trans}_{\mathbb{B}}(a)), \text{Pack}_{\mathbb{B}}(b) \rangle) = \langle a, b \rangle \in \mathbb{B}.$$

The ring inner product is computed in $\mathbb{R}_{\mathbb{B}}$ using the inclusion $\mathbb{R}_{\mathbb{F}} \subseteq \mathbb{R}_{\mathbb{B}}$.

Proof. Set $s := \delta n/d$. Recall that $\beta = (\beta_1, \dots, \beta_\delta)$ is the fixed ordered $\mathbb{F}$ -basis of $\mathbb{B}$. For each $i \in [n]$, uniquely write $b_i = \sum_{k=1}^{\delta} b_{i,k} \beta_k$ with $b_{i,k} \in \mathbb{F}$. Let

$$\begin{aligned} a \otimes \beta &= (\beta_1 a_1, \dots, \beta_\delta a_1, \dots, \beta_1 a_n, \dots, \beta_\delta a_n) = (u_1, \dots, u_s), \\ \text{Flat}_{\mathbb{B}}(b) &= (b_{1,1}, \dots, b_{1,\delta}, \dots, b_{n,1}, \dots, b_{n,\delta}) = (v_1, \dots, v_s), \end{aligned}$$

where $u_j \in \mathbb{B}^d$ and $v_j \in \mathbb{F}^d \subseteq \mathbb{B}^d$ are the corresponding d -sized blocks for every $j \in [s]$. By Definitions 13, 14, and 15,

$$\begin{aligned} \text{Trans}_{\mathbb{B}}(a) &= (T_{\mathbb{B}}^{-1} u_1, \dots, T_{\mathbb{B}}^{-1} u_s), \\ \text{Pack}_{\mathbb{B}}(b) &= (\text{Emb}_{\mathbb{F}}(v_1), \dots, \text{Emb}_{\mathbb{F}}(v_s)). \end{aligned}$$

Since $v_j \in \mathbb{F}^d \subseteq \mathbb{B}^d$, we have $\text{Emb}_{\mathbb{F}}(v_j) = \text{Emb}_{\mathbb{B}}(v_j)$. Therefore, by the $\mathbb{B}$ -linearity of ct and Lemma 6 applied to each pair (u_j, v_j) ,

$$\begin{aligned} \text{ct}(\langle \text{Emb}_{\mathbb{B}}(\text{Trans}_{\mathbb{B}}(a)), \text{Pack}_{\mathbb{B}}(b) \rangle) &= \sum_{j=1}^s \text{ct}(\text{Emb}_{\mathbb{B}}(T_{\mathbb{B}}^{-1} u_j) \cdot \text{Emb}_{\mathbb{B}}(v_j)) \\ &= \sum_{j=1}^s \langle u_j, v_j \rangle = \sum_{i=1}^n \sum_{k=1}^{\delta} (\beta_k a_i) b_{i,k} = \sum_{i=1}^n a_i \left(\sum_{k=1}^{\delta} b_{i,k} \beta_k \right) = \langle a, b \rangle. \end{aligned}$$

□

Theorem 9 (Matrix-Vector Product Transform). *Let $n \in \mathbb{N}_{\geq 1}$ satisfy $d \mid \delta n$, and set $s := \delta n/d$. Recall from Definitions 13, 14, and 15 the maps*

$$\text{Trans}_{\mathbb{B}} : \mathbb{B}^{m \times n} \rightarrow \mathbb{B}^{m \times ds}, \quad \text{Emb}_{\mathbb{B}} : \mathbb{B}^{m \times ds} \rightarrow \mathbb{R}_{\mathbb{B}}^{m \times s}, \quad \text{Pack}_{\mathbb{B}} : \mathbb{B}^n \rightarrow \mathbb{R}_{\mathbb{F}}^s.$$

For every $M \in \mathbb{B}^{m \times n}$ and $z \in \mathbb{B}^n$,

$$\text{ct}(\text{Emb}_{\mathbb{B}}(\text{Trans}_{\mathbb{B}}(M)) \cdot \text{Pack}_{\mathbb{B}}(z)) = Mz \in \mathbb{B}^m.$$

The matrix-vector product inside ct is computed in $\mathbb{R}_{\mathbb{B}}$ using the inclusion $\mathbb{R}_{\mathbb{F}} \subseteq \mathbb{R}_{\mathbb{B}}$.

Proof. Let $M_1, \dots, M_m \in \mathbb{B}^n$ be the rows of M . By Definitions 13 and 15,

$$(\text{Emb}_{\mathbb{B}}(\text{Trans}_{\mathbb{B}}(M)))_i = \text{Emb}_{\mathbb{B}}(\text{Trans}_{\mathbb{B}}(M_i))$$

for every $i \in [m]$. Hence, by Theorem 8, the i -th coordinate of the left-hand side is

$$\text{ct}(\langle \text{Emb}_{\mathbb{B}}(\text{Trans}_{\mathbb{B}}(M_i)), \text{Pack}_{\mathbb{B}}(z) \rangle) = \langle M_i, z \rangle = (Mz)_i.$$

Since this holds for every $i \in [m]$, $\text{ct}(\text{Emb}_{\mathbb{B}}(\text{Trans}_{\mathbb{B}}(M)) \cdot \text{Pack}_{\mathbb{B}}(z)) = Mz$. □

Theorem 10 (Matrix-Vector Product Evaluation). *Let $\mathbb{B} \in \{\mathbb{F}, \mathbb{L}\}$ be a degree- δ extension of $\mathbb{F}$. Let $m, n \in \mathbb{N}_{\geq 1}$, with m a power of two and $d \mid \delta n$, and consider $M \in \mathbb{B}^{m \times n}$, $z \in \mathbb{B}^n$, and $r \in \mathbb{K}^{\log m}$. Define*

$$h := \text{Emb}_{\mathbb{B}}(\text{Trans}_{\mathbb{B}}(M)) \cdot \text{Pack}_{\mathbb{B}}(z) \in \mathbb{R}_{\mathbb{B}}^m \subseteq \mathbb{R}_{\mathbb{A}}^m, \quad y := \widetilde{h}(r) \in \mathbb{R}_{\mathbb{A}}.$$

Then, for every $\ell \in [d]$, $\text{cf}(y)_{\ell} = \widetilde{\text{cf}(h)}_{\ell}(r) \in \mathbb{A}$. In particular, $\text{ct}(y) = \widetilde{Mz}(r) \in \mathbb{A}$.

Proof. Consider an arbitrary ring element $a := \sum_{\ell=1}^d a_{\ell} X^{\ell-1} \in \mathbb{R}_{\mathbb{B}}$ and scalar $c \in \mathbb{K}$. Multiplying a by c scales each of its coefficients by c : $ca = \sum_{\ell=1}^d (ca_{\ell}) X^{\ell-1} \in \mathbb{R}_{\mathbb{A}}$. Hence, by linearity of the coefficient map, $\text{cf}(ca)_{\ell} = c \cdot \text{cf}(a)_{\ell} \in \mathbb{A}$ for every $\ell \in [d]$. By definition, $y = \widetilde{h}(r) = \sum_{\vec{x} \in \{0,1\}^{\log m}} \text{eq}(r, \vec{x}) \cdot h_{\vec{x}} \in \mathbb{R}_{\mathbb{A}}$. Since each $\text{eq}(r, \vec{x})$ lies in $\mathbb{K}$, applying the observation above termwise and using linearity of the coefficient map gives $\text{cf}(y)_{\ell} = \widetilde{\text{cf}(h)}_{\ell}(r) \in \mathbb{A}$ for every $\ell \in [d]$. Taking $\ell = 1$ gives $\text{ct}(y) = \text{ct}(h)(r) \in \mathbb{A}$. By Theorem 9, $\text{ct}(h) = Mz \in \mathbb{B}^m$, and therefore $\text{ct}(y) = \widetilde{Mz}(r) \in \mathbb{A}$. □

Theorem 11 (Evaluation Homomorphism). Recall from Definition 1 the fields $\mathbb{F}, \mathbb{L}, \mathbb{K}, \mathbb{A}$ and rings $R_{\mathbb{F}}, R_{\mathbb{L}}, R_{\mathbb{K}}, R_{\mathbb{A}}$ satisfying

$$\mathbb{F} \subseteq \mathbb{L} \subseteq \mathbb{A}, \quad \mathbb{F} \subseteq \mathbb{K} \subseteq \mathbb{A}, \quad R_{\mathbb{F}} \subseteq R_{\mathbb{L}} \subseteq R_{\mathbb{A}}, \quad R_{\mathbb{F}} \subseteq R_{\mathbb{K}} \subseteq R_{\mathbb{A}}.$$

Let $\mathbb{B} \in \{\mathbb{F}, \mathbb{L}\}$ be a degree- δ extension of $\mathbb{F}$, and let $m, n, s, s_{\text{in}} \in \mathbb{N}_{\geq 1}$ satisfy $\delta n = ds$, with m a power of two. Consider $M \in \mathbb{B}^{m \times n}$, vectors $z_1, \dots, z_{\ell} \in \mathbb{B}^n$, scalars $\rho_1, \dots, \rho_{\ell} \in R_{\mathbb{F}}$, and $r \in \mathbb{K}^{\log m}$. Recall the maps

$$\text{Trans}_{\mathbb{B}} : \mathbb{B}^{m \times n} \rightarrow \mathbb{B}^{m \times ds}, \quad \text{Emb}_{\mathbb{B}} : \mathbb{B}^{m \times ds} \rightarrow R_{\mathbb{B}}^{m \times s}, \quad \text{Pack}_{\mathbb{B}} : \mathbb{B}^n \rightarrow R_{\mathbb{F}}^s.$$

Let $\mathcal{L} : R_{\mathbb{F}}^s \rightarrow \mathbb{C}$ and $\mathcal{L}_{\text{in}} : R_{\mathbb{F}}^s \rightarrow R_{\mathbb{F}}^{s_{\text{in}}}$ be $R_{\mathbb{F}}$ -module homomorphisms. Define

$$\begin{aligned} \forall i \in [\ell], \quad & \mathbf{z}_i := \text{Pack}_{\mathbb{B}}(z_i) \in R_{\mathbb{F}}^s, & c_i &:= \mathcal{L}(\mathbf{z}_i) \in \mathbb{C}, \\ & h_i := \text{Emb}_{\mathbb{B}}(\text{Trans}_{\mathbb{B}}(M)) \cdot \mathbf{z}_i \in R_{\mathbb{B}}^m, & y_i &:= \widetilde{h_i}(r) \in R_{\mathbb{A}}, \\ & \mathbf{x}_i := \mathcal{L}_{\text{in}}(\mathbf{z}_i) \in R_{\mathbb{F}}^{s_{\text{in}}}, \\ & \mathbf{z} := \sum_{i=1}^{\ell} \rho_i \mathbf{z}_i \in R_{\mathbb{F}}^s, & c &:= \sum_{i=1}^{\ell} \rho_i c_i \in \mathbb{C}, \\ & h := \sum_{i=1}^{\ell} \rho_i h_i \in R_{\mathbb{B}}^m, & y &:= \sum_{i=1}^{\ell} \rho_i y_i \in R_{\mathbb{A}}, \\ & \mathbf{x} := \sum_{i=1}^{\ell} \rho_i \mathbf{x}_i \in R_{\mathbb{F}}^{s_{\text{in}}}. \end{aligned}$$

Then, we must have

$$\begin{aligned} h &= \text{Emb}_{\mathbb{B}}(\text{Trans}_{\mathbb{B}}(M)) \cdot \mathbf{z} \in R_{\mathbb{B}}^m & c &= \mathcal{L}(\mathbf{z}) \in \mathbb{C}, \\ \mathbf{x} &= \mathcal{L}_{\text{in}}(\mathbf{z}) \in R_{\mathbb{F}}^{s_{\text{in}}} & y &= \widetilde{h}(r) \in R_{\mathbb{A}} \\ & \quad \wedge \\ & \quad \forall i \in [\ell], \quad \text{ct}(y_i) = \widetilde{Mz_i}(r) \in \mathbb{A}, \\ & \quad \text{ct}(y) = \widetilde{Mz}(r) \in \mathbb{A} \end{aligned}$$

for $z := \text{Pack}_{\mathbb{B}}^{-1}(\mathbf{z}) \in \mathbb{B}^n$. All evaluations are computed in $R_{\mathbb{A}}$.

Proof. Since $\mathcal{L}$ and $\mathcal{L}_{\text{in}}$ are $R_{\mathbb{F}}$ -module homomorphisms,

$$\begin{aligned} c &= \sum_{i=1}^{\ell} \rho_i \mathcal{L}(\mathbf{z}_i) = \mathcal{L}\left(\sum_{i=1}^{\ell} \rho_i \mathbf{z}_i\right) = \mathcal{L}(\mathbf{z}), \\ \mathbf{x} &= \sum_{i=1}^{\ell} \rho_i \mathcal{L}_{\text{in}}(\mathbf{z}_i) = \mathcal{L}_{\text{in}}\left(\sum_{i=1}^{\ell} \rho_i \mathbf{z}_i\right) = \mathcal{L}_{\text{in}}(\mathbf{z}). \end{aligned}$$

By distributivity,

$$h = \sum_{i=1}^{\ell} \rho_i h_i = \sum_{i=1}^{\ell} \rho_i (\text{Emb}_{\mathbb{B}}(\text{Trans}_{\mathbb{B}}(M)) \cdot \mathbf{z}_i) = \text{Emb}_{\mathbb{B}}(\text{Trans}_{\mathbb{B}}(M)) \cdot \sum_{i=1}^{\ell} \rho_i \mathbf{z}_i = \text{Emb}_{\mathbb{B}}(\text{Trans}_{\mathbb{B}}(M)) \cdot \mathbf{z}.$$

Multilinear evaluation at r is $R_{\mathbb{A}}$ -linear and hence $R_{\mathbb{F}}$ -linear. Therefore,

$$y = \sum_{i=1}^{\ell} \rho_i y_i = \sum_{i=1}^{\ell} \rho_i \widetilde{h_i}(r) = \widetilde{\sum_{i=1}^{\ell} \rho_i h_i}(r) = \widetilde{h}(r).$$

By Theorem 10, $\text{ct}(y_i) = \widetilde{Mz_i}(r)$ for every $i \in [\ell]$. Finally, for $z := \text{Pack}_{\mathbb{B}}^{-1}(\mathbf{z})$, we have $\mathbf{z} = \text{Pack}_{\mathbb{B}}(z)$, so another application of Theorem 10 gives $\text{ct}(y) = \widetilde{Mz}(r)$. $\square$

Remark 4 (Reductions). In our reductions, we instantiate $(\mathbb{B}, \delta, n, \beta)$ with

$$(\mathbb{F}, 1, n_{\mathbb{F}}, (1)) \quad \text{and} \quad (\mathbb{L}, \tau, n_{\mathbb{L}}, (\beta_1, \dots, \beta_{\tau})).$$

From Definition 1, $n_{\mathbb{F}} = \tau n_{\mathbb{L}} = dn_{\mathbb{R}}$. Remark 2 gives two variants of the witness: $z \in \mathbb{L}^{n_{\mathbb{L}}}$ as the native relation vector for CCS, and $z^{\flat} := \text{Flat}_{\mathbb{L}}(z) \in \mathbb{F}^{n_{\mathbb{F}}}$ as the flattened base field vector required for norm checking. These variants share the same packed representation as a ring vector, namely

$$\mathbf{z} := \text{Pack}_{\mathbb{L}}(z) = \text{Pack}_{\mathbb{F}}(z^{\flat}) \in R_{\mathbb{F}}^{n_{\mathbb{R}}}.$$

Looking ahead, we must check that z has low norm and that z satisfies the CCS relation over matrices $M_j \in \mathbb{L}^{m \times n_L}$. To do so, we will need to fold evaluations of a padded version of z^b and of $M_j z$. Assume $n_F \leq m$ and m is a power of two. Define

$$\text{Pad} := \begin{bmatrix} I_{n_F} \\ 0_{(m-n_F) \times n_F} \end{bmatrix} \in \mathbb{F}^{m \times n_F}, \quad z^{b, \text{pad}} := \text{Pad } z^b = \begin{bmatrix} z^b \\ 0_{m-n_F} \end{bmatrix} \in \mathbb{F}^m.$$

Since padding only appends zero coordinates, $\|z^{b, \text{pad}}\|_\infty = \|z^b\|_\infty$. For $r \in \mathbb{K}^{\log m}$, define

$$\begin{aligned} h &:= \text{Emb}_{\mathbb{F}}(\text{Trans}_{\mathbb{F}}(\text{Pad})) \cdot z \in \mathbb{R}_{\mathbb{F}}^m, & y &:= \widetilde{h}(r) \in \mathbb{R}_{\mathbb{K}}, \\ \forall j \in [t], \quad h_j &:= \text{Emb}_{\mathbb{L}}(\text{Trans}_{\mathbb{L}}(M_j)) \cdot z \in \mathbb{R}_{\mathbb{L}}^m, & y_j &:= \widetilde{h}_j(r) \in \mathbb{R}_{\mathbb{A}}. \end{aligned}$$

By Theorem 10,

$$\text{ct}(y) = \widetilde{z^{b, \text{pad}}}(r) \in \mathbb{K}, \quad \forall j \in [t], \quad \text{ct}(y_j) = \widetilde{M_j z}(r) \in \mathbb{A}.$$

Thus, Theorem 11 lets us fold both kinds of evaluation claims consistently with folds of the same committed ring vector z .

6 Strong and weak interactive reductions

Definition 16 (Uniqueness Adversaries). A uniqueness adversary $\mathcal{A}_{\text{uniq}}^{\mathcal{O}}$ is a probabilistic expected polynomial-time oracle algorithm that may query its oracle $\mathcal{O}$ adaptively. It outputs either $\perp$ or a pair of responses returned by two different calls to $\mathcal{O}$. Each call to $\mathcal{O}$ uses fresh independent randomness. The running time includes all computations performed by $\mathcal{O}$.

Definition 17 (Weak Interactive Reductions). Consider relations $\mathcal{R}_1$, $\mathcal{R}'_1$, and $\mathcal{R}_2$ over public parameters, structure, instance, and witness tuples such that $\mathcal{R}_1 \subseteq \mathcal{R}'_1$. Let $\mathcal{U}_1$ be the ambient instance space of $\mathcal{R}_1$.

An interactive reduction $\Pi : \mathcal{R}_1 \rightarrow \mathcal{R}_2$, defined by PPT algorithms $(\mathcal{G}, \mathcal{K}, \mathcal{P}, \mathcal{V})$ (Definition 9), is **weak** if it is complete, public coin, and there exists a function $\phi : \mathcal{U}_1 \rightarrow \mathbb{C}$ (for an arbitrary space $\mathbb{C}$) such that for any EPT adversary $(\mathcal{A}, \mathcal{P}^*)$, there exists an EPT extractor $\mathcal{E}$ such that

(i) If the success probability of the adversary $\epsilon(\mathcal{A}, \mathcal{P}^*) \geq 1/\text{poly}(\lambda)$, then

$$\Pr \left[(\text{pp}, s, u_1, w_1) \in \mathcal{R}'_1 \mid \begin{array}{l} \text{pp} \leftarrow \mathcal{G}(1^\lambda, \text{sz}) \\ (s, u_1, \text{st}) \leftarrow \mathcal{A}(\text{pp}) \\ (\text{pk}, \text{vk}) \leftarrow \mathcal{K}(\text{pp}, s) \\ w_1 \leftarrow \mathcal{E}(\text{pp}, s, u_1, \text{st}) \end{array} \right] \geq \epsilon(\mathcal{A}, \mathcal{P}^*) - \text{negl}(\lambda).$$

(ii) If $\mathcal{A} := (\mathcal{B}, \mathcal{B}')$ such that

$$\Pr \left[\begin{array}{c} u_1, u'_1 \neq \perp \\ \downarrow \\ \phi(u_1) = \phi(u'_1) \end{array} \mid \begin{array}{l} \text{pp} \leftarrow \mathcal{G}(1^\lambda, \text{sz}) \\ (s, \text{st}^*) \leftarrow \mathcal{B}(\text{pp}) \\ (u_1, \text{st}) \leftarrow \mathcal{B}'(\text{st}^*) \\ (u'_1, \text{st}') \leftarrow \mathcal{B}'(\text{st}^*) \end{array} \right] = 1,$$

then the following holds. Define the oracle

$$\begin{aligned} \mathcal{O}_{\text{weak}}(\text{pp}, s, \text{st}^*) &\rightarrow (u_1, w_1) : \\ (a) \quad &(u_1, \text{st}) \leftarrow \mathcal{B}'(\text{st}^*). \\ (b) \quad &w_1 \leftarrow \mathcal{E}(\text{pp}, s, u_1, \text{st}). \\ (c) \quad &\text{Output } (u_1, w_1). \end{aligned}$$

For any uniqueness adversary $\mathcal{W}_{\text{uniq}}^{\mathcal{O}_{\text{weak}}}$ (Definition 16), we have

$$\Pr \left[\begin{array}{c} w_1, w'_1 \neq \perp \\ \wedge w_1 \neq w'_1 \\ \text{in} \leftarrow (\text{pp}, s, \text{st}^*) \\ (u_1, w_1), (u'_1, w'_1) \leftarrow \mathcal{W}_{\text{uniq}}^{\mathcal{O}_{\text{weak}}(\text{in})}(\text{in}) \end{array} \right] \leq \text{negl}(\lambda)$$

Definition 18 (Strong Interactive Reductions). Consider relations $\mathcal{R}_1$, $\mathcal{R}_2$, and $\mathcal{R}'_2$ over public parameters, structure, instance, and witness tuples such that $\mathcal{R}_2 \subseteq \mathcal{R}'_2$. Let $\mathcal{U}_2$ be the ambient instance space of $\mathcal{R}_2$.

An interactive reduction $\Pi : \mathcal{R}_1 \rightarrow \mathcal{R}_2$, defined by PPT algorithms $(\mathcal{G}, \mathcal{K}, \mathcal{P}, \mathcal{V})$ (Definition 9), is **strong** if it is complete, public coin, and there exists a function $\phi : \mathcal{U}_2 \rightarrow \mathbb{C}$ (for an arbitrary space $\mathbb{C}$) such that

(i) For any EPT adversary $(\mathcal{A}, \mathcal{P}^*)$,

$$\Pr \left[\begin{array}{c|c} u_2, u'_2 \neq \perp & \text{pp} \leftarrow \mathcal{G}(1^\lambda, \text{sz}) \\ \downarrow & (\text{s}, u_1, \text{st}_1) \leftarrow \mathcal{A}(\text{pp}) \\ \phi(u_2) = \phi(u'_2) & (\text{pk}, \text{vk}) \leftarrow \mathcal{K}(\text{pp}, \text{s}) \\ & (u_2, w_2) \leftarrow \langle \mathcal{P}^*, \mathcal{V} \rangle((\text{pk}, \text{vk}), u_1, \text{st}_1) \\ & (u'_2, w'_2) \leftarrow \langle \mathcal{P}^*, \mathcal{V} \rangle((\text{pk}, \text{vk}), u_1, \text{st}_1) \end{array} \right] = 1$$

(ii) For any EPT adversary $(\mathcal{A}, \mathcal{P}^*)$, define the oracle

$\mathcal{O}_{\text{str}}(\text{pp}, \text{s}, u_1, \text{st}) \rightarrow (u_2, w_2) :$
(a) $(\text{pk}, \text{vk}) \leftarrow \mathcal{K}(\text{pp}, \text{s})$.
(b) $(u_2, w_2) \leftarrow \langle \mathcal{P}^*, \mathcal{V} \rangle((\text{pk}, \text{vk}), u_1, \text{st})$.
(c) Output (u_2, w_2) .

