

Maltese: Succinct Polynomial Commitment from Lattices

Katarina Cheng
University of Washington

Wilson Nguyen
*Microsoft Research**

Nirvan Tyagi
University of Washington

Abstract. Succinct polynomial commitment schemes are a key building block in succinct non-interactive arguments of knowledge (SNARKs). Existing succinct lattice-based polynomial commitment schemes take a “split-and-fold” strategy making use of homomorphism to compress proof size to $O(\text{polylog } N)$ for polynomials of size N . Prior works either (1) incur superlogarithmic verifier work (e.g., $O(\sqrt{N})$), (2) incur concretely large proof sizes and verifier work (e.g., hundreds of MB), or (3) rely on non-standard lattice assumptions. We propose a new succinct polynomial commitment scheme Maltese that operates over a Merkle tree commitment of the lattice-based Ajtai hash function. We employ a folding strategy and propose new sum-check-based reductions for managing norm growth of the tree commitment over folding rounds. Maltese is secure under the standard Module-SIS assumption and produces opening proofs of 335KB for multilinear polynomials of size $N = 2^{30}$; proofs are around $300\times$ smaller than prior work with polylogarithmic verifier complexity but between $2\text{-}6\times$ larger than prior work with larger verifier complexity or stronger structured assumptions.

*This work was partially completed while this author was at New York University.

Contents

1	Introduction	3
2	Technical Overview	6
2.1	Homomorphic Tree Commitments	6
2.2	Maltese’s Folding Cycle via Sum-Checks	7
2.2.1	Overview of Security Analysis	9
2.2.2	Overview of Sum-Check Protocols	9
3	Preliminaries	13
3.1	Notation	13
3.2	Algebra	13
3.3	Interactive Reductions	15
3.4	Challenge Sets and Coordinate-Wise Special Soundness	17
3.5	Ajtai Commitment Scheme	20
4	Sum-Checks Over Rings Using Isomorphisms	22
4.1	Batched Sum-Checks Over $R_{\mathbb{F}}$	23
4.2	Batched Sum-Checks Over $R_{\mathbb{K}}$	24
5	Polynomial Commitment Scheme	25
5.1	Norm Check Reduction	26
5.1.1	Norm Check Sum-Check Claims	26
5.1.2	Norm Check Reduction Protocol Description	27
5.2	Folding Reduction	29
5.2.1	Folding Sum-Check Claims	30
5.2.2	Folding Reduction Protocol Description	32
5.3	Norm Decomposition Reduction	34
5.3.1	Decomposition Sum-Check Claims	35
5.3.2	Norm Decomposition Reduction Protocol Description	37
5.4	Greyhound-Based Finishing Protocol	39
5.4.1	Finishing Protocol Relations	39
5.4.2	Preparing to Compose with Greyhound	41
5.4.3	A Modified Greyhound	45
6	Evaluation	46
6.1	Parameter Selection	46
6.2	Proof Size Accounting	47
A	Applying the PCS to different polynomials	51
A.1	Potentially high norm $R_{\mathbb{F}}$ -polynomials	51
A.2	Potentially high norm $\mathbb{F}$ -polynomials	53
B	Deferred Proofs	56
B.1	From Section 3: Schwartz-Zippel for Approximate Strong Sampling Sets	56
B.2	From Section 4: Sum-Checks Over Rings Using Isomorphisms	58
B.3	From Section 5.1: Norm Check	62
B.4	From Section 5.2: Folding	65
B.5	From Section 5.3: Norm Decomposition	81
B.6	From Section 5.4: Finishing Protocol	84
C	Average Approximate Strong Sampling Sets	87

1 Introduction

Succinct *polynomial commitment* schemes (PC) [KZG10] enable provers to commit to a polynomial (e.g., $f \in \mathbb{F}[X]$) via a small commitment and produce non-interactive *evaluation proofs* convincing a verifier of an evaluation (e.g., $f(u) = v$) that are small and fast to verify. PC schemes are a core ingredient in modern recipes for succinct non-interactive arguments of knowledge (SNARKs) in which an idealized information-theoretic proof system known as a *polynomial interactive oracle proof* (polyIOP) [CHM⁺20, BFS20, CFF⁺21, LPS25, BDF⁺25, CGKY25] is instantiated via a succinct PC scheme. Efficiency properties of SNARKs constructed through this recipe are directly tied to the efficiency properties (i.e., prover time and memory-usage, proof size, and verifier time) of the underlying PC scheme. In this work, we propose a new PC scheme with desirable efficiency properties based on lattice assumptions.

Why lattice-based cryptography? There are a number of properties that make lattice-based cryptography a particularly attractive choice for our polynomial commitment design goals.

Plausibly post-quantum secure. Structured lattice problems that underly lattice-based cryptography are considered to be plausibly post-quantum secure. Hash-based constructions provide an alternative to lattices in achieving post-quantum security, and there exists a long line of work building PC schemes via linear error-correcting codes and collision-resistant hash functions [BBHR18, BCG20, XZS22, KPV22, GLS⁺23, ZCF24, BFK⁺24, ACFY24, BCF⁺25, ACFY25, BFRW25].

Commitments that admit partial homomorphism. On the other hand, while hash-based constructions enable fast provers, they result in concretely large proof sizes. Commitment homomorphism has been a powerful tool in constructing PCs with small proof size that remain prover-efficient. PC openings employ a “split-and-fold” strategy: to obtain a $O(\text{polylog } N)$ size proof for a claim of size N , protocols perform logarithmic rounds of folding, splitting up the initial claim of size N into k (say $k = 2$) claims of size N/k , homomorphically combining them with random challenges, and then recursing [BCC⁺16, BBB⁺18, AC20, BMM⁺21, Lee21]. Lattice-based commitments can benefit from this strategy; they offer a limited form of homomorphism in which commitments are only binding for messages of low norm, meaning only a fixed number of homomorphic operations can be applied before the norm bound is exceeded.

Small field arithmetic. Lattice-based constructions can be instantiated using cyclotomic polynomial rings over small fields (e.g., 32-bit). One benefit of small fields is that they can allow more efficient arithmetization of constraints in a SNARK: if you want to prove operations over 32-bit fields, it is more efficient to encode those operations using 32-bit fields instead of, say, 256-bit fields. Another benefit is that lattice operations (i.e., matrix multiplications) over small primes are fast! The workload is highly amenable to parallel hardware optimizations such as CPU vectorization and GPU acceleration.

Existing lattice-based PCs fall short. Existing lattice-base PCs fail to simultaneously achieve both (1) provers on par in efficiency to hash-based protocols and (2) proof sizes that are concretely small and fast to verify as in group-based protocols, while (3) relying on standard lattice assumptions. A summary is provided in Figure 1. Almost all prior constructions adapt some version of the “split-and-fold” strategy [BLNS20, ACK21, AL21, BCS23, CLM23, FMN24, NS24, HSS24, AFLN24, CMNW24, KLNO24, KLNO25, KLOT25, HLSS26, KLN⁺26]. However, whenever a protocol performs a homomorphic operation (e.g., in each folding round) there is also an accompanying growth in norm. These works can be broadly categorized into a few different folding strategies for handling norm growth:

Folding without norm-fixing. One line of works [BLNS20, ACK21, BCS23, FMN24, AFLN24, CMNW24] has culminated in asymptotically succinct $O(\text{polylog } N)$ proof sizes and verifier times [CMNW24]; these works make use of *low-norm invertible subtractive sets*. That is, random challenges used for homomorphic linear combinations are drawn from a special set that ensures extractability of low-norm inputs. These sets can be chosen so that norms grow at a slow enough rate to allow for logarithmic rounds of folding (e.g., from $\{-1, 0, 1\}$). Unfortunately, such sets can only be $\text{poly}(\lambda)$ in size for security parameter λ [AL21], so soundness must be amplified via repetition. The combination of repetition and parameter selection that can accommodate norm growth results in high concrete costs for prover time and proof size.

Folding with norm-fixing. A second line of works [BS23, NS24, NOZ26, BBL⁺26] addresses norm growth by periodically (e.g. after every folding round) recommitting to a low-norm decomposition of the folded commitment, thereby “fixing” the norm for the next round of folding. Constraints to prove that the recommitment are (1) low-norm and (2) well-formed as a decomposition of the folded high-norm commitment are tied in to the claim on the new recommitment. The next round of folding proceeds on the new claim over the low-norm recommitment, and the cycle of folding and

	Protocol	Prover	Verifier	Proof Size ($N = 2^{30}$)		Assumption
<i>no norm fixing</i>	SLAP [AFLN24]	$\tilde{O}(N)$	$O(\text{polylog } N)$	$O(\text{polylog } N)$	(767MB)	MSIS
	CMNW [CMNW24]	$\tilde{O}(N)$	$O(\text{polylog } N)$	$O(\text{polylog } N)$	(>100MB)	SIS
	CMNW [CMNW24]	$\tilde{O}(N)$	$O(\sqrt[3]{N})$	$O(\sqrt[3]{N})$	(5296KB)	MSIS
<i>norm fixing</i>	LaBRADOR [BS23]	$\tilde{O}(N)$	$O(N)$	$O(\text{polylog } N)$	(60KB)	MSIS
	Greyhound [NS24]	$\tilde{O}(N)$	$O(\sqrt{N})$	$O(\text{polylog } N)$	(53KB)	MSIS
	Hachi [NOZ26]	$\tilde{O}(N)$	$O(\sqrt{N})$	$O(\text{polylog } N)$	(55KB)	MSIS
	Akita [DBK ⁺ 26]	$\tilde{O}_K(N)$	$O(N^{1/K})$	$O(\text{polylog } N, K)$	(63KB)	MSIS
<i>succinct norm fixing</i>	RoK and Roll [KLNO25]	$\tilde{O}(N)$	$O(\text{polylog } N)$	$O(\text{polylog } N)$	(3152KB)	VSIS
	SALSAA [KLOT25]	$O(N)$	$O(\text{polylog } N)$	$O(\text{polylog } N)$	(1123KB)	VSIS
	RoKoko [KLN ⁺ 26]	$O(N)$	$O(\text{polylog } N)$	$O(\text{polylog } N)$	(187KB)	VSIS
	Maltese (this work)	$O(N)$	$O(\log^3 N)$	$O(\log^2 N)$	(335KB)	MSIS

Figure 1: Summary table of costs for lattice-based polynomial commitment opening proofs categorized by norm fixing strategy. For proof size, a concrete proof size is additionally provided for a polynomial instance of size 2^{30} (i.e., represented by 2^{30} $\mathbb{F}$ elements or $2^{30}/d$ $R_{\mathbb{F}}$ elements where d is the degree of the cyclotomic polynomial).

norm-fixing continues until a constant-size base case is reached.

A downside of these works is that the verifier costs for constructing the new constraints over the low-norm recommitment are large (typically linear in the instance size). There are two approaches for proving the well-formedness and low-norm properties of the recommitment. One approach uses *random linear projections* with respect to a random matrix of low-norm elements. A vector is shown to be low-norm by taking the matrix-product of the vector with the random matrix and checking whether the projection result is low-norm [Ach03, BL17, LNS21, LNP22, GHL22]. The other approach uses the sum-check protocol [LFKN92] to check low-norm by arithmetizing the norm-check as a sum-check claim [BC25a, BC25b, NS25, KLOT25, NOZ26, KLN⁺26]. Both approaches yield verifier work linear in the instance size resulting from handling large projection matrices and/or large commitment keys.¹

Nevertheless, with norm-fixing, these constructions are open to sample folding challenges from a large invertible subtractive set dispensing with the need for repetition. Thus, they result in succinct $O(\text{polylog } N)$ proof sizes that are concretely the smallest in the literature.

Folding with succinctly-verifiable norm-fixing. Lastly, a recent line of work [KLNO24, KLNO25, KLOT25, KLN⁺26, KK26] extends folding with norm-fixing to achieve succinct $O(\text{polylog } N)$ verification. To do this, additional structure is incorporated to realize well-formedness constraints that can be verified in $O(\text{polylog } N)$ time. With regards to commitments, all of these works use a structured commitment scheme based on a newly-proposed variant of the *Module Short Integer Solution* assumption (MSIS) called the *Vanishing SIS* (VSIS) assumption [CLM23, JL25] in which the matrix used for commitment has structured rows consisting of powers of a random value. In contrast, the standard MSIS assumption consists of a matrix of random rows. One way to view the assumption is that it is hard to find a low-norm vector that, when interpreted as a univariate polynomial, evaluates to zero (i.e., vanishes) on the random points encoded in each row. The power structure of the commitment key (matrix) is well-suited for folding as a folded commitment key can be computed concisely without linear work.

The works also address succinct proofs of low-norm. RoK and Roll [KLNO25] introduces the use of a structured projection matrix that is amenable to folding. SALSAA [KLOT25], RoKoko [KLN⁺26], and Grand Danois [KK26] further incorporate sum-check protocols to achieve a linear-time prover and improve concrete efficiency.

Our contributions. We propose a new polynomial commitment scheme, Maltese, that admits a concretely fast prover, small proofs, and fast verifier. Asymptotically, Maltese has a linear-time $O(N)$ prover and $O(\text{polylog } N)$ proof size and verifier time and is secure under the standard MSIS assumption. Prior works to achieve these asymptotic properties are either (1) based on new non-standard assumptions (VSIS) [KLNO24, KLNO25, KLOT25, KLN⁺26], or (2) incur prohibitive concrete costs from repetition [AFLN24, CMNW24].

Our starting point is the “Merkle tree” Ajtai commitment scheme used by Cini et al. [PSTY13, LLNW16, CMNW24] secure under MSIS. Maltese adopts the folding with norm-fixing strategy; our main contributions are new techniques for folding tree commitments and producing succinctly-verifiable proofs for decomposition and low-norm over these tree commitments:

¹Greyhound [NS24] and subsequent work [NOZ26, BBL⁺26] achieve $O(\sqrt{N})$ verifier time for an instance of size N by performing an initial folding step of arity $k = \sqrt{N}$.

- *Sum-check-based low-norm check for tree commitments*: Previous work has employed sum-check for proving low-norm in single-layer commitments [BC25a, BC25b, NS25, KLOT25, NOZ26]. We extend the sum-check norm checking strategy to work over the tree commitment structure. Here, the norm bound needs to be enforced for the opening of *every layer* of the tree. This protocol ultimately reduces to a polynomial evaluation claim on *every layer* of the tree.
- *Sum-check-based folding for tree commitments*: Cini et al. propose a folding strategy for an evaluation claim on the leaf layer of a tree commitment [CMNW24]. To be compatible with our low-norm check, we propose a new folding strategy to efficiently fold evaluation claims over every layer of the tree commitment.
- *Sum-check-based decomposition for tree commitments*: After folding, we fix the norm growth by decomposing the folded commitment into a low-norm decomposition. Decomposition in previous works using single-layer commitments is often straightforward, requiring only a linear recomposition check of the claimed decomposition [BS23, BC25a, NS25]. Unfortunately, the tree commitment structure does not maintain decomposition linearity across layers. Instead we propose a new strategy for decomposing every layer of the tree commitment by recommitting to the layer decompositions together in the leaves of a new tree commitment. A sum-check protocol is then used to verify the well-formedness of the recommitment.

Overall, our new polynomial commitment scheme enjoys the following desirable properties that make it a good candidate for instantiating polyIOP-based SNARKs:

- *Small proof size*: The proof consists of $O(\log^2 N)$ ring elements. For a polynomial of size 2^{30} , the proof size is 335KB.
- *Efficient verifier*: The verifier performs $O(\log^3 N)$ ring operations. Outside of that, the verifier performs a relatively lightweight $O(\log^2 N)$ extension field operations for sum-check verification.
- *Linear-time efficient prover*: The prover is concretely efficient incurring $O(N)$ ring operations.
- *Transparent setup with constant-size public parameters*: The public parameters of our protocol consist only of a constant-size ($O(\lambda)$) matrix for an Ajtai commitment. The security is based on the binding of the commitment which relies on the standard MSIS assumption.
- *Slack-free extraction*: Our opening protocol extracts an exact commitment opening, i.e., without slack, allowing for immediate composition with higher level polyIOPs.
- *Efficient encoding for small field polynomials*: As mentioned before, small field arithmetizations can be more efficient for encoding many common types of statements in SNARKs, e.g., the ones that arise in proving virtual machine execution [AST24]. Our PC supports efficient encoding of polynomials over small fields (e.g., 32-bit) into ring elements.
- *Extension-field evaluation*: PolyIOPs over small fields either require to repeat evaluation challenges or, more efficiently, sample challenges from a larger extension field for soundness [BGKS20]. Our PC supports providing opening proofs for evaluations on extension field points.
- *Free sparse commitment*: Commitment to the witness (i.e., the polynomial) is often a significant cost in SNARK provers. For many applications, it is common for a witness to be sparse, i.e., include primarily zero values [ST25]. Our PC enables “free” commitment to zero values by omitting matrix-row multiplications during commitment.