There exists an EPT extractor $\mathcal{E}$ such that if

$$\epsilon'(\mathcal{A}, \mathcal{P}^*) := \Pr \left[(\text{pp}, \text{s}, \langle \mathcal{P}^*, \mathcal{V} \rangle((\text{pk}, \text{vk}), u_1, \text{st})) \in \mathcal{R}'_2 \mid \begin{array}{c} \text{pp} \leftarrow \mathcal{G}(1^\lambda, \text{sz}) \\ (\text{s}, u_1, \text{st}) \leftarrow \mathcal{A}(\text{pp}) \\ (\text{pk}, \text{vk}) \leftarrow \mathcal{K}(\text{pp}, \text{s}) \end{array} \right]$$

$\geq 1/\text{poly}(\lambda)$, and, for any uniqueness adversary $\mathcal{S}_{\text{uniq}}^{\mathcal{O}_{\text{str}}}$ (Definition 16),

$$\Pr \left[\begin{array}{c|c} w_2, w'_2 \neq \perp & \text{pp} \leftarrow \mathcal{G}(1^\lambda, \text{sz}) \\ \wedge & (\text{s}, u_1, \text{st}) \leftarrow \mathcal{A}(\text{pp}) \\ w_2 \neq w'_2 & \text{in} \leftarrow (\text{pp}, \text{s}, u_1, \text{st}) \\ & (u_2, w_2), (u'_2, w'_2) \leftarrow \mathcal{S}_{\text{uniq}}^{\mathcal{O}_{\text{str}}(\text{in})}(\text{in}) \end{array} \right] \leq \text{negl}(\lambda)$$

then we have that

$$\Pr \left[(\text{pp}, \text{s}, u_1, w_1) \in \mathcal{R}_1 \mid \begin{array}{c} \text{pp} \leftarrow \mathcal{G}(1^\lambda, \text{sz}) \\ (\text{s}, u_1, \text{st}) \leftarrow \mathcal{A}(\text{pp}) \\ (\text{pk}, \text{vk}) \leftarrow \mathcal{K}(\text{pp}, \text{s}) \\ w_1 \leftarrow \mathcal{E}(\text{pp}, \text{s}, u_1, \text{st}) \end{array} \right] \geq \epsilon'(\mathcal{A}, \mathcal{P}^*) - \text{negl}(\lambda).$$

Theorem 12 (Strong-Weak Composition). Consider relations $\mathcal{R}_1$, $\mathcal{R}_2$, $\mathcal{R}'_2$ and $\mathcal{R}_3$ over public parameters, structure, instance, and witness tuples such that $\mathcal{R}_2 \subseteq \mathcal{R}'_2$. Let $\mathcal{U}_2$ be the ambient instance space of $\mathcal{R}_2$. Consider interactive reductions (Definition 9) $\Pi_1 : \mathcal{R}_1 \rightarrow \mathcal{R}_2$ ($\mathcal{R}'_2$), $\Pi_2 : \mathcal{R}_2$ ($\mathcal{R}'_2$) $\rightarrow \mathcal{R}_3$ such that

- (i) Π_1 is **strong** (Definition 18) with respect to a function $\phi : \mathcal{U}_2 \rightarrow \mathbb{C}$ and
- (ii) Π_2 is **weak** (Definition 17) with respect to the **same** function ϕ ,

then the sequential composition $\Pi_2 \circ \Pi_1 : \mathcal{R}_1 \rightarrow \mathcal{R}_3$ is a **reduction of knowledge**.

Proof. For brevity, we defer the proof to Appendix B.1. □

7 SuperNeo's folding scheme for CCS

7.1 Relations

Definition 19 (Structure). We define a **structure** as a collection of elements

$$\text{s} := \left\{ \{M_j \in \mathbb{L}^{m \times n_L}\}_{j \in [t]}, f \in \mathbb{L}[X_1, \dots, X_t] \text{ with } \deg(f) < u \right\},$$

which consists of matrices and a polynomial of total degree less than u .

Definition 20 (Norm-bounded CCS). Let $\mathcal{L} : \mathbb{R}_{\mathbb{F}}^{n_{\mathbb{R}}} \rightarrow \mathbb{C}$ be an arbitrary $\mathbb{R}_{\mathbb{F}}$ -module homomorphism. Let $\mathbf{s}$ be a structure as defined in Definition 19. We define the **norm-bounded CCS relation**, $\text{CCS}(b, \mathcal{L})$, as follows:

$$\left\{ \begin{array}{l} \text{For } z := [x, w] \in \mathbb{L}^{n_L}, \\ z^b := \text{Flat}_{\mathbb{L}}(z) \in \mathbb{F}^{n_{\mathbb{F}}}, \\ \mathbf{z} := \text{Pack}_{\mathbb{L}}(z) \\ \quad = \text{Pack}_{\mathbb{F}}(z^b) \in \mathbb{R}_{\mathbb{F}}^{n_{\mathbb{R}}}, \\ c = \mathcal{L}(\mathbf{z}) \wedge \|z^b\|_{\infty} < b \wedge \\ f(\widetilde{M_1 z}, \dots, \widetilde{M_t z}) \in \text{ZS}_{\log m}(\mathbb{L}) \end{array} \right\}$$

Definition 21 (Norm-bounded CCS Evaluation Relation). Let $\mathbf{s}$ be a structure as defined in Definition 19. Let $\mathcal{L} : \mathbb{R}_{\mathbb{F}}^{n_{\mathbb{R}}} \rightarrow \mathbb{C}$ be an arbitrary $\mathbb{R}_{\mathbb{F}}$ -module homomorphism. Define $\mathcal{L}_{\text{in}} : \mathbb{R}_{\mathbb{F}}^{n_{\mathbb{R}}} \rightarrow \mathbb{R}_{\mathbb{F}}^{n_{\mathbb{R}, \text{in}}}$ to be the trivial $\mathbb{R}_{\mathbb{F}}$ -module homomorphism that projects the first $n_{\mathbb{R}, \text{in}}$ indices. Recall from Definition 1 that $n_{\mathbb{F}} \leq m$ and $n_{\text{pad}} := m - n_{\mathbb{F}}$. Define

$$\text{Pad} := \begin{bmatrix} I_{n_{\mathbb{F}}} \\ 0_{n_{\text{pad}} \times n_{\mathbb{F}}} \end{bmatrix} \in \mathbb{F}^{m \times n_{\mathbb{F}}}.$$

We define the **norm-bounded CCS evaluation relation**, $\text{CE}(b, \mathcal{L})$, as follows:

$$\left\{ \begin{array}{l} z^b := \text{Flat}_{\mathbb{L}}(z) \in \mathbb{F}^{n_{\mathbb{F}}}, \\ \mathbf{x} := \text{Pack}_{\mathbb{L}}(x) \in \mathbb{R}_{\mathbb{F}}^{n_{\mathbb{R}, \text{in}}}, \\ \mathbf{z} := \text{Pack}_{\mathbb{L}}(z) \\ \quad = \text{Pack}_{\mathbb{F}}(z^b) \in \mathbb{R}_{\mathbb{F}}^{n_{\mathbb{R}}}, \\ c = \mathcal{L}(\mathbf{z}) \wedge \mathbf{x} = \mathcal{L}_{\text{in}}(\mathbf{z}) \\ \wedge \|z^b\|_{\infty} < b \wedge \\ h := \text{Emb}_{\mathbb{F}}(\text{Trans}_{\mathbb{F}}(\text{Pad})) \cdot \mathbf{z} \in \mathbb{R}_{\mathbb{F}}^m, \\ y = \widetilde{h}(r) \wedge \\ \forall j \in [t], \\ h_j := \text{Emb}_{\mathbb{L}}(\text{Trans}_{\mathbb{L}}(M_j)) \cdot \mathbf{z} \in \mathbb{R}_{\mathbb{L}}^m, \\ y_j = \widetilde{h_j}(r). \end{array} \right\}$$

7.2 A folding scheme for CCS via interactive reductions

Definition 22 (Global Reduction Parameters).

Here, we define the global parameters used in our reductions:

- Define the fields $\mathbb{F}, \mathbb{L}, \mathbb{K}, \mathbb{A}$, extension degrees τ, ν , and rings $\mathbb{R}_{\mathbb{F}}, \mathbb{R}_{\mathbb{L}}, \mathbb{R}_{\mathbb{K}}, \mathbb{R}_{\mathbb{A}}$ as in Definition 1. The dimensions $m, n_L, n_{L, \text{in}}, n_{\mathbb{F}}, n_{\mathbb{F}, \text{in}}, n_{\mathbb{R}}, n_{\mathbb{R}, \text{in}}$, CCS parameters u, t , instance counts k, K , and norm bounds b, B are also as in Definition 1.
- Let $\mathcal{C} \subseteq \mathbb{R}_{\mathbb{F}}$ be a strong sampling set (Definition 6) with expansion factor T such that $(K + k)T(b - 1) < B$ and $1/|\mathcal{C}| = \text{negl}(\lambda)$.
- Let $\text{com} := (\text{Setup}, \text{Commit})$ be a ring commitment scheme (Definition 7), which is homomorphic and $(2B, \mathcal{C})$ -relaxed binding. For $\text{pp} \leftarrow \text{Setup}(1^\lambda, n_{\mathbb{R}})$, define $\mathcal{L} := \text{Commit}(\text{pp}, \cdot) : \mathbb{R}_{\mathbb{F}}^{n_{\mathbb{R}}} \rightarrow \mathbb{C}$, which is a $\mathbb{R}_{\mathbb{F}}$ -module homomorphism.
- Let $\mathcal{L}_{\text{in}} : \mathbb{R}_{\mathbb{F}}^{n_{\mathbb{R}}} \rightarrow \mathbb{R}_{\mathbb{F}}^{n_{\mathbb{R}, \text{in}}}$ be the trivial $\mathbb{R}_{\mathbb{F}}$ -module homomorphism that projects the first $n_{\mathbb{R}, \text{in}}$ indices.
- Let $\mathbf{s}$ denote a structure as defined in Definition 19.

In Section 8, we instantiate these parameters with concrete values.

7.3 Interactive reduction for CCS – Π_{CCS}

Overview. The interactive reduction Π_{CCS} checks that the K incoming CCS instances (Definition 20) indeed satisfy the required CCS constraints, the k *evaluation* claims (Definition 21), from the prior folding step, hold for point r , and checks that the norms of all of the flattened witness vectors (all $K + k$ of them) involved are less than b . To do so, Π_{CCS} relies on the classic sum-check protocol (Definition 11). The approach is inspired by similar reductions from [14, 56]. Π_{CCS} defines helper polynomials that, when used in the sum-check protocol, will perform the previously specified checks. $F(\vec{X})$ encodes the CCS constraints (all K of them). $\text{NC}(\vec{X})$ encodes the norm constraints (all $K + k$ of them). $\text{Eval}_{\mathbb{K}}(\vec{X})$ encodes the flattened-witness evaluation claims (all k of them). $\text{Eval}_{\mathbb{A}}(\vec{X})$ encodes the relation evaluation claims (all kt of them). Finally, $Q(\vec{X})$ is defined such that if its sum over the boolean hypercube $\{0, 1\}^{\log(m)}$ equals to the constructed sum T , then all the respective checks hold.

CCS reduction Π_{CCS}
Parameters: Refer to Definition 22. Input $\in \text{CCS}(b, \mathcal{L})^K \times \text{CE}(b, \mathcal{L})^k$ $(s; \quad (c_i \in \mathbb{C}, x_i \in \mathbb{L}^{n_{L, \text{in}}}); \quad w_i \in \mathbb{L}^{n_L - n_{L, \text{in}}})_{i=1}^K,$ $(s; \quad c_i \in \mathbb{C}, x_i \in \mathbb{L}^{n_{L, \text{in}}}, r \in \mathbb{K}^{\log m}, y_i \in \mathbb{R}_{\mathbb{K}}, \{y_{i,j} \in \mathbb{R}_{\mathbb{A}}\}_{j \in [t]}; \quad z_i \in \mathbb{L}^{n_L})_{i=K+1}^{K+k}$ Output $\in \text{CE}(b, \mathcal{L})^{K+k}$ $(s; \quad c_i \in \mathbb{C}, x_i \in \mathbb{L}^{n_{L, \text{in}}}, r' \in \mathbb{K}^{\log m}, y'_i \in \mathbb{R}_{\mathbb{K}}, \{y'_{i,j} \in \mathbb{R}_{\mathbb{A}}\}_{j \in [t]}; \quad z_i \in \mathbb{L}^{n_L})_{i \in [K+k]}$ Setup $\mathcal{G}(1^\lambda, n_R) \rightarrow \text{pp}$: Output $\text{pp} \leftarrow \text{Setup}(1^\lambda, n_R)$. Encoder $\mathcal{K}(\text{pp}, s) \rightarrow (\text{pk}, \text{vk})$: Output $((\text{pp}, s), \perp)$. Reduction $\langle \mathcal{P}, \mathcal{V} \rangle((\text{pk}, \text{vk}), u_1, w_1) \rightarrow (u_2; w_2)$: $\mathcal{V}$: Send challenges $\alpha \xleftarrow{\\$} \mathbb{K}^{\log m}$ and $\gamma \xleftarrow{\\$} \mathbb{K}$ to $\mathcal{P}$. $\mathcal{V} \leftrightarrow \mathcal{P}$: For all $i \in [K]$, define $z_i := [x_i, w_i] \in \mathbb{L}^{n_L}$. Recall from Definition 21 the matrix $\text{Pad} \in \mathbb{F}^{m \times n_F}$. For every $i \in [K + k]$, define $z_i^b := \text{Flat}_{\mathbb{L}}(z_i) \in \mathbb{F}^{n_F},$ $z_i^{b, \text{pad}} := \text{Pad} z_i^b \in \mathbb{F}^m,$ $z_i := \text{Pack}_{\mathbb{L}}(z_i) = \text{Pack}_{\mathbb{F}}(z_i^b) \in \mathbb{R}_{\mathbb{F}}^{n_R},$ $h_i := \text{Emb}_{\mathbb{F}}(\text{Trans}_{\mathbb{F}}(\text{Pad})) \cdot z_i \in \mathbb{R}_{\mathbb{F}}^m.$ For every $i \in [K + k]$ and $j \in [t]$, define $h_{i,j} := \text{Emb}_{\mathbb{L}}(\text{Trans}_{\mathbb{L}}(M_j)) \cdot z_i \in \mathbb{R}_{\mathbb{L}}^m.$ Finally, define $\vec{X} := (X_1, \dots, X_{\log m})$. For $i \in \{K + 1, \dots, K + k\}$, $j \in [t]$, and $\ell \in [d]$, define $\text{I}_{\mathbb{K}}(i, \ell) := (i - K - 1) + k(\ell - 1),$ $\text{I}_{\mathbb{A}}(i, j, \ell) := (i - K - 1) + k(j - 1) + kt(\ell - 1).$ $F(\vec{X}) := \sum_{i=1}^K \gamma^{i-1} \cdot f(\widetilde{M_1 z_i}, \dots, \widetilde{M_t z_i}) \in \mathbb{A}[\vec{X}],$ $\text{NC}(\vec{X}) := \sum_{i=1}^{K+k} \gamma^{i-1} \cdot \prod_{a=-b+1}^{b-1} (\widetilde{z_i^{b, \text{pad}}} - a) \in \mathbb{K}[\vec{X}],$

$$\text{Eval}_{\mathbb{K}}(\vec{X}) := \text{eq}(\vec{X}, r) \cdot \sum_{i=K+1}^{K+k} \sum_{\ell=1}^d \gamma^{\text{I}_{\mathbb{K}}(i,\ell)} \cdot \widetilde{\text{cf}(h_i)}_{\ell} \in \mathbb{K}[\vec{X}],$$

$$\text{Eval}_{\mathbb{A}}(\vec{X}) := \text{eq}(\vec{X}, r) \cdot \sum_{i=K+1}^{K+k} \sum_{j=1}^t \sum_{\ell=1}^d \gamma^{\text{I}_{\mathbb{A}}(i,j,\ell)} \cdot \widetilde{\text{cf}(h_{i,j})}_{\ell} \in \mathbb{A}[\vec{X}].$$

Define the sum-check polynomial

$$Q(\vec{X}) := \text{Eval}_{\mathbb{K}}(\vec{X}) + \gamma^{kd} \cdot \text{Eval}_{\mathbb{A}}(\vec{X}) + \gamma^{kd(t+1)} \cdot \text{eq}(\vec{X}, \alpha) (\text{F}(\vec{X}) + \gamma^K \cdot \text{NC}(\vec{X})) \in \mathbb{A}[\vec{X}].$$

Define claimed sum of Q over $\{0, 1\}^{\log m}$ as

$$\text{T} := \sum_{i=K+1}^{K+k} \sum_{\ell=1}^d \gamma^{\text{I}_{\mathbb{K}}(i,\ell)} \cdot \text{cf}(y_i)_{\ell} + \gamma^{kd} \cdot \sum_{i=K+1}^{K+k} \sum_{j=1}^t \sum_{\ell=1}^d \gamma^{\text{I}_{\mathbb{A}}(i,j,\ell)} \cdot \text{cf}(y_{i,j})_{\ell} \in \mathbb{A}.$$

Perform **SumCheck**($\text{T}; Q$) (Definition 11) which reduces the claim that

$$\text{T} = \sum_{\vec{x} \in \{0,1\}^{\log m}} Q(\vec{x})$$

to a new evaluation claim $v \stackrel{?}{=} Q(r') \in \mathbb{A}$ for a new evaluation point $r' \in \mathbb{K}^{\log m}$.

3. $\mathcal{P}$: Send the claimed evaluations

$$\begin{aligned} \forall i \in [K+k], \quad y'_i &\leftarrow \widetilde{h_i}(r') \in \mathbb{R}_{\mathbb{K}}, \\ \forall i \in [K+k], j \in [t], \quad y'_{i,j} &\leftarrow \widetilde{h_{i,j}}(r') \in \mathbb{R}_{\mathbb{A}}. \end{aligned}$$

4. $\mathcal{V}$: By Theorem 10, interpret $\text{ct}(y'_i) \in \mathbb{K}$ and $\text{ct}(y'_{i,j}) \in \mathbb{A}$ as the claimed evaluations $\widetilde{z_i^{b,\text{pad}}}(r')$ and $\widetilde{M_j z_i}(r')$, respectively, and derive the claimed intermediate evaluations

$$\text{F} := \sum_{i=1}^K \gamma^{i-1} \cdot f(\text{ct}(y'_{i,1}), \dots, \text{ct}(y'_{i,t})) \in \mathbb{A},$$

$$\text{N} := \sum_{i=1}^{K+k} \gamma^{i-1} \cdot \prod_{a=-b+1}^{b-1} (\text{ct}(y'_i) - a) \in \mathbb{K},$$

$$\text{E}_{\mathbb{K}} := \text{eq}(r', r) \cdot \sum_{i=K+1}^{K+k} \sum_{\ell=1}^d \gamma^{\text{I}_{\mathbb{K}}(i,\ell)} \cdot \text{cf}(y'_i)_{\ell} \in \mathbb{K},$$

$$\text{E}_{\mathbb{A}} := \text{eq}(r', r) \cdot \sum_{i=K+1}^{K+k} \sum_{j=1}^t \sum_{\ell=1}^d \gamma^{\text{I}_{\mathbb{A}}(i,j,\ell)} \cdot \text{cf}(y'_{i,j})_{\ell} \in \mathbb{A}.$$

Check the evaluation claim $v \stackrel{?}{=} Q(r')$ as follows,

$$v \stackrel{?}{=} \text{E}_{\mathbb{K}} + \gamma^{kd} \cdot \text{E}_{\mathbb{A}} + \gamma^{kd(t+1)} \cdot \text{eq}(r', \alpha) (\text{F} + \gamma^K \cdot \text{N}).$$

5. Output ($\mathbf{s}; c_i, x_i, r', y'_i, \{y'_{i,j}\}_{j \in [t]}; z_i\}_{i \in [K+k]}$)

Lemma 7 (Π_{CCS} is strong). *The interactive reduction $\Pi_{\text{CCS}} : \text{CCS}(b, \mathcal{L})^K \times \text{CE}(b, \mathcal{L})^k \rightarrow \text{CE}(b, \mathcal{L})^{K+k}$ ($\text{CE}(q/2, \mathcal{L})^{K+k}$) is **strong** (Definition 18) for the function ϕ , which projects commitments $(c_i)_{i \in [K+k]}$ from the instance.*

Proof. For brevity, we defer the proof to Appendix B.2. □

7.4 Random linear combination reduction – Π_{RLC}

The interactive reduction Π_{RLC} does exactly as the name suggests. Given $K + k$ input CCS evaluation claims whose flattened witness vectors have norm less than b , it outputs a single CCS evaluation claim of larger norm bound B , which is a random linear combination of the input claims using challenges from a strong sampling set $\mathcal{C}$ (Definition 6). Using the same challenges, Π_{RLC} combines the commitments, packed inputs and witnesses, evaluations of the flattened witnesses in $\mathbb{R}_{\mathbb{K}}$, and relation evaluations in $\mathbb{R}_{\mathbb{A}}$.

Random linear combination reduction Π_{RLC}
Parameters: Refer to Definition 22. Input $\in \text{CE}(b, \mathcal{L})^{K+k}$ $(s; c_i \in \mathbb{C}, x_i \in \mathbb{L}^{n_{L,\text{in}}}, r \in \mathbb{K}^{\log m}, y_i \in \mathbb{R}_{\mathbb{K}}, \{y_{i,j} \in \mathbb{R}_{\mathbb{A}}\}_{j \in [t]}; z_i \in \mathbb{L}^{n_L})_{i \in [K+k]}$ Output $\in \text{CE}(B, \mathcal{L})$ $(s; c \in \mathbb{C}, x \in \mathbb{L}^{n_{L,\text{in}}}, r \in \mathbb{K}^{\log m}, y \in \mathbb{R}_{\mathbb{K}}, \{y_j \in \mathbb{R}_{\mathbb{A}}\}_{j \in [t]}; z \in \mathbb{L}^{n_L})$ Setup $\mathcal{G}(1^\lambda, n_{\text{R}}) \rightarrow \text{pp}$: Output $\text{pp} \leftarrow \text{Setup}(1^\lambda, n_{\text{R}})$. Encoder $\mathcal{K}(\text{pp}, s) \rightarrow (\text{pk}, \text{vk})$: Output $((\text{pp}, s), \perp)$. Reduction $\langle \mathcal{P}, \mathcal{V} \rangle((\text{pk}, \text{vk}), u_1, w_1) \rightarrow (u_2; w_2)$: Recall from Definition 21 the matrix $\text{Pad} \in \mathbb{F}^{m \times n_{\mathbb{F}}}$. For every $i \in [K + k]$, define $\begin{aligned} z_i^b &:= \text{Flat}_{\mathbb{L}}(z_i), \\ z_i^{b, \text{pad}} &:= \text{Pad} z_i^b, \\ x_i &:= \text{Pack}_{\mathbb{L}}(x_i), \\ z_i &:= \text{Pack}_{\mathbb{L}}(z_i) = \text{Pack}_{\mathbb{F}}(z_i^b), \\ h_i &:= \text{Emb}_{\mathbb{F}}(\text{Trans}_{\mathbb{F}}(\text{Pad})) \cdot z_i. \end{aligned}$ For every $i \in [K + k]$ and $j \in [t]$, define $h_{i,j} := \text{Emb}_{\mathbb{L}}(\text{Trans}_{\mathbb{L}}(M_j)) \cdot z_i.$ 1. $\mathcal{V}$: Sample $\rho_1, \dots, \rho_{K+k} \xleftarrow{\\$} \mathcal{C}$ and compute: $\begin{aligned} c &\leftarrow \sum_{i \in [K+k]} \rho_i c_i, & x &\leftarrow \sum_{i \in [K+k]} \rho_i x_i, & x &\leftarrow \text{Pack}_{\mathbb{L}}^{-1}(x), \\ y &\leftarrow \sum_{i \in [K+k]} \rho_i y_i, & \forall j \in [t], & y_j &\leftarrow \sum_{i \in [K+k]} \rho_i y_{i,j}. \end{aligned}$ Send $\rho_1, \dots, \rho_{K+k}$ to $\mathcal{P}$. 2. $\mathcal{P}$: Compute $z \leftarrow \sum_{i \in [K+k]} \rho_i z_i, \quad z \leftarrow \text{Pack}_{\mathbb{L}}^{-1}(z).$ 3. Output $(s; c, x, r, y, \{y_j\}_{j \in [t]}; z)$.

Lemma 8 (Π_{RLC} is weak). *The interactive reduction $\Pi_{\text{RLC}} : \text{CE}(b, \mathcal{L})^{K+k} (\text{CE}(q/2, \mathcal{L})^{K+k}) \rightarrow \text{CE}(B, \mathcal{L})$ is **weak** (Definition 17) for the function ϕ , which projects commitments $(c_i)_{i \in [K+k]}$ from the instance.*

Proof. For brevity, we defer the proof to Appendix B.3. □

7.5 Decomposition reduction Π_{DEC}

Inspired by folklore techniques [12, 15, 72], our final reduction aims to reduce the norm of the flattened witness vector from $B = b^k$ to b , which will allow us to continually fold CCS claims without increasing the norms of the flattened representations z_i^b of the openings $(z_i)_i$ to the commitments. The prover applies the signed base b decomposition (Definition 1) to z^b to obtain lower norm vectors $z_1^b, \dots, z_k^b$, and recovers the corresponding native vectors $z_i := \text{Flat}_{\mathbb{L}}^{-1}(z_i^b)$. The prover then sends commitments, evaluations of the flattened witnesses in $\mathbb{R}_{\mathbb{K}}$, and relation evaluations in $\mathbb{R}_{\mathbb{A}}$. The verifier checks that the commitments and both evaluation families recombine to the original claim.

Decomposition reduction Π_{DEC}
Parameters: Refer to Definition 22. Input $\in \text{CE}(B, \mathcal{L})$ $(s; \quad c \in \mathbb{C}, x \in \mathbb{L}^{n_{L,\text{in}}}, r \in \mathbb{K}^{\log m}, y \in \mathbb{R}_{\mathbb{K}}, \{y_j \in \mathbb{R}_{\mathbb{A}}\}_{j \in [t]}; \quad z \in \mathbb{L}^{n_L})$ Output $\in \text{CE}(b, \mathcal{L})^k$ $(s; \quad c_i \in \mathbb{C}, x_i \in \mathbb{L}^{n_{L,\text{in}}}, r \in \mathbb{K}^{\log m}, y_i \in \mathbb{R}_{\mathbb{K}}, \{y_{i,j} \in \mathbb{R}_{\mathbb{A}}\}_{j \in [t]}; \quad z_i \in \mathbb{L}^{n_L})_{i \in [k]}$ Setup $\mathcal{G}(1^\lambda, n_R) \rightarrow \text{pp}$: Output $\text{pp} \leftarrow \text{Setup}(1^\lambda, n_R)$. Encoder $\mathcal{K}(\text{pp}, s) \rightarrow (\text{pk}, \text{vk})$: Output $((\text{pp}, s), \perp)$. Reduction $\langle \mathcal{P}, \mathcal{V} \rangle((\text{pk}, \text{vk}), u_1, w_1) \rightarrow (u_2; w_2)$: 1. $\mathcal{P}$: Compute $(c_i, y_i, \{y_{i,j}\}_{j \in [t]}; z_i)_{i \in [k]}$ as follows. Recall from Definition 21 the matrix $\text{Pad} \in \mathbb{F}^{m \times n_F}$. First, compute $z^b \leftarrow \text{Flat}_{\mathbb{L}}(z), \quad (z_1^b, \dots, z_k^b) \leftarrow \text{split}_b(z^b).$ For every $i \in [k]$, compute $\begin{aligned} z_i &\leftarrow \text{Flat}_{\mathbb{L}}^{-1}(z_i^b), \\ z_i &\leftarrow \text{Pack}_{\mathbb{L}}(z_i) = \text{Pack}_{\mathbb{F}}(z_i^b), \\ h_i &\leftarrow \text{Emb}_{\mathbb{F}}(\text{Trans}_{\mathbb{F}}(\text{Pad})) \cdot z_i, \\ c_i &\leftarrow \mathcal{L}(z_i), \\ y_i &\leftarrow \widetilde{h}_i(r), \\ \forall j \in [t], \quad h_{i,j} &\leftarrow \text{Emb}_{\mathbb{L}}(\text{Trans}_{\mathbb{L}}(M_j)) \cdot z_i, \\ \forall j \in [t], \quad y_{i,j} &\leftarrow \widetilde{h}_{i,j}(r). \end{aligned}$ Send $(c_i, y_i, \{y_{i,j}\}_{j \in [t]})_{i \in [k]}$ to $\mathcal{V}$. 2. $\mathcal{V}$: Compute $x^b \leftarrow \text{Flat}_{\mathbb{L}}(x)$. If $\ x^b\ _{\infty} \geq B$, output $\perp$. Compute $(x_1^b, \dots, x_k^b) \leftarrow \text{split}_b(x^b), \quad x_i \leftarrow \text{Flat}_{\mathbb{L}}^{-1}(x_i^b) \quad \text{for every } i \in [k].$ Check the following equations, $c \stackrel{?}{=} \sum_{i \in [k]} b^{i-1} \cdot c_i, \quad y \stackrel{?}{=} \sum_{i \in [k]} b^{i-1} \cdot y_i, \quad \forall j \in [t], \quad y_j \stackrel{?}{=} \sum_{i \in [k]} b^{i-1} \cdot y_{i,j}.$ where b is treated as a constant element of $\mathbb{R}_{\mathbb{F}} \subseteq \mathbb{R}_{\mathbb{K}}, \mathbb{R}_{\mathbb{A}}$. If any check fails, the verifier outputs $\perp$. 3. Output $(s; \quad c_i, x_i, r, y_i, \{y_{i,j}\}_{j \in [t]}; \quad z_i)_{i \in [k]}$

Theorem 13. $\Pi_{\text{DEC}}: \text{CE}(B, \mathcal{L}) \rightarrow \text{CE}(b, \mathcal{L})^k$ is a **reduction of knowledge** (Definition 9).

Proof. For brevity, we defer the proof to Appendix B.4. □

8 Concrete parameters

This section provides three efficient parameterizations over ≤ 64 -bit fields. For all three parameterizations, we set $\tau = 1$, so that $\mathbb{L} = \mathbb{F}$, and take $\mathbb{K} := \mathbb{F}_{q^2}$ and $\mathbb{A} := \mathbb{K}$. Additionally, Appendix B.5 and Appendix B.6 provide the corresponding sage scripts that we used to determine valid parameterizations. In Definition 22, we require the commitment scheme to be $(2B, \mathcal{C})$ -relaxed binding (Definition 7). Thus, we need the commitment scheme to be $4TB$ -binding (Definition 7). Finally, Ajtai’s commitment scheme is $4TB$ -binding if $\text{MSIS}_{n_R, 8TB}^{\infty, \kappa, q}$ is hard. We estimate the hardness of Module-SIS using the lattice estimator library provided by [4] using our script (Appendix B.6), and compute the invertibility bounds that determine challenge set size using Appendix B.5.

8.1 Almost Goldilocks: $(2^{64} - 2^{32} + 1) - 32$

We provide a new field, which we refer to as *Almost Goldilocks*. This field’s order is $q = (2^{64} - 2^{32} + 1) - 32$, which is close to the order of the Goldilocks field $2^{64} - 2^{32} + 1$. Because of this, the field admits an efficient implementation with a small change to the Solinas prime reduction algorithm (which is typically used for the Goldilocks field).

$\eta = 128$, $\Phi = X^{64} + 1$, $d = 64$, $R_{\mathbb{F}} := \mathbb{F}[X]/(\Phi)$, $\kappa = 15$, $m = n_L = n_F = 2^{33}$, $n_R = 2^{27}$, $n_{\text{pad}} = 0$, $b = 2$, $k = 13$, $K \in [50]$, $B = 2^{13}$. Define $\mathcal{C}$ to be the set polynomials in $R_{\mathbb{F}}$ whose coefficients belong to $[-1, 0, 1, 2]$. By Theorem 5, $T = 128$. By Theorem 4, $b_{\text{inv}} \approx 4$.

$|\mathcal{C}| = 2^{128}$, $|\mathbb{K}| \approx 2^{128}$, $\text{MSIS}_{n_R, 8TB}^{\infty, \kappa, q} \approx 129$ bits of security.

8.2 Goldilocks: $(2^{64} - 2^{32} + 1)$

This is a popular choice of field for SNARKs as the field admits an efficient implementation: field operations can be implemented with essentially only bit-shifts and the field has high 2-adicity ($2^{32} \mid (p - 1)$), which is useful for compressing Neo and SuperNeo IVC proofs with SNARKs.

$\eta = 81$, $\Phi = X^{54} + X^{27} + 1$, $d = 54$, $R_{\mathbb{F}} := \mathbb{F}[X]/(\Phi)$, $\kappa = 18$, $m = 2^{30}$, $n_L = n_F = 2^{30} - 46$, $n_R = 19,884,107$, $n_{\text{pad}} = 46$, $b = 2$, $k = 14$, $K \in [61]$, $B = 2^{14}$. Define $\mathcal{C}$ to be the set polynomials in $R_{\mathbb{F}}$ whose coefficients belong to $[-2, -1, 0, 1, 2]$. By Theorem 5, $T = 216$. By Theorem 4, $b_{\text{inv}} \approx 2.5 \cdot 10^9$.

$|\mathcal{C}| \approx 2^{125}$, $|\mathbb{K}| \approx 2^{128}$, $\text{MSIS}_{n_R, 8TB}^{\infty, \kappa, q} \approx 129$ bits of security.

Remark 5 (Incompatibility with LatticeFold [14]). In LatticeFold [14], the constructions and analysis are limited to power-of-two cyclotomic polynomials, namely of the form $X^d + 1$ with d being a power-of-two. Since the Goldilocks field has high 2-adicity, the cyclotomic polynomial completely factors into linear terms. This means that the ring $R_{\mathbb{F}}$ is isomorphic to $\mathbb{F}_q^d$ (the NTT representation). The security of LatticeFold’s construction depends on the size of the field in the NTT representation [14, Sec 3.3], which here is only 64 bits.

8.3 Mersenne 61: $2^{61} - 1$

This field admits an incredibly efficient implementation as it is only one off from a power-of-two. Specifically, modular arithmetic over this field can be implemented with simple bit-shifts with an algorithm more efficient than Goldilocks.

$\eta = 81$, $\Phi = X^{54} + X^{27} + 1$, $d = 54$, $R_{\mathbb{F}} := \mathbb{F}[X]/(\Phi)$, $\kappa = 18$, $m = 2^{28}$, $n_L = n_F = 2^{28} - 52$, $n_R = 4,971,026$, $n_{\text{pad}} = 52$, $b = 2$, $k = 14$, $K \in [61]$, $B = 2^{14}$. Define $\mathcal{C}$ to be the set polynomials in $R_{\mathbb{F}}$ whose coefficients belong to $[-2, -1, 0, 1, 2]$. By Theorem 5, $T = 216$. By Theorem 4, $b_{\text{inv}} \approx 383$.

$|\mathcal{C}| \approx 2^{125}$, $|\mathbb{K}| \approx 2^{122}$, $\text{MSIS}_{n_R, 8TB}^{\infty, \kappa, q} \approx 129$ bits of security.

Remark 6 (Incompatibility with LatticeFold [14]). As stated earlier, LatticeFold’s constructions and analysis are limited to power-of-two cyclotomic polynomials, namely of the form $X^d + 1$ for d being a power-of-two. For Mersenne 61, there is no choice of power-of-two cyclotomic polynomials, which satisfies the requirements of Theorem 4. Hence, it cannot be determined whether a choice of parameters with $\Phi = X^d + 1$ leads to a secure construction.

References

- [1] Aardal, M.A., Aranha, D.F., Boudgoust, K., Kolby, S., Takahashi, A.: Aggregating falcon signatures with LaBRADOR. In: Reyzin, L., Stebila, D. (eds.) *Advances in Cryptology – CRYPTO 2024, Part I*. Lecture Notes in Computer Science, vol. 14920, pp. 71–106. Springer, Cham, Switzerland, Santa Barbara, CA, USA (Aug 18–22, 2024). https://doi.org/10.1007/978-3-031-68376-3_3
- [2] Ajtai, M.: Generating hard instances of lattice problems (extended abstract). In: 28th Annual ACM Symposium on Theory of Computing. pp. 99–108. ACM Press, Philadelphia, PA, USA (May 22–24, 1996). <https://doi.org/10.1145/237814.237838>
- [3] Albrecht, M.R., Lai, R.W.F.: Subtractive sets over cyclotomic rings - limits of Schnorr-like arguments over lattices. In: Malkin, T., Peikert, C. (eds.) *Advances in Cryptology – CRYPTO 2021, Part II*. Lecture Notes in Computer Science, vol. 12826, pp. 519–548. Springer, Cham, Switzerland, Virtual Event (Aug 16–20, 2021). https://doi.org/10.1007/978-3-030-84245-1_18
- [4] Albrecht, M.R., Player, R., Scott, S.: On the concrete hardness of learning with errors. *Journal of Mathematical Cryptology* **9**(3), 169–203 (2015)
- [5] Alkim, E., Ducas, L., Pöppelmann, T., Schwabe, P.: Post-quantum key exchange - a new hope. *Cryptology ePrint Archive*, Report 2015/1092 (2015), <https://eprint.iacr.org/2015/1092>
- [6] Arnon, G., Chiesa, A., Fenzi, G., Yogev, E.: WHIR: Reed-solomon proximity testing with super-fast verification. *Cryptology ePrint Archive*, Report 2024/1586 (2024), <https://eprint.iacr.org/2024/1586>
- [7] Attema, T., Cramer, R., Kohl, L.: A compressed Σ -protocol theory for lattices. In: Malkin, T., Peikert, C. (eds.) *Advances in Cryptology – CRYPTO 2021, Part II*. Lecture Notes in Computer Science, vol. 12826, pp. 549–579. Springer, Cham, Switzerland, Virtual Event (Aug 16–20, 2021). https://doi.org/10.1007/978-3-030-84245-1_19
- [8] Attema, T., Klooß, M., Lai, R.W.F., Yatsyna, P.: Adaptive special soundness: Improved knowledge extraction by adaptive useful challenge sampling. *Cryptology ePrint Archive*, Report 2024/2038 (2024), <https://eprint.iacr.org/2024/2038>
- [9] Attema, T., Lyubashevsky, V., Seiler, G.: Practical product proofs for lattice commitments. In: Micciancio, D., Ristenpart, T. (eds.) *Advances in Cryptology – CRYPTO 2020, Part II*. Lecture Notes in Computer Science, vol. 12171, pp. 470–499. Springer, Cham, Switzerland, Santa Barbara, CA, USA (Aug 17–21, 2020). https://doi.org/10.1007/978-3-030-56880-1_17
- [10] Ben-Sasson, E., Bentov, I., Horesh, Y., Riabzev, M.: Scalable zero knowledge with no trusted setup. In: *CRYPTO* (2019)
- [11] Ben-Sasson, E., Chiesa, A., Tromer, E., Virza, M.: Scalable zero knowledge via cycles of elliptic curves. In: *CRYPTO* (2014)
- [12] Beullens, W., Seiler, G.: LaBRADOR: Compact proofs for R1CS from module-SIS. In: Handschuh, H., Lysyanskaya, A. (eds.) *Advances in Cryptology – CRYPTO 2023, Part V*. Lecture Notes in Computer Science, vol. 14085, pp. 518–548. Springer, Cham, Switzerland, Santa Barbara, CA, USA (Aug 20–24, 2023). https://doi.org/10.1007/978-3-031-38554-4_17
- [13] Bitansky, N., Canetti, R., Chiesa, A., Tromer, E.: Recursive composition and bootstrapping for SNARKs and proof-carrying data. In: *STOC* (2013)
- [14] Boneh, D., Chen, B.: LatticeFold: A lattice-based folding scheme and its applications to succinct proof systems. *Cryptology ePrint Archive*, Paper 2024/257 (2024)
- [15] Boneh, D., Chen, B.: LatticeFold: A lattice-based folding scheme and its applications to succinct proof systems. In: Hanaoka, G., Yang, B.Y. (eds.) *Advances in Cryptology – ASIACRYPT 2025, Part III*. Lecture Notes in Computer Science, vol. 16247, pp. 330–362. Springer, Singapore, Singapore, Melbourne, VIC, Australia (Dec 8–12, 2025). https://doi.org/10.1007/978-981-95-5099-9_11
- [16] Boneh, D., Chen, B.: LatticeFold+: Faster, simpler, shorter lattice-based folding for succinct proof systems. In: Kalai, Y.T., Kamara, S.F. (eds.) *Advances in Cryptology – CRYPTO 2025, Part VII*. Lecture Notes in Computer Science, vol. 16006, pp. 327–361. Springer, Cham, Switzerland, Santa Barbara, CA, USA (Aug 17–21, 2025). https://doi.org/10.1007/978-3-032-01907-3_11

- [17] Boneh, D., Chen, B.: LatticeFold+: Faster, simpler, shorter lattice-based folding for succinct proof systems. Cryptology ePrint Archive, Report 2025/247 (2025), <https://eprint.iacr.org/2025/247>
- [18] Boneh, D., Lynn, B., Shacham, H.: Short signatures from the Weil pairing. In: Advances in Cryptology—ASIACRYPT 2001 (2001)
- [19] Bootle, J., Lyubashevsky, V., Seiler, G.: Algebraic techniques for short(er) exact lattice-based zero-knowledge proofs. In: Boldyreva, A., Micciancio, D. (eds.) Advances in Cryptology – CRYPTO 2019, Part I. Lecture Notes in Computer Science, vol. 11692, pp. 176–202. Springer, Cham, Switzerland, Santa Barbara, CA, USA (Aug 18–22, 2019). https://doi.org/10.1007/978-3-030-26948-7_7
- [20] Bünz, B., Chen, B.: Protostar: Generic efficient accumulation/folding for special sound protocols. Cryptology ePrint Archive, Paper 2023/620 (2023)
- [21] Bünz, B., Chiesa, A., Fenzi, G., Wang, W.: Linear-time accumulation schemes. In: Applebaum, B., Lin, H.R. (eds.) TCC 2025: 23rd Theory of Cryptography Conference, Part I. Lecture Notes in Computer Science, vol. 16268, pp. 369–399. Springer, Cham, Switzerland, Aarhus, Denmark (Dec 1–5, 2025). https://doi.org/10.1007/978-3-032-12287-2_13
- [22] Bunz, B., Fenzi, G., Rothblum, R., Wang, W.: Tensorswitch: Nearly optimal polynomial commitments from tensor codes. Cryptology ePrint Archive, Paper 2025/2065 (2025), <https://eprint.iacr.org/2025/2065>, <https://eprint.iacr.org/2025/2065>
- [23] Bünz, B., Fisch, B., Szepieniec, A.: Transparent SNARKs from DARK compilers. In: EUROCRYPT (2020)
- [24] Bunz, B., Mishra, P., Nguyen, W., Wang, W.: Arc: Accumulation for reed–solomon codes. Cryptology ePrint Archive, Paper 2024/1731 (2024)
- [25] Bünz, B., Mishra, P., Nguyen, W., Wang, W.: Accumulation without homomorphism. Cryptology ePrint Archive, Paper 2024/474 (2024)
- [26] Chen, B.: Symphony: Scalable SNARKs in the random oracle model from lattice-based high-arity folding. Cryptology ePrint Archive, Report 2025/1905 (2025), <https://eprint.iacr.org/2025/1905>
- [27] Chen, B., Bünz, B., Boneh, D., Zhang, Z.: Hyperplonk: Plonk with linear-time prover and high-degree custom gates. In: EUROCRYPT (2023)
- [28] Chen, S., Cheon, J.H., Kim, D., Park, D.: Verifiable computing for approximate computation. Cryptology ePrint Archive, Report 2019/762 (2019), <https://eprint.iacr.org/2019/762>
- [29] Chen, W., Chiesa, A., Dauterman, E., Ward, N.P.: Reducing participation costs via incremental verification for ledger systems. Cryptology ePrint Archive, Report 2020/1522 (2020)
- [30] Chiesa, A., Hu, Y., Maller, M., Mishra, P., Vesely, N., Ward, N.: Marlin: Preprocessing zkSNARKs with universal and updatable SRS. In: EUROCRYPT (2020)
- [31] Chiesa, A., Ojha, D., Spooner, N.: Fractal: Post-quantum and transparent recursive proofs from holography. In: EUROCRYPT (2020)
- [32] Cini, V., Lai, R.W.F., Malavolta, G.: Lattice-based succinct arguments from vanishing polynomials - (extended abstract). In: Handschuh, H., Lysyanskaya, A. (eds.) Advances in Cryptology – CRYPTO 2023, Part II. Lecture Notes in Computer Science, vol. 14082, pp. 72–105. Springer, Cham, Switzerland, Santa Barbara, CA, USA (Aug 20–24, 2023). https://doi.org/10.1007/978-3-031-38545-2_3
- [33] Cini, V., Malavolta, G., Nguyen, N.K., Wee, H.: Polynomial commitments from lattices: Post-quantum security, fast verification and transparent setup. In: Reyzin, L., Stebila, D. (eds.) Advances in Cryptology – CRYPTO 2024, Part X. Lecture Notes in Computer Science, vol. 14929, pp. 207–242. Springer, Cham, Switzerland, Santa Barbara, CA, USA (Aug 18–22, 2024). https://doi.org/10.1007/978-3-031-68403-6_7
- [34] Coratger, T., Setty, S.: Post Quantum Signature Aggregation: a Folding Approach. <https://ethresear.ch/t/post-quantum-signature-aggregation-a-folding-approach/23639> (2025)
- [35] Eagen, L., Gabizon, A.: Protogalaxy: Efficient protostar-style folding of multiple instances. Cryptology ePrint Archive, Paper 2023/1106 (2023)
- [36] Esgin, M.F., Nguyen, N.K., Seiler, G.: Practical exact proofs from lattices: New techniques to exploit fully-splitting rings. In: Moriai, S., Wang, H. (eds.) Advances in Cryptology – ASIACRYPT 2020, Part II. Lecture Notes in Computer Science, vol. 12492, pp. 259–288. Springer, Cham, Switzerland, Daejeon, South Korea (Dec 7–11, 2020). https://doi.org/10.1007/978-3-030-64834-3_9