Concurrent work. Akita [DBK⁺26] presents an alternative approach to achieving concretely efficient verification in lattice-based polynomial commitments. As in Maltese, Akita achieves this relying only on the standard MSIS assumption. Akita builds on the Greyhound [NS24] square root folding strategy. Instead of folding $\sqrt{N}$ commitments of size $\sqrt{N}$, Akita folds $N^{1/K}$ commitments of size $N/N^{1/K}$ for a fixed small constant K , then provides techniques to recurse on the smaller instance of size $N/N^{1/K}$. In Figure 1, we categorize Akita in the “non-succinct norm-fixing” category as the verifier does not achieve asymptotic polylogarithmic time, as K must be chosen to be small. Further, Akita relies on a public setup that takes work $O(N/N^{1/K})$ to generate. Nevertheless, in practice, implementation and evaluation of Akita demonstrate concretely that verifier time is comparable to RoKoko [KLN⁺26] (and, we expect, to Maltese).

2 Technical Overview

2.1 Homomorphic Tree Commitments

To begin, consider the homomorphic Ajtai commitment scheme [Ajt96] instantiated over the cyclotomic polynomial ring $R_{\mathbb{F}} := \mathbb{F}[X]/(\phi(X))$ where $\phi(X)$ is a cyclotomic polynomial of degree d . To commit to a vector $f \in R_{\mathbb{F}}^n$, commitment t and opening s are generated as:

$$t := A \cdot G_n^{-1}(f), \quad s := G_n^{-1}(f),$$

where $A \in R_{\mathbb{F}}^{\kappa \times n\alpha}$ is a public uniformly-random matrix, $G_n^{-1} : R_{\mathbb{F}}^n \rightarrow R_{\mathbb{F}}^{n\alpha}$ is the gadget decomposition function that decomposes each ring element into its base- b low-norm representation with $\alpha := \log_b(|\mathbb{F}|)$, and $\kappa := O(\lambda)$ is derived from the security parameter λ (see preliminaries in Section 3). To open a commitment t , one reveals message $f' \in R_{\mathbb{F}}^n$ and opening $s' \in R_{\mathbb{F}}^{n\alpha}$ such that:

$$t = A \cdot s', \quad f' = G \cdot s', \quad \|s'\|_{\infty} \leq b,$$

where $G \in R_{\mathbb{F}}^{n\alpha \times n}$ is the gadget recomposition matrix; the commitment is binding under the MSIS assumption with respect to norm-bound b' . Note this commitment is homomorphic in that given (t_0, s_0, f_0) and (t_1, s_1, f_1) and low-norm $c \in R_{\mathbb{F}}$, one can compute commitment $t' := t_0 + c \cdot t_1$ to message $f' := f_0 + c \cdot f_1$ with opening $s' := s_0 + c \cdot s_1$ that is valid as long as $\|s'\|_{\infty} < b'$.

This commitment scheme can be naturally extended to a tree structure that commits to data values at the “leaves” and compresses commitments up the tree to a single “root” commitment; one can view this as a Merkle tree using hash function $H(m) = A \cdot G^{-1}(m)$. Consider a binary tree and public parameters $A \in R_{\mathbb{F}}^{n \times 2n\alpha}$ (our constructions generalize to arbitrary arity > 2). To commit to a vector $f \in R_{\mathbb{F}}^{2^\ell n}$, we first partition f into $2^{\ell-1}$ pairs of size- $2n$ vectors, commit to each pair using the Ajtai commitment scheme to get $2^{\ell-1}$ size- n commitments, then repeat this process $\ell - 1$ more times until we get a single size- n root commitment t :

$$\begin{aligned} \text{TCom.Commit}(A \in R_{\mathbb{F}}^{n \times 2n\alpha}, f \in R_{\mathbb{F}}^{2^\ell n}) &\rightarrow (t \in R_{\mathbb{F}}^n, [s_i \in R_{\mathbb{F}}^{2^{i-1}n\alpha}]_{i \in [\ell]}): \\ s_\ell &:= G_{2^\ell n}^{-1}(f), \quad \forall i \in [\ell-1, \dots, 1] \quad s_i := G_{2^i n}^{-1}((I_{2^i} \otimes A) \cdot s_{i+1}), \quad t := A \cdot s_1. \end{aligned}$$

Now an opening of t to f consists of the full set of opening vectors $[s_i]_{i \in [\ell]}$:

$$\begin{aligned} \text{TCom.Open}_b(A, t, f, [s_i]_{i \in [\ell]}) &\rightarrow b \in \{0, 1\}: \\ f &= G \cdot s_\ell, \quad t = A \cdot s_1, \quad \forall i \in [\ell-1] \quad (I_{2^i} \otimes A) \cdot s_{i+1} = G \cdot s_i, \quad \forall i \in [\ell] \quad \|s_i\|_{\infty} \leq b. \end{aligned}$$

Further, the homomorphic property still holds, in that, the openings for each layer of the tree can be linearly combined to create openings for a homomorphically derived root commitment.

Polynomial evaluations via folding. Define $\tilde{f} \in R_{\mathbb{F}}[X_1, \dots, X_{\ell+\log n}]$ to be the unique multilinear extension polynomial of $f \in R_{\mathbb{F}}^{2^\ell n}$ with evaluations defined over the hypercube $\{0, 1\}^{\ell+\log n}$. A polynomial evaluation proof that $\tilde{f}(u) = v$ for a tree commitment is a proof for the relation ²:

$$\text{PE}(\ell, b) := \left\{ \begin{array}{l} \text{x} := (t \in R_{\mathbb{F}}^n, u \in R_{\mathbb{F}}^{\ell+\log n}, v \in R_{\mathbb{F}}) \\ \text{w} := ((s_i \in R_{\mathbb{F}}^{2^{i-1}n\alpha})_{i \in [\ell]}, f \in R_{\mathbb{F}}^{2^\ell n}) \end{array} : \begin{array}{l} \text{TCom.Open}_b(A, t, f, [s_i]_{i \in [\ell]}) = 1 \\ \tilde{f}(u) = v \end{array} \right\} \quad (1)$$

Prior works employ a FRI-based folding strategy [BBHR18] to prove evaluation claims over polynomials committed within homomorphic tree commitments [FMN24, CMNW24]. To see how this works, consider a simplified presentation without worrying about norm bounds for witness extraction; we will address norm bounds shortly.

Denote $\text{x} := (t, u, v)$, $\text{w} := [s_i]_{i \in [\ell]}$, and $f = G \cdot s_{1,\ell}$, indicating the inputs for the first round of folding. The prover sends the opening s_1 for the first layer consisting of the two children of the root commitment t to the verifier. The verifier checks $t = A \cdot s_1$ and $\|s_1\|_{\infty} \leq b$ then computes $(t'_0, t'_1) \in R_{\mathbb{F}}^{2 \times n} = G \cdot s_1$ as the left and right children of the root, respectively. Observe two things about t'_0 and t'_1 :

1. By the recursive tree commitment structure, t'_0 and t'_1 are tree commitments to the left and right subtrees of the original tree commitment committing to the left-right vectors $(f'_0, f'_1) \in R_{\mathbb{F}}^{2 \times 2^{\ell-1}n} \leftarrow f$, respectively.

²For the purposes of the overview, we use simplified relations. The complete relations are defined in Section 5.

2. By the properties of multilinear extensions:

$$\tilde{f}(X_1, X_2, \dots, X_{\ell+\log n}) = (1 - X_1) \cdot \tilde{f}'_0(X_2, \dots, X_{\ell+\log n}) + X_1 \cdot \tilde{f}'_1(X_2, \dots, X_{\ell+\log n}).$$

This gives rise to the following folding protocol. Define evaluation point $(u_1 \in R_{\mathbb{F}}, u' \in R_{\mathbb{F}}^{\ell+\log n-1})$ by partitioning u into its first coordinate u_1 and the remaining coordinates u' . The prover computes and sends claimed evaluations $v'_0 := \tilde{f}'_0(u')$ and $v'_1 := \tilde{f}'_1(u')$ to the verifier. The verifier checks that the claimed evaluations are consistent with the original evaluation claim by checking $(1 - u_1) \cdot v'_0 + u_1 \cdot v'_1 = v$. Then, the verifier samples and sends a folding challenge $c \leftarrow \mathcal{C}$ to the prover (from some challenge space that we will specify later) to combine the two new polynomial evaluation claims into a single folded claim $(x', w') \in \text{PE}(\ell - 1, b')$ of half the size of the original claim:

$$\begin{aligned} x'_0 &:= (t'_0, u', v'_0), & w'_0 &:= [s_{i,0}]_{i \in [2, \ell]} \\ x'_1 &:= (t'_1, u', v'_1), & w'_1 &:= [s_{i,1}]_{i \in [2, \ell]} \\ x' &:= \begin{pmatrix} t' \leftarrow t'_0 + c \cdot t'_1, \\ u', \\ v' \leftarrow v'_0 + c \cdot v'_1 \end{pmatrix}, & w' &:= [s'_{i-1} \leftarrow s_{i,0} + c \cdot s_{i,1}]_{i \in [2, \ell]} \end{aligned}$$

In the above, openings $s_{i,0}$ and $s_{i,1}$ correspond to the left and right halves of openings s_i for the original commitment. The new claim is derived by the prover and verifier homomorphically combining the left and right subtree commitments. With high probability (if the folding challenge is sampled from a sufficiently large space), the new claim is valid only if the original claim is valid. The folding protocol repeats for ℓ rounds until the claim is reduced to a single evaluation claim over a size- n polynomial, which can be checked by directly opening the commitment.

Addressing norm growth while folding. The challenge in completing the above protocol intuition is in managing norm growth. At every folding round, the norm of the witnesses grows from b to $b + Tb$ where T is the expansion factor of the challenge space; if the norm grows too large, binding of the commitment is lost and the protocol becomes unsound. The expansion factor of a space is the maximum norm increase of a ring element when multiplied by any element from that space. Cini et al. [CMNW24] use a challenge space of $\mathcal{C} := \{-1, 0, 1\}$ to ensure the norm does not grow too large, but due to the small challenge space, the protocol requires repetition to achieve soundness, resulting in concretely expensive provers and large proof sizes.

Alternatively, one may draw challenges from an exponential-sized challenge set, but the set must be chosen to ensure witness extraction is still possible. With folding, extraction corresponds to solving a linear system of equations using $(c - c')^{-1}$ where c and c' are two distinct challenges acquired from rewinding. It is possible to select such a set so that every $c \in \mathcal{C}$ is of low-norm and every difference $c - c'$ has an inverse, however, the inverse may be of arbitrary norm. This has two consequences: (1) The norm $\|c\|_{\infty}$ is small but not so small that norm growth is manageable for many rounds of folding, and (2) the norm $\|(c - c')^{-1}\|_{\infty}$ may be large meaning the extracted witness may be of large norm and additional norm-checking techniques are needed to ensure soundness (e.g., random projections and/or sum-checks). Indeed, Cini et al. [CMNW24] propose a variant of their protocol in this paradigm but are only able to support constant $\ell = 3$ folding rounds, resulting in verifier complexity and proof size of $O(\sqrt[3]{N} = 2^{\ell}n)$.

To support logarithmic (or more) folding rounds, prior works [BS23, KLNO24, KLNO25, KLOT25, KLN⁺26] employ a cycle of three reductions that resets norm growth between folding rounds. The cycle consists of (1) a norm checking reduction, (2) a folding reduction, and (3) a norm decomposition reduction:

- *Norm check:* The norm check is important for extraction (e.g., backwards execution) and ensures the high-norm witness extracted from folding is “fixed” to a low-norm witness to continue extraction for the previous folding round.
- *Folding:* Folding enables succinctness. The folding reduction takes an evaluation claim and folds it into a claim of smaller size but with larger norm.
- *Norm decomposition:* Norm decomposition is important for forward execution and ensures the high-norm witness output from folding is “fixed” by recommitting into a new low-norm commitment for the next folding round.

Maltese instantiates this cycle of reductions for the homomorphic tree commitment described above.

2.2 Maltese’s Folding Cycle via Sum-Checks

A key challenge in Maltese in working over the tree commitment is that norm checking and norm decomposition must be performed over the openings for *every layer of the tree*. As such, in addition to considering the polynomial

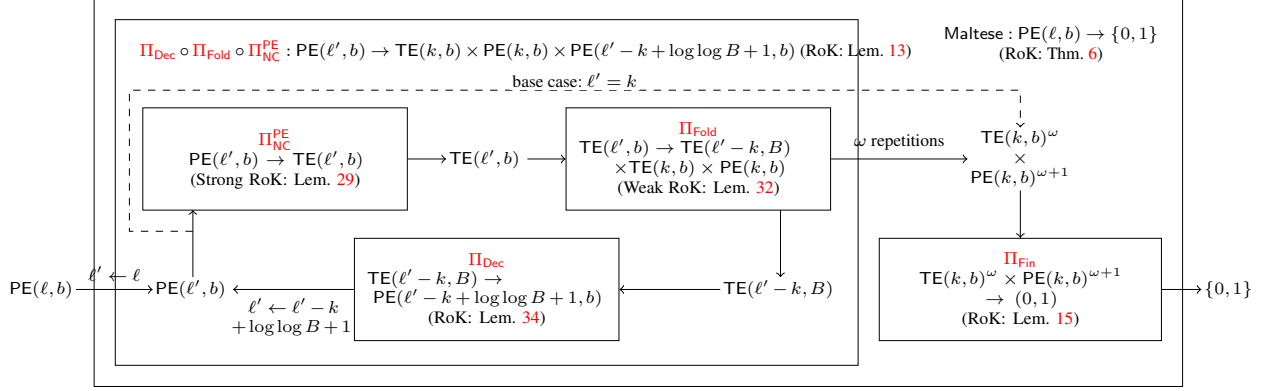


Figure 2: Overview of the Maltese polynomial commitment opening protocol as a composition of reductions of knowledge.

evaluation relation $\text{PE}(\ell, b)$ (Eq. 1), we also consider the following *tree evaluation* relation $\text{TE}(\ell, b)$ for checking polynomial evaluation claims for each layer of the tree:

$$\text{TE}(\ell, b) := \left\{ \begin{array}{l} x := \left(\begin{array}{l} t \in R_{\mathbb{F}}^n, \\ u \in R_{\mathbb{F}}^{\ell+1+\log n}, \\ v \in R_{\mathbb{F}} \end{array} \right), \\ w := ((s_i \in R_{\mathbb{F}}^{2^i n})_{i \in [\ell]}, f \in R_{\mathbb{F}}^{2^\ell n}) \end{array} \right\} : \left\{ \begin{array}{l} \text{TCom.Open}_b(A, t, f, [s_i]_{i \in [\ell]}) = 1 \\ s := 0^{2n} \| s_1 \| \dots \| s_\ell \in R_{\mathbb{F}}^{2^{\ell+1} n} \\ \tilde{s}(u) = v \end{array} \right\} \quad (2)$$

Our proving strategy in Maltese makes heavy use of the sum-check protocol [LFKN92] to perform norm checking, high-arity folding, and norm decomposition. Sum-check protocols allow us to check expressive properties over committed values. The sum-check protocol outputs a polynomial evaluation claim on the committed values. As such, our reductions will pass PE and TE claims as input and output relations. These can be thought of as the output of a sum-check protocol checking some property over the leaf openings and internal layer openings of the tree commitment, respectively.