- [37] Fenzi, G., Knabenhans, C., Nguyen, N.K., Pham, D.T.: Lova: Lattice-based folding scheme from unstructured lattices. Cryptology ePrint Archive, Paper 2024/1964 (2024)
- [38] Fenzi, G., Knabenhans, C., Nguyen, N.K., Pham, D.T.: Lova: Lattice-based folding scheme from unstructured lattices. In: Chung, K.M., Sasaki, Y. (eds.) *Advances in Cryptology – ASIACRYPT 2024*, Part IV. Lecture Notes in Computer Science, vol. 15487, pp. 303–326. Springer, Singapore, Singapore, Kolkata, India (Dec 9–13, 2024). https://doi.org/10.1007/978-981-96-0894-2_10
- [39] Fenzi, G., Moghaddas, H., Nguyen, N.K.: Lattice-based polynomial commitments: Towards asymptotic and concrete efficiency. *Journal of Cryptology* **37**(3), 31 (Jul 2024). <https://doi.org/10.1007/s00145-024-09511-8>
- [40] Flynn, M.J.: Very high-speed computing systems. *Proceedings of the IEEE* **54**(12), 1901–1909 (2005)
- [41] Flynn, M.J.: Some computer organizations and their effectiveness. *IEEE transactions on computers* **100**(9), 948–960 (2009)
- [42] Garreta, A., Lipmaa, H., Luhaäär, U., Osadnik, M.: Cyclo: Lightweight lattice-based folding via partial range checks. Cryptology ePrint Archive (2026)
- [43] Gentry, C., Halevi, S., Lyubashevsky, V.: Practical non-interactive publicly verifiable secret sharing with thousands of parties. In: Dunkelman, O., Dziembowski, S. (eds.) *Advances in Cryptology – EUROCRYPT 2022*, Part I. Lecture Notes in Computer Science, vol. 13275, pp. 458–487. Springer, Cham, Switzerland, Trondheim, Norway (May 30 – Jun 3, 2022). https://doi.org/10.1007/978-3-031-06944-4_16
- [44] Grassi, L., Khovratovich, D., Rechberger, C., Roy, A., Schofnegger, M.: Poseidon: A new hash function for zero-knowledge proof systems. Cryptology ePrint Archive, Paper 2019/458 (2019)
- [45] Hülsing, A., Butin, D., Gazdag, S.L., Rijneveld, J., Mohaisen, A.: XMSS: eXtended Merkle signature scheme. RFC 8391 (2018)
- [46] Jyrkinen, K., Lai, R.W.F.: Vanishing short integer solution, revisited - reductions, trapdoors, homomorphic signatures for low-degree polynomials. In: Jager, T., Pan, J. (eds.) *PKC 2025: 28th International Conference on Theory and Practice of Public Key Cryptography*, Part II. Lecture Notes in Computer Science, vol. 15675, pp. 273–300. Springer, Cham, Switzerland, Røros, Norway (May 12–15, 2025). https://doi.org/10.1007/978-3-031-91823-0_9
- [47] Kate, A., Zaverucha, G.M., Goldberg, I.: Constant-size commitments to polynomials and their applications. In: *ASIACRYPT*. pp. 177–194 (2010)
- [48] Kilian, J.: A note on efficient zero-knowledge proofs and arguments (extended abstract). In: *STOC* (1992)
- [49] Klooß, M., Lai, R.W.F., Nguyen, N.K., Osadnik, M.: RoK, paper, SIsors toolkit for lattice-based succinct arguments - (extended abstract). In: Chung, K.M., Sasaki, Y. (eds.) *Advances in Cryptology – ASIACRYPT 2024*, Part V. Lecture Notes in Computer Science, vol. 15488, pp. 203–235. Springer, Singapore, Singapore, Kolkata, India (Dec 9–13, 2024). https://doi.org/10.1007/978-981-96-0935-2_7
- [50] Klooß, M., Lai, R.W.F., Nguyen, N.K., Osadnik, M.: RoK and roll - verifier-efficient random projection for $\tilde{O}(\lambda)$ -size lattice arguments - (extended abstract). In: Hanaoka, G., Yang, B.Y. (eds.) *Advances in Cryptology – ASIACRYPT 2025*, Part III. Lecture Notes in Computer Science, vol. 16247, pp. 297–329. Springer, Singapore, Singapore, Melbourne, VIC, Australia (Dec 8–12, 2025). https://doi.org/10.1007/978-981-95-5099-9_10
- [51] Kothapalli, A.: A Theory of Composition for Proofs of Knowledge. Ph.D. thesis, Carnegie Mellon University (2024)
- [52] Kothapalli, A., Parno, B.: Algebraic reductions of knowledge. In: Handschuh, H., Lysyanskaya, A. (eds.) *Advances in Cryptology – CRYPTO 2023*, Part IV. Lecture Notes in Computer Science, vol. 14084, pp. 669–701. Springer, Cham, Switzerland, Santa Barbara, CA, USA (Aug 20–24, 2023). https://doi.org/10.1007/978-3-031-38551-3_21
- [53] Kothapalli, A., Parno, B.: Algebraic reductions of knowledge. In: *CRYPTO* (2023)
- [54] Kothapalli, A., Setty, S.: SuperNova: Proving universal machine executions without universal circuits. Cryptology ePrint Archive (2022)
- [55] Kothapalli, A., Setty, S.: Cyclefold: Folding-scheme-based recursive arguments over a cycle of elliptic curves. Cryptology ePrint Archive, Paper 2023/1192 (2023), <https://eprint.iacr.org/2023/1192>, <https://eprint.iacr.org/2023/1192>

- [56] Kothapalli, A., Setty, S.: HyperNova: Recursive arguments for customizable constraint systems. In: CRYPTO (2024)
- [57] Kothapalli, A., Setty, S.: NeutronNova: Folding everything that reduces to zero-check. Cryptology ePrint Archive (2024)
- [58] Kothapalli, A., Setty, S., Tzialla, I.: Nova: Recursive Zero-Knowledge Arguments from Folding Schemes. In: CRYPTO (2022)
- [59] Kothapalli, A., Setty, S.T.V.: HyperNova: Recursive arguments for customizable constraint systems. In: Reyzin, L., Stebila, D. (eds.) *Advances in Cryptology – CRYPTO 2024, Part X. Lecture Notes in Computer Science*, vol. 14929, pp. 345–379. Springer, Cham, Switzerland, Santa Barbara, CA, USA (Aug 18–22, 2024). https://doi.org/10.1007/978-3-031-68403-6_11
- [60] Kuriyama, S., Lai, R., Osadnik, M., Tucci, L.: Salsaa - sumcheck-aided lattice-based succinct arguments and applications. Cryptology ePrint Archive, Paper 2025/2124 (2025), <https://eprint.iacr.org/2025/2124>, <https://eprint.iacr.org/2025/2124>
- [61] Langlois, A., Stehlé, D.: Worst-case to average-case reductions for module lattices. *Designs, Codes and Cryptography* **75**(3), 565–599 (2015). <https://doi.org/10.1007/s10623-014-9938-4>
- [62] Longa, P., Naehrig, M.: Speeding up the number theoretic transform for faster ideal lattice-based cryptography. In: Foresti, S., Persiano, G. (eds.) *CANS 16: 15th International Conference on Cryptology and Network Security. Lecture Notes in Computer Science*, vol. 10052, pp. 124–139. Springer, Cham, Switzerland, Milan, Italy (Nov 14–16, 2016). https://doi.org/10.1007/978-3-319-48965-0_8
- [63] Lund, C., Fortnow, L., Karloff, H., Nisan, N.: Algebraic methods for interactive proof systems. In: FOCS (Oct 1990)
- [64] Lyubashevsky, V., Micciancio, D.: Generalized compact Knapsacks are collision resistant. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) *ICALP 2006: 33rd International Colloquium on Automata, Languages and Programming, Part II. Lecture Notes in Computer Science*, vol. 4052, pp. 144–155. Springer Berlin Heidelberg, Germany, Venice, Italy (Jul 10–14, 2006). https://doi.org/10.1007/11787006_13
- [65] Lyubashevsky, V., Nguyen, N.K., Plançon, M.: Lattice-based zero-knowledge proofs and applications: Shorter, simpler, and more general. In: Dodis, Y., Shrimpton, T. (eds.) *Advances in Cryptology – CRYPTO 2022, Part II. Lecture Notes in Computer Science*, vol. 13508, pp. 71–101. Springer, Cham, Switzerland, Santa Barbara, CA, USA (Aug 15–18, 2022). https://doi.org/10.1007/978-3-031-15979-4_3
- [66] Lyubashevsky, V., Seiler, G.: Short, invertible elements in partially splitting cyclotomic rings and applications to lattice-based zero-knowledge proofs. In: Nielsen, J.B., Rijmen, V. (eds.) *Advances in Cryptology – EUROCRYPT 2018, Part I. Lecture Notes in Computer Science*, vol. 10820, pp. 204–224. Springer, Cham, Switzerland, Tel Aviv, Israel (Apr 29 – May 3, 2018). https://doi.org/10.1007/978-3-319-78381-9_8
- [67] Micali, S.: CS proofs. In: FOCS (1994)
- [68] Nethermind Research: Lattice-based operations performance report. <https://nethermind.notion.site/Latticefold-and-lattice-based-operations-performance-report-153360fc38d080ac930cdeeffed69559> (2025)
- [69] Nguyen, N.K., O’Rourke, G., Zhang, J.: Hachi: Efficient lattice-based multilinear polynomial commitments over extension fields. Cryptology ePrint Archive, Paper 2026/156 (2026), <https://eprint.iacr.org/2026/156>, <https://eprint.iacr.org/2026/156>
- [70] Nguyen, N.K., Seiler, G.: Greyhound: Fast polynomial commitments from lattices. In: Reyzin, L., Stebila, D. (eds.) *Advances in Cryptology – CRYPTO 2024, Part X. Lecture Notes in Computer Science*, vol. 14929, pp. 243–275. Springer, Cham, Switzerland, Santa Barbara, CA, USA (Aug 18–22, 2024). https://doi.org/10.1007/978-3-031-68403-6_8
- [71] Nguyen, W., Setty, S.: Neo: Lattice-based folding scheme for CCS over small fields and pay-per-bit commitments. Cryptology ePrint Archive, Report 2025/294 (2025), <https://eprint.iacr.org/2025/294>
- [72] Papamanthou, C., Shi, E., Tamassia, R., Yi, K.: Streaming authenticated data structures. In: Johansson, T., Nguyen, P.Q. (eds.) *Advances in Cryptology – EUROCRYPT 2013. Lecture Notes in Computer Science*, vol. 7881, pp. 353–370. Springer Berlin Heidelberg, Germany, Athens, Greece (May 26–30, 2013). https://doi.org/10.1007/978-3-642-38348-9_22
- [73] Pedersen, T.P.: Non-interactive and information-theoretic secure verifiable secret sharing. In: Feigenbaum, J. (ed.) *Advances in Cryptology – CRYPTO’91. Lecture Notes in Computer Science*, vol. 576,

- pp. 129–140. Springer Berlin Heidelberg, Germany, Santa Barbara, CA, USA (Aug 11–15, 1992). https://doi.org/10.1007/3-540-46766-1_9
- [74] Peikert, C., Rosen, A.: Efficient collision-resistant hashing from worst-case assumptions on cyclic lattices. In: Halevi, S., Rabin, T. (eds.) TCC 2006: 3rd Theory of Cryptography Conference. Lecture Notes in Computer Science, vol. 3876, pp. 145–166. Springer Berlin Heidelberg, Germany, New York, NY, USA (Mar 4–7, 2006). https://doi.org/10.1007/11681878_8
 - [75] Polygon Zero Team: Plonky2: Fast recursive arguments with PLONK and FRI (2022), <https://docs.rs/crate/plonky2/latest/source/plonky2.pdf>, <https://docs.rs/crate/plonky2/latest/source/plonky2.pdf>
 - [76] Regev, O.: Lattice-based cryptography. In: Annual International Cryptology Conference. pp. 131–141. Springer (2006)
 - [77] Schwartz, J.T.: Fast probabilistic algorithms for verification of polynomial identities. J. ACM **27**(4) (1980)
 - [78] Seiler, G.: Faster avx2 optimized ntt multiplication for ring-lwe lattice cryptography. Cryptology ePrint Archive (2018)
 - [79] Setty, S.: Spartan: Efficient and general-purpose zkSNARKs without trusted setup. In: CRYPTO (2020)
 - [80] Setty, S., Thaler, J., Wahby, R.: Customizable constraint systems for succinct arguments. Cryptology ePrint Archive (2023)
 - [81] Shor, P.W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. vol. 26, pp. 1484–1509 (1997)
 - [82] Thaler, J.: The sum-check protocol. <https://people.cs.georgetown.edu/jthaler/sumcheck.pdf> (Sep 2017)
 - [83] Valiant, P.: Incrementally verifiable computation or proofs of knowledge imply time/space efficiency. In: TCC. pp. 552–576 (2008)
 - [84] Zeilberger, H., Chen, B., Fisch, B.: BaseFold: Efficient field-agnostic polynomial commitment schemes from foldable codes. In: Reyzin, L., Stebila, D. (eds.) Advances in Cryptology – CRYPTO 2024, Part X. Lecture Notes in Computer Science, vol. 14929, pp. 138–169. Springer, Cham, Switzerland, Santa Barbara, CA, USA (Aug 18–22, 2024). https://doi.org/10.1007/978-3-031-68403-6_5
 - [85] Zhao, J., Setty, S.T.V., Cui, W., Zaverucha, G.: MicroNova: Folding-based arguments with efficient (on-chain) verification. In: Blanton, M., Enck, W., Nita-Rotaru, C. (eds.) 2025 IEEE Symposium on Security and Privacy. pp. 1964–1982. IEEE Computer Society Press, San Francisco, CA, USA (May 12–15, 2025). <https://doi.org/10.1109/SP61157.2025.00168>
 - [86] Zhou, Z., Zhang, Z., Dong, J.: Proof-carrying data from multi-folding schemes. Cryptology ePrint Archive, Paper 2023/1282 (2023)

Supplementary Material

A AI Disclaimer

Portions of this manuscript were edited with the assistance of an AI writing tool (Github copilot), which was used to improve grammar, wording, and formatting consistency. All technical content-including definitions, theorems, and proofs-was produced and verified by the authors, who take full responsibility for the correctness and originality of the work.

B Deferred theorems and proofs

B.1 Proof of Composition Theorem (Theorem 12)

Proof. Consider an arbitrary expected polynomial time adversary $(\mathcal{A}, \mathcal{P}^*)$ for the composition $\Pi := \Pi_2 \circ \Pi_1$ with success probability $\epsilon(\mathcal{A}, \mathcal{P}^*) \geq 1/\text{poly}(\lambda)$. Without loss of generality, the adversary $\mathcal{P}^*$ can be split into two adversaries $(\mathcal{P}_1^*, \mathcal{P}_2^*)$ such that given $\text{pp} \leftarrow \mathcal{G}(1^\lambda, \text{sz})$, $(s, u_1, \text{st}_1) \leftarrow \mathcal{A}(\text{pp})$, and $(\text{pk}, \text{vk}) \leftarrow \mathcal{K}(\text{pp}, s)$,

- $\langle \mathcal{P}_1^*, \mathcal{V}_1 \rangle((\text{pk}, \text{vk}), u_1, \text{st}_1) \rightarrow (u_2, \text{st}_2)$
- $\langle \mathcal{P}_2^*, \mathcal{V}_2 \rangle((\text{pk}, \text{vk}), u_2, \text{st}_2) \rightarrow (u_3, w_3)$

Furthermore, we assume that $\mathcal{A}$ outputs st_1 which contains (s, pp, u_1) ; otherwise, we could trivially construct an adversary $\mathcal{A}'$ with an identical distribution of prior outputs that does so. First, we construct an adversary $\mathcal{A}_2 := (\mathcal{B}_2, \mathcal{B}_2')$ for Π_2 :

$\mathcal{B}_2(\text{pp}) \rightarrow (s, \text{st}_1)$:

1. $(s, u_1, \text{st}_1) \leftarrow \mathcal{A}(\text{pp})$.
2. Output (s, st_1) .

$\mathcal{B}_2'(\text{st}_1) \rightarrow (u_2, \text{st}_2)$:

1. Parse st_1 to obtain (s, pp, u_1) .
2. $(\text{pk}, \text{vk}) \leftarrow \mathcal{K}(\text{pp}, s)$.
3. Simulate $(u_2, \text{st}_2) \leftarrow \langle \mathcal{P}_1^*, \mathcal{V}_1 \rangle((\text{pk}, \text{vk}), u_1, \text{st}_1)$.
4. Output (u_2, st_2) .

$\mathcal{A}_2(\text{pp}) \rightarrow (s, u_2, \text{st}_2)$:

1. $(s, \text{st}_1) \leftarrow \mathcal{B}_2(\text{pp})$.
2. $(u_2, \text{st}_2) \leftarrow \mathcal{B}_2'(\text{st}_1)$.
3. Output (s, u_2, st_2) .

The adversary $\mathcal{A}_2$ runs in expected polynomial time since $\mathcal{A}$ and $\mathcal{P}_1^*$ do, and all remaining computations run in polynomial time. Observe that, by construction, the success probability $\epsilon(\mathcal{A}_2, \mathcal{P}_2^*)$ of adversary $(\mathcal{A}_2, \mathcal{P}_2^*)$ for Π_2 is equal to the success probability $\epsilon(\mathcal{A}, \mathcal{P}^*)$ of adversary $(\mathcal{A}, \mathcal{P}^*)$ for Π . By construction, $\mathcal{B}_2$ samples the input and state according to $\mathcal{A}$, while each invocation of $\mathcal{B}_2'$ simulates one execution of $\langle \mathcal{P}_1^*, \mathcal{V}_1 \rangle$ on that input and state. Thus, the distribution of (u_2, u_2') in the following experiment is identical to that in condition (i) for Π_1 with respect to adversary $(\mathcal{A}, \mathcal{P}_1^*)$. By assumption, Π_1 is a strong interactive reduction (Definition 18). As such, it satisfies condition (i); thus, we must have

$$\Pr \left[\begin{array}{c|c} u_2, u_2' \neq \perp & \text{pp} \leftarrow \mathcal{G}(1^\lambda, \text{sz}) \\ \downarrow & (s, \text{st}_1) \leftarrow \mathcal{B}_2(\text{pp}) \\ \phi(u_2) = \phi(u_2') & (u_2, \text{st}_2) \leftarrow \mathcal{B}_2'(\text{st}_1) \\ & (u_2', \text{st}_2') \leftarrow \mathcal{B}_2'(\text{st}_1) \end{array} \right] = 1, \quad (1)$$

By assumption, Π_2 is a weak interactive reduction (Definition 17). Thus, with respect to adversary $(\mathcal{A}_2, \mathcal{P}_2^*)$, there exists an expected polynomial time extractor $\mathcal{E}_2$ such that, since $\epsilon(\mathcal{A}_2, \mathcal{P}_2^*) = \epsilon(\mathcal{A}, \mathcal{P}^*) \geq 1/\text{poly}(\lambda)$,

$$\Pr \left[\begin{array}{c|c} (\text{pp}, s, u_2, w_2) \in \mathcal{R}_2' & \text{pp} \leftarrow \mathcal{G}(1^\lambda, \text{sz}) \\ & (s, u_2, \text{st}_2) \leftarrow \mathcal{A}_2(\text{pp}) \\ & (\text{pk}, \text{vk}) \leftarrow \mathcal{K}(\text{pp}, s) \\ & w_2 \leftarrow \mathcal{E}_2(\text{pp}, s, u_2, \text{st}_2) \end{array} \right] \geq \epsilon(\mathcal{A}, \mathcal{P}^*) - \text{negl}(\lambda) \quad (2)$$

Moreover, since $\mathcal{A}_2 = (\mathcal{B}_2, \mathcal{B}'_2)$ satisfies (1), the same extractor $\mathcal{E}_2$ satisfies condition (ii) of weak interactive reductions (Definition 17). Let $\mathcal{O}_{\text{weak}}$ be the oracle from Definition 17 instantiated with $\mathcal{B}'_2$ and $\mathcal{E}_2$. Thus, for any uniqueness adversary $\mathcal{W}_{\text{uniq}}^{\mathcal{O}_{\text{weak}}}$,

$$\Pr \left[\begin{array}{l} w_2, w'_2 \neq \perp \\ \wedge w_2 \neq w'_2 \end{array} \middle| \begin{array}{l} \text{pp} \leftarrow \mathcal{G}(1^\lambda, \text{sz}) \\ (\text{s}, \text{st}_1) \leftarrow \mathcal{B}_2(\text{pp}) \\ \text{in} \leftarrow (\text{pp}, \text{s}, \text{st}_1) \\ (u_2, w_2), (u'_2, w'_2) \leftarrow \mathcal{W}_{\text{uniq}}^{\mathcal{O}_{\text{weak}}(\text{in})}(\text{in}) \end{array} \right] \leq \text{negl}(\lambda) \quad (3)$$

Next, we will construct an adversary $\mathcal{P}_1^{**}$ for Π_1 . Recall that $\mathcal{V}_1$ is the verifier of Π_1 . During the interaction $\langle \mathcal{P}_1^{**}, \mathcal{V}_1 \rangle$, $\mathcal{P}_1^{**}$ internally runs $\mathcal{P}_1^*$ and relays messages between $\mathcal{P}_1^*$ and $\mathcal{V}_1$:

$\mathcal{P}_1^{**}(\text{pk}, u_1, \text{st}_1) \rightarrow w_2$:

1. Parse st_1 to obtain (s, pp) .
2. $(\text{pk}, \text{vk}) \leftarrow \mathcal{K}(\text{pp}, \text{s})$.
3. Run $\mathcal{P}_1^*(\text{pk}, u_1, \text{st}_1)$, relaying each message from $\mathcal{V}_1$ to $\mathcal{P}_1^*$ and each response from $\mathcal{P}_1^*$ back to $\mathcal{V}_1$. Let tr be the resulting transcript and st_2 the final state of $\mathcal{P}_1^*$.
4. Since Π_1 is public coin, compute the verifier's output u_2 by emulating $\mathcal{V}_1(\text{vk}, u_1)$ on tr .
5. $w_2 \leftarrow \mathcal{E}_2(\text{pp}, \text{s}, u_2, \text{st}_2)$.
6. Output w_2 .