All together, Maltese's folding cycle consists of the following reductions (see Figure 2):

- $\Pi_{\text{NC}}^{\text{PE}} : \text{PE}(\ell, b) \rightarrow \text{TE}(\ell, b)$: The norm check reduction performs sum-checks to check norm constraints $\|s_i\|_\infty < b$ for every layer $i \in [\ell]$ and also checks the original evaluation claim $\tilde{f}(u) = v$. As so, the sum-check outputs polynomial evaluation claims on every layer of the tree which corresponds to a new claim in $\text{TE}(\ell, b)$.
- $\Pi_{\text{Fold}} : \text{TE}(\ell, b) \rightarrow \text{TE}(\ell - k, B) \times \text{TE}(k, b) \times \text{PE}(k, b)$: The folding reduction performs folding over the tree commitment and evaluation claims to reduce the size of the claim to a tree of height $\ell - k$ for some folding parameter k (i.e., a tree of size $2^k \times$ smaller). Our protocol departs from the folding protocol of Cini et al. [CMNW24] in two ways. First, we support higher arity folding by opening k layers of the tree at once and folding together 2^k subtrees of the original tree commitment. Second, since this is a tree relation, evaluation claims are over the concatenated openings so the simple linear combination approach of Cini et al. no longer applies. We provide a special sum-check-based folding protocol to separate the tree evaluation claim on the initial tree into a claim on the top k layers of the tree and a claim on the bottom $\ell - k$ layers of the tree. The claim on the top layers is in $\text{TE}(k, b)$. The claim on the bottom layers, representing 2^k subtrees, are folded together to produce a new claim for the folded tree in $\text{TE}(\ell - k, B)$ of higher norm $B > b$. Lastly an additional claim in $\text{PE}(k, b)$ is created to aid in folding the 2^k subtrees. The folding cycle continues with the folded claim in $\text{TE}(\ell - k, B)$ and the auxiliary claims in $\text{TE}(k, b)$ and $\text{PE}(k, b)$ are set aside to be handled at the end of folding.
- $\Pi_{\text{Dec}} : \text{TE}(\ell - k, B) \rightarrow \text{PE}(\ell - k + \log \log(B) + 1, b)$: The folded claim is over a tree commitment with high-norm openings B . Consider the flattened vector of all layer openings for the tree commitment $(s_1, \dots, s_{\ell-k}) \in R_{\mathbb{F}}^{2^{\ell-k+1} n}$. The norm decomposition reduction commits in a fresh low-norm tree commitment to the decomposition of the flattened vector of openings of size $2^{\ell-k+1} n \cdot \log B$; a sum-check is used to prove the correctness of the decomposition with respect to the original tree commitment. This results in a polynomial evaluation claim on the new low-norm tree commitment in $\text{PE}(\ell - k + \log \log(B) + 1, b)$. This claim is passed back to the norm check reduction, and the folding cycle continues. Note that as long as $k > \log \log(B) + 1$, the folding cycle has made progress in reducing the size of the claim.

- $\Pi_{\text{Fin}} : \text{TE}(k, b)^\omega \times \text{PE}(k, b)^{\omega+1} \rightarrow (0, 1)$: Finally, after ω cycles of folding, the norm-decomposition protocol will output a base case claim in $\text{PE}(k, b)$. Note if k does not divide perfectly, the final cycle may pad the tree to output a tree of the correct shape. The finishing protocol gathers this claim together with the ω auxiliary claims output in each folding round. A sum-check is performed to unify the claims onto the same evaluation point so that they may be linearly-combined into a single claim in $\text{TE}(k, B')$. From here, depending on k , the claim can be recursed on via a folding cycle with a smaller folding parameter k' , or the claim can be directly checked by opening the tree commitment and checking the evaluation, or another proving strategy can be taken for the small base case claim. In our evaluation, we make use of the Greyhound proof system [NS24] to resolve the base claim.

2.2.1 Overview of Security Analysis

The security analysis takes a modular approach using the reductions of knowledge (RoK) framework for interactive reductions [KP23, Kot24]. We consider the extended properties of *weak* and *strong* RoKs introduced by Nguyen and Setty [NS25] to abstract the security analysis commonly seen in lattice-based protocols.

Recall from the above discussion, folding challenges are drawn from a set $\mathcal{C}$ such that every difference $c - c'$ for challenges $c, c' \in \mathcal{C}$ has an inverse, but the inverse may be of large norm. Therefore, extracted witnesses are not in the original low-norm relation TE but instead belong to a related high-norm *relaxed* relation TE^* . This *relaxed extraction* property along with a *unique extraction* property that we derive from the relaxed binding of Ajtai commitments provides us:

Lemma 32 (Informal). $\Pi_{\text{Fold}} : \text{TE}(\ell, b) \rightarrow (\text{TE}^*(\ell, B')) \rightarrow \text{TE}(\ell - k, B) \times \text{TE}(k, b) \times \text{PE}(k, b)$ is a *weak RoK*.

Looking forward, to instantiate the protocol and optimize parameters, as in prior work, we can consider a challenge space $\mathcal{C}$ where not every difference $c - c'$ has an inverse, but it is unlikely to find a non-invertible pair [GLLO26, KLN⁺26]. We identify a subtle bug in the extraction analysis of prior work [GLLO26] in which the joint distribution of accepting challenges and non-invertible challenges is not handled properly. We provide a new analysis approach to handle this subtlety; see Section 3.4 for further remarks.

Returning to the Maltese folding cycle, to handle weak extraction of the folding reduction, weak RoKs are composed with strong RoKs that can extract a witness of the original relation even from a prover and witness for the relaxed relation. Our sum-check-based norm checking protocol has this *strong extraction* property to fix the high-norm witness:

Lemma 29 (Informal). $\Pi_{\text{NC}}^{\text{PE}} : \text{PE}(\ell, b) \rightarrow \text{TE}(\ell, b) \rightarrow (\text{TE}^*(\ell, B'))$ is a *strong RoK*.

We show that the decomposition protocol stands on its own as a knowledge-sound RoK. Then the full folding cycle, using the weak-strong composition theorem of Nguyen and Setty [NS25], is a knowledge-sound RoK as well:

Lemma 34 (Informal). $\Pi_{\text{Dec}} : \text{TE}(\ell - k, B) \rightarrow \text{PE}(\ell - k + \log \log(B) + 1, b)$ is a *RoK*.

Lemma 13 (Informal). $\Pi_{\text{Dec}} \circ \Pi_{\text{Fold}} \circ \Pi_{\text{NC}}^{\text{PE}} : \text{PE}(\ell, b) \rightarrow \text{PE}(\ell - k + \log \log(B) + 1, b) \times \text{TE}(k, b) \times \text{PE}(k, b)$ is a *RoK*.

Lastly, we show the finishing protocol is a knowledge-sound RoK. The knowledge-sound finishing protocol, together with repeated application of the knowledge-sound folding cycle, gives us the knowledge-soundness of the Maltese polynomial evaluation opening protocol:

Lemma 15 (Informal). $\Pi_{\text{Fin}} : \text{TE}(k, b)^\omega \times \text{PE}(k, b)^{\omega+1} \rightarrow (0, 1)$ is a *RoK*.

Theorem 6 (Informal). *Maltese* : $\text{PE}(\ell, b) \rightarrow (0, 1)$ is a *RoK*.

2.2.2 Overview of Sum-Check Protocols

Maltese largely uses existing arithmetization techniques to encode the many constraints over the tree commitments in the described reductions. The full arithmetization of the constraints are extensive and are provided in detail in the protocol descriptions of Section 5. In this section, we will provide an overview of the sum-check constraints and where they come from.

In terms of efficiency, this section will propose many sum-checks over polynomials in $R_{\mathbb{F}}[X]$. Naively, each would incur a proof including a logarithmic (in the size of the polynomial) number of $R_{\mathbb{F}}$ elements. When compiling the protocol we make use of standard batching and NTT mapping techniques to reduce the costs of sum-checks over $R_{\mathbb{F}}$ polynomials, instead performing a much more efficient sum-check over an extension field polynomial [NS25, BC25a, KLN⁺26]. We

detail this approach in Section 4; we define a procedure BatchSC which batches sum-check claims over $R_{\mathbb{F}}$, runs a batched sum-check over the extension field, and outputs an evaluation claim over $R_{\mathbb{F}}$.

Sum-check-based norm check. The first reduction checks the norm bounds on the openings of the tree commitment:

$$\forall i \in [\ell] \quad \|s_i\|_{\infty} < b.$$

It proceeds via two sum-checks. We first employ the norm-checking strategy of Nguyen and Setty [NS25] to check the infinity norm of $s_i \in R_{\mathbb{F}}^{2^i n \alpha}$ by constraining that each $\mathbb{F}$ element in the d -coefficient representation of each $R_{\mathbb{F}}$ element is bounded. Recall that for cyclotomic rings, each element $z \in R_{\mathbb{F}} := \mathbb{F}[X]/\phi(X)$ can be represented as a vector of elements $(a_1, \dots, a_d) \in \mathbb{F}^d$ (i.e., $a_0 + a_1 X + \dots + a_{d-1} X^{d-1}$) for cyclotomic polynomial $\phi(X) = X^d + 1$ of degree d . Consider $s \leftarrow 0^{2n\alpha} \|s_1\| \dots \|s_{\ell} \in R_{\mathbb{F}}^{2^{\ell+1} n \alpha}$. Denote $\text{cf}(s) \in \mathbb{F}^{2^{\ell+1} n \alpha \times d}$ as the matrix of coefficients and $\text{cf}(s, j) \in \mathbb{F}^{2^{\ell+1} n \alpha}$ as the vector of all the j^{th} coefficients for $j \in [d]$ for vector s . A batched sum-check is used to test that the norm-check polynomial applied to each coefficient vector evaluates to zero over the evaluation domain (i.e., a zero check). Note the norm-check polynomial NC evaluates to zero only if the value lies in the range $[-b, b]$.

$$\text{NC}(X) := X \cdot \prod_{b=1}^{b-1} ((X-b) \cdot (X+b)), \quad \text{NC}(\widetilde{\text{cf}(s, j)}(X)) = 0 \quad \text{over } X \in \{0, 1\}^{\ell+1+\log n \alpha d}.$$

To achieve proper soundness this sum-check is performed over a sufficiently large extension field $\mathbb{K}$ of $\mathbb{F}$ (e.g., quadratic extension) resulting in evaluation claims for $\widetilde{\text{cf}(s, j)}(X)$ over some point $r_{\mathbb{K}} \in \mathbb{K}^{\ell+1+\log n \alpha d}$. Unfortunately, it is not immediately clear how to handle evaluation claims over these coefficient polynomials; the folding strategy introduced earlier for homomorphic trees operates over commitments and claims on $R_{\mathbb{F}}$.

We make the following observation about the structure of these algebraic elements. Consider the analogous cyclotomic ring $R_{\mathbb{K}} := \mathbb{K}[X]/\phi(X)$ where $R_{\mathbb{F}} \subseteq R_{\mathbb{K}}$ and $\mathbb{K} \subseteq R_{\mathbb{K}}$ are subrings of $R_{\mathbb{K}}$. We show that the d evaluation claims on $\mathbb{K}[X]$ polynomials on each $\widetilde{\text{cf}(s, j)}(r_{\mathbb{K}})$ for $j \in [d]$ can be “packed back” into a single evaluation claim on $\widetilde{s}(r_{\mathbb{K}})$ interpreted as a $R_{\mathbb{K}}[X]$ polynomial claim. Here, the j^{th} coefficient in $\mathbb{K}$ of the resulting evaluation in $R_{\mathbb{K}}$ is itself the evaluation of the j^{th} coefficient polynomial:

Lemma 14 (Informal). $\forall j \in [d], \text{cf}(\widetilde{s}(r_{\mathbb{K}}), j) = \widetilde{\text{cf}(s, j)}(r_{\mathbb{K}}).$

This packing results in a single evaluation claim over $R_{\mathbb{K}}$. To get us back to an evaluation claim over $R_{\mathbb{F}}$, we perform a second sum-check simply checking the pending $R_{\mathbb{K}}$ evaluation claim on s (including the original claim over f) ultimately resulting in new polynomial evaluation claims over $\widetilde{s}$ unified over the same point. This second sum-check claim is over $R_{\mathbb{K}}$ due to the evaluation point $r_{\mathbb{K}} \in \mathbb{K}$ and the coefficients $s \in R_{\mathbb{F}}^{2^{\ell+1} n \alpha}$. However, if the sum-check challenges are drawn from $R_{\mathbb{F}} \subseteq R_{\mathbb{K}}$, the polynomial evaluation claims produced at the end are over $R_{\mathbb{F}}$. We show how to shift this sum-check claim from $R_{\mathbb{K}}$ into two sum-check claims over $R_{\mathbb{F}}$ in a new subprotocol we call ShiftSC (Section 4.2). These are in turn batched and decomposed into sum-checks over $\mathbb{K}$ via BatchSC; this avoids sending $R_{\mathbb{K}}$ and $R_{\mathbb{F}}$ elements as part of the sum-check proof. All together, the norm check outputs a unified $R_{\mathbb{F}}$ evaluation claim on $\widetilde{s}$ which directly corresponds to a claim in $\text{TE}(\ell, b)$.

Sum-check-based high-arity folding. The folding protocol departs from the 2-to-1 folding protocol given in Section 2.1 in two ways: (1) To support evaluation claims on every layer of the tree (i.e., a claim in TE instead of PE), and (2) to support opening k layers of the tree and folding 2^k -to-1 subtrees.

First consider the following high-arity strawman protocol which makes clear the connection to the original 2-to-1 folding protocol. Given a folding parameter k , the prover opens k layers of the tree commitment t to produce $[t'_j]_{j \in [2^k]}$ commitments to the 2^k subtrees each of size $N/2^k$ where $N = 2^{\ell} n$ is the size of the original tree of height ℓ . For a claim in PE, it can be checked by the same multilinear structure on the 2^k chunks of the original vector $f \in R_{\mathbb{F}}^{2^{\ell} n}$, where each chunk $f'_j \in R_{\mathbb{F}}^{2^{\ell-k} n}$ is committed to in the subtree of t'_j . Below, $\text{bits}(j)$ is the length k bit vector representation of $j \in [2^k]$:

$$\widetilde{f}(X_1, \dots, X_{\ell+\log n}) = \sum_{j \in [2^k]} \widetilde{f}'_j(X_{k+1}, \dots, X_{\ell+\log n}) \cdot \prod_{b \in [k]} \left(\text{bits}(j)_b X_b + (1 - \text{bits}(j)_b)(1 - X_b) \right) \quad (3)$$

Folding proceeds as before. The prover provides claimed evaluations v'_j for all $\widetilde{f}'_j$. The verifier checks the original claim via the multilinear structure above and then provides folding challenges $(c_1, \dots, c_{2^k})$ to combine the subtrees and

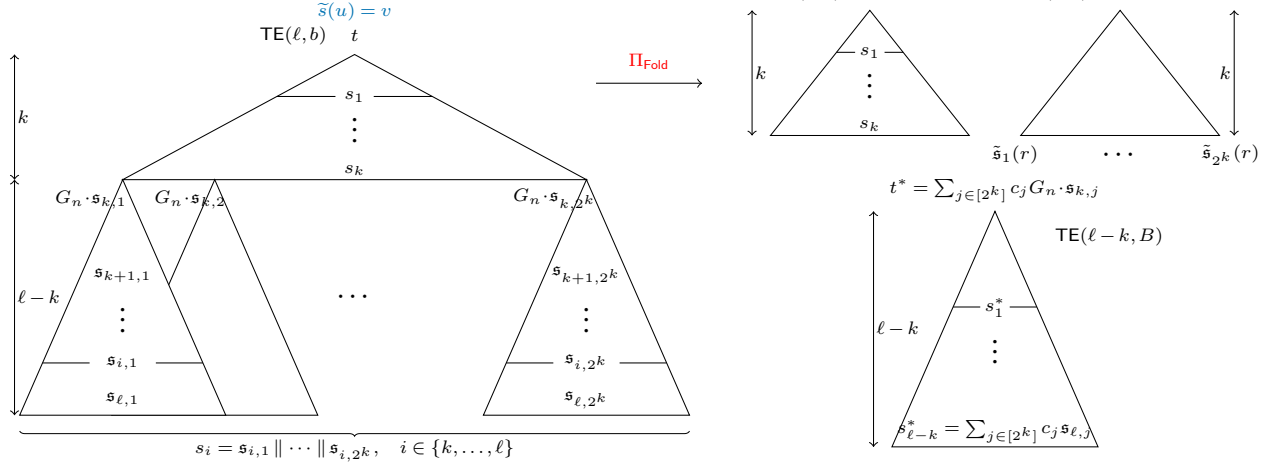


Figure 3: Visualization of $\Pi_{\text{Fold}} : \text{TE}(\ell, b) \rightarrow \text{TE}(\ell-k, B) \times \text{TE}(k, b) \times \text{PE}(k, b)$. The initial tree of ℓ layers is committed to in t and the initial tree evaluation claim $\tilde{s}(u) = v$ is for the layer openings $s = 0^{2^{n\alpha}} \| s_1 \| \dots \| s_\ell$. The folding protocol produces claims on (1) the folded subtrees with $\ell-k$ layers committed to in new commitment t^* , (2) the top k layers of the initial tree committed to in t , and (3) a new commitment t_{subs} of k layers that commits to evaluations of the 2^k subtrees.

claims into a single claim on folded subtree t_k :

$$t' := \sum_{j \in [2^k]} c_j \cdot t'_j, \quad \forall i \in [k+1, \ell] \quad v' := \sum_{j \in [2^k]} c_j \cdot v'_j. \quad (4)$$

Unfortunately, this strawman generalization of Cini et al. folding does not work for claims in TE. The evaluation claim in TE is over the concatenated openings of all layers of the tree, not just the leaf openings, and the multilinear structure of Eq. (3) does not hold for the concatenated openings. Instead, we take an entirely different approach for folding the tree evaluation claim by separating out components of the evaluation claim for the top k layers of the tree and the bottom $\ell-k$ layers of the tree. Consider the following writing the evaluation of the concatenated s in its multilinear extension form where $s^{(j)}$ denotes the j^{th} entry of s :

$$\begin{aligned} \tilde{s}(u) &= \sum_{j=0}^{2^{\ell+1}n\alpha-1} s^{(j)} \cdot \tilde{eq}(\text{bits}(j), u) \\ &= \underbrace{\sum_{j=0}^{2^{k+1}n\alpha-1} s^{(j)} \cdot \tilde{eq}(\text{bits}(j), u)}_{\text{top } k \text{ layers indices}} + \underbrace{\sum_{j=2^{k+1}n\alpha}^{2^{\ell+1}n\alpha-1} s^{(j)} \cdot \tilde{eq}(\text{bits}(j), u)}_{\text{bottom } \ell-k \text{ layers indices}} \\ &= \tilde{eq}(0^{\ell-k}, u^{(:, \ell-k)}) \cdot \underbrace{\tilde{s}_{\text{top}}(u^{(\ell-k,:)})}_{\text{top tree claim}} + \underbrace{\sum_{j=2^{k+1}n\alpha}^{2^{\ell+1}n\alpha-1} s^{(j)} \cdot \tilde{eq}(\text{bits}(j), u)}_{\text{bottom subtrees claim}} \end{aligned} \quad (5)$$

As so, a claim on the top k layers of the tree can be factored out where $s_{\text{top}} = 0^{2^{n\alpha}} \| s_1 \| \dots \| s_k$ and the claim lies in $\text{TE}(k, b)$. The second summation represents the bottom $\ell-k$ layers of the tree which can be further factored into claims on each of the 2^k subtrees:

$$\sum_{j=2^{k+1}n\alpha}^{2^{\ell+1}n\alpha-1} s^{(j)} \cdot \tilde{eq}(\text{bits}_{\ell+1+m}(j), u) = \sum_{j=1}^{2^k} \sum_{i=1}^{\ell-k} \tilde{eq}(0^{\ell-(k+i)} \| 1 \| \text{bits}_k(j-1), u^{(:, \ell-i+1)}) \cdot \tilde{s}_{k+i,j}(u^{(\ell-i+1,:)}) \quad (6)$$

In the above, $s_{i,j}$ for $i \in [\ell-k]$ and $j \in [2^k]$ denotes the opening for the i^{th} layer of the j^{th} subtree.

$$\forall i \in [0, \ell-k] \quad s_{k+i} = s_{k+i,1} \in R_{\mathbb{F}}^{2^{i n\alpha}} \| \dots \| s_{k+i,2^k} \in R_{\mathbb{F}}^{2^{i n\alpha}}$$

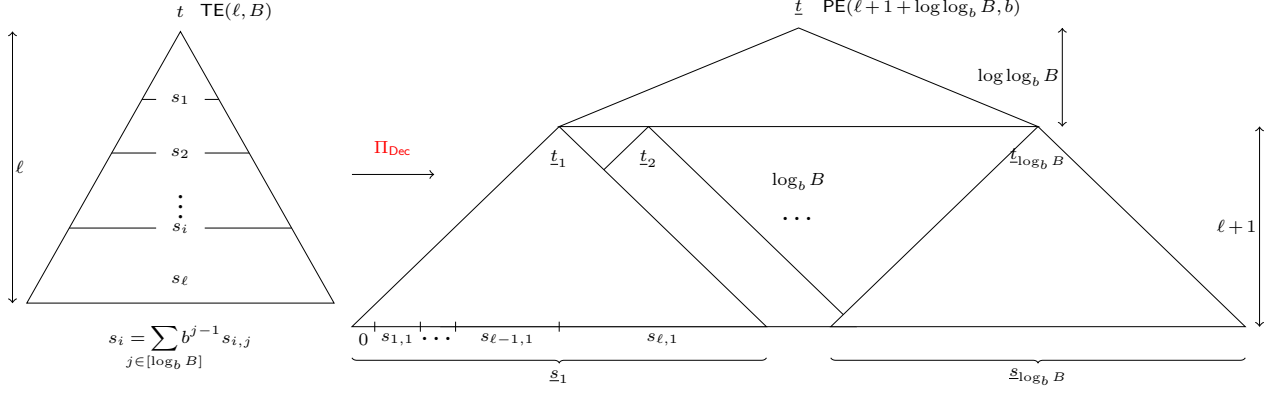


Figure 4: Visualization of $\Pi_{\text{Dec}}: \text{TE}(\ell, B) \rightarrow \text{PE}(\ell + \log \log(B) + 1, b)$. The high norm layer openings of the original commitment are flattened and the decomposition of the flattened vector is recommitted to.

We craft a sum-check argument that matches the above form in Eq. 6 to reduce the check to tree evaluation claims on the subtrees $\mathfrak{s}_j = 0^{2n\alpha} \|\mathfrak{s}_{1,j}\| \cdots \|\mathfrak{s}_{\ell-k,j}\|$. These claims are folded together into a single claim in $\text{TE}(\ell - k, B)$ of higher norm $B > b$. Lastly, to perform the folding, the prover additionally commits to 2^k evaluations for each of the subtrees. This results in an additional claim in $\text{PE}(k, b)$. A high level depiction of the folding structure with the top tree and subtrees is given in Figure 3.

Sum-check-based norm decomposition. First consider a strategy for norm decomposition taken in prior works [BC25a, BC25b, NS25]. Define the b -decomposition of a vector $s \in R_{\mathbb{F}}^{2n\alpha}$ of norm bound $\|s\|_\infty < B$ as the vectors $(s_1, \dots, s_{\log_b B}) \in R_{\mathbb{F}}^{2n\alpha \times \log_b B}$ where

$$s = \sum_{j \in [\log_b B]} b^{j-1} \cdot s_j, \quad \text{and} \quad \|s_j\|_\infty < b.$$

Consider the single-level Ajtai commitment to s as $t \leftarrow A \cdot s$. Observe that the Ajtai commitments to each of the decomposed vectors $t_j \leftarrow A s_j$, recomposes to t under b -recomposition:

$$\sum_{j \in [\log_b B]} b^{j-1} \cdot t_j = \sum_{j \in [\log_b B]} b^{j-1} \cdot A s_j = A \left(\sum_{j \in [\log_b B]} b^{j-1} \cdot s_j \right) = A s = t.$$

A norm decomposition protocol consists of simply sending commitments t_j and having the verifier check the b -recomposition. This simple recomposition check combined with a norm check on the values s_j committed in t_j suffice to prove that s_j is a valid b -decomposition for s .

The same strategy does not extend to tree commitments. Consider a tree commitment t with openings $(s_1, \dots, s_\ell)$. One can b -decompose the bottom layer s_ℓ into $(s_{\ell,1}, \dots, s_{\ell, \log_b B})$ and recommit to the decomposed vectors in tree commitments $(t_1, \dots, t_{\log_b B})$. It is true that the tree commitments to the decomposed vectors will b -recompose to the original tree commitment t . However, the b -decomposition is only valid at the bottom layer s_ℓ . The openings of the decomposed tree commitments for layers $i \in [1, \ell - 1]$, e.g., $(s_{i,1}, \dots, s_{i, \log_b B})$, are not valid low-norm b -decompositions of the corresponding layer in the original tree commitment:

$$\forall i \in [1, \ell - 1] \quad \sum_{j \in [\log_b B]} b^{j-1} \cdot s_{i,j} = s_i, \quad \text{but} \quad \|s_{i,j}\|_\infty \not\leq b.$$

Instead, we must take a different approach. Our strategy is to flatten *all of the layers together* into a single vector s and b -decompose and recommit to these decompositions $(\underline{s}_1, \dots, \underline{s}_{\log_b B})$ as the bottom layer in fresh tree commitments $(\underline{t}_1, \dots, \underline{t}_{\log_b B})$:

$$s = 0_{2n\alpha} \|\underbrace{s_1}_{2n\alpha} \parallel \underbrace{s_2}_{4n\alpha} \parallel \underbrace{s_3}_{8n\alpha} \parallel \cdots \parallel \underbrace{s_\ell}_{2^\ell n\alpha}\| \in R_{\mathbb{F}}^{2^{\ell+1} n\alpha}$$

The prover uses a sum-check to show that the claimed decomposition correctly recomposes to open the original commitment t . This is captured by the following opening constraints which consider the recomposition of s_i from the

particular offset of s :

$$t = A \cdot s_1, \quad \forall i \in [\ell-1] \quad G_{2^i n} \cdot s_i = (I_{2^i} \otimes A) \cdot s_{i+1}$$

$$\text{where } \forall i \in [\ell] \quad s_i = \sum_{j \in [\log_b B]} b^{j-1} \cdot \underline{s}_j [2^j n \alpha : 2^{j+1} n \alpha].$$

A key observation for succinct verification is that, due to the tree commitment structure, the commitment key A is of small constant size. Thus, the sum-check arithmetization ends up including polynomials related to the multilinear extension $\tilde{A}$ which can be efficiently evaluated locally by the verifier. This leaves polynomial evaluation claims on each of the $\underline{s}_j$ decomposition vectors which correspond to $\log_b B$ claims in $\text{PE}(\ell+1, b)$ on $(\underline{t}_1, \dots, \underline{t}_{\log_b B})$, respectively. The tree commitments $\underline{t}_j$ are one layer taller than the original commitment t since all of the layers from the original tree are flattened. We provide one further optimization and instead commit to all $(\underline{s}_1, \dots, \underline{s}_{\log_b B})$ together in one fresh tree commitment. This results in a single polynomial evaluation claim in $\text{PE}(\ell+1+\log_2 \log_b B, b)$ that is larger but saves in proof size as only one root commitment is sent instead of $\log_b B$ commitments. Figure 4 provides a high-level depiction of the decomposition strategy.

Full protocol details. Section 3 expands on preliminaries mentioned here and used in the full description of the protocol. Section 5 presents the full details of each reduction, proves the required security properties for each reduction, and proves the composition of the reductions into our full protocol. Section 5 also presents a candidate finishing protocol based on Greyhound [NS24] which resolves the smaller claims of size k deferred from folding. In Section 6, we consider parameterization of our protocol and report on asymptotic costs of our protocol and concrete costs of a tuned main cycle and finishing protocol for an initial claim of size 2^{30} .

3 Preliminaries

3.1 Notation

Sets and vectors. For $m, n \in \mathbb{Z}$ let $[m, n]$ denote the vector $[m, \dots, n]$ and $[n]$ be shorthand for $[1, n]$. $[\cdot]$ and $(\cdot)$ to denote ordered vectors, and $\{\cdot\}$ denotes an unordered set. For a vector $x = (x^{(1)}, \dots, x^{(n)})$ and $a, b \in [n]$ where $b \geq a$, let $x^{(a:b)}$ denote the subvector $(x^{(a)}, \dots, x^{(b)})$. The notation $x^{(:b)}$ is shorthand for $x^{(1:b)}$. Similarly, $x^{(a:)}$ is shorthand for $x^{(a:n)}$.

Polynomials. For a vector $s \in R^m$, $\tilde{s}$ denotes the multilinear extension interpreting s as evaluations of $\tilde{s}$ over the Boolean hypercube $\{0, 1\}^{\log m}$ in a canonical order, i.e. $s = [\tilde{s}(0 \dots 0), \dots, \tilde{s}(1 \dots 1)]$. We denote the polynomial $\tilde{eq}(x, y) = \prod_{i=1}^{\ell} (x_i \cdot y_i + (1 - x_i) \cdot (1 - y_i))$, which outputs 1 if $x = y$ and 0 otherwise for $x, y \in \{0, 1\}$.

3.2 Algebra

Fields and Rings. Let d be a power of two and let $e \mid d$ be a power of two with $d/e > 1$. Let $\phi(X) = X^d + 1$ be the $2d$ -th cyclotomic polynomial. Let $\mathbb{F}$ be a finite field of prime order q with $q \equiv 2(d/e) + 1 \pmod{4(d/e)}$. Then, ϕ factors over $\mathbb{F}$ into d/e irreducible degree- e polynomials:

$$\phi(X) = \prod_{j=1}^{d/e} (X^e - r_j)$$

for distinct $r_j \in \mathbb{F}^*$. Set $\nu := r_1$. Since $Y^e - \nu$ is irreducible over $\mathbb{F}$, we realize the degree- e extension as $\mathbb{K} := \mathbb{F}[Y]/(Y^e - \nu)$. We define the cyclotomic rings $R_{\mathbb{F}} := \mathbb{F}[X]/\phi(X)$ and $R_{\mathbb{K}} := \mathbb{K}[X]/\phi(X)$, so that $\mathbb{F} \subseteq R_{\mathbb{F}} \subseteq R_{\mathbb{K}}$ and $\mathbb{F} \subseteq \mathbb{K} \subseteq R_{\mathbb{K}}$ as subrings.

Isomorphisms. We use two ring isomorphisms via the Number Theoretic Transform (NTT) and the Chinese Remainder Theorem (CRT)³. First, CRT applied to $\phi(X) = \prod_{j=1}^{d/e} (X^e - r_j)$ over $\mathbb{F}$ gives the following map. For $a \in R_{\mathbb{F}}$, let $a(X) \in \mathbb{F}[X]$ denote its unique representative of degree less than d .

$$R_{\mathbb{F}} \xrightarrow{\sim} \prod_{j=1}^{d/e} \mathbb{F}[X]/(X^e - r_j) \cong \mathbb{K}^{d/e}, \quad a \xrightarrow{\text{NTT}} (a(X) \bmod (X^e - r_j))_{j=1}^{d/e},$$

³Both maps are instances of CRT. We reserve NTT for the split induced by factoring the cyclotomic modulus $\phi(X)$ over $\mathbb{F}$, matching standard lattice-cryptography usage, and CRT for the split induced by factoring the extension modulus $Y^e - \nu$ over $R_{\mathbb{F}}$.

where each factor $\mathbb{F}[X]/(X^e - r_j)$ is a degree- e extension of $\mathbb{F}$, and hence isomorphic to $\mathbb{K}$.

Second, observe $R_{\mathbb{K}}$ can be identified with $R_{\mathbb{F}}[Y]/(Y^e - \nu)$, so consider CRT applied to $Y^e - \nu \in R_{\mathbb{F}}[Y]$. Consider first the irreducible polynomial $Y^e - \nu \in \mathbb{F}[Y]$ that splits over $\mathbb{K}[Y]$. Let $u_1, \dots, u_e \in \mathbb{K}$ be its roots. For each $i \in [e]$, define $\rho_i := \text{NTT}^{-1}(u_i, \dots, u_i) \in R_{\mathbb{F}}$. Notice the constant $\text{NTT}(\nu \in \mathbb{F}) = (\nu, \dots, \nu)$. Thus, the ρ_i are the roots of $Y^e - \nu$ in $R_{\mathbb{F}}$: $Y^e - \nu = \prod_{i=1}^e (Y - \rho_i)$. The factors $Y - \rho_i$ are pairwise comaximal in $R_{\mathbb{F}}[Y]$. Then for $h \in R_{\mathbb{K}}$, let $h(Y) \in R_{\mathbb{F}}[Y]$ denote its unique representative of degree less than e . Applying CRT gives

$$R_{\mathbb{K}} \xrightarrow{\sim} \prod_{i=1}^e R_{\mathbb{F}}[Y]/(Y - \rho_i) \cong R_{\mathbb{F}}^e, \quad h \xrightarrow{\text{CRT}} (h(Y) \bmod (Y - \rho_i))_{i=1}^e.$$

We write $\text{NTT} : R_{\mathbb{F}} \rightarrow \mathbb{K}^{d/e}$ and $\text{CRT} : R_{\mathbb{K}} \rightarrow R_{\mathbb{F}}^e$ for the two isomorphisms above, and NTT^{-1} and CRT^{-1} for their inverses. We implicitly extend these maps to vectors and matrices over $R_{\mathbb{F}}/R_{\mathbb{K}}$ by applying them coordinate-wise, and to polynomials over $R_{\mathbb{F}}/R_{\mathbb{K}}$ by applying them coefficient-wise. The latter is well-defined with respect to any choice of basis (e.g. the monomial, Lagrange, or multilinear basis).

Norm. For an element $a \in \mathbb{F}$, we define $\|a\|_{\infty}$ as follows: let $a' \in [0, q-1]$ denote the integer representation of $a \bmod q$. If $a' \leq (q-1)/2$, then $\|a\|_{\infty} = a'$; otherwise, $\|a\|_{\infty} = |a' - q|$. For a vector $z \in \mathbb{F}^n$, define $\|z\|_{\infty} := \max_i \|z_i\|_{\infty}$. For an element $a \in R_{\mathbb{F}}$, define $\|a\|_{\infty}$ to be the ℓ_{∞} -norm of its coefficient vector. For a vector $z \in R_{\mathbb{F}}^m$, define $\|z\|_{\infty} := \max_i \|z_i\|_{\infty}$.