The adversary $\mathcal{P}_1^{**}$ runs in expected polynomial time since $\mathcal{P}_1^*$ and $\mathcal{E}_2$ do, and all remaining computations run in polynomial time. Conditioned on $(\text{pp}, \text{s}, u_1, \text{st}_1)$, $\mathcal{B}'_2(\text{st}_1)$ and the interaction $\langle \mathcal{P}_1^{**}, \mathcal{V}_1 \rangle$ generate (u_2, st_2) according to the same distribution: both run $\mathcal{P}_1^*$ against $\mathcal{V}_1$ using the same keys produced by the deterministic encoder. Moreover, since Π_1 is public coin, $\mathcal{P}_1^{**}$ can reconstruct from tr the output u_2 produced by $\mathcal{V}_1$. Hence, the event in (2) is exactly the event defining $\epsilon'(\mathcal{A}, \mathcal{P}_1^{**})$, and therefore $\epsilon'(\mathcal{A}, \mathcal{P}_1^{**}) \geq \epsilon(\mathcal{A}, \mathcal{P}^*) - \text{negl}(\lambda) \geq 1/\text{poly}(\lambda)$.

Next, consider any uniqueness adversary $\mathcal{S}_{\text{uniq}}^{\mathcal{O}_{\text{str}}}$ (Definition 16), where $\mathcal{O}_{\text{str}}$ is the oracle from condition (ii) of strong interactive reductions (Definition 18) instantiated with $\mathcal{P}_1^{**}$. Recall that st_1 contains $(\text{s}, \text{pp}, u_1)$. Define $\text{in}_{\text{weak}} := (\text{pp}, \text{s}, \text{st}_1)$ and $\text{in}_{\text{str}} := (\text{pp}, \text{s}, u_1, \text{st}_1)$, where u_1 is parsed from st_1 . Define

$$\mathcal{W}_{\text{uniq}}^{\mathcal{O}_{\text{weak}}(\text{in}_{\text{weak}})}(\text{in}_{\text{weak}}) := \mathcal{S}_{\text{uniq}}^{\mathcal{O}_{\text{weak}}(\text{in}_{\text{weak}})}(\text{in}_{\text{str}}).$$

In particular, $\mathcal{W}_{\text{uniq}}$ runs the code of $\mathcal{S}_{\text{uniq}}$, but answers each oracle query using $\mathcal{O}_{\text{weak}}(\text{in}_{\text{weak}})$. By definition,

$$\begin{array}{ll} \mathcal{O}_{\text{weak}}(\text{in}_{\text{weak}}) \rightarrow (u_2, w_2) : & \mathcal{O}_{\text{str}}(\text{in}_{\text{str}}) \rightarrow (u_2, w_2) : \\ \begin{array}{l} 1. (u_2, \text{st}_2) \leftarrow \mathcal{B}'_2(\text{st}_1). \\ 2. w_2 \leftarrow \mathcal{E}_2(\text{pp}, \text{s}, u_2, \text{st}_2). \\ 3. \text{Output } (u_2, w_2). \end{array} & \begin{array}{l} 1. (\text{pk}, \text{vk}) \leftarrow \mathcal{K}(\text{pp}, \text{s}). \\ 2. (u_2, w_2) \leftarrow \langle \mathcal{P}_1^{**}, \mathcal{V}_1 \rangle((\text{pk}, \text{vk}), u_1, \text{st}_1). \\ 3. \text{Output } (u_2, w_2). \end{array} \end{array}$$

By construction, $\mathcal{B}'_2$ runs $\mathcal{P}_1^*$ against $\mathcal{V}_1$. The prover $\mathcal{P}_1^{**}$ runs the same interaction, reconstructs u_2 , and runs $\mathcal{E}_2$ on $(\text{pp}, \text{s}, u_2, \text{st}_2)$. Since the encoder is deterministic, the two oracles have the same output distribution. Since each oracle call uses fresh independent randomness, the view and output of $\mathcal{S}_{\text{uniq}}$ have the same distribution under either oracle. Furthermore, $\mathcal{W}_{\text{uniq}}^{\mathcal{O}_{\text{weak}}}$ runs in expected polynomial time. Since $\mathcal{S}_{\text{uniq}}^{\mathcal{O}_{\text{str}}}$ is EPT, its computation outside oracle calls and its expected number of oracle calls are polynomial. The adversary $\mathcal{W}_{\text{uniq}}$ runs the same computation but replaces each call to $\mathcal{O}_{\text{str}}$ with one call to $\mathcal{O}_{\text{weak}}$. By the definitions of $\mathcal{B}'_2$ and $\mathcal{P}_1^{**}$, each replacement runs the same interaction between $\mathcal{P}_1^*$ and $\mathcal{V}_1$ followed by $\mathcal{E}_2$, with only polynomial time differences. Thus, $\mathcal{W}_{\text{uniq}}^{\mathcal{O}_{\text{weak}}}$ is EPT. Thus, the following experiment has the same distribution as the experiment in (3); by (3), we must have

$$\Pr \left[\begin{array}{l} w_2, w'_2 \neq \perp \\ \wedge \\ w_2 \neq w'_2 \end{array} \middle| \begin{array}{l} \text{pp} \leftarrow \mathcal{G}(1^\lambda, \text{sz}) \\ (\text{s}, u_1, \text{st}_1) \leftarrow \mathcal{A}(\text{pp}) \\ \text{in}_{\text{str}} \leftarrow (\text{pp}, \text{s}, u_1, \text{st}_1) \\ (u_2, w_2), (u'_2, w'_2) \leftarrow \mathcal{S}_{\text{uniq}}^{\mathcal{O}_{\text{str}}(\text{in}_{\text{str}})}(\text{in}_{\text{str}}) \end{array} \right] \leq \text{negl}(\lambda) \quad (4)$$

Since $\mathcal{S}_{\text{uniq}}$ was arbitrary, (4) holds for any uniqueness adversary. Thus, by (4), the bound $\epsilon'(\mathcal{A}, \mathcal{P}_1^{**}) \geq 1/\text{poly}(\lambda)$ established above, and condition (ii) of Definition 18 for Π_1 , there exists an expected polynomial time extractor $\mathcal{E}_1$ such that

$$\Pr \left[(\text{pp}, \text{s}, u_1, w_1) \in \mathcal{R}_1 \left| \begin{array}{l} \text{pp} \leftarrow \mathcal{G}(1^\lambda, \text{sz}) \\ (\text{s}, u_1, \text{st}_1) \leftarrow \mathcal{A}(\text{pp}) \\ (\text{pk}, \text{vk}) \leftarrow \mathcal{K}(\text{pp}, \text{s}) \\ w_1 \leftarrow \mathcal{E}_1(\text{pp}, \text{s}, u_1, \text{st}_1) \end{array} \right. \right] \geq \epsilon'(\mathcal{A}, \mathcal{P}_1^{**}) - \text{negl}(\lambda) \quad (5)$$

By (2) and the preceding argument,

$$\epsilon'(\mathcal{A}, \mathcal{P}_1^{**}) \geq \epsilon(\mathcal{A}, \mathcal{P}^*) - \text{negl}(\lambda).$$

Therefore, combining these inequalities and absorbing the sum of the two negligible functions into $\text{negl}(\lambda)$, we have

$$\Pr \left[(\text{pp}, \text{s}, u_1, w_1) \in \mathcal{R}_1 \left| \begin{array}{l} \text{pp} \leftarrow \mathcal{G}(1^\lambda, \text{sz}) \\ (\text{s}, u_1, \text{st}_1) \leftarrow \mathcal{A}(\text{pp}) \\ (\text{pk}, \text{vk}) \leftarrow \mathcal{K}(\text{pp}, \text{s}) \\ w_1 \leftarrow \mathcal{E}_1(\text{pp}, \text{s}, u_1, \text{st}_1) \end{array} \right. \right] \geq \epsilon(\mathcal{A}, \mathcal{P}^*) - \text{negl}(\lambda).$$

In conclusion, for the arbitrary adversary $(\mathcal{A}, \mathcal{P}^*)$ fixed above, set $\mathcal{E} := \mathcal{E}_1$. The preceding inequality shows that $\mathcal{E}$ extracts a valid witness for $\mathcal{R}_1$ with probability at least $\epsilon(\mathcal{A}, \mathcal{P}^*) - \text{negl}(\lambda)$. Thus, $\Pi := \Pi_2 \circ \Pi_1$ is knowledge sound. Since Π_1 and Π_2 are complete and public coin, their sequential composition Π is also complete and public coin. Therefore, Π is a reduction of knowledge (Definition 9). $\square$

B.2 Proofs for Π_{CCS}

We first provide a lemma that will be helpful for both the security and completeness of the interactive reduction.

Lemma 9. *Consider the following arbitrary items:*

$$\begin{aligned} &\text{structure } \text{s}, \quad \text{vectors } z_1, \dots, z_{K+k} \in \mathbb{L}^{n_L}, \\ &\text{point } r \in \mathbb{K}^{\log m}, \quad \text{evaluations } (y_i \in \mathbb{R}_{\mathbb{K}}, \{y_{i,j} \in \mathbb{R}_{\mathbb{A}}\}_{j \in [t]})_{i=K+1}^{K+k} \end{aligned}$$

Recall from Definition 21 the matrix $\text{Pad} \in \mathbb{F}^{m \times n_{\mathbb{F}}}$. For every $i \in [K+k]$, define

$$\begin{aligned} z_i^b &:= \text{Flat}_{\mathbb{L}}(z_i) \in \mathbb{F}^{n_{\mathbb{F}}} \\ z_i^{b, \text{pad}} &:= \text{Pad } z_i^b \in \mathbb{F}^m \\ z_i &:= \text{Pack}_{\mathbb{L}}(z_i) = \text{Pack}_{\mathbb{F}}(z_i^b) \in \mathbb{R}_{\mathbb{F}}^{n_{\mathbb{R}}} \\ h_i &:= \text{Emb}_{\mathbb{F}}(\text{Trans}_{\mathbb{F}}(\text{Pad})) \cdot z_i \in \mathbb{R}_{\mathbb{F}}^m \end{aligned}$$

For every $i \in [K+k]$ and $j \in [t]$, define

$$h_{i,j} := \text{Emb}_{\mathbb{L}}(\text{Trans}_{\mathbb{L}}(M_j)) \cdot z_i \in \mathbb{R}_{\mathbb{L}}^m$$

Similarly to Π_{CCS} (Section 7.3), define

$$\begin{aligned} \text{F}(\vec{X}, C) &:= \sum_{i=1}^K C^{i-1} \cdot f(\widetilde{M_1 z_i}, \dots, \widetilde{M_t z_i}) \\ \text{NC}(\vec{X}, C) &:= \sum_{i=1}^{K+k} C^{i-1} \cdot \prod_{a=-b+1}^{b-1} \left(\widetilde{z_i^{b, \text{pad}}} - a \right) \end{aligned}$$

$$\begin{aligned}
\text{Eval}_{\mathbb{K}}(\vec{X}, C) &:= \text{eq}(\vec{X}, r) \cdot \sum_{i=K+1}^{K+k} \sum_{\ell=1}^d C^{\mathbf{I}_{\mathbb{K}}(i, \ell)} \cdot \widetilde{\text{cf}(h_i)}_{\ell} \\
\text{Eval}_{\mathbb{A}}(\vec{X}, C) &:= \text{eq}(\vec{X}, r) \cdot \sum_{i=K+1}^{K+k} \sum_{j=1}^t \sum_{\ell=1}^d C^{\mathbf{I}_{\mathbb{A}}(i, j, \ell)} \cdot \widetilde{\text{cf}(h_{i,j})}_{\ell} \\
Q(\vec{X}, \vec{A}, C) &:= \text{Eval}_{\mathbb{K}}(\vec{X}, C) + C^{kd} \cdot \text{Eval}_{\mathbb{A}}(\vec{X}, C) + C^{kd(t+1)} \cdot \text{eq}(\vec{X}, \vec{A}) \left(\text{F}(\vec{X}, C) + C^K \cdot \text{NC}(\vec{X}, C) \right) \\
T(C) &:= \sum_{i=K+1}^{K+k} \sum_{\ell=1}^d C^{\mathbf{I}_{\mathbb{K}}(i, \ell)} \cdot \text{cf}(y_i)_{\ell} + C^{kd} \cdot \sum_{i=K+1}^{K+k} \sum_{j=1}^t \sum_{\ell=1}^d C^{\mathbf{I}_{\mathbb{A}}(i, j, \ell)} \cdot \text{cf}(y_{i,j})_{\ell}
\end{aligned}$$

where $\vec{X} := (X_1, \dots, X_{\log m})$, and the challenges $\alpha \in \mathbb{K}^{\log m}$ and $\gamma \in \mathbb{K}$ are replaced by the indeterminates $\vec{A} := (A_1, \dots, A_{\log m})$ and C , respectively.

We must have $T(C) = \sum_{\vec{x} \in \{0,1\}^{\log m}} Q(\vec{x}, \vec{A}, C)$ if and only if

1. $f(\widetilde{M_1 z_i}, \dots, \widetilde{M_t z_i}) \in \text{ZS}_{\log m}(\mathbb{L})$ for all $i \in [K]$,
2. $\|z_i^b\|_{\infty} < b$ for all $i \in [K+k]$,
3. $y_i = \widetilde{h_i}(r)$ for all $i \in [K+1, K+k]$,
4. $y_{i,j} = \widetilde{h_{i,j}}(r)$ for all $i \in [K+1, K+k]$ and $j \in [t]$.

Proof. By definition of $T(C)$,

$$T(C) = \sum_{\vec{x} \in \{0,1\}^{\log m}} Q(\vec{x}, \vec{A}, C)$$

if and only if

$$\sum_{i=K+1}^{K+k} \sum_{\ell=1}^d C^{\mathbf{I}_{\mathbb{K}}(i, \ell)} \cdot \text{cf}(y_i)_{\ell} + C^{kd} \cdot \sum_{i=K+1}^{K+k} \sum_{j=1}^t \sum_{\ell=1}^d C^{\mathbf{I}_{\mathbb{A}}(i, j, \ell)} \cdot \text{cf}(y_{i,j})_{\ell} = \sum_{\vec{x} \in \{0,1\}^{\log m}} Q(\vec{x}, \vec{A}, C) \quad (6)$$

Since powers of C are linearly independent, Equation (6) occurs if and only if

$$\forall i \in [K], \quad 0 = \sum_{\vec{x} \in \{0,1\}^{\log m}} \text{eq}(\vec{x}, \vec{A}) \cdot f(\widetilde{M_1 z_i}(\vec{x}), \dots, \widetilde{M_t z_i}(\vec{x})), \quad (7)$$

$$\forall i \in [K+k], \quad 0 = \sum_{\vec{x} \in \{0,1\}^{\log m}} \text{eq}(\vec{x}, \vec{A}) \cdot \prod_{a=-b+1}^{b-1} \left(\widetilde{z_i^{b, \text{pad}}}(\vec{x}) - a \right), \quad (8)$$

$$\forall i \in [K+1, K+k], \quad \forall \ell \in [d], \quad \text{cf}(y_i)_{\ell} = \sum_{\vec{x} \in \{0,1\}^{\log m}} \text{eq}(\vec{x}, r) \cdot \widetilde{\text{cf}(h_i)}_{\ell}(\vec{x}) \quad (9)$$

$$\forall i \in [K+1, K+k], \quad \forall j \in [t], \quad \forall \ell \in [d], \quad \text{cf}(y_{i,j})_{\ell} = \sum_{\vec{x} \in \{0,1\}^{\log m}} \text{eq}(\vec{x}, r) \cdot \widetilde{\text{cf}(h_{i,j})}_{\ell}(\vec{x}) \quad (10)$$

By Lemma 3, Equation (7) and Equation (8) occur if and only if

1. $f(\widetilde{M_1 z_i}, \dots, \widetilde{M_t z_i}) \in \text{ZS}_{\log m}(\mathbb{L})$ for all $i \in [K]$ (Item 1),
2. $\prod_{a=-b+1}^{b-1} \left(\widetilde{z_i^{b, \text{pad}}} - a \right) \in \text{ZS}_{\log m}(\mathbb{F})$ for all $i \in [K+k]$.

Since $z_i^{\mathbf{b}, \text{pad}} = \text{Pad } z_i^{\mathbf{b}}$ only appends zero coordinates and $0 \in \{-b+1, \dots, b-1\}$, Item 2 holds if and only if $\|z_i^{\mathbf{b}}\|_\infty < b$ for all $i \in [K+k]$ (Item 2). By definition of multilinear extension and bijectivity of the coefficient map, Equation (9) and Equation (10) hold if and only if

$$\begin{aligned} y_i &= \widetilde{h_i}(r) & \text{for all } i \in [K+1, K+k] & \quad (\text{Item 3}), \\ y_{i,j} &= \widetilde{h_{i,j}}(r) & \text{for all } i \in [K+1, K+k], j \in [t] & \quad (\text{Item 4}). \end{aligned}$$

In conclusion, we have shown

$$T(C) = \sum_{\vec{x} \in \{0,1\}^{\log m}} Q(\vec{x}, \vec{A}, C)$$

if and only if (Item 1), (Item 2), (Item 3), and (Item 4). $\square$

Lemma 10. *The interactive reduction $\Pi_{\text{CCS}} : \text{CCS}(b, \mathcal{L})^K \times \text{CE}(b, \mathcal{L})^k \rightarrow \text{CE}(b, \mathcal{L})^{K+k}$ is **complete** and **public coin**.*

Proof. Completeness. Assume the original input tuples belong to relations $\text{CCS}(b, \mathcal{L})$ (Definition 20) and $\text{CE}(b, \mathcal{L})$ (Definition 21). We will first argue that the sum-check verifier in step 2 passes. Then, we will argue that the evaluation claim check in step 4 passes. Finally, we will argue that output tuples belong to $\text{CE}(b, \mathcal{L})^{K+k}$.

By the definition of relations $\text{CCS}(b, \mathcal{L})$ (Definition 20) and $\text{CE}(b, \mathcal{L})$ (Definition 21), we must have that (Item 1), (Item 2), (Item 3), and (Item 4) from Lemma 9 hold. Therefore, we must have that

$$T(C) = \sum_{\vec{x} \in \{0,1\}^{\log m}} Q(\vec{x}, \vec{A}, C).$$

Thus, for any choice of challenges $\alpha \in \mathbb{K}^{\log m}$ and $\gamma \in \mathbb{K}$ chosen in step 1,

$$T(\gamma) = \sum_{\vec{x} \in \{0,1\}^{\log m}} Q(\vec{x}, \alpha, \gamma).$$

Thus, by the completeness of the sum-check protocol (Definition 11), we must have that the sum-check verifier (step 2) always passes.

By step 3 and Theorem 10, we must have that

$$\begin{aligned} \text{ct}(y'_i) &= \widetilde{z_i^{\mathbf{b}, \text{pad}}}(r') & \text{for all } i \in [K+k], \\ \text{ct}(y'_{i,j}) &= \widetilde{M_j z_i}(r') & \text{for all } i \in [K+k], j \in [t]. \end{aligned}$$

Finally, by the same theorem, we must have that

$$\begin{aligned} \text{cf}(y'_i)_\ell &= \widetilde{\text{cf}(h_i)_\ell}(r') & \text{for all } i \in [K+1, K+k], \ell \in [d], \\ \text{cf}(y'_{i,j})_\ell &= \widetilde{\text{cf}(h_{i,j})_\ell}(r') & \text{for all } i \in [K+1, K+k], j \in [t], \ell \in [d]. \end{aligned}$$

By definition of $Q(\vec{X})$ in step 2, we must have that

$$Q(r') = E_{\mathbb{K}} + \gamma^{kd} \cdot E_{\mathbb{A}} + \gamma^{kd(t+1)} \cdot \text{eq}(r', \alpha)(F + \gamma^K \cdot N)$$

for values F , N , $E_{\mathbb{K}}$, and $E_{\mathbb{A}}$ derived in step 4. Since the honest sum-check execution yields $v = Q(r')$, the verifier check in step 4 passes.

Observe that Π_{CCS} outputs exactly the original structure s , commitments $(c_i)_{i \in [K+k]}$, vectors $(z_i)_{i \in [K+k]}$, and instances $(x_i)_{i \in [K+k]}$. Thus, by the definition of $\text{CCS}(b, \mathcal{L})$, we must have immediately that every condition in $\text{CE}(b, \mathcal{L})$ is satisfied for all the $K+k$ tuples, except that

$$\begin{aligned} y'_i &= \widetilde{h_i}(r') & \text{for all } i \in [K+k], \\ y'_{i,j} &= \widetilde{h_{i,j}}(r') & \text{for all } i \in [K+k], j \in [t]. \end{aligned}$$

However, these are exactly the evaluations computed by the honest prover in step 3. Therefore, the output tuples do belong to $\text{CE}(b, \mathcal{L})^{K+k}$ as required.

Public coin. The sum-check protocol itself is a public-coin protocol. The remaining randomness from the verifier are the challenges $\alpha \in \mathbb{K}^{\log m}$, $\gamma \in \mathbb{K}$, which are sampled uniformly at random and sent to the prover. $\square$

We prove conditions (i) and (ii) of strong interactive reductions (Definition 18).

Proof.

Proof of (i) By construction, the verifier trivially sets the commitments in the output instance u_2 to be the original commitments $(c_i)_{i \in [K+k]}$ from the input instance u_1 . Hence, for repeated executions with respect to the same input instance u_1 with output instances u_2, u'_2 , the commitments in these output instances must be the same.

Proof of (ii) Consider an arbitrary expected polynomial-time adversary $(\mathcal{A}, \mathcal{P}^*)$, such that the relaxed success probability of the adversary $\epsilon'(\mathcal{A}, \mathcal{P}^*) \geq 1/\text{poly}(\lambda)$ and, letting $\mathcal{O}_{\text{str}}$ be the oracle from condition (ii) of strong interactive reductions (Definition 18) instantiated with $\mathcal{P}^*$, for any uniqueness adversary $\mathcal{S}_{\text{uniq}}^{\mathcal{O}_{\text{str}}}$ (Definition 16),

$$\Pr \left[\begin{array}{c} w_2, w'_2 \neq \perp \\ \wedge \\ w_2 \neq w'_2 \end{array} \middle| \begin{array}{l} \text{pp} \leftarrow \mathcal{G}(1^\lambda, \text{sz}) \\ (\text{s}, u_1, \text{st}) \leftarrow \mathcal{A}(\text{pp}) \\ \text{in} \leftarrow (\text{pp}, \text{s}, u_1, \text{st}) \\ (u_2, w_2), (u'_2, w'_2) \leftarrow \mathcal{S}_{\text{uniq}}^{\mathcal{O}_{\text{str}}(\text{in})}(\text{in}) \end{array} \right] \leq \text{negl}(\lambda) \quad (11)$$

then we will show that there exists an expected polynomial-time extractor $\mathcal{E}$ such that

$$\Pr \left[(\text{pp}, \text{s}, u_1, w_1) \in \mathcal{R}_1 \middle| \begin{array}{l} \text{pp} \leftarrow \mathcal{G}(1^\lambda, \text{sz}) \\ (\text{s}, u_1, \text{st}) \leftarrow \mathcal{A}(\text{pp}) \\ (\text{pk}, \text{vk}) \leftarrow \mathcal{K}(\text{pp}, \text{s}) \\ w_1 \leftarrow \mathcal{E}(\text{pp}, \text{s}, u_1, \text{st}) \end{array} \right] \geq \epsilon'(\mathcal{A}, \mathcal{P}^*) - \text{negl}(\lambda).$$

Namely, the following extractor $\mathcal{E}$,

- $\mathcal{E}(\text{pp}, \text{s}, u_1, \text{st}) \rightarrow w_1 :$
1. $(\text{pk}, \text{vk}) \leftarrow \mathcal{K}(\text{pp}, \text{s})$.
 2. Simulate $(u_2, w_2) \leftarrow \langle \mathcal{P}^*, \mathcal{V} \rangle((\text{pk}, \text{vk}), u_1, \text{st})$ with fresh randomness.
 3. If $(\text{pp}, \text{s}, u_2, w_2) \notin \mathcal{R}'_2$, then output $\perp$.
 4. Parse $(z_1, \dots, z_K, z_{K+1}, \dots, z_{K+k}) \leftarrow w_2$.
 5. For all $i \in [K]$, assign $w_i^{\text{CCS}} \leftarrow z_i[n_{L, \text{in}} :]$.
 6. Output $w_1 := (w_1^{\text{CCS}}, \dots, w_K^{\text{CCS}}, z_{K+1}, \dots, z_{K+k})$.