Definition 1. Define $\text{split}_b : R_{\mathbb{F}}^m \rightarrow (R_{\mathbb{F}}^m)^*$ to be the b -ary decomposition map, which performs the b -ary decomposition of a vector $z \in R_{\mathbb{F}}^m$ into vectors $(z_1, z_2, \dots, z_t) \in (R_{\mathbb{F}}^m)^*$. For example, if $z \in R_{\mathbb{F}}^m$ and $\|z\|_{\infty} < b^t$, then we have

$$\text{split}_b(z) := (z_1, \dots, z_t) \quad \text{such that} \quad z = \sum_{i=1}^t b^{i-1} \cdot z_i \quad \text{and} \quad \|z\|_{\infty} < b$$

Now we recall a few standard definitions and lemmas pertaining to the classic sum-check protocol (Def. 9).

Definition 2 (Sampling set [CCKP19]). Let R be a ring. $S \subseteq R$ is a sampling set of R if S is a finite set such that $\forall s_1 \neq s_2 \in S, s_1 - s_2 \in R$ is not a zero-divisor.

Lemma 1 (Schwartz-Zippel [Sch80]). Let $\mathbb{F}$ be a field, $g \in \mathbb{F}[X_1, \dots, X_m]$ be an m -variate polynomial of total degree at most d over $\mathbb{F}$. Then, on any finite set $S \subseteq \mathbb{F}$, $\Pr_{x \leftarrow_{\$} S^m} [g(x) = 0] \leq d/|S|$.

We now define some useful maps for our interactive reductions.

Definition 3. Let $z = [z_1, \dots, z_n] \in R_{\mathbb{F}}^n$. For each $z_i \in R_{\mathbb{F}}, i \in [n]$, let $(z_{i,0}, \dots, z_{i,d-1}) \in \mathbb{F}^d$ be the coefficients of $z_i = z_{i,0} + z_{i,1} \cdot \overline{X} + \dots + z_{i,d-1} \cdot \overline{X}^{d-1}$. Define the map $\text{cf} : R_{\mathbb{F}}^n \rightarrow \mathbb{F}^{n \times d}$ to map a ring vector to the map of its field coefficients

$$\text{cf}(z) := \begin{bmatrix} z_{1,0} & \dots & z_{1,d-1} \\ \vdots & \ddots & \vdots \\ z_{n,0} & \dots & z_{n,d-1} \end{bmatrix}$$

to match the powers of $\overline{X}$, index the columns by $h \in [0, d-1]$. Then denote the h th column of $\text{cf}(z)$ by

$$\text{cf}(z)_h := \begin{bmatrix} z_{1,h} \\ \vdots \\ z_{n,h} \end{bmatrix} \in \mathbb{F}^n$$

If $z \in R_{\mathbb{F}}$ (i.e. $n = 1$), $\text{cf}(z)_h$ denotes the h -th element of a row vector in $\mathbb{F}^{1 \times d}$.

The map is defined in the same way for $R_{\mathbb{K}}$ producing coefficients in $\mathbb{K}$.

3.3 Interactive Reductions

Definition 4 (Interactive reduction [KP23, Kot24]). A reduction of knowledge from $\mathcal{R}_1$ to $\mathcal{R}_2$ is defined by PPT algorithms $(\mathcal{G}, \mathcal{P}, \mathcal{V})$ called the generator, prover, and verifier

- $\mathcal{G}(1^\lambda) \rightarrow pp$: takes as input security parameter 1^λ and outputs public parameters pp .
- $\mathcal{P}(pp, x_1, w_1) \rightarrow (x_2, w_2)$: takes as input public parameters, statement-witness pair (x_1, w_1) and interactively reduces the task of checking $(pp, x_1, w_1) \in \mathcal{R}_1$ to the task of checking $(pp, x_2, w_2) \in \mathcal{R}_2$.
- $\mathcal{V}(pp, x_1) \rightarrow x_2$: takes as input public parameters, statement x_1 and interactively reduces the task of checking x_1 to the task of checking x_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 (pp, x_1, w_1) and runs the interaction on prover input (pp, x_1, w_1) and verifier input (pp, x_1) . At the end of the interaction, $\langle \mathcal{P}, \mathcal{V} \rangle$ outputs the verifier's statement x_2 and prover's witness w_2 . A **reduction of knowledge** is an interactive reduction $(\mathcal{G}, \mathcal{P}, \mathcal{V})$ which satisfies

1. **Completeness:** For any PPT adversary $\mathcal{A}$, given $pp \leftarrow \mathcal{G}(1^\lambda)$ and $(x_1, w_1) \leftarrow \mathcal{A}(pp)$ such that $(pp, x_1, w_1) \in \mathcal{R}_1$, we have that the prover's output statement is equal to the verifier's output statement and that

$$(pp, \langle \mathcal{P}, \mathcal{V} \rangle(pp, x_1, w_1)) \in \mathcal{R}_2.$$

2. **Knowledge Soundness:** For any expected polynomial time adversaries $\mathcal{A}$ and $\mathcal{P}^*$, there exists an expected polynomial time extractor $\mathcal{E}$ such that if the success probability of the adversary

$$\epsilon(\mathcal{A}, \mathcal{P}^*) := \Pr \left[(pp, \langle \mathcal{P}^*, \mathcal{V} \rangle(x_1, st)) \in \mathcal{R}_2 \mid \begin{array}{l} pp \leftarrow \mathcal{G}(1^\lambda) \\ (x_1, st) \leftarrow \mathcal{A}(pp) \end{array} \right] \geq 1/\text{poly}(\lambda)$$

then

$$\Pr \left[(pp, x_1, w_1) \in \mathcal{R}_1 \mid \begin{array}{l} pp \leftarrow \mathcal{G}(1^\lambda) \\ (x_1, st) \leftarrow \mathcal{A}(pp) \\ w_1 \leftarrow \mathcal{E}(pp, x_1, st) \end{array} \right] \geq \epsilon(\mathcal{A}, \mathcal{P}^*) - \text{negl}(\lambda)$$

3. **Public Coin:** All the verifier's messages are uniformly random challenges of some prescribed length, and the messages contain all the randomness used by the verifier.

Lemma 2 (Sequential composition [KP23, Kot24]). For reductions of knowledge $\Pi_1 = (\mathcal{G}, \mathcal{P}_1, \mathcal{V}_1) : \mathcal{R}_1 \rightarrow \mathcal{R}_2$ and $\Pi_2 = (\mathcal{G}, \mathcal{P}_2, \mathcal{V}_2) : \mathcal{R}_2 \rightarrow \mathcal{R}_3$, we have that $\Pi_2 \circ \Pi_1 = (\mathcal{G}, \mathcal{P}, \mathcal{V}) : \mathcal{R}_1 \rightarrow \mathcal{R}_3$ is a reduction of knowledge where

$$\begin{aligned} \mathcal{P}(pp, x_1, w_1) &= \mathcal{P}_2(pp, \mathcal{P}_1(pp, x_1, w_1)) \\ \mathcal{V}(pp, x_1) &= \mathcal{V}_2(pp, \mathcal{V}_1(pp, x_1)) \end{aligned}$$

Definition 5 (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 6 (Weak Interactive Reductions [NS25]). 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 $(\mathcal{G}, \mathcal{P}, \mathcal{V})$ (Def. 4), 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

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

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

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

then the following holds. Define the oracle

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

For any uniqueness adversary $\mathcal{W}_{\text{uniq}}^{\mathcal{O}_{\text{weak}}}$ (Def. 5), we have

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

Definition 7 (Strong Interactive Reductions [NS25]). 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 $(\mathcal{G}, \mathcal{P}, \mathcal{V})$ (Def. 4), 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 $(\mathcal{A}, \mathcal{P}^*)$,

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

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

$$\begin{aligned} \mathcal{O}_{\text{str}}(pp, u_1, \text{st}) &\rightarrow (u_2, w_2) : \\ (a) \quad &(u_2, w_2) \leftarrow \langle \mathcal{P}^*, \mathcal{V} \rangle(pp, u_1, \text{st}). \\ (b) \quad &\text{Output } (u_2, w_2). \end{aligned}$$

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

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

$\geq 1/\text{poly}(\lambda)$, and, for any uniqueness adversary $\mathcal{S}_{\text{uniq}}^{\mathcal{O}_{\text{str}}}$ (Def. 5),

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

then we have that

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

Theorem 1 (Strong-Weak Composition [NS25]). 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 (Def. 4) $\Pi_1 : \mathcal{R}_1 \rightarrow \mathcal{R}_2$ ($\mathcal{R}'_2$), $\Pi_2 : \mathcal{R}_2$ ($\mathcal{R}'_2$) $\rightarrow \mathcal{R}_3$ such that

1. Π_1 is **strong** (Def. 7) with respect to a function $\phi : \mathcal{U}_2 \rightarrow \mathbb{C}$ and
2. Π_2 is **weak** (Def. 6) 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**.

Definition 8 (Polynomial interactive oracle reduction (informal) [BBHR19, BMNW25]). *Polynomial interactive oracle reductions (polyIORs) are a special case of interactive reductions (Def. 4). A polyIOR allows polynomial oracles to be made available to $\mathcal{V}$ as part of the statement and prover messages. Oracles, denoted by $\llbracket f \rrbracket^{\leq D}$, can be queried by the verifier at arbitrary points. D denotes the degree bound of f , and will be dropped from the notation of f if degree 1.*

We will also allow for $\mathcal{V}$ to make queries to virtual oracles which are queries to polynomials that $\mathcal{V}$ is able to derive from oracles that $\mathcal{V}$ has already received (e.g. linear combinations, NTT).

A key component of our reductions of knowledge will be the sum-check protocol [LFKN92], which we frame as a polyIOR here.

Definition 9 (Sum-check protocol [LFKN92] (informal)). *The sum-check protocol $(r \in \mathbb{K}^\ell, y \in \mathbb{K}) \leftarrow \text{SC}(v, \ell, G)$ is a polyIOR that checks that the sum of evaluations of a ℓ -variate, degree at most d polynomial $G \in \mathbb{K}^{\leq d}[X_1, \dots, X_\ell]$ on the Boolean hypercube results in some claimed value $v \in \mathbb{K}$. In this protocol, $\mathcal{V}$ has query access to (potentially virtual) $\llbracket G \rrbracket$. The output of the sum-check protocol is a claim that $y \stackrel{?}{=} G(r)$ for some random point $r \in \mathbb{K}^\ell$ and claimed evaluation $y \in \mathbb{K}$.*

Lemma 3 ([LFKN92] (informal)). *The sum-check protocol (Def. 9) is public-coin, has a completeness error of 0, and has a soundness error of $\leq \ell d / |\mathbb{K}|$.*

Lemma 4 (Zero-check via sum-check). *Let $Q \in \mathbb{K}[X_1, \dots, X_m]$ be an arbitrary multivariate polynomial. Then*

$$0 = \sum_{x \in \{0,1\}^m} \tilde{e}q(x, Z) \cdot Q(x) \quad \text{if and only if} \quad Q(x) = 0 \quad \forall x \in \{0,1\}^m$$

3.4 Challenge Sets and Coordinate-Wise Special Soundness

Theorem 2 (Low norm invertibility [LS18, Corollary 1.2]). *Let $R_{\mathbb{F}} = \mathbb{F}[X] / \phi(X)$ with $\phi(X) = \prod_{j=1}^{d/e} (X^e - r_j)$. Any $y \in R_{\mathbb{F}}$ that satisfies the following norm bounds has an inverse in $R_{\mathbb{F}}$:*

$$0 < \|y\|_{\infty} < \frac{1}{\sqrt{d/e}} \cdot q^{e/d}.$$

Definition 10 (Strong sampling sets [CCKP19, AL21, GLLO26]). *Let R be a ring and let $\mathcal{C} \subseteq R$ be a finite set with $|\mathcal{C}| \geq 2$. Let $R^\times$ denote the group of units of R (i.e., invertible elements), and define*

$$\epsilon_{\text{inv}}(\mathcal{C}) := \max_{a \in \mathcal{C}} \Pr_{b \leftarrow_{\S} \mathcal{C} \setminus \{a\}} [a - b \notin R^\times].$$

We say that $\mathcal{C}$ is a strong sampling set of R if $\epsilon_{\text{inv}}(\mathcal{C}) \leq \text{negl}(\lambda)$. If $\epsilon_{\text{inv}}(\mathcal{C}) = 0$, then $\mathcal{C}$ is an exact strong sampling set.

Remark. A previous notion of approximate strong sampling sets [GLLO26] uses a weaker “average” notion of non-invertibility requiring that for a uniformly sampled pair of distinct challenges $a, b \leftarrow_{\S} \mathcal{C}$, the difference $a - b$ is invertible with high probability. In contrast, our definition requires a “local” notion that for every fixed challenge $a \in \mathcal{C}$, the difference $a - b$ is invertible with high probability over a random challenge $b \leftarrow_{\S} \mathcal{C} \setminus \{a\}$. Looking forward, this stronger notion is useful for tight extraction bounds. For dense ternary challenges, the per-slot min-entropy analysis of prior work [ALS20] implies this stronger local property.

Definition 11 (Dense ternary challenge set [ALS20, Section 3]). *We define the dense ternary challenge set*

$$\mathcal{C}_{\text{ter}} := \left\{ \rho(X) = \sum_{i=0}^{d-1} \rho_i X^i \in R_{\mathbb{F}} : \rho_i \in \{-1, 0, 1\} \right\}.$$

Thus, $|\mathcal{C}_{\text{ter}}| = 3^d$, and sampling uniformly from $\mathcal{C}_{\text{ter}}$ is equivalent to sampling every coefficient independently and uniformly from $\{-1, 0, 1\}$.

Every pair of distinct $a, b \in \mathcal{C}_{\text{ter}}$ satisfies $0 < \|a - b\|_\infty \leq 2$. Thus, $\mathcal{C}_{\text{ter}}$ is an exact strong sampling set when within the deterministic bound of Theorem 2. For parameters that do not satisfy this deterministic bound, we refer to the following results.

Definition 12 (Biased ternary distribution). *Let χ_p denote the biased ternary distribution with bias $p \in [0, 1]$ over $\{-1, 0, 1\} \subseteq \mathbb{F}$. It samples 0 with probability p and samples each of -1 and 1 with probability $(1-p)/2$. We write χ_p^d for the product distribution over $R_{\mathbb{F}}$ obtained by sampling each coefficient independently from χ_p .*

Definition 13 (ϵ -min-entropy in every NTT slot [GLLO26, Lemma 8]). *Let $\mathcal{D}$ be a distribution over $R_{\mathbb{F}}$ and let $\epsilon \in [0, 1]$. We say that $\mathcal{D}$ has ϵ -min-entropy in every NTT slot if, for every $j \in [d/e]$ and every $y \in \mathbb{F}[X]/(X^e - r_j)$,*

$$\Pr_{x \leftarrow \mathcal{D}} [x \bmod (X^e - r_j) = y] \leq \epsilon.$$

Lemma 5 (Biased ternary NTT-slot min-entropy [GLLO26, Lemma 8], [BL26, Lemma 32], [ALS20, Lemma 3.2]). *Let $R_{\mathbb{F}} = \mathbb{F}[X]/\phi(X)$ with $\phi(X) = \prod_{j=1}^{d/e} (X^e - r_j)$. Define*

$$\epsilon_\chi := \max_{j \in [d/e]} \left(\frac{1}{q} + \frac{1}{q} \sum_{t \in [q-1]} \prod_{i=0}^{d/e-1} \left| p + (1-p) \cos \left(\frac{2\pi t r_j^i}{q} \right) \right| \right)^e.$$

Then χ_p^d has ϵ_χ -min-entropy in every NTT slot in the sense of Definition 13.

Lemma 6 (Dense ternary invertibility). *Let $R_{\mathbb{F}} = \mathbb{F}[X]/\phi(X)$ with $\phi(X) = \prod_{j=1}^{d/e} (X^e - r_j)$, and let ϵ_χ be as in Lemma 5 for $p = 1/3$. Then*

$$\epsilon_{\text{inv}}(\mathcal{C}_{\text{ter}}) \leq \frac{d}{e} \cdot \frac{3^d \epsilon_\chi - 1}{3^d - 1} \leq \frac{d}{e} \epsilon_\chi.$$

Proof. Set $N := |\mathcal{C}_{\text{ter}}| = 3^d$ and fix $a \in \mathcal{C}_{\text{ter}}$. For each $j \in [d/e]$, the ϵ_χ -min-entropy bound in Lemma 5 implies

$$|\{b \in \mathcal{C}_{\text{ter}} : b \bmod (X^e - r_j) = a \bmod (X^e - r_j)\}| \leq N \epsilon_\chi.$$

This set contains a , so a uniformly sampled $b \in \mathcal{C}_{\text{ter}} \setminus \{a\}$ collides with a modulo this factor with probability at most $(N \epsilon_\chi - 1)/(N - 1)$. By the NTT decomposition of $R_{\mathbb{F}}$, the difference $a - b$ is noninvertible only if such a collision occurs modulo at least one of the d/e factors. A union bound over the factors and a maximum over a prove the first inequality. The second follows from $\epsilon_\chi \leq 1$. $\square$

Lemma 7 (Dense ternary heuristic [GLLO26, Lemma 9]). *Let $R_{\mathbb{F}} = \mathbb{F}[X]/\phi(X)$ with $\phi(X) = \prod_{j=1}^{d/e} (X^e - r_j)$. For the uniform distribution $\chi_{1/3}^d$ over $\mathcal{C}_{\text{ter}}$, the parameter from Lemma 5 heuristically satisfies*

$$\epsilon_\chi \approx \max \left\{ \frac{1}{q^e}, \frac{1}{3^d} \right\}.$$

Note, for small primes q , e.g. 32-bit, the above heuristic can be validated by explicitly computing ϵ_χ .

Definition 14 (Sparse coefficient challenge set [NS24, Section 5]). *Let $\tau_1, \tau_2 \in \mathbb{Z}_{\geq 0}$ satisfy $1 \leq \tau_1 + \tau_2 < d$. We define*

$$\mathcal{C}_{\text{sp}}(\tau_1, \tau_2) := \left\{ \sum_{i \in S_1} \varepsilon_i X^i + 2 \sum_{i \in S_2} \varepsilon_i X^i : \begin{array}{l} S_1, S_2 \subseteq [0, d-1], \quad S_1 \cap S_2 = \emptyset, \\ |S_1| = \tau_1, \quad |S_2| = \tau_2, \quad \varepsilon_i \in \{-1, 1\} \end{array} \right\}.$$

In other words, a challenge has exactly τ_1 coefficients in $\{-1, 1\}$, exactly τ_2 coefficients in $\{-2, 2\}$, and all other coefficients equal to zero. Its cardinality is

$$|\mathcal{C}_{\text{sp}}(\tau_1, \tau_2)| = \binom{d}{\tau_1} \binom{d - \tau_1}{\tau_2} 2^{\tau_1 + \tau_2}.$$

Every pair of distinct $a, b \in \mathcal{C}_{\text{sp}}(\tau_1, \tau_2)$ satisfies $0 < \|a - b\|_\infty \leq 4$. Thus, Theorem 2 also gives a deterministic exact-strong-sampling criterion for this set when within bounds.

Definition 15 (Expansion factors [CCKP19, AL21]). *The expansion factor $T_{\mathcal{S}}$ of a set $\mathcal{S} \subseteq R_{\mathbb{F}}$ bounds the norm growth due to multiplication by any $\rho \in \mathcal{S}$:*

$$T_{\mathcal{S}} := \max_{\substack{v \in R_{\mathbb{F}} \setminus \{0\} \\ \rho \in \mathcal{S}}} \frac{\|\rho v\|_{\infty}}{\|v\|_{\infty}}.$$

Next we consider the expansion factors of candidate challenge sets along with their difference sets. We write $\mathcal{C} - \mathcal{C} := \{a - b : a, b \in \mathcal{C}\}$ to denote the difference set.

Theorem 3 (Expansion factors of the dense ternary challenge set [GLLO26, Remark 7 and Theorem 3]). *The dense ternary challenge set from Definition 11 and its difference set satisfy*

$$T_{\mathcal{C}_{\text{ter}}} \leq d \quad \text{and} \quad T_{\mathcal{C}_{\text{ter}} - \mathcal{C}_{\text{ter}}} \leq 2d.$$

Theorem 4 (Expansion factors of the sparse coefficient challenge set [NS24, Lemmas 3.1–3.2 and Section 5]). *The sparse coefficient challenge set from Definition 14 and its difference set satisfy*

$$T_{\mathcal{C}_{\text{sp}}(\tau_1, \tau_2)} \leq \tau_1 + 2\tau_2 \quad \text{and} \quad T_{\mathcal{C}_{\text{sp}}(\tau_1, \tau_2) - \mathcal{C}_{\text{sp}}(\tau_1, \tau_2)} \leq 2\tau_1 + 4\tau_2.$$

Definition 16 (ℓ -coordinate-wise special soundness for Σ -protocols [FMN24, Definition 2.30]). *Let $\Pi = (\mathcal{G}, \mathcal{P}, \mathcal{V}) : \mathcal{R}_1 \rightarrow \mathcal{R}_2$ be public-coin three-round interactive reduction, and suppose the challenge space of $\mathcal{V}$ is $\mathcal{C}^{\ell}$. We say that Π is ℓ -coordinate-wise (2-)special sound if there exists a polynomial time algorithm that on statement x_1 and $\ell + 1$ accepting transcripts $(a_i, \vec{c}_i, w_{2,i})_{i \in [\ell+1]}$ with $(\vec{c}_i)_{i \in [\ell+1]}$ are of the following form*

$$\begin{aligned} \vec{c}_1 &= (c^{(1)}, c^{(2)}, \dots, c^{(\ell)}) \\ \vec{c}_2 &= (d^{(1)}, c^{(2)}, \dots, c^{(\ell)}) \\ \vec{c}_3 &= (c^{(1)}, d^{(2)}, \dots, c^{(\ell)}) \\ &\vdots \\ \vec{c}_{\ell+1} &= (c^{(1)}, c^{(2)}, \dots, d^{(\ell)}) \end{aligned}$$

$d^{(i)} \neq c^{(i)}, i \in [\ell]$, and common first message a , $E(\text{pp}, x_1, (a_i, \vec{c}_i, w_{2,i})_{i \in [\ell+1]}) \rightarrow w_1$ outputs a witness w_1 such that $(x_1, w_1) \in \mathcal{R}_1$.

Theorem 5 (Coordinate-wise extraction [FMN24, Lemma 7.1]). *Let $\ell \in \mathbb{N}$ and $\mathcal{C}$ be a finite set. Let $\mathcal{A} : \mathcal{C}^{\ell} \rightarrow \{0, 1\}^*$ be an arbitrary (probabilistic) expected polynomial time algorithm (adversary), and $\mathcal{V} : \mathcal{C}^{\ell} \times \{0, 1\}^* \rightarrow \{0, 1\}$ be an arbitrary (probabilistic) polynomial time algorithm (verifier). Define the success probability of adversary $\mathcal{A}$ as*

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

Then there exists an expected polynomial time oracle algorithm $E^{\mathcal{A}}$ that requires an expected number of at most $\ell + 1$ queries to $\mathcal{A}$ with probability at least

$$\epsilon^{\mathcal{V}}(\mathcal{A}) - \frac{\ell}{|\mathcal{C}|}$$

and outputs $\ell + 1$ pairs $(\vec{c}_i, w_i)_{i \in [\ell+1]}$ such that

- *For all $i \in [\ell + 1]$, $\mathcal{V}(\vec{c}_i, w_i) = 1$,*
- *and $\{\vec{c}_i\}_{i \in [\ell+1]}$ satisfy the challenge set property described in Def. 16.*

Corollary 1 (of Thm. 5). *Let $\Pi = (\mathcal{G}, \mathcal{P}, \mathcal{V})$ be public-coin 3-round interactive proof system for relation $\mathcal{R}$ and suppose the challenge space of $\mathcal{V}$ in each round is $\mathcal{C}^{\ell}$. If Π is ℓ -coordinate-wise special sound (and $\ell = \text{poly}(\lambda)$), then it is knowledge sound with knowledge error $\ell/|\mathcal{C}|$.*

Remark. The preceding coordinate-wise extraction results are stated for arbitrary finite challenge sets, but in the context of lattice-based protocols, are useful only when the challenge set is an exact strong sampling set. In particular, the

extractor from Theorem 5 guarantees distinct coordinate-wise accepting transcripts but does not guarantee that their challenge differences are invertible.

Prior analyses that use the abstraction of approximate strong sampling sets first apply coordinate-wise extraction and then use the “average” notion of approximate strong sampling to argue that the extracted differences are invertible with high probability [GLLO26]. However, conditioned on both transcripts being accepting, the extracted challenge pair need not be uniformly distributed over distinct pairs from $\mathcal{C}$, so the average property does not apply directly. We show in Appendix C (Lemma 37) that the coordinate-wise extractor can be modified to extract accepting transcripts with invertible differences when the challenge set is an average approximate strong sampling set. However, our analysis incurs a square root loss in the average non-invertibility probability. Instead, in Appendix B.4 (Lemma 30), we extend the extraction result to approximate strong sampling sets under our stronger “local” property from Definition 10 with a tighter linear loss. Note that Attema et al. [ALS20] employ a similar tight extraction strategy but do not go through the abstraction of approximate strong sampling sets, instead analyzing the specific ternary challenge set directly.

Lastly, we give a generalization of the Schwartz-Zippel lemma over rings [BCPS18] when the random inputs come from approximate strong sampling sets.

Lemma 8 (Univariate Schwartz-Zippel with approximate strong sampling sets). *Let R be a ring, $p \in R[X]$ be an univariate non-zero polynomial of degree at most d over R . Let $\mathcal{C} \subseteq R$ be an approximate sampling set of R with local inversion error $\epsilon_{\text{inv}}(\mathcal{C})$ (Def. 10), and $|\mathcal{C}| \geq 2$. Then*

$$\Pr_{s \leftarrow \mathcal{C}}[p(s) = 0] \leq d \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right)$$

where $p(s) \in R$.

Proof. We defer the proof to Appendix B.1. □

Lemma 9 (Multivariate Schwartz-Zippel with approximate strong sampling sets). *Let R be a ring, $p \in R[X_1, \dots, X_m]$ be an m -variate non-zero polynomial of total degree at most d over R . Let $\mathcal{C} \subseteq R$ be an approximate sampling set of R with local inversion error $\epsilon_{\text{inv}}(\mathcal{C})$ (Def. 10) and $|\mathcal{C}| \geq d$. Then*

$$\Pr_{s \leftarrow \mathcal{C}^m}[p(s) = 0] \leq d \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right)$$

where $p(s) \in R$.

Proof. We defer the proof to Appendix B.1. □

3.5 Ajtai Commitment Scheme

Definition 17 (Module short integer solution problem [LS15, LM06, PR06]). *Define $R_{\mathbb{Z}} := \mathbb{Z}[X]/(\phi(X))$. For a probabilistic algorithm $\mathcal{B}$, define its advantage in solving $\text{MSIS}_{q,n,d,m,B}$ as*

$$\text{Adv}_A^{\text{MSIS}_{q,n,d,m,B}}(\mathcal{B}, \lambda) := \Pr \left[\begin{array}{l} z \in R_{\mathbb{Z}}^m \setminus \{0\} \\ \wedge \quad Az = 0 \pmod{q} \\ \wedge \quad \|z\|_{\infty} < B \end{array} \middle| \begin{array}{l} A \leftarrow_{\$} R_{\mathbb{F}}^{n \times m} \\ z \leftarrow \mathcal{B}(1^{\lambda}, A) \end{array} \right].$$

The probability is over the uniform choice of A and the internal randomness of $\mathcal{B}$. We say that the $\text{MSIS}_{q,n,d,m,B}$ assumption holds if, for every PPT algorithm $\mathcal{B}$,

$$\text{Adv}_A^{\text{MSIS}_{q,n,d,m,B}}(\mathcal{B}, \lambda) \leq \text{negl}(\lambda).$$

Definition 18 (Gadget decomposition and recomposition). *Let $b \geq 2$ be an integer gadget base and let $\alpha := \lceil \log_b q \rceil$. For $m \in \mathbb{N}$, define the gadget row vector*

$$g_b^{\text{T}} := [1 \quad b \quad b^2 \quad \dots \quad b^{\alpha-1}] \in R_{\mathbb{F}}^{1 \times \alpha}$$

and the gadget recomposition matrix

$$G_m := I_m \otimes g_b^\top \in R_{\mathbb{F}}^{m \times m\alpha}.$$

Thus, for $z = (z_{i,j})_{i \in [m], j \in [0, \alpha-1]} \in R_{\mathbb{F}}^{m\alpha}$,

$$(G_m \cdot z)_i = \sum_{j=0}^{\alpha-1} b^j \cdot z_{i,j} \quad \text{for every } i \in [m].$$

The gadget decomposition map $G_m^{-1} : R_{\mathbb{F}}^m \rightarrow R_{\mathbb{F}}^{m\alpha}$ decomposes each coefficient of each ring element into its canonical base- b digits modulo q and groups equal digit positions into ring elements. Equivalently, for every $x \in R_{\mathbb{F}}^m$,

$$G_m \cdot G_m^{-1}(x) = x \quad \text{and} \quad \|G_m^{-1}(x)\|_\infty < b.$$

Definition 19 (Tree-variant for Ajtai commitments [PSTY13, LLNW16, CMNW24]). *The commitment scheme TCom is a tuple of algorithms (Setup, Commit, OpenF). We provide an additional helper algorithm Open that checks the validity of the openings without checking the underlying committed value. Let $\ell, r, n, \alpha, b \in \mathbb{N}$ where ℓ is the tree height, r is the tree branching factor, n is the dimension of the underlying lattice, α is the gadget decomposition parameter, and b is a norm bound. We overload Commit to be able to commit directly to a low-norm bottom layer openings on the condition that $\|s_\ell\|_\infty < b$.*

- $\text{Setup}(1^\lambda) \rightarrow pp$: Sample a random matrix $A \leftarrow_{\$} R_{\mathbb{F}}^{n \times rn\alpha}$ where dimension n is chosen appropriately for security parameter λ . Return $pp \leftarrow \{A\}$.
- $\text{Commit}(A \leftarrow pp, f \in R_{\mathbb{F}}^{\ell n}) \rightarrow (t, st)$: Compute commitment and openings:

$$\begin{aligned} t &\leftarrow A \cdot G_{rn}^{-1} \left((I_r \otimes A) \cdot G_{r2n}^{-1} \left(\cdots G_{r^{\ell-1}n}^{-1} \left((I_{r^{\ell-1}} \otimes A) \cdot G_{r^{\ell}n}^{-1}(f) \right) \right) \right) \\ s_i &\leftarrow G_{rjn}^{-1} \left((I_{rj} \otimes A) \cdot G_{rj+1n}^{-1} \left(\cdots G_{r^{\ell-1}n}^{-1} \left((I_{r^{\ell-1}} \otimes A) \cdot G_{r^{\ell}n}^{-1}(f) \right) \right) \right) \end{aligned}$$

Return $(t, (s_i)_{i \in [\ell]})$.

- $\text{Commit}(A \leftarrow pp, s_\ell \in R_{\mathbb{F}}^{r\ell n\alpha}) \rightarrow (t, st)$: Compute commitment and openings:

$$\begin{aligned} t &\leftarrow A \cdot G_{rn}^{-1} \left((I_r \otimes A) \cdot G_{r2n}^{-1} \left(\cdots G_{r^{\ell-1}n}^{-1} \left((I_{r^{\ell-1}} \otimes A) \cdot s_\ell \right) \right) \right) \\ s_i &\leftarrow G_{rjn}^{-1} \left((I_{rj} \otimes A) \cdot G_{rj+1n}^{-1} \left(\cdots G_{r^{\ell-1}n}^{-1} \left((I_{r^{\ell-1}} \otimes A) \cdot s_\ell \right) \right) \right) \end{aligned}$$

Return $(t, (s_i)_{i \in [\ell]})$.

- $\text{Open}(A \leftarrow pp, b, t, (s_i)_{i \in [\ell]} \leftarrow st) \rightarrow 0/1$: Return 1 if

$$t = A \cdot s_1 \quad \forall i \in [\ell-1], G_{r^{i+1}n} \cdot s_i = (I_{r^{i+1}} \otimes A) \cdot s_{i+1}, \quad \forall i \in [\ell], \|s_i\|_\infty < b.$$

Else, return 0.

- $\text{OpenF}(A \leftarrow pp, b, t, f \in R_{\mathbb{F}}^{\ell n}, (s_i)_{i \in [\ell]} \leftarrow st) \rightarrow 0/1$: Return 1 if $\text{Open}(A, b, t, (s_i)_{i \in [\ell]})$ and $G_{r^{\ell}n} \cdot s_\ell = f$. Else, return 0.