Extractor runtime. The extractor makes one call to the expected polynomial-time prover $\mathcal{P}^*$ and otherwise performs only polynomial-time computations. Hence, it runs in expected polynomial time.

Extractor success probability. Let $\text{in} := (\text{pp}, \text{s}, u_1, \text{st})$ denote everything fixed before the protocol execution, and compute $(\text{pk}, \text{vk}) \leftarrow \mathcal{K}(\text{pp}, \text{s})$. Conditioned on in , let (u_2, w_2) be the response from the extractor's interaction, and let (u'_2, w'_2) be one fresh response from $\mathcal{O}_{\text{str}}(\text{in})$:

$$\begin{aligned} (u_2, w_2) &\leftarrow \langle \mathcal{P}^*, \mathcal{V} \rangle((\text{pk}, \text{vk}), u_1, \text{st}), \\ (u'_2, w'_2) &\leftarrow \mathcal{O}_{\text{str}}(\text{in}). \end{aligned}$$

Let w_1 denote the extractor's output when its protocol call returns (u_2, w_2) . Define the following events:

- $\text{Succ}_1 := \{(\text{pp}, \text{s}, u_2, w_2) \in \mathcal{R}'_2\}$: relaxed success of the first response.

- $\text{Succ}_2 := \{(\text{pp}, \text{s}, u'_2, w'_2) \in \mathcal{R}'_2\}$: relaxed success of the fresh response.
- $\text{Err}_1 := \text{Succ}_1 \cap \{(\text{pp}, \text{s}, u_1, w_1) \notin \mathcal{R}_1\}$: extraction error for the first response.
- $\text{Agree} := \{w'_2 = w_2\}$: agreement between the two witnesses.¹⁰

Thus, conditioned on in , Succ_1 and Succ_2 each occur with probability

$$p_{\text{in}} := \Pr[\text{Succ}_1 \mid \text{in}] = \Pr[\text{Succ}_2 \mid \text{in}],$$

and Err_1 occurs with probability

$$a_{\text{in}} := \Pr[\text{Err}_1 \mid \text{in}].$$

Since $\text{Err}_1 \subseteq \text{Succ}_1$, we have $0 \leq a_{\text{in}} \leq p_{\text{in}}$. By the law of total probability,

$$\epsilon'(\mathcal{A}, \mathcal{P}^*) = \mathbb{E}_{\text{in}}[\Pr[\text{Succ}_1 \mid \text{in}]] = \mathbb{E}_{\text{in}}[p_{\text{in}}]. \quad (12)$$

Consider the following uniqueness adversary:

$\mathcal{S}_{\text{uniq}}^{\mathcal{O}_{\text{str}}(\text{in})}(\text{in})$:

1. Parse $(\text{pp}, \text{s}, u_1, \text{st}) \leftarrow \text{in}$ and query $\mathcal{O}_{\text{str}}(\text{in})$ to obtain (u_2, w_2) .
2. If $(\text{pp}, \text{s}, u_2, w_2) \notin \mathcal{R}'_2$, output $\perp$.
3. Parse $(z_1, \dots, z_K, z_{K+1}, \dots, z_{K+k}) \leftarrow w_2$. For all $i \in [K]$, assign $w_i^{\text{CCS}} \leftarrow z_i[n_{L,\text{in}} :]$, and assign $w_1 \leftarrow (w_1^{\text{CCS}}, \dots, w_K^{\text{CCS}}, z_{K+1}, \dots, z_{K+k})$.
4. If $(\text{pp}, \text{s}, u_1, w_1) \in \mathcal{R}_1$, output $\perp$.
5. Repeatedly query $\mathcal{O}_{\text{str}}(\text{in})$ until obtaining (u'_2, w'_2) such that $(\text{pp}, \text{s}, u'_2, w'_2) \in \mathcal{R}'_2$.
6. Output $(u_2, w_2), (u'_2, w'_2)$.

By construction, the uniqueness adversary reaches step 5 exactly when Err_1 occurs. Let t_{in} be the expected running time of one oracle call together with the relation checks and witness construction, conditioned on in . If $p_{\text{in}} = 0$, then $a_{\text{in}} = 0$, so step 5 is never reached. The expected running time conditioned on in is at most t_{in} . Otherwise, fresh oracle calls imply that step 5 terminates with probability one and has expected running time $t_{\text{in}}/p_{\text{in}}$. Since step 5 is reached with probability a_{in} and $a_{\text{in}} \leq p_{\text{in}}$, the expected running time conditioned on in is at most

$$t_{\text{in}} + \frac{a_{\text{in}}}{p_{\text{in}}} t_{\text{in}}.$$

Since $\mathcal{P}^*$ is EPT and all other computations run in polynomial time, $\mathbb{E}_{\text{in}}[t_{\text{in}}]$ is polynomial. Thus, $\mathcal{S}_{\text{uniq}}^{\mathcal{O}_{\text{str}}}$ runs in expected polynomial time, and (11) gives

$$\epsilon_{\text{uniq}} := \Pr \left[\begin{array}{l} w_2, w'_2 \neq \perp \\ \wedge w_2 \neq w'_2 \end{array} \middle| \begin{array}{l} \text{pp} \leftarrow \mathcal{G}(1^\lambda, \text{sz}) \\ (\text{s}, u_1, \text{st}) \leftarrow \mathcal{A}(\text{pp}) \\ \text{in} \leftarrow (\text{pp}, \text{s}, u_1, \text{st}) \\ (u_2, w_2), (u'_2, w'_2) \leftarrow \mathcal{S}_{\text{uniq}}^{\mathcal{O}_{\text{str}}(\text{in})}(\text{in}) \end{array} \right] \leq \text{negl}(\lambda).$$

To bound the probability that $\mathcal{S}_{\text{uniq}}$ returns equal witnesses, fix in and a first response (u_2, w_2) for which Err_1 occurs. Consider one call $(u'_2, w'_2) \leftarrow \mathcal{O}_{\text{str}}(\text{in})$ made in step 5. We first bound $\text{Succ}_2 \cap \text{Agree}$ for this call. By definition of $\mathcal{R}'_2 = \text{CE}(q/2, \mathcal{L})^{K+k}$, for all $i \in [K+k]$, define

$$\mathbf{x}_i := \text{Pack}_{\mathbb{L}}(x_i), \quad \mathbf{z}_i := \text{Pack}_{\mathbb{L}}(z_i).$$

We must have

$$\mathbf{x}_i = \mathcal{L}_{\text{in}}(\mathbf{z}_i) \quad \text{and} \quad c_i = \mathcal{L}(\mathbf{z}_i). \quad (13)$$

By the definitions of $\text{Pack}_{\mathbb{L}}$ (Definition 14) and $\mathcal{L}_{\text{in}}$ (Definition 22), together with $n_{\text{F},\text{in}} = \tau n_{L,\text{in}} = d n_{\text{R},\text{in}}$ (Definition 1), Equation (13) also implies that for all $i \in [K+k]$, the first $n_{L,\text{in}}$ entries of z_i are equal to x_i (from input instance u_1).

¹⁰ Here, in $\text{Agree} := \{w'_2 = w_2\}$, (u'_2, w'_2) denotes the response from the single call to $\mathcal{O}_{\text{str}}(\text{in})$ above.

Since Err_1 occurs for the fixed first response, $(\text{pp}, \mathbf{s}, u_1, w_1) \notin \mathcal{R}_1$. Recall that $\mathcal{R}_1 := \text{CCS}(b, \mathcal{L})^K \times \text{CE}(b, \mathcal{L})^k$. By Equation (13), we know that all conditions in $\text{CCS}(b, \mathcal{L})$ and $\text{CE}(b, \mathcal{L})$, except for the norm bound $\|z^b\|_\infty < b$, the evaluations $y = \widetilde{h}(r)$ and $y_j = \widetilde{h}_j(r)$ for $j \in [t]$, or the CCS requirements $f(\widetilde{M_1 z}, \dots, \widetilde{M_t z}) \in \text{ZS}_{\log m}(\mathbb{L})$, are satisfied. Thus, using notation from Lemma 9, $(\text{pp}, \mathbf{s}, u_1, w_1) \notin \mathcal{R}_1$ implies that either:

1. there exists an $i \in [K]$, $f(\widetilde{M_1 z_i}, \dots, \widetilde{M_t z_i}) \notin \text{ZS}_{\log m}(\mathbb{L})$,
2. OR there exists an $i \in [K + k]$, $\|z_i^b\|_\infty \geq b$,
3. OR there exists an $i \in [K + 1, K + k]$, $y_i \neq \widetilde{h}_i(r)$,
4. OR there exists an $i \in [K + 1, K + k]$ and $j \in [t]$, $y_{i,j} \neq \widetilde{h}_{i,j}(r)$.

By Lemma 9, we must have

$$T(C) \neq \sum_{\vec{x} \in \{0,1\}^{\log m}} Q(\vec{x}, \vec{A}, C).$$

Now focus on the oracle call, whose randomness is independent of the fixed first response. Write $\alpha', \gamma', r', v', y'_i$, and $y'_{i,j}$ for its challenges, sum-check point and value, and claimed evaluations.

On **Agree**, $w_2 = w'_2$. Consequently, the oracle call uses the same vectors z_i , and hence the same h_i and $h_{i,j}$, as the fixed first response.

On **Succ**₂, the oracle call has relaxed success, so $(\text{pp}, \mathbf{s}, u'_2, w'_2) \in \mathcal{R}'_2$. Therefore, the verifier did not abort in protocol steps 2 and 4, and the prover's claimed evaluations in protocol step 3 are true. Namely,

$$\begin{aligned} y'_i &= \widetilde{h}_i(r') && \text{for all } i \in [K + k], \\ y'_{i,j} &= \widetilde{h}_{i,j}(r') && \text{for all } i \in [K + k], j \in [t]. \end{aligned}$$

Thus, by Theorem 10, the definition of Q , and the construction of the verifier's checks in protocol step 4, we must have that the oracle call's sum-check evaluation claim $v' = Q(r', \alpha', \gamma')$ is true.

Thus, in order for the sum-check verifier to have passed in the oracle call, either the adversary $\mathcal{P}^*$

- succeeded in the sum-check protocol, despite $T(\gamma') \neq \sum_{\vec{x} \in \{0,1\}^{\log m}} Q(\vec{x}, \alpha', \gamma')$ (in other words, violated the soundness of the sum-check protocol)
- OR the non-zero polynomial

$$P(C, \vec{A}) := T(C) - \sum_{\vec{x} \in \{0,1\}^{\log m}} Q(\vec{x}, \vec{A}, C)$$

evaluated to zero on the random point $(\gamma', \alpha') \in \mathbb{K}^{1+\log m}$ of the oracle call.

By the soundness error of the sum-check protocol (Definition 11), the first event occurs with probability at most $\epsilon_{\text{SC}} := \max(u, 2b, 2) \cdot \log m / |\mathbb{K}|$, where u is the strict total-degree bound for f , b is the norm bound, and 2 comes from $\text{Eval}_{\mathbb{K}}(\vec{X})$ and $\text{Eval}_{\mathbb{A}}(\vec{X})$. By the Schwartz–Zippel lemma (Lemma 2), the second event occurs with probability at most $\epsilon_{\text{SZ}} := (kd(t+1) + 2K + k - 1 + \log m) / |\mathbb{K}|$. By Definition 1, $1/|\mathbb{K}| = \text{negl}(\lambda)$. Since u, b, t, k, K, d , and $\log m$ are polynomially bounded in λ , both ϵ_{SC} and ϵ_{SZ} are negligible. For the fixed in and first response for which Err_1 occurs, the union bound over the randomness of the oracle call gives

$$\Pr[\text{Succ}_2 \cap \text{Agree}] \leq \epsilon_{\text{SC}} + \epsilon_{\text{SZ}}. \tag{14}$$

Let $\epsilon_{\text{test}} := \epsilon_{\text{SC}} + \epsilon_{\text{SZ}}$. For fixed in and first response in Err_1 , the response returned by step 5¹¹ is distributed as one oracle call conditioned on Succ_2 . By Equation (14) and conditional probability,

$$\Pr[\text{Agree} \mid \text{Succ}_2] = \frac{\Pr[\text{Succ}_2 \cap \text{Agree}]}{\Pr[\text{Succ}_2]} \leq \frac{\epsilon_{\text{test}}}{p_{\text{in}}}.$$

¹¹ Hereafter, in **Agree** := $\{w'_2 = w_2\}$, (u'_2, w'_2) denotes the response returned by step 5.

If $p_{\text{in}} = 0$, then $a_{\text{in}} = 0$, so step 5 is never reached and

$$\Pr[\text{Err}_1 \cap \text{Agree} \mid \text{in}] = 0.$$

If $p_{\text{in}} > 0$, then

$$\Pr[\text{Err}_1 \cap \text{Agree} \mid \text{in}] \leq a_{\text{in}} \frac{\epsilon_{\text{test}}}{p_{\text{in}}} \leq \epsilon_{\text{test}},$$

since $a_{\text{in}} \leq p_{\text{in}}$. Thus, the bound holds for every in. Averaging over in gives

$$\Pr[\text{Err}_1 \cap \text{Agree}] \leq \epsilon_{\text{test}}.$$

By construction, $\mathcal{S}_{\text{uniq}}$ reaches step 5 exactly when Err_1 occurs, and step 5 terminates with probability one. Its returned witnesses either agree or differ. Therefore,

$$\Pr[\text{Err}_1] = \Pr[\text{Err}_1 \cap \text{Agree}] + \Pr[\text{Err}_1 \cap \text{Agree}^c].$$

The first term is at most ϵ_{test} by the bound above. The second term equals ϵ_{uniq} , since the uniqueness adversary returns different witnesses exactly when $\text{Err}_1 \cap \text{Agree}^c$ occurs. Hence,

$$\mathbb{E}_{\text{in}}[a_{\text{in}}] = \Pr[\text{Err}_1] \leq \epsilon_{\text{test}} + \epsilon_{\text{uniq}}.$$

For every fixed in, relaxed success occurs with probability p_{in} and extraction error occurs with probability a_{in} . The extractor outputs a valid witness exactly when relaxed success occurs without an extraction error, namely on $\text{Succ}_1 \setminus \text{Err}_1$. Since $\text{Err}_1 \subseteq \text{Succ}_1$, this event has probability $p_{\text{in}} - a_{\text{in}}$. Consequently, by the law of total probability, linearity of expectation, and Equation (12),

$$\begin{aligned} \Pr[(\text{pp}, \mathbf{s}, u_1, w_1) \in \mathcal{R}_1] &= \Pr[\text{Succ}_1 \setminus \text{Err}_1] \\ &= \mathbb{E}_{\text{in}}[p_{\text{in}} - a_{\text{in}}] \\ &= \mathbb{E}_{\text{in}}[p_{\text{in}}] - \mathbb{E}_{\text{in}}[a_{\text{in}}] \\ &= \epsilon'(\mathcal{A}, \mathcal{P}^*) - \mathbb{E}_{\text{in}}[a_{\text{in}}] \\ &\geq \epsilon'(\mathcal{A}, \mathcal{P}^*) - \epsilon_{\text{test}} - \epsilon_{\text{uniq}}. \end{aligned}$$

□

B.3 Proofs for Π_{RLC}

Lemma 11. *The interactive reduction $\Pi_{\text{RLC}} : \text{CE}(b, \mathcal{L})^{K+k} \rightarrow \text{CE}(B, \mathcal{L})$ is **complete** and **public coin**.*

Proof. Completeness. By definition of $\text{CE}(b, \mathcal{L})$ (Definition 21), the input tuples satisfy the conditions in Theorem 11 instantiated with $(\mathbb{B}, M, z_i, y_i) = (\mathbb{F}, \text{Pad}, z_i^b, y_i)$ and, for every $j \in [t]$, with $(\mathbb{B}, M, z_i, y_i) = (\mathbb{L}, M_j, z_i, y_{i,j})$. Since the protocol decodes x and z from their packed linear combinations, we have

$$\begin{aligned} c &= \mathcal{L}(\mathbf{z}), \quad \mathbf{x} = \mathcal{L}_{\text{in}}(\mathbf{z}), \\ y &= \tilde{h}(r), \quad y_j = \tilde{h}_j(r) \quad \text{for every } j \in [t]. \end{aligned}$$

Here, $h := \text{Emb}_{\mathbb{F}}(\text{Trans}_{\mathbb{F}}(\text{Pad})) \cdot \mathbf{z}$ and $h_j := \text{Emb}_{\mathbb{L}}(\text{Trans}_{\mathbb{L}}(M_j)) \cdot \mathbf{z}$ for every $j \in [t]$. Therefore, the output tuple satisfies every requirement of $\text{CE}(B, \mathcal{L})$ except $\|\mathbf{z}^b\|_{\infty} < B = b^k$.

However, we show that this bound follows from the expansion factor T of $\mathcal{C}$ chosen in Definition 22:

$$\begin{aligned} \|\mathbf{z}^b\|_{\infty} &= \|\mathbf{z}\|_{\infty} = \left\| \sum_{i=1}^{K+k} \rho_i \mathbf{z}_i \right\|_{\infty} \\ &\leq \sum_{i=1}^{K+k} \|\rho_i \mathbf{z}_i\|_{\infty} \leq \sum_{i=1}^{K+k} T \|\mathbf{z}_i\|_{\infty} \\ &= \sum_{i=1}^{K+k} T \|\mathbf{z}_i^b\|_{\infty} \leq (K+k)T(b-1) < B. \end{aligned}$$

The equalities $\|z^b\|_\infty = \|z\|_\infty$ and $\|z_i^b\|_\infty = \|z_i\|_\infty$ follow from the definitions of Pack_F and the norm. The first inequality is the triangle inequality, the second inequality follows from the expansion factor of $\mathcal{C}$ being T , the third inequality follows from the definition of $\text{CE}(b, \mathcal{L})$, which enforces a norm bound of b , and the last inequality holds by assumption (Definition 22). Hence, the output tuple must belong to $\text{CE}(B, \mathcal{L})$.

Public coin. The verifier's randomness consists of challenges $\rho_1, \dots, \rho_{k+K}$, which are sampled uniformly at random from $\mathcal{C}$ and sent to the prover. $\square$

We prove the conditions of weak interactive reductions (Definition 17).

Proof. Consider an arbitrary expected-polynomial time adversary $(\mathcal{A}, \mathcal{P}^*)$ for Π_{RLC} . First, we can construct an adversary and verification function for Theorem 7,

$A_{(\text{pp}, s, u_1, \text{st})}(\vec{c}) :$

1. If $u_1 = \perp$, output $\perp$.
2. Execute encoder $(\text{pk}, \text{vk}) \leftarrow \mathcal{K}(\text{pp}, s)$.
3. Simulate $(u_2, w_2) \leftarrow \langle \mathcal{P}^*(\text{pk}, u_1, \text{st}), \mathcal{V}(\text{vk}, u_1) \rangle$ with verifier randomness $\vec{c}$.
4. Output w_2

$V_{(\text{pp}, s, u_1, \text{st})}(\vec{c}, w_2) \rightarrow \{0, 1\} :$

1. If $u_1 = \perp$, output reject.
2. Execute encoder $(\text{pk}, \text{vk}) \leftarrow \mathcal{K}(\text{pp}, s)$.
3. Simulate $(u_2, -) \leftarrow \langle \mathcal{P}^*(\text{pk}, u_1, \text{st}), \mathcal{V}(\text{vk}, u_1) \rangle$ with verifier randomness $\vec{c}$.
4. Output accept if and only if $(u_2, w_2) \in \text{CE}(B, \mathcal{L})$.

Let $E_{(\text{pp}, s, u_1, \text{st})}$ be the corresponding extractor from Theorem 7. We define $E(\text{pp}, s, u_1, \text{st})$ as the trivial algorithm that executes $E_{(\text{pp}, s, u_1, \text{st})}$ by simulating calls to $A_{(\text{pp}, s, u_1, \text{st})}$. Without loss of generality, E outputs $\perp$ whenever its output does not satisfy the three conditions in Theorem 7. We construct an extractor for adversary $(\mathcal{A}, \mathcal{P}^*)$ as follows:

$\mathcal{E}(\text{pp}, s, u_1, \text{st}) :$

1. **result** $\leftarrow E(\text{pp}, s, u_1, \text{st})$.
2. If $u_1 = \perp$ or **result** $= \perp$, output $\perp$.
3. Parse $(\vec{c}, w'), (\vec{c}_1, w'_1), \dots, (\vec{c}_{K+k}, w'_{K+k}) \leftarrow \text{result}$.
4. Parse $z \leftarrow w'$ and $\rho_1, \dots, \rho_{K+k} \leftarrow \vec{c}$.
5. For $i \in [K+k]$,
 - (a) Parse $z^{(i)} \leftarrow w'_i$ and $\rho_1^{(i)}, \dots, \rho_{K+k}^{(i)} \leftarrow \vec{c}_i$.
 - (b) Assign $z_i \leftarrow (\rho_i - \rho_i^{(i)})^{-1} \cdot (z - z^{(i)})$.
 - (c) Assign $z_i \leftarrow \text{Pack}_{\mathcal{L}}^{-1}(z_i)$.
6. Parse $(c_i, x_i, r, y_i, \{y_{i,j}\}_{j \in [t]})_{i \in [K+k]} \leftarrow u_1$.
7. Output $w_1 := (z_i)_{i \in [K+k]}$ if and only if

$$(s; c_i, x_i, r, y_i, \{y_{i,j}\}_{j \in [t]}; z_i)_{i \in [K+k]} \in \text{CE}(q/2, \mathcal{L})^{K+k}$$

Extractor runtime. By Theorem 7, we are guaranteed $E_{(\text{pp}, s, u_1, \text{st})}$ makes in expectation at most $(K+k)+1$ calls to $A_{(\text{pp}, s, u_1, \text{st})}$. Hence, our overall extractor $\mathcal{E}$ runs in expected polynomial time.

Extractor success probability. If $u_1 = \perp$, then $A_{(\text{pp}, s, u_1, \text{st})}$ outputs $\perp$ and $V_{(\text{pp}, s, u_1, \text{st})}$ rejects, so both the conditional success probability and $\Pr[\text{result} \neq \perp \mid \text{pp}, s, u_1, \text{st}]$ are zero. For every fixed $(\text{pp}, s, u_1, \text{st})$, by Theorem 7, we are guaranteed that $E(\text{pp}, s, u_1, \text{st})$ outputs $(K+k)+1$ pairs $(\vec{c}, w'), (\vec{c}_1, w'_1), \dots, (\vec{c}_{K+k}, w'_{K+k})$ such that

- $V_{(\text{pp}, s, u_1, \text{st})}(\vec{c}, w') = 1$,
- for all $i \in [K+k]$, $V_{(\text{pp}, s, u_1, \text{st})}(\vec{c}_i, w'_i) = 1$, and

- $(\vec{c}, \vec{c}_1, \dots, \vec{c}_{K+k}) \in \text{SS}(\mathcal{C}, K+k)$

with probability at least $\epsilon^{V_{(\text{pp}, \text{s}, u_1, \text{st})}}(A_{(\text{pp}, \text{s}, u_1, \text{st})}) - \frac{K+k}{|\mathcal{C}|}$. Hence, for every fixed $(\text{pp}, \text{s}, u_1, \text{st})$,

$$\Pr[\text{result} \neq \perp \mid \text{pp}, \text{s}, u_1, \text{st}] \geq \epsilon^{V_{(\text{pp}, \text{s}, u_1, \text{st})}}(A_{(\text{pp}, \text{s}, u_1, \text{st})}) - \frac{K+k}{|\mathcal{C}|}.$$

Since the verifier's output u_2 in Π_{RLC} is determined by u_1 and the challenge vector $\vec{c}$, for every fixed $(\text{pp}, \text{s}, u_1, \text{st})$,

$$\epsilon^{V_{(\text{pp}, \text{s}, u_1, \text{st})}}(A_{(\text{pp}, \text{s}, u_1, \text{st})}) = \Pr[(\text{pp}, \text{s}, u_2, w_2) \in \text{CE}(B, \mathcal{L}) \mid \text{pp}, \text{s}, u_1, \text{st}].$$

By definition, $\epsilon(\mathcal{A}, \mathcal{P}^*)$ is the overall probability that the output belongs to $\text{CE}(B, \mathcal{L})$. Therefore, by the law of total probability and the fixed-tuple equality above,

$$\begin{aligned} \epsilon(\mathcal{A}, \mathcal{P}^*) &= \Pr[(\text{pp}, \text{s}, u_2, w_2) \in \text{CE}(B, \mathcal{L})] \\ &= \mathbb{E}_{(\text{pp}, \text{s}, u_1, \text{st})} [\Pr[(\text{pp}, \text{s}, u_2, w_2) \in \text{CE}(B, \mathcal{L}) \mid \text{pp}, \text{s}, u_1, \text{st}]] \\ &= \mathbb{E}_{(\text{pp}, \text{s}, u_1, \text{st})} [\epsilon^{V_{(\text{pp}, \text{s}, u_1, \text{st})}}(A_{(\text{pp}, \text{s}, u_1, \text{st})})]. \end{aligned} \quad (15)$$

Averaging this pointwise guarantee over the distribution of $(\text{pp}, \text{s}, u_1, \text{st})$ induced by $\text{pp} \leftarrow \mathcal{G}(1^\lambda, \text{sz})$ and $(\text{s}, u_1, \text{st}) \leftarrow \mathcal{A}(\text{pp})$ gives

$$\begin{aligned} \Pr[\text{result} \neq \perp] &= \mathbb{E}_{(\text{pp}, \text{s}, u_1, \text{st})} [\Pr[\text{result} \neq \perp \mid \text{pp}, \text{s}, u_1, \text{st}]] \\ &\geq \mathbb{E}_{(\text{pp}, \text{s}, u_1, \text{st})} \left[\epsilon^{V_{(\text{pp}, \text{s}, u_1, \text{st})}}(A_{(\text{pp}, \text{s}, u_1, \text{st})}) - \frac{K+k}{|\mathcal{C}|} \right] \\ &= \mathbb{E}_{(\text{pp}, \text{s}, u_1, \text{st})} [\epsilon^{V_{(\text{pp}, \text{s}, u_1, \text{st})}}(A_{(\text{pp}, \text{s}, u_1, \text{st})})] - \frac{K+k}{|\mathcal{C}|} \\ &= \epsilon(\mathcal{A}, \mathcal{P}^*) - \frac{K+k}{|\mathcal{C}|}. \end{aligned} \quad (16)$$

The first equality follows from the law of total probability, the inequality from the fixed-tuple guarantee above, and the next equality from linearity of expectation. The final equality follows from Equation (15). Assume that $\text{result} \neq \perp$. Since $V_{(\text{pp}, \text{s}, u_1, \text{st})}(\vec{c}, w') = 1$ and $V_{(\text{pp}, \text{s}, u_1, \text{st})}$ executes Π_{RLC} 's $\mathcal{V}$ and outputs accept if and only if $(u_2, w_2) \in \text{CE}(B, \mathcal{L})$, we must have for $\mathbf{x} := \sum_{i=1}^{K+k} \rho_i \mathbf{x}_i$ and $x := \text{Pack}_{\mathbb{L}}^{-1}(\mathbf{x})$ that

$$\left(\begin{array}{c} c := \sum_{i=1}^{K+k} \rho_i c_i, \\ \text{s}; \quad x, r, y := \sum_{i=1}^{K+k} \rho_i y_i, \quad ; \quad z \\ \left\{ y_j := \sum_{i=1}^{K+k} \rho_i y_{i,j} \right\}_{j \in [t]} \end{array} \right) \in \text{CE}(B, \mathcal{L}) \quad (17)$$

where $(c_i, x_i, r, y_i, \{y_{i,j}\}_j)_i$ are the instance elements in u_1 (parsed in step 6) and $z \leftarrow w'$ and $(\rho_i)_i \leftarrow \vec{c}$ are the elements parsed in step 4.

For all $i \in [K+k]$, we will make a similar argument to the one directly above. Namely, since

$$V_{(\text{pp}, \text{s}, u_1, \text{st})}(\vec{c}_i, w'_i) = 1,$$

we must have, for

$$\mathbf{x}^{(i)} := \sum_{a=1}^{K+k} \rho_a^{(i)} \mathbf{x}_a, \quad x^{(i)} := \text{Pack}_{\mathbb{L}}^{-1}(\mathbf{x}^{(i)}),$$

that

$$\left(\begin{array}{c} c^{(i)} := \sum_{a=1}^{K+k} \rho_a^{(i)} c_a, \\ \text{s}; \quad x^{(i)}, r, y^{(i)} := \sum_{a=1}^{K+k} \rho_a^{(i)} y_a, \quad ; \quad z^{(i)} \\ \left\{ y_j^{(i)} := \sum_{a=1}^{K+k} \rho_a^{(i)} y_{a,j} \right\}_{j \in [t]} \end{array} \right) \in \text{CE}(B, \mathcal{L}) \quad (18)$$

where $(c_i, x_i, r, y_i, \{y_{i,j}\}_j)_i$ are in u_1 and $z^{(i)} \leftarrow w'_i$ and $(\rho_a^{(i)})_a \leftarrow \vec{c}_i$ are the elements parsed in step 5a. By definition of $\text{CE}(B, \mathcal{L})$ (Definition 21), we must have

$$c = \mathcal{L}(\mathbf{z}), \quad c^{(i)} = \mathcal{L}(\mathbf{z}^{(i)}), \quad \mathbf{x} = \mathcal{L}_{\text{in}}(\mathbf{z}), \quad \mathbf{x}^{(i)} = \mathcal{L}_{\text{in}}(\mathbf{z}^{(i)}) \quad (19)$$

Since $(\vec{c}, \vec{c}_1, \dots, \vec{c}_{K+k}) \in \text{SS}(\mathcal{C}, K+k)$, we must have for all $i \in [K+k]$ that

$$(\rho_1, \dots, \rho_{K+k}) \equiv_i (\rho_1^{(i)}, \dots, \rho_{K+k}^{(i)}) \quad (20)$$

which means the challenges differ only on index i . By definition of strong sampling set (Definition 6), we must have $(\rho_i - \rho_i^{(i)})$ is invertible for all $i \in [K+k]$.

Thus, by Equation (19) and Equation (20), we have for all $i \in [K+k]$,

$$\begin{aligned} c - c^{(i)} &= \mathcal{L}(\mathbf{z}) - \mathcal{L}(\mathbf{z}^{(i)}) \\ \mathbf{x} - \mathbf{x}^{(i)} &= \mathcal{L}_{\text{in}}(\mathbf{z}) - \mathcal{L}_{\text{in}}(\mathbf{z}^{(i)}) \end{aligned}$$

$$\begin{aligned} \sum_{a=1}^{K+k} \rho_a c_a - \sum_{a=1}^{K+k} \rho_a^{(i)} c_a &= \mathcal{L}(\mathbf{z}) - \mathcal{L}(\mathbf{z}^{(i)}), \\ \sum_{a=1}^{K+k} \rho_a \mathbf{x}_a - \sum_{a=1}^{K+k} \rho_a^{(i)} \mathbf{x}_a &= \mathcal{L}_{\text{in}}(\mathbf{z}) - \mathcal{L}_{\text{in}}(\mathbf{z}^{(i)}) \end{aligned} \quad (21)$$

$$\begin{aligned} (\rho_i - \rho_i^{(i)}) \cdot c_i &= \mathcal{L}(\mathbf{z}) - \mathcal{L}(\mathbf{z}^{(i)}), \\ (\rho_i - \rho_i^{(i)}) \cdot \mathbf{x}_i &= \mathcal{L}_{\text{in}}(\mathbf{z}) - \mathcal{L}_{\text{in}}(\mathbf{z}^{(i)}) \end{aligned} \quad (22)$$

$$\begin{aligned} c_i &= \mathcal{L}\left((\rho_i - \rho_i^{(i)})^{-1} \cdot (\mathbf{z} - \mathbf{z}^{(i)})\right), \\ \mathbf{x}_i &= \mathcal{L}_{\text{in}}\left((\rho_i - \rho_i^{(i)})^{-1} \cdot (\mathbf{z} - \mathbf{z}^{(i)})\right) \end{aligned} \quad (23)$$

$$c_i = \mathcal{L}(\mathbf{z}_i), \quad \mathbf{x}_i = \mathcal{L}_{\text{in}}(\mathbf{z}_i) \quad (24)$$

where equation (21) follows from (17), (18), and Equation (19). Equation (22) follows from the equivalence in Equation (20). Equation (23) follows from $\mathcal{L}, \mathcal{L}_{\text{in}}$ being $\mathbb{R}_{\mathbb{F}}$ -module homomorphisms and $\mathcal{C}$ being a strong sampling set (Definition 6) which because $\rho_i \neq \rho_i^{(i)}$ (guaranteed by (20)) means $\rho_i - \rho_i^{(i)}$ is invertible. Equation (24) follows from the packed extraction in step 5b and the subsequent assignment $z_i := \text{Pack}_{\mathbb{L}}^{-1}(\mathbf{z}_i)$.

We make a similar argument for the evaluations. Define

$$\begin{aligned} h &:= \text{Emb}_{\mathbb{F}}(\text{Trans}_{\mathbb{F}}(\text{Pad})) \cdot \mathbf{z}, & h^{(i)} &:= \text{Emb}_{\mathbb{F}}(\text{Trans}_{\mathbb{F}}(\text{Pad})) \cdot \mathbf{z}^{(i)}, \\ h_j &:= \text{Emb}_{\mathbb{L}}(\text{Trans}_{\mathbb{L}}(M_j)) \cdot \mathbf{z}, & h_j^{(i)} &:= \text{Emb}_{\mathbb{L}}(\text{Trans}_{\mathbb{L}}(M_j)) \cdot \mathbf{z}^{(i)} \quad \text{for every } j \in [t]. \end{aligned}$$

In particular, by the definition of $\text{CE}(B, \mathcal{L})$ (Definition 21), Equation (17), and Equation (18), we must have

$$y = \widetilde{h}(r), \quad y^{(i)} = \widetilde{h^{(i)}}(r), \quad y_j = \widetilde{h_j}(r), \quad y_j^{(i)} = \widetilde{h_j^{(i)}}(r) \quad \text{for every } j \in [t]. \quad (25)$$

Thus, we must have for all $i \in [K+k]$,

$$y - y^{(i)} = \widetilde{h}(r) - \widetilde{h^{(i)}}(r) \quad (26)$$

$$\sum_{a=1}^{K+k} \rho_a y_a - \sum_{a=1}^{K+k} \rho_a^{(i)} y_a = \overline{\text{Emb}_{\mathbb{F}}(\text{Trans}_{\mathbb{F}}(\text{Pad})) \cdot (\mathbf{z} - \mathbf{z}^{(i)})}(r) \quad (27)$$

$$(\rho_i - \rho_i^{(i)}) \cdot y_i = \overline{\text{Emb}_{\mathbb{F}}(\text{Trans}_{\mathbb{F}}(\text{Pad})) \cdot (\mathbf{z} - \mathbf{z}^{(i)})}(r) \quad (28)$$

$$y_i = \overline{\text{Emb}_{\mathbb{F}}(\text{Trans}_{\mathbb{F}}(\text{Pad})) \cdot \left((\rho_i - \rho_i^{(i)})^{-1} \cdot (\mathbf{z} - \mathbf{z}^{(i)})\right)}(r) \quad (29)$$

$$= \widetilde{h_i}(r)$$

where Equation (26) follows from Equation (25), Equation (27) follows from Equation (17) and Equation (18), the definitions of h and $h^{(i)}$, and the linearity of multilinear evaluation, Equation (28) follows from the equivalence (20), and (29) follows from $\mathcal{C}$ being a strong sampling set (Definition 6) which because $\rho_i \neq \rho_i^{(i)}$ (guaranteed by (20)) means $\rho_i - \rho_i^{(i)}$ is invertible, together with the $\mathbb{R}_F$ -linearity of the transform and multilinear evaluation. The final equality follows from the construction of $\mathbf{z}_i$ and the definition of h_i . Thus, we must have for all $i \in [K+k]$ and $j \in [t]$,

$$y_j - y_j^{(i)} = \widetilde{h_j}(r) - \widetilde{h_j^{(i)}}(r) \quad (30)$$

$$\sum_{a=1}^{K+k} \rho_a y_{a,j} - \sum_{a=1}^{K+k} \rho_a^{(i)} y_{a,j} = \overline{\text{Emb}_{\mathbb{L}}(\text{Trans}_{\mathbb{L}}(M_j)) \cdot (\mathbf{z} - \mathbf{z}^{(i)})}(r) \quad (31)$$

$$(\rho_i - \rho_i^{(i)}) \cdot y_{i,j} = \overline{\text{Emb}_{\mathbb{L}}(\text{Trans}_{\mathbb{L}}(M_j)) \cdot (\mathbf{z} - \mathbf{z}^{(i)})}(r) \quad (32)$$

$$\begin{aligned} y_{i,j} &= \overline{\text{Emb}_{\mathbb{L}}(\text{Trans}_{\mathbb{L}}(M_j)) \cdot \left((\rho_i - \rho_i^{(i)})^{-1} \cdot (\mathbf{z} - \mathbf{z}^{(i)}) \right)}(r) \\ &= \widetilde{h_{i,j}}(r) \end{aligned} \quad (33)$$

where Equation (30) follows from Equation (25), Equation (31) follows from Equation (17) and Equation (18), the definitions of h_j and $h_j^{(i)}$, and the linearity of multilinear evaluation, Equation (32) follows from the equivalence (20), and (33) follows from $\mathcal{C}$ being a strong sampling set (Definition 6) which because $\rho_i \neq \rho_i^{(i)}$ (guaranteed by (20)) means $\rho_i - \rho_i^{(i)}$ is invertible, together with the $\mathbb{R}_F$ -linearity of matrix multiplication and multilinear evaluation. The final equality follows from the construction of $\mathbf{z}_i$ and the definition of $h_{i,j}$.

Therefore, by Equation (16), Equation (24), Equation (29), and Equation (33), we must have with probability at least $\epsilon(\mathcal{A}, \mathcal{P}^*) - (K+k)/|\mathcal{C}|$, the extractor outputs witness elements $z_1, \dots, z_{K+k}$ such that

$$(\mathbf{s}; c_i, x_i, r, y_i, \{y_{i,j}\}_{j \in [t]}; z_i)_{i \in [K+k]} \in \text{CE}(q/2, \mathcal{L})^{K+k},$$

The norm bound $q/2$ is trivial because every coordinate of $z_i^b \in \mathbb{F}^{n_F}$ has norm at most $(q-1)/2 < q/2$. By Definition 22, $1/|\mathcal{C}| = \text{negl}(\lambda)$, so this probability is at least $\epsilon(\mathcal{A}, \mathcal{P}^*) - \text{negl}(\lambda)$. Thus, the extractor satisfies condition (i) of weak interactive reductions (Definition 17).

Now, assume that $\mathcal{A} := (\mathcal{B}, \mathcal{B}')$ such that

$$\Pr \left[\begin{array}{c|c} u_1, u'_1 \neq \perp & \text{pp} \leftarrow \mathcal{G}(1^\lambda, \text{sz}) \\ \downarrow & (\mathbf{s}, \text{st}^*) \leftarrow \mathcal{B}(\text{pp}) \\ \phi(u_1) = \phi(u'_1) & \begin{array}{l} (u_1, \text{st}) \leftarrow \mathcal{B}'(\text{st}^*) \\ (u'_1, \text{st}') \leftarrow \mathcal{B}'(\text{st}^*) \end{array} \end{array} \right] = 1 \quad (34)$$

Let $\mathcal{O}_{\text{weak}}$ be the oracle from condition (ii) of weak interactive reductions (Definition 17), instantiated with $\mathcal{B}'$ and $\mathcal{E}$. Consider any uniqueness adversary $\mathcal{W}_{\text{uniq}}^{\mathcal{O}_{\text{weak}}}$. We will show that

$$\Pr \left[\begin{array}{c|c} w_1, w'_1 \neq \perp & \text{pp} \leftarrow \mathcal{G}(1^\lambda, \text{sz}) \\ \wedge w_1 \neq w'_1 & \begin{array}{l} (\mathbf{s}, \text{st}^*) \leftarrow \mathcal{B}(\text{pp}) \\ \text{in} \leftarrow (\text{pp}, \mathbf{s}, \text{st}^*) \\ (u_1, w_1), (u'_1, w'_1) \leftarrow \mathcal{W}_{\text{uniq}}^{\mathcal{O}_{\text{weak}}(\text{in})}(\text{in}) \end{array} \end{array} \right] \leq \text{negl}(\lambda) \quad (35)$$

To bound the probability in Equation (35), consider a relaxed-binding adversary $\mathcal{A}_{\text{rlx}}$. The generator $\mathcal{G}(1^\lambda, n_{\text{R}})$ outputs $\text{pp} \leftarrow \text{Setup}(1^\lambda, n_{\text{R}})$, so $\mathcal{A}_{\text{rlx}}$ receives public parameters with the same distribution as in Equation (35).

$\mathcal{A}_{\text{rlx}}(\text{pp}) :$

1. $(\text{s}, \text{st}^*) \leftarrow \mathcal{B}(\text{pp})$ and $\text{in} \leftarrow (\text{pp}, \text{s}, \text{st}^*)$.
2. Run $\mathcal{W}_{\text{uniq}}^{\mathcal{O}_{\text{weak}}(\text{in})}$. For every query to $\mathcal{O}_{\text{weak}}$:
 - (a) Compute $(u, \text{st}) \leftarrow \mathcal{B}'(\text{st}^*)$.
 - (b) Run $\mathcal{E}(\text{pp}, \text{s}, u, \text{st})$ to obtain w , retaining its internal value **result**.
 - (c) Return (u, w) and store **result** for this call.
3. If $\mathcal{W}_{\text{uniq}}$ outputs $\perp$, a $\perp$ witness, or equal witnesses, output $\perp$.
4. Otherwise, let (u_1, w_1) and (u'_1, w'_1) be the selected responses, and retrieve **result** and **result'** stored for the two calls that returned them.
5. Parse $w_1 = (z_i)_{i \in [K+k]}$ and $w'_1 = (z'_i)_{i \in [K+k]}$, and choose $i \in [K+k]$ such that $z_i \neq z'_i$.
6. Parse **result** and **result'** to obtain $\rho_i, \rho_i^{(i)}, z, z^{(i)}$ and $\rho'_i, \rho'_i^{(i)'}, z', z^{(i)'}$.
7. Assign

$$\begin{aligned} \Delta_1 &\leftarrow \rho_i - \rho_i^{(i)}, & v_1 &\leftarrow \text{Pack}_{\mathbb{L}}(z) - \text{Pack}_{\mathbb{L}}(z^{(i)}), \\ \Delta_2 &\leftarrow \rho'_i - \rho'_i^{(i)'}, & v_2 &\leftarrow \text{Pack}_{\mathbb{L}}(z') - \text{Pack}_{\mathbb{L}}(z^{(i)'}). \end{aligned}$$

8. Parse c_i from u_1 and output $(c_i, \Delta_1, \Delta_2, v_1, v_2)$.

Assume that, during an execution of $\mathcal{A}_{\text{rlx}}$, $\mathcal{W}_{\text{uniq}}$ returns $(u_1, w_1), (u'_1, w'_1)$ such that the event in Equation (35) occurs. Every call to $\mathcal{B}'(\text{st}^*)$ that $\mathcal{A}_{\text{rlx}}$ makes to answer $\mathcal{W}_{\text{uniq}}$'s oracle queries uses fresh randomness. Therefore, by Equation (34), any two such outputs $u_1, u'_1 \neq \perp$ satisfy $\phi(u_1) = \phi(u'_1)$ with probability one. In particular,

1. Since $w_1, w'_1 \neq \perp$, we have $u_1, u'_1 \neq \perp$. Hence, $\phi(u_1) = \phi(u'_1)$, which guarantees that the selected instances share identical commitments $(c_i)_{i \in [K+k]}$.
2. Write $w_1 = (z_i)_{i \in [K+k]}$ and $w'_1 = (z'_i)_{i \in [K+k]}$. Since $w_1 \neq w'_1$, there exists $i \in [K+k]$ such that $z_i \neq z'_i$.