Definition 20 (Perfect completeness). *A commitment scheme is perfectly complete if, for every message f ,*

$$\Pr \left[\text{OpenF}(pp, b, t, f, st) = 1 \mid \begin{array}{l} pp \leftarrow \text{Setup}(1^\lambda) \\ (t, st) \leftarrow \text{Commit}(pp, f) \end{array} \right] = 1.$$

Definition 21 (Binding). *A commitment scheme is binding if, for every PPT adversary $\mathcal{A}$,*

$$\Pr \left[\begin{array}{l} (f, st) \neq (f', st') \\ \wedge \text{OpenF}(pp, b, t, f, st) = 1 \\ \wedge \text{OpenF}(pp, b, t, f', st') = 1 \end{array} \mid \begin{array}{l} pp \leftarrow \text{Setup}(1^\lambda) \\ (t, f, st, f', st') \leftarrow \mathcal{A}(pp) \end{array} \right] \leq \text{negl}(\lambda).$$

Lemma 10 ([CMNW24, Theorem 3]). *The commitment scheme TCom is perfectly complete for $b \geq b$.*

Lemma 11 ([CMNW24, Theorem 4]). *The commitment scheme TCom is binding assuming $\text{MSIS}_{q,n,d,rn\alpha,b'}$ with $b' \geq 2b$.*

Definition 22 ($(B, \mathcal{C}, k)$ -relaxed binding).

$$\Pr \left[\begin{array}{l} \text{Open}(A, |\mathbb{F}|/2, t, s) = 1 \\ \wedge \text{Open}(A, |\mathbb{F}|/2, t, s') = 1 \\ \wedge \forall i \in [\ell], j \in [\min(2^{i-1}, 2^k)] \\ \quad \|\Delta_j \cdot \mathfrak{s}_{i,j}\|_\infty < B, \\ \quad \|\Delta'_j \cdot \mathfrak{s}'_{i,j}\|_\infty < B \\ \wedge s \neq s' \end{array} \middle| \begin{array}{l} pp \leftarrow \text{Setup}(1^\lambda, m) \\ t \in R_{\mathbb{F}}^n, \\ \Delta, \Delta' \in ((\mathcal{C} - \mathcal{C}) \cap R_{\mathbb{F}}^\times)^{2^k}, \\ s := \left(s_i \in R_{\mathbb{F}}^{2^i n \alpha} \right)_{i \in [\ell]}, \\ s' := \left(s'_i \in R_{\mathbb{F}}^{2^i n \alpha} \right)_{i \in [\ell]} \end{array} \leftarrow \mathcal{A}(pp) \right] \leq \epsilon_{\text{rlx}}(B, \mathcal{C}, k)$$

where for all $i \in [\ell]$, chunk $s_i := \mathfrak{s}_{i,1} \parallel \dots \parallel \mathfrak{s}_{i,\min(2^{i-1}, 2^k)}$.

Lemma 12. *Assuming $\text{MSIS}_{q,n,d,2n\alpha,2 \cdot T_{\text{bck}} \cdot B}$, the commitment scheme TCom satisfies $(B, \mathcal{C}, k)$ -relaxed binding with $\epsilon_{\text{rlx}}(B, \mathcal{C}, k) \leq \text{negl}(\lambda)$.*

Proof. Suppose there is a relaxed binding collision $(s, \Delta), (s', \Delta')$. Pick the smallest $i \in [\ell]$ such that $s_i \neq s'_i$ and $s_j = s'_j$ for all $1 \leq j < i$. Pick $j \in [\min(2^{i-1}, 2^k)]$ such that $\mathfrak{s}_{i,j} \neq \mathfrak{s}'_{i,j}$. Denote the length h to be such that $\mathfrak{s}_{i,j} \in R_{\mathbb{F}}^{2^h \cdot (2n\alpha)}$. Then we have

$$\begin{aligned} (I_{2^h} \otimes A) \cdot \mathfrak{s}_{i,j} &= (I_{2^h} \otimes A) \cdot \mathfrak{s}'_{i,j} \implies (I_{2^h} \otimes A) \cdot (\mathfrak{s}_{i,j} - \mathfrak{s}'_{i,j}) = 0 \\ &\implies (I_{2^h} \otimes A) \cdot (\Delta'_j \cdot \Delta_j \cdot (\mathfrak{s}_{i,j} - \mathfrak{s}'_{i,j})) = 0 \end{aligned}$$

Observe that because Δ_j, Δ'_j are invertible for $j \in [\min(2^{i-1}, 2^k)]$, $z := \Delta'_j \cdot \Delta_j \cdot (\mathfrak{s}_{i,j} - \mathfrak{s}'_{i,j}) \neq 0$, and so

$$\|\Delta'_j \cdot \Delta_j \cdot (\mathfrak{s}_{i,j} - \mathfrak{s}'_{i,j})\|_\infty \leq \|\Delta'_j \cdot (\Delta_j \cdot \mathfrak{s}_{i,j})\|_\infty + \|\Delta_j \cdot (\Delta'_j \cdot \mathfrak{s}'_{i,j})\|_\infty \leq 2 \cdot T_{\text{bck}} \cdot B$$

where T_{bck} is the expansion factor (Def. 15) of $\mathcal{C} - \mathcal{C}$. So z is a short solution to $(I_{2^h} \otimes A) \cdot x = 0$, which implies a $\text{MSIS}_{q,n,d,2n\alpha,2 \cdot T_{\text{bck}} \cdot B}$ solution on matrix A . $\square$

4 Sum-Checks Over Rings Using Isomorphisms

Our reductions make repeated use of the sum-check protocol over polynomials in $R_{\mathbb{F}}[X]$ and $R_{\mathbb{K}}[X]$ [LFKN92, BCS21]. Naively, each such sum-check would incur a transcript containing logarithmically many $R_{\mathbb{F}}$ and $R_{\mathbb{K}}$ elements, respectively. To compile these checks more efficiently, we employ isomorphisms between $R_{\mathbb{F}}$ and $\mathbb{K}^{d/e}$ and between $R_{\mathbb{K}}$ and $R_{\mathbb{F}}^e$ to batch many sum-check claims into a single sum-check over $\mathbb{K}$. See preliminaries Section 3.2 for details on the NTT and CRT isomorphisms used.

Relations. We define input and output relations used in both BatchSC (Prot. 1) with $S = R_{\mathbb{F}}, R = R_{\mathbb{F}}$ and ShiftSC (Prot. 2) with $S = R_{\mathbb{K}}, R = R_{\mathbb{F}}$.

$$\begin{aligned} \text{BSC}_{\text{in}}(S, a, b) &:= \left\{ \begin{array}{l} x := \left(\left(\begin{array}{l} v_i \in S, \\ \ell_i \in \mathbb{N}, \\ G_i \in S^{\leq D}[X_{[1:b]}] \\ (\ell_j \in \mathbb{N}, \llbracket Q_j \rrbracket)_{j \in [b]} \end{array} \right)_{i \in [a]}, \quad \forall i \in [a], \\ v_i = \sum_{X \in \{0,1\}^{\ell_i}} G_i(Q_1(X), \dots, Q_b(X)) \\ w := (Q_j \in S[X_{[1:\ell_j]}])_{j \in [b]} \end{array} \right\} \\ \text{BSC}_{\text{out}}(S, R, b) &:= \left\{ \begin{array}{l} x := \left(\begin{array}{l} \ell \in \mathbb{N}, r \in R^\ell, \\ (y_j \in S, \llbracket Q_j \rrbracket)_{j \in [b]} \end{array} \right), \quad \forall j \in [b], \\ y_j = Q_j(r_{[1:\ell_j]}) \\ w := (Q_j \in S[X_{[1:\ell_j]}])_{j \in [b]} \end{array} \right\} \end{aligned}$$

4.1 Batched Sum-Checks Over $R_{\mathbb{F}}$

We first present the BatchSC protocol, which batches many sum-check claims over $R_{\mathbb{F}}$ into a single sum-check over $\mathbb{K}$. The techniques used in BatchSC are collected from recent works [NS25, BC25a, KLN⁺26]; the BatchSC protocol presented in Protocol 1 unifies these techniques into a single procedure that will be run as a subroutine in the following reductions.

BatchSC takes as input sum-check claims on $G_i(Q_1(X), \dots, Q_b(X))$, where $\mathcal{V}$ has access to public multivariate polynomials ($G_i \in R_{\mathbb{F}}^{\leq D}[X_{[1:b]}]$) and to oracles for multilinear polynomials ($Q_j \in R_{\mathbb{F}}[X_{[1:\ell_j]}]_{j \in [b]}$). Note that the oracles to multilinear ($Q_j \in R_{\mathbb{F}}[X_{[1:\ell_j]}]_{j \in [b]}$) may either be polynomials that are publicly computable by $\mathcal{V}$ (e.g. $\tilde{eq}(X, r)$) or polynomials for which $\mathcal{V}$ only has query access to (e.g. witnesses $\tilde{s}(X)$). We do not formalize this distinction in the BatchSC protocol.

For each G_i , consider the subset $S_i \subseteq [b]$ for which G_i has a non-zero coefficient for the corresponding Q_j oracle. Let ℓ_i denote the variable count for $G_i(Q_1(X), \dots, Q_b(X)) \in R_{\mathbb{F}}[X_{[1:\ell_i]}]$. For this to be well-defined, we assume that for all $j \in S_i$, the variable count $\ell_j = \ell_i$.

Lastly, note that when calling BatchSC (and ShiftSC) in later protocols 3, 4, and 5, we abuse notation and pass in the sum-check claims rather than explicitly defining the combiner polynomials G_i and multilinear polynomials Q_j for simplicity. In Step 5a, the prover also does not need to send claims $y_j \in R_{\mathbb{F}}$ for oracles that the verifier can compute on its own, so we omit these claims from the protocol descriptions.

Protocol 1: BatchSC: Batched Sum-Check Subroutine over $R_{\mathbb{F}}$
Procedure: $\text{BatchSC}((v_i, \ell_i, G_i)_{i \in [a]}, (\ell_j, Q_j)_{j \in [b]}) \rightarrow (r, (y_j)_{j \in [b]})$ Input: Sum-check claims $v_i \stackrel{?}{=} \sum_{X \in \{0,1\}^{\ell_i}} G_i(Q_1(X), \dots, Q_b(X))$ for $i \in [a]$: $\left(\text{claimed sum } v_i \in R_{\mathbb{F}}, \text{ variable count } \ell_i \in \mathbb{N}, \text{ multivariate polynomial } G_i \in R_{\mathbb{F}}^{\leq D}[X_{[1:b]}] \right)_{i \in [a]},$ $\left(\text{variable count } \ell_j \in \mathbb{N}, \text{ oracles to multilinear polynomials } \llbracket Q_j \rrbracket \text{ where } Q_j \in R_{\mathbb{F}}[X_{[1:\ell_j]}] \right)_{j \in [b]}$ Output: random point $r \in R_{\mathbb{F}}^{\ell}$ (where $\ell := \max_i \ell_i$) and evaluations $(y_j \in R_{\mathbb{F}})_{j \in [b]}$. 1. $\mathcal{P}$ and $\mathcal{V}$ combine a sum-check claims into a single claim on $v \in R_{\mathbb{F}}$ and $G \in R_{\mathbb{F}}[X_{[1:\ell]}]$: (a) $\mathcal{V}$ samples $\eta \leftarrow_{\mathbb{S}} \mathcal{C}$ and sends it to $\mathcal{P}$. (b) $\mathcal{P}$ and $\mathcal{V}$ produce a new sum-check claim $v \stackrel{?}{=} \sum_{X \in \{0,1\}^{\ell}} G(Q_1(X), \dots, Q_b(X))$ where $v := \sum_{i=1}^a \eta^{i-1} \cdot 2^{\ell-\ell_i} \cdot v_i \in R_{\mathbb{F}}, \quad G(X) := \sum_{i=1}^a \eta^{i-1} \cdot G_i(Q_1(X), \dots, Q_b(X)) \in R_{\mathbb{F}}[X_{[1:\ell]}].$ 2. $\mathcal{P}$ applies NTT to produce an equivalent d/e independent (extension-field) sum-check claims on $v^{(k)} \in \mathbb{K}$ and $G^{(k)}(Q_1^{(k)}(X), \dots, Q_b^{(k)}(X)) \in \mathbb{K}[X_{[1:\ell]}]$: $v^{(k)} \stackrel{?}{=} \sum_{X \in \{0,1\}^{\ell}} G^{(k)}(Q_1^{(k)}(X), \dots, Q_b^{(k)}(X)) \in \mathbb{K} \quad \forall k \in [d/e],$ where $(v^{(1)}, \dots, v^{(d/e)}) \in \mathbb{K} \leftarrow \text{NTT}(v)$, $(Q_j^{(1)}, \dots, Q_j^{(d/e)}) \in \mathbb{K}[X_{[1:\ell]}] \leftarrow \text{NTT}(Q_j)$ for $j \in [b]$, and $(G^{(1)}, \dots, G^{(d/e)}) \in \mathbb{K}^{\leq D}[X_{[1:b]}] \leftarrow \text{NTT}(G)$. Note $\mathcal{V}$'s oracle $\llbracket Q_j \rrbracket$ acts as an oracle for $Q_j^{(k)}$, $k \in [d/e]$. 3. $\mathcal{P}$ and $\mathcal{V}$ combine the d/e extension-field claims into a single claim on $v^* \in \mathbb{K}$ and $G^* \in \mathbb{K}[X_{[1:\ell]}]$: (a) $\mathcal{V}$ samples $\gamma \leftarrow_{\mathbb{S}} \mathbb{K}$ and sends it to $\mathcal{P}$.

(b) $\mathcal{P}$ and $\mathcal{V}$ produce a new (extension-field) sum-check claim $v^* \stackrel{?}{=} \sum_{X \in \{0,1\}^\ell} G^*(X)$ where

$$v^* := \sum_{k=1}^{d/e} \gamma^{k-1} \cdot v^{(k)} \in \mathbb{K}, \quad G^*(X) := \sum_{k=1}^{d/e} \gamma^{k-1} \cdot G^{(k)}(Q_1^{(k)}(X), \dots, Q_b^{(k)}(X)) \in \mathbb{K}[X_{1:\ell}].$$

4. $\mathcal{P}$ and $\mathcal{V}$ run the sum-check protocol to get claim $(h \in \mathbb{K}^\ell, y \in \mathbb{K}) \leftarrow \text{SC}(v^*, \ell, G^*)$ over $\mathbb{K}$ (see Def. 9).
5. $\mathcal{P}$ sends $\mathcal{V}$ claimed evaluations from which $\mathcal{P}$ and $\mathcal{V}$ produce output evaluation claim $(r \in R_{\mathbb{F}}^\ell, (y_j \in R_{\mathbb{F}})_{j \in [b]})$:
 - (a) $\mathcal{P}$ sends $y_j := Q_j(r_{[1:\ell_j]}) \in R_{\mathbb{F}}$ for each $j \in [b]$, where $r \leftarrow \text{NTT}^{-1}(h, \dots, h) \in R_{\mathbb{F}}^\ell$.
 - (b) $\mathcal{V}$ computes $(y^{(1)}, \dots, y^{(d/e)}) \in \mathbb{K}^{d/e} \leftarrow \text{NTT}\left(\sum_{i=1}^m \eta^{i-1} \cdot G_i(y_1, \dots, y_b)\right)$ and checks $y \stackrel{?}{=} \sum_{k=1}^{d/e} \gamma^{k-1} \cdot y^{(k)} \in \mathbb{K}$. Note the isomorphism implies that $y^{(k)} = G^{(k)}(Q_1^{(k)}(h), \dots, Q_b^{(k)}(h))$ for each $k \in [d/e]$.
 - (c) $\mathcal{P}$ and $\mathcal{V}$ output $(r, (y_j)_{j \in [b]})$.

Lemma 23 (Informal). $\text{BatchSC} : \text{BSC}_{\text{in}}(R_{\mathbb{F}}, a, b) \rightarrow \text{BSC}_{\text{out}}(R_{\mathbb{F}}, R_{\mathbb{F}}, b)$ is complete with a linear time prover, logarithmic time verifier, and logarithmic proof size.

Proof. We defer the proof to Appendix B.2. □

Lemma 24 (Informal). $\text{BatchSC} : \text{BSC}_{\text{in}}(R_{\mathbb{F}}, a, b) \rightarrow \text{BSC}_{\text{out}}(R_{\mathbb{F}}, R_{\mathbb{F}}, b)$ is sound with negligible soundness error.

Proof. We defer the proof to Appendix B.2. □

4.2 Batched Sum-Checks Over $R_{\mathbb{K}}$

Next, we present the ShiftSC in Protocol 2, which batches sum-check claims over $R_{\mathbb{K}}$ into a single sum-check over $\mathbb{K}$ by applying $\text{CRT} : R_{\mathbb{K}} \rightarrow R_{\mathbb{F}}^e$ to the $R_{\mathbb{K}}$ -claims and then invoking BatchSC on the resulting $R_{\mathbb{F}}$ -claims. We write a more general formulation here, but when we invoke ShiftSC in Prot. 3, inputs Q_j are multilinear polynomial over the smaller ring $R_{\mathbb{F}}$ allowing the ShiftSC output claims to be over $R_{\mathbb{F}}$ as well.