The values parsed by $\mathcal{A}_{\text{rlx}}$ from **result** and **result'** for the two selected calls satisfy the following

$$\begin{aligned} z_i \neq z'_i &\iff \text{Pack}_{\mathbb{L}}(z_i) \neq \text{Pack}_{\mathbb{L}}(z'_i) \\ &\iff (\rho_i - \rho_i^{(i)})^{-1} \cdot (\text{Pack}_{\mathbb{L}}(z) - \text{Pack}_{\mathbb{L}}(z^{(i)})) \\ &\quad \neq (\rho'_i - \rho'_i^{(i)'})^{-1} \cdot (\text{Pack}_{\mathbb{L}}(z') - \text{Pack}_{\mathbb{L}}(z^{(i)'})) \end{aligned} \tag{36}$$

$$\|\text{Flat}_{\mathbb{L}}(z)\|_{\infty} \|\text{Flat}_{\mathbb{L}}(z^{(i)})\|_{\infty} \|\text{Flat}_{\mathbb{L}}(z')\|_{\infty} \|\text{Flat}_{\mathbb{L}}(z^{(i)'})\|_{\infty} < B \tag{37}$$

Equation (36) follows from (Item 2), the bijectivity of $\text{Pack}_{\mathbb{L}}$, and the construction of $\mathcal{E}$. Equation (37) follows from the construction of $\mathcal{E}$, which only outputs $w_1, w'_1 \neq \perp$ when the internal extractor E (from Theorem 7) succeeds in the two selected calls. In particular, the internal extractor E succeeding guarantees that the verification functions $V_{(\text{pp}, \text{s}, u_1, \text{st})}$ and $V_{(\text{pp}, \text{s}, u'_1, \text{st}')}$ accept. These verification functions check that output tuples (corresponding to Equation (17) and Equation (18)) belong to $\text{CE}(B, \mathcal{L})$, which exactly checks the required norm bound on the flattened vectors. Since $\rho_i, \rho_i^{(i)}, \rho'_i, \rho'_i^{(i)'}$ are in $\mathcal{C}$, we have $\Delta_1, \Delta_2 \in \mathcal{C} - \mathcal{C}$. By Equation (37), the definitions of $\text{Pack}_{\mathbb{L}}$ and the norm, and the triangle inequality,

$$\begin{aligned} \|v_1\|_{\infty} &\leq \|\text{Pack}_{\mathbb{L}}(z)\|_{\infty} + \|\text{Pack}_{\mathbb{L}}(z^{(i)})\|_{\infty} < 2B, \\ \|v_2\|_{\infty} &\leq \|\text{Pack}_{\mathbb{L}}(z')\|_{\infty} + \|\text{Pack}_{\mathbb{L}}(z^{(i)'})\|_{\infty} < 2B. \end{aligned}$$

Applying (23) to the two selected calls and using Item 1, we must have

$$c_i = \mathcal{L}(\Delta_1^{-1} \cdot v_1) = \mathcal{L}(\Delta_2^{-1} \cdot v_2).$$

Thus, since $\mathcal{L}$ is a $\mathbb{R}_{\mathbb{F}}$ -module homomorphism, we have

$$\Delta_1 \cdot c_i = \mathcal{L}(v_1) \wedge \Delta_2 \cdot c_i = \mathcal{L}(v_2) \tag{38}$$

Moreover, since Δ_1 and Δ_2 are invertible in the commutative ring $R_{\mathbb{F}}$, multiplication by $\Delta_1 \Delta_2$ is injective. Therefore, Equation (36) implies $\Delta_1 v_2 \neq \Delta_2 v_1$. All together, by (36), (37), and (38), we have that $(c_i, \Delta_1, \Delta_2, v_1, v_2)$ is a $(2B, \mathcal{C})$ -relaxed binding collision (Definition 7). Furthermore, $\mathcal{A}_{\text{rlx}}$ runs in expected polynomial time. The execution of $\mathcal{W}_{\text{uniq}}^{\mathcal{O}_{\text{weak}}}$, including its oracle calls, is EPT. Recording each **result** and processing the two selected results add only polynomial time computations. Thus, $\mathcal{A}_{\text{rlx}}$ is EPT.

By assumption (Definition 22), the ring commitment scheme defining $\mathcal{L}$ satisfies $(2B, \mathcal{C})$ -relaxed binding. Since $\mathcal{A}_{\text{rlx}}$ is EPT and outputs a valid relaxed-binding collision whenever the event in Equation (35) occurs,

$$\Pr \left[\begin{array}{l} w_1, w'_1 \neq \perp \\ \wedge w_1 \neq w'_1 \end{array} \middle| \begin{array}{l} \text{pp} \leftarrow \mathcal{G}(1^\lambda, \text{sz}) \\ (\text{s}, \text{st}^*) \leftarrow \mathcal{B}(\text{pp}) \\ \text{in} \leftarrow (\text{pp}, \text{s}, \text{st}^*) \\ (u_1, w_1), (u'_1, w'_1) \leftarrow \mathcal{W}_{\text{uniq}}^{\mathcal{O}_{\text{weak}}(\text{in})}(\text{in}) \end{array} \right] \leq \epsilon_{\text{rlx}}(2B, \mathcal{C}) \leq \text{negl}(\lambda).$$

Thus, the extractor satisfies condition (ii) of weak interactive reductions (Definition 17). $\square$

B.4 Π_{DEC} is a Reduction of Knowledge (Theorem 13)

Proof. Completeness: First, we show that the verifier's checks in step 2 pass. Then, we will show that the output tuples belong to $\text{CE}(b, \mathcal{L})^k$.

By the definition of $\text{CE}(B, \mathcal{L})$, let $z^b := \text{Flat}_{\mathbb{L}}(z)$ and $x^b := \text{Flat}_{\mathbb{L}}(x)$. We must have $\|z^b\|_{\infty} < B = b^k$ (Definition 22), so $\text{split}_b(z^b)$ is defined and $z^b = \sum_{i=1}^k b^{i-1} \cdot z_i^b$. Moreover, $\text{Pack}_{\mathbb{L}}(x) = \mathcal{L}_{\text{in}}(\text{Pack}_{\mathbb{L}}(z))$. Since $\mathcal{L}_{\text{in}}$ projects the first $n_{\text{R}, \text{in}}$ packed coordinates, x^b is the corresponding prefix of z^b . Therefore, $\|x^b\|_{\infty} \leq \|z^b\|_{\infty} < B$, so $\text{split}_b(x^b)$ is also defined. Since split_b is coordinatewise, each x_i^b is the corresponding prefix of z_i^b . Hence, $\text{Pack}_{\mathbb{L}}(x_i) = \mathcal{L}_{\text{in}}(\text{Pack}_{\mathbb{L}}(z_i))$ for every $i \in [k]$. By linearity,

$$\begin{aligned} z^b &= \sum_{i=1}^k b^{i-1} \cdot z_i^b \\ z &= \sum_{i=1}^k b^{i-1} \cdot z_i \\ \text{Pack}_{\mathbb{L}}(z) &= \sum_{i=1}^k b^{i-1} \cdot \text{Pack}_{\mathbb{L}}(z_i) \end{aligned} \tag{39}$$

$$\begin{aligned} \mathcal{L}(\text{Pack}_{\mathbb{L}}(z)) &= \mathcal{L}(\sum_{i=1}^k b^{i-1} \cdot \text{Pack}_{\mathbb{L}}(z_i)) \\ c &= \sum_{i=1}^k b^{i-1} \cdot \mathcal{L}(\text{Pack}_{\mathbb{L}}(z_i)), \end{aligned} \tag{40}$$

$$c = \sum_{i=1}^k b^{i-1} \cdot c_i, \tag{41}$$

Equation (40) follows directly from $\mathcal{L}$ being a $R_{\mathbb{F}}$ -module homomorphism. Equation (41) follows by construction of $c_i \leftarrow \mathcal{L}(\text{Pack}_{\mathbb{L}}(z_i))$ in step 1. Therefore, the verifier's commitment check passes. Starting from equation (39), we must have

$$\begin{aligned} \text{Pack}_{\mathbb{L}}(z) &= \sum_{i=1}^k b^{i-1} \cdot \text{Pack}_{\mathbb{L}}(z_i) \\ \text{Emb}_{\mathbb{F}}(\text{Trans}_{\mathbb{F}}(\text{Pad})) \cdot \text{Pack}_{\mathbb{L}}(z) &= \sum_{i=1}^k b^{i-1} \cdot \text{Emb}_{\mathbb{F}}(\text{Trans}_{\mathbb{F}}(\text{Pad})) \cdot \text{Pack}_{\mathbb{L}}(z_i) \\ h &= \sum_{i=1}^k b^{i-1} \cdot h_i \\ \widetilde{h}(r) &= \sum_{i=1}^k b^{i-1} \cdot \widetilde{h}_i(r) \end{aligned} \tag{42}$$

$$y = \sum_{i=1}^k b^{i-1} \cdot y_i \tag{43}$$

Similarly, for every $j \in [t]$,

$$\begin{aligned} \text{Pack}_{\mathbb{L}}(z) &= \sum_{i=1}^k b^{i-1} \cdot \text{Pack}_{\mathbb{L}}(z_i) \\ \text{Emb}_{\mathbb{L}}(\text{Trans}_{\mathbb{L}}(M_j)) \cdot \text{Pack}_{\mathbb{L}}(z) &= \sum_{i=1}^k b^{i-1} \cdot \text{Emb}_{\mathbb{L}}(\text{Trans}_{\mathbb{L}}(M_j)) \cdot \text{Pack}_{\mathbb{L}}(z_i) \\ h_j &= \sum_{i=1}^k b^{i-1} \cdot h_{i,j} \end{aligned}$$

$$\widetilde{h}_j(r) = \sum_{i=1}^k b^{i-1} \cdot \widetilde{h_{i,j}}(r) \quad (44)$$

$$y_j = \sum_{i=1}^k b^{i-1} \cdot y_{i,j} \quad (45)$$

Equation (42) and Equation (44) follow from the linearity of matrix multiplication and multilinear evaluation. Equation (43) and Equation (45) follow from the definition of $\text{CE}(B, \mathcal{L})$ and the construction of y_i and $y_{i,j}$ in step 1. Thus, by (41), (43), and (45), the verifier's checks pass.

Next, we show that the output tuple, $(s; \{c_i, x_i, r, y_i, \{y_{i,j}\}_{j \in [t]}\}_{i \in [k]}; \{z_i\}_{i \in [k]})$, belongs to $\text{CE}(b, \mathcal{L})^k$. By the definition of split_b , we must have that $\|z_i^b\|_\infty < b$ for all $i \in [k]$. Since $\mathcal{L}_{\text{in}}$ is the trivial $\mathbb{R}_F$ -module homomorphism which projects the first n_{Rin} coordinates, and split_b is coordinatewise, we must have that, by construction in step 2, $\text{Pack}_{\mathbb{L}}(x_i) = \mathcal{L}_{\text{in}}(\text{Pack}_{\mathbb{L}}(z_i))$ for all $i \in [k]$. Thus, in total, we must have, along with the construction of $(c_i, y_i, \{y_{i,j}\}_{j \in [t]})_{i \in [k]}$ in step 1, that the output tuples belong to $\text{CE}(b, \mathcal{L})^k$.

Public coin: The verifier uses no randomness in this protocol. Thus, the protocol is trivially public coin.

Knowledge soundness: Consider an arbitrary expected-polynomial time adversary $(\mathcal{A}, \mathcal{P}^*)$ for Π_{DEC} with success probability, $\epsilon(\mathcal{A}, \mathcal{P}^*) \geq 1/\text{poly}(\lambda)$. We construct an extractor $\mathcal{E}$ for Π_{DEC} as follows,

$\mathcal{E}(\text{pp}, s, u_1 := (c, x, r, y, (y_j)_{j \in [t]}), \text{st}):$

1. Execute encoder $(\text{pk}, \text{vk}) \leftarrow \mathcal{K}(\text{pp}, s)$.
2. Simulate $(u_2, w_2) \leftarrow \langle \mathcal{P}^*(\text{pk}, u_1, \text{st}), \mathcal{V}(\text{vk}, u_1) \rangle$.
3. If $u_2 = \perp$, output $\perp$.
4. Parse $(z_1, \dots, z_k) \leftarrow w_2$.
5. Output $w_1 := \sum_{i=1}^k b^{i-1} z_i$.

Extractor runtime: The extractor runs in expected polynomial time, since it simulates only one execution between the adversary $\mathcal{P}^*$ and verifier $\mathcal{V}$, which both run in expected polynomial time.

Extractor success probability: Assume that the simulated adversary $(\mathcal{A}, \mathcal{P}^*)$ succeeds in convincing the verifier $\mathcal{V}$ and the parties jointly output $(s, u_2, w_2) \in \text{CE}(b, \mathcal{L})^k$; note that this occurs with probability $\epsilon(\mathcal{A}, \mathcal{P}^*)$. Define

$$(c_i, x_i, r, y_i, (y_{i,j})_{j \in [t]})_{i \in [k]} := u_2 \text{ and } z_1, \dots, z_k := w_2.$$

By the definition of $\text{CE}(b, \mathcal{L})$, for every $i \in [k]$, define $z_i^b := \text{Flat}_{\mathbb{L}}(z_i)$, $h_i := \text{Emb}_F(\text{Trans}_F(\text{Pad})) \cdot \text{Pack}_{\mathbb{L}}(z_i)$, and $h_{i,j} := \text{Emb}_{\mathbb{L}}(\text{Trans}_{\mathbb{L}}(M_j)) \cdot \text{Pack}_{\mathbb{L}}(z_i)$ for every $j \in [t]$. We must have

$$\begin{aligned} c_i &= \mathcal{L}(\text{Pack}_{\mathbb{L}}(z_i)), \\ \text{Pack}_{\mathbb{L}}(x_i) &= \mathcal{L}_{\text{in}}(\text{Pack}_{\mathbb{L}}(z_i)), \\ \|z_i^b\|_\infty &< b, \\ y_i &= \widetilde{h_i}(r), \\ \forall j \in [t], \quad y_{i,j} &= \widetilde{h_{i,j}}(r) \end{aligned} \quad (46)$$

Since the adversary convinces the verifier, we must have

$$\begin{aligned} c &= \sum_{i=1}^k b^{i-1} \cdot c_i, \\ y &= \sum_{i=1}^k b^{i-1} \cdot y_i, \\ \forall j \in [t], \quad y_j &= \sum_{i=1}^k b^{i-1} \cdot y_{i,j} \end{aligned} \quad (47)$$

By construction in step 2 (i.e. definition of split_b), we also must have $x = \sum_{i=1}^k b^{i-1} \cdot x_i$. By defining $z := \sum_{i=1}^k b^{i-1} z_i$, observe that $x = \sum_{i=1}^k b^{i-1} \cdot x_i$, (46), and (47) satisfy the remaining conditions stated in

Theorem 11, applied first with $\mathbb{B} = \mathbb{F}$, $M = \text{Pad}$, vectors $(z_i^b)_{i \in [k]}$, and evaluations $(y_i)_{i \in [k]}$, and then, for every $j \in [t]$, with $\mathbb{B} = \mathbb{L}$, $M = M_j$, vectors $(z_i)_{i \in [k]}$, and evaluations $(y_{i,j})_{i \in [k]}$. In both applications, the scalars are $(b^{i-1})_{i \in [k]}$, where each b^{i-1} is treated as a constant element of $\mathbb{R}_{\mathbb{F}}$. We must have $c = \mathcal{L}(\text{Pack}_{\mathbb{L}}(z))$, $\text{Pack}_{\mathbb{L}}(x) = \mathcal{L}_{\text{in}}(\text{Pack}_{\mathbb{L}}(z))$, $y = \widetilde{h}(r)$, and, for all $j \in [t]$, $y_j = \widetilde{h_j}(r)$, where $h := \text{Emb}_{\mathbb{F}}(\text{Trans}_{\mathbb{F}}(\text{Pad})) \cdot \text{Pack}_{\mathbb{L}}(z)$ and $h_j := \text{Emb}_{\mathbb{L}}(\text{Trans}_{\mathbb{L}}(M_j)) \cdot \text{Pack}_{\mathbb{L}}(z)$. Since in (46), we have $\|z_i^b\|_{\infty} < b$ for every $i \in [k]$, by the $\mathbb{F}$ -linearity of $\text{Flat}_{\mathbb{L}}$, we must also have $z^b = \sum_{i=1}^k b^{i-1} z_i^b$ and $\|z^b\|_{\infty} \leq \sum_{i=1}^k b^{i-1}(b-1) = b^k - 1 < B$. These are exactly the conditions for $(s; u_1 := \{c, x, r, y, \{y_j\}_{j \in [t]}\}; w_1 := z)$ to belong to $\text{CE}(B, \mathcal{L})$. The simulated interaction in $\mathcal{E}$ has the same distribution as the interaction defining $\epsilon(\mathcal{A}, \mathcal{P}^*)$. Whenever this interaction succeeds, the preceding argument shows that the extracted witness w_1 satisfies the input relation. Therefore, $\mathcal{E}$ outputs a satisfying witness with probability at least $\epsilon(\mathcal{A}, \mathcal{P}^*)$. $\square$

B.5 Hardness and Inversion Bound calculation scripts

We defer to the lattice estimator script for hardness. The inversion bounds are calculated with the following script.

```
# [LS18, eprint 2017-523] pg 6
# m is the cyclotomic polynomial index
def tau(m):
    return m if (m % 2) != 0 else m / 2

# [LS18, eprint 2017-523] Thm 1.1, pg 4
# m is the cyclotomic polynomial index
# p is the prime
# z is any divisor of m
# This tests for the condition for thm 1.1 to hold
def thm1_1_cond(m, p, z):
    cond1 = (p % z) == 1
    cond2 = Mod(p,m).multiplicative_order() == m/z
    return cond1 and cond2

# [LS18, eprint 2017-523] Thm 1.1, pg 4
# p is the prime
# z is any divisor of m
# lInf bound for elements to be invertible
# given that m,p,z satisfy thm 1.1 cond
def thm1_1_inv_bound(p, z):
    return (1/s1(z)*p^(1/euler_phi(z))).n()

def thm1_1_num_factors(z):
    return euler_phi(z)

# Output divisors of m
def divisors(m):
    zs = list()
    for i in range(1,m+1):
        if m % i == 0:
            zs.append(i)
    return zs

# [LS18, eprint 2017-523] pg 6, pg 9
# We only consider prime power cyclotomics
# m is the cyclotomic polynomial index
def s1(m):
    return sqrt(tau(m))
```

```

# checks if cyclotomic index m is power of two
def is_pow2(m):
    return sum(m.digits(2)) == 1

# [AL21] eprint Prop 2. 2021/202
# for all  $u, v$  in  $\mathbb{R}$ ,  $|u \cdot v| / |v| \leq \gamma \cdot |u|$ 
# outputs  $T = \gamma \cdot |u|$ 
# assumes we are only testing prime powers
def expansion_factor(m, norm):
    if is_pow2(m):
        return euler_phi(m) * norm
    else:
        return 2 * euler_phi(m) * norm

# p is prime
# max_idx is max cyclotomic index
# outputs list of (m, z)
def candidates(p, min_idx=10, max_idx=200):
    # prime powers
    possible_indices = [i for i in range(min_idx, max_idx) if len(factor(i)) == 1]
    c = list()
    for m in possible_indices:
        zs = divisors(m)
        for z in zs:
            if thm1_1_cond(m, p, z):
                c.append((Integer(m), Integer(z)))
    return c

def pre_filter(q, cyclotomic_index, z, chals):
    chals_max_diff = chals[-1] - chals[0]
    phi = cyclotomic_polynomial(cyclotomic_index) # index cyclotomic polynomial
    d = phi.degree() # degree of cyclotomic

    return chals_max_diff < thm1_1_inv_bound(q, z) and log(len(chals)^d, 2).n() >= 120

def info(q, cyclotomic_index, z, chals):
    chals_norm = max({abs(c) for c in chals})
    chals_max_diff = chals[-1] - chals[0]
    phi = cyclotomic_polynomial(cyclotomic_index) # index cyclotomic polynomial
    d = phi.degree() # degree of cyclotomic
    T = expansion_factor(cyclotomic_index, chals_norm)

    print("####")
    print("Cyclotomic idx:", cyclotomic_index)
    print("Cyclotomic Poly:", phi)
    print("z:", z)
    print("Csmall norm is small enough?", chals_max_diff < thm1_1_inv_bound(q, z))
    print("Csmall large enough?", log(len(chals)^d, 2).n() >= 120)
    print("Degree of Cyclotomic:", d)
    print("Expansion Factor T:", T)
    print("Invertible Norm bound:", thm1_1_inv_bound(q, z))
    print("log(|C_Small|):", log(len(chals)^d, 2).n())
    print("Factors of Cyclotomic:", thm1_1_num_factors(z))
    print()

def possible_settings(q, chals):

```

```

for (cyclotomic_index, z) in candidates(q):
    if pre_filter(q, cyclotomic_index, z, chals):
        info(q, cyclotomic_index, z, chals)
    else:
        d = cyclotomic_polynomial(cyclotomic_index).degree()
        print(
            "[Does not satisfy challenge-set requirements] "
            "index: {}, degree: {}, z: {}, "
            "log(|C_Small|): {}, Invertible Norm bound: {}".format(
                cyclotomic_index,
                d,
                z,
                log(len(chals)^d,2).n(),
                thm1_1_inv_bound(q, z),
            )
        )

# Primes:
GL = 2^64 - 2^32 + 1
AGL = GL - 32
print("#####")
print("AGL #####")
print("#####")
# Small Challenge set
chals = [-1, 0, 1, 2]
possible_settings(AGL, chals)
print("#####")
print("M61 #####")
print("#####")
# Small Challenge set
chals = [-2, -1, 0, 1, 2]
possible_settings(2^61-1, chals)
print("#####")
print("GL #####")
print("#####")
# Small Challenge set
chals = [-2, -1, 0, 1, 2]
possible_settings(GL, chals)
print("#####")

```

B.6 Lattice Estimator Script

```

from pathlib import Path
import sys

# Initialize this dependency with: git submodule update --init
sys.path.insert(0, str(Path(__file__).resolve().parent / "lattice-estimator"))

from estimator import *
Logging.set_level(Logging.LEVEL0)

M61 = 2^61 - 1
GL = 2^64 - 2^32 + 1
AGL = GL - 32
b = 2

kappa = 15

```

```

d = 64
k = 13
K = 50
B = b^k
n_R = 2^27
n_F = 2^33
assert n_F == n_R*d

T = 128
q = AGL

n_sis = kappa*d
m_sis = n_R*d
B_l2 = sqrt(n_R*d)*(8*T*B)

params = SIS.Parameters(n=n_sis, q=q, m=m_sis,length_bound=B_l2, norm=2)
_ = SIS.estimate(params)
print((K+k)*T*(b-1) < B)

kappa = 18
d = 54
k = 14
K = 61
B = b^k
n_R = 19884107
n_F = 2^30 - 46
assert n_F == n_R*d

T=216
q = GL

n_sis = kappa*d
m_sis = n_R*d
B_l2 = sqrt(n_R*d)*(8*T*B)

params = SIS.Parameters(n=n_sis, q=q, m=m_sis,length_bound=B_l2, norm=2)
_ = SIS.estimate(params)
print((K+k)*T*(b-1) < B)

kappa = 18
d = 54
k = 14
K = 61
B = b^k
n_R = 4971026
n_F = 2^28 - 52
assert n_F == n_R*d

T = 216
q = M61

n_sis = kappa*d
m_sis = n_R*d
B_l2 = sqrt(n_R*d)*(8*T*B)

params = SIS.Parameters(n=n_sis, q=q, m=m_sis,length_bound=B_l2, norm=2)

```

```
_ = SIS.estimate(params)
print((K+k)*T*(b-1) < B)
```