Protocol 2: ShiftSC: Batched Sum-Check Subroutine over $R_{\mathbb{K}}$

Procedure: $\text{ShiftSC}((v_i, \ell_i, G_i)_{i \in [a]}, (\ell_j, Q_j)_{j \in [b]}) \rightarrow (r, (y_j)_{j \in [b]})$

Input: Sum-check claims $v_i \stackrel{?}{=} \sum_{X \in \{0,1\}^{\ell_i}} G_i(Q_1(X), \dots, Q_b(X))$ for $i \in [a]$:

$$\left(\text{claimed sum } v_i \in R_{\mathbb{K}}, \text{ variable count } \ell_i \in \mathbb{N}, \text{ multivariate polynomial } G_i \in R_{\mathbb{K}}^{\leq D}[X_{1:b}] \right)_{i \in [a]},$$

$$\left(\text{variable count } \ell_j \in \mathbb{N}, \text{ oracles to multilinear polynomials } \llbracket Q_j \rrbracket \text{ where } Q_j \in R_{\mathbb{K}}[X_{1:\ell_j}] \right)_{j \in [b]}$$

Output: random point $r \in R_{\mathbb{F}}^\ell$ (where $\ell := \max_i \ell_i$) and evaluations $(y_j \in R_{\mathbb{K}})_{j \in [b]}$.

1. $\mathcal{P}$ applies CRT to each sum-check claim to produce equivalent $e \cdot a$ independent $R_{\mathbb{F}}$ sum-check claims on $v_i^{(k)} \in R_{\mathbb{F}}$ and $G_i^{(k)}(Q_1^{(k)}(X), \dots, Q_b^{(k)}(X)) \in R_{\mathbb{F}}^{\leq D}[X_{1:\ell_i}]$:

$$v_i^{(k)} \stackrel{?}{=} \sum_{X \in \{0,1\}^\ell} G_i^{(k)}(Q_1^{(k)}(X), \dots, Q_b^{(k)}(X)) \in R_{\mathbb{F}} \quad \forall i \in [a], k \in [e],$$

where

$$\begin{aligned} (v_i^{(1)}, \dots, v_i^{(e)} \in R_{\mathbb{F}}) &\leftarrow \text{CRT}(v) \quad \forall i \in [a] \\ (G_i^{(1)}, \dots, G_i^{(e)} \in R_{\mathbb{F}}^{\leq D}[X_{1:b}]) &\leftarrow \text{CRT}(G) \quad \forall i \in [a] \\ (Q_j^{(1)}, \dots, Q_j^{(e)} \in R_{\mathbb{F}}[X_{1:\ell_j}]) &\leftarrow \text{CRT}(Q_j) \quad \forall j \in [b] \end{aligned}$$

Note $\mathcal{V}$'s oracle $\llbracket Q_j \rrbracket$ acts as an oracle for $Q_j^{(k)}$, $k \in [e]$.

2. $\mathcal{P}$ and $\mathcal{V}$ invoke BatchSC (Prot. 1) on the $e \cdot a$ claims (across $i \in [a]$ and $k \in [e]$) over b multilinear polynomials.

$$\begin{aligned} \left(r, (y_j^{(k)})_{j \in [b], k \in [e]} \right) &\leftarrow \text{BatchSC} \left((v_i^{(k)}, \ell_i, G_i^{(k)})_{i \in [a], k \in [e]}, \right. \\ &\quad \left. (\ell_j, Q_j^{(k)})_{j \in [b], k \in [e]} \right). \end{aligned}$$

3. $\mathcal{P}$ and $\mathcal{V}$ compute

$$y_j := \text{CRT}^{-1} \left(y_j^{(1)}, \dots, y_j^{(e)} \right) \in R_{\mathbb{K}} \quad j \in [b]$$

4. Output $(r \in R_{\mathbb{R}}^\ell, (y_j \in R_{\mathbb{K}})_{j \in [b]})$.

Lemma 25 (Informal). $\text{ShiftSC} : \text{BSC}_{\text{in}}(R_{\mathbb{K}}, a, b) \rightarrow \text{BSC}_{\text{out}}(R_{\mathbb{K}}, R_{\mathbb{F}}, b)$ is complete with a linear time prover, logarithmic time verifier, and logarithmic proof size.

Proof. We defer the proof to Appendix B.2. □

Lemma 26 (Informal). $\text{ShiftSC} : \text{BSC}_{\text{in}}(R_{\mathbb{K}}, a, b) \rightarrow \text{BSC}_{\text{out}}(R_{\mathbb{K}}, R_{\mathbb{F}}, b)$ is sound with negligible soundness error.

Proof. We defer the proof to Appendix B.2. □

5 Polynomial Commitment Scheme

We present our polynomial commitment scheme as a sequence of interactive reductions. Here, we describe the relevant relations. The subsequent subsections describe our 3 main reductions $\Pi_{\text{NC}}^{\text{PE}}$, Π_{Fold} , Π_{Dec} .

Relations. Let ℓ denote the commitment tree (see Def. 19) depth, n the commitment size, $\alpha = |g^\top|$ for gadget vector $g^\top$, and let $m := \log(n\alpha)$. $\text{PE}(L, \ell, \vec{b})$ describes L Ajtai tree commitments with evaluation claims on the bottom layer $\vec{s}_\ell$ on the same input point u . Here, each b_l , $l \in [L]$ in $\vec{b} = (b_1, \dots, b_L)$ denotes the norm bound for openings to commitment t_l .

$$\text{PE}(L, \ell, \vec{b}) := \left\{ \begin{array}{l} \mathbf{x} := \begin{pmatrix} (t_l \in R_{\mathbb{F}}^n)_{l \in [L]}, \\ u \in R_{\mathbb{F}}^{\ell+m}, \\ (v_l \in R_{\mathbb{F}})_{l \in [L]} \end{pmatrix}, \quad \forall l \in [L], \\ \mathbf{w} := (s_{i,l} \in R_{\mathbb{F}}^{2^i n \alpha})_{\substack{i \in [\ell], \\ l \in [L]}}, \end{array} : \begin{array}{l} \text{Open}(A, b_l, t_l, (s_{i,l})_{i \in [\ell]}) \\ \tilde{s}_{\ell,l}(u) = v_l \end{array} \right\}$$

$\text{TE}(L, \ell, \vec{b})$ describes L Ajtai tree commitments, each with an evaluation claim on the multilinear extension of the concatenated opening vector

$$s_l := 0^{2n\alpha} \|s_{1,l}\| \cdots \|s_{\ell,l}\| \in R_{\mathbb{F}}^{2^{\ell+1} n \alpha}$$

on the same input point $u \in R_{\mathbb{F}}^{\ell+1+m}$.

$$\text{TE}(L, \ell, \vec{b}) := \left\{ \begin{array}{l} \mathbf{x} := \begin{pmatrix} (t_l \in R_{\mathbb{F}}^n)_{l \in [L]}, \\ u \in R_{\mathbb{F}}^{\ell+1+m}, \\ (v_l \in R_{\mathbb{F}})_{l \in [L]} \end{pmatrix}, \quad \forall l \in [L], \\ \mathbf{w} := (s_{i,l} \in R_{\mathbb{F}}^{2^i n \alpha})_{\substack{i \in [\ell], \\ l \in [L]}}, \end{array} : \begin{array}{l} \text{Open}(A, b_l, t_l, (s_{i,l})_{i \in [\ell]}) \\ s_l := 0^{2n\alpha} \|s_{1,l}\| \cdots \|s_{\ell,l}\| \in R_{\mathbb{F}}^{2^{\ell+1} n \alpha} \\ \tilde{s}_l(u) = v_l \end{array} \right\}$$

Proving security. To prove completeness and knowledge soundness of the polynomial commitment scheme, we employ the strategy from [NS24] and prove each interactive reduction to be strong (Def. 7) or weak (Def. 6). To prove

the strong and weak properties, we require a *relaxed* notion of TE which has no norm requirements.

$$\text{TE}^*(L, \ell) := \left\{ \begin{array}{l} x := \left(\begin{array}{l} (t_l \in R_{\mathbb{F}}^n)_{l \in [L]}, \\ u \in R_{\mathbb{F}}^{\ell+1+m}, \\ (v_l \in R_{\mathbb{F}})_{l \in [L]} \end{array} \right), \quad \forall l \in [L], \\ w := \left(s_{i,l} \in R_{\mathbb{F}}^{2^{i n \alpha}} \right)_{\substack{i \in [\ell], \\ l \in [L]}}, \quad \begin{array}{l} \text{Open} \left(A, \frac{|\mathbb{F}|}{2}, t_l, (s_{i,l})_{i \in [\ell]} \right) \\ s_l := 0^{2^{n \alpha}} \|s_{1,l}\| \cdots \|s_{\ell,l}\| \in R_{\mathbb{F}}^{2^{\ell+1} n \alpha} \\ \tilde{s}_l(u) = v_l \end{array} \end{array} \right\}$$

In the next subsections, we will describe our 3 main reductions $\Pi_{\text{NC}}^{\text{PE}}$ (Sec. 5.1), Π_{Fold} (Sec. 5.2), Π_{Dec} (Sec. 5.3), and claim that their composition is a reduction of knowledge. Composing this cycle with a finishing protocol Π_{Fin} (Sec. 5.4) that is also a reduction of knowledge, we claim that our polynomial commitment scheme Maltese is public coin, complete, and knowledge sound.

Lemma 13. *Let $B := 2^k \cdot T \cdot b$, $h := \log_b B$. $\Pi_{\text{Dec}} \circ \Pi_{\text{Fold}} \circ \Pi_{\text{NC}}^{\text{PE}} : \text{PE}(1, \ell, b) \rightarrow \text{TE}(1, k, b) \times \text{PE}(1, \ell - k + 1 + \log h, b) \times \text{PE}(1, k, b)$ is a reduction of knowledge (Def. 4).*

Proof. By Thm. 1 (strong-weak reduction composition), Lem. 28 and 29 ($\Pi_{\text{NC}}^{\text{PE}}$ is a strong interactive reduction), Lem. 31 and 32 (Π_{Fold} is a weak interactive reduction), $\Pi_{\text{Fold}} \circ \Pi_{\text{NC}}^{\text{PE}}$ is a reduction of knowledge (Def. 4). By Lem. 2 (reduction of knowledge sequential composition) and Lem. 33 and Lem. 34 (Π_{Dec} is a reduction of knowledge), the cycle $\Pi_{\text{Dec}} \circ \Pi_{\text{Fold}} \circ \Pi_{\text{NC}}^{\text{PE}}$ is a reduction of knowledge. $\square$

Theorem 6. *Maltese : $\text{PE}(1, \ell, b) \rightarrow (0, 1)$ is a reduction of knowledge (Def. 4).*

Proof. By Lem. 13, the main cycle $\Pi_{\text{Dec}} \circ \Pi_{\text{Fold}} \circ \Pi_{\text{NC}}^{\text{PE}}$ is a reduction of knowledge (Def. 4). By Lem. 15, Π_{Fin} is a reduction of knowledge. Thus by Lem. 2, Maltese (see Fig. 2) is a reduction of knowledge. $\square$

5.1 Norm Check Reduction

This section presents the $\Pi_{\text{NC}}^{\text{PE}}$ reduction which reduces a polynomial evaluation claim to a tree evaluation claim while performing a norm check on the commitment opening. Performing a norm check allows this reduction to “fix” the relaxed norm openings of the folding reduction during extraction. We first present the arithmetization of the norm check and evaluation claim as sum-checks and then present the full reduction in Protocol 3.

5.1.1 Norm Check Sum-Check Claims

For input $(t_l, (s_{i,l})_{i \in [\ell]})_{l \in [L]} \in \text{PE}(L, \ell, \vec{b})$, $\Pi_{\text{NC}}^{\text{PE}}$ proves that $\|s_{i,l}\|_{\infty} < b_l$ for all $i \in [\ell]$ and $\tilde{s}_{\ell,l}(u) = v_l$ for all $l \in [L]$.

Norm check sum-check claim. The first sum-check claim directly checks infinity norm bounds of the tree openings $((s_{i,l})_{i \in [\ell]})_{l \in [L]}$. First consider the concatenated opening vector $s_l := 0^{2^{n \alpha}} \|s_{1,l}\| s_{2,l} \| \cdots \| s_{\ell,l} \in R_{\mathbb{F}}^{2^{\ell+1} n \alpha}$ for each $l \in [L]$ and observe that $\|s_{i,l}\|_{\infty} < b_l$ for all $i \in [\ell], l \in [L]$ if and only if $\|s_l\|_{\infty} < b_l$ for all $l \in [L]$. Thus, we focus on proving $\|s_l\|_{\infty} < b_l$ for all $l \in [L]$.

Define $N_b \in \mathbb{F}[X]$ to be a degree- $(2b-1)$ univariate polynomial for norm bound $b \in \mathbb{N}$:

$$N_b(X) := X \cdot \prod_{j=1}^{b-1} ((X-j) \cdot (X+j)). \quad (7)$$

Recall the coefficient map $\text{cf} : R_{\mathbb{F}}^n \rightarrow \mathbb{F}^{n \times d}$ (Def. 3) which maps elements in $R_{\mathbb{F}}$ to their coefficients in $\mathbb{F}^d$. Indexing by k corresponds to selecting the k^{th} coefficients of the ring elements $z \in R_{\mathbb{F}}^n$: $\text{cf}(z)_k \in \mathbb{F}^n$ for $k \in [d]$. The infinity norm $\|s_l\|_{\infty} < b_l$ for $l \in [L]$ if all the coefficient vectors $k \in [d]$ have infinity norm $\|\text{cf}(s_l)_k\|_{\infty} < b_l$. Finally, observe that the infinity norm over coefficient vectors in $\mathbb{F}$ can be checked via the following zero-check claim over the boolean hypercube, which can be checked via a sum-check (see Lem. 4):

$$N_{b_l}(\widetilde{\text{cf}(s_l)_k}(X)) = 0 \quad \text{over } X \in \{0, 1\}^{\ell+1+m} \quad (8)$$

Ring shift sum-check claim. The first sum-check claim above is over polynomials $\widetilde{\text{cf}(s_l)_k}(X) \in \mathbb{F}[X]$ and results in an evaluation claim $r \in \mathbb{K}^{\ell+1+m}$ over the larger extension field $\mathbb{K}$ (for soundness). We prove the following lemma that shows that we can interpret the d evaluation claims in $\mathbb{K}^{\ell+1+m}$ over each coefficient polynomial in $\mathbb{F}[X]$ as a single evaluation claim over the recomposed vector in $s_l \in R_{\mathbb{F}}^{2^{\ell+1}n\alpha}$ when interpreted as a polynomial in $R_{\mathbb{K}}[X]$. The new claim on $\tilde{s}_l \in R_{\mathbb{F}}[X]$ (as an $R_{\mathbb{K}}[X]$ polynomial) at the same point $r_N \in \mathbb{K}^{\ell+1+m}$ (as an $R_{\mathbb{K}}$ vector) is over $R_{\mathbb{K}}$.

Lemma 14. *Let $z \in R_{\mathbb{K}}^n$ and $r \in \mathbb{K}^{\log n}$, then for all $k \in [d]$, it holds that $\text{cf}(\tilde{z}(r))_k = \widetilde{\text{cf}(z)_k}(r) \in \mathbb{K}$.*

Proof. A multilinear evaluation $\tilde{z}(r)$ is computed by

$$\tilde{z}(r) = \langle z, \hat{r} \rangle = \sum_{i \in [n]} \hat{r}_i \cdot z_i, \quad \hat{r} := \bigotimes_{j=1}^{\log n} \begin{bmatrix} 1 - r_j \\ r_j \end{bmatrix} \in \mathbb{K}^n.$$

Consider the k^{th} coefficient of the evaluation:

$$\text{cf}(\tilde{z}(r))_k = \text{cf}\left(\sum_{i \in [n]} \hat{r}_i \cdot z_i\right)_k \stackrel{(1)}{=} \sum_{i \in [n]} \text{cf}(\hat{r}_i \cdot z_i)_k \stackrel{(2)}{=} \sum_{i \in [n]} \hat{r}_i \cdot \text{cf}(z_i)_k = \widetilde{\text{cf}(z)_k}(r).$$

Recall $R_{\mathbb{K}} := \mathbb{K}[X]/X^d + 1$. Equality (1) holds because adding $R_{\mathbb{K}}$ elements together corresponds to element-wise addition of their coefficient vectors. Equality (2) holds because $\hat{r}_i \in \mathbb{K}$ is a constant in $R_{\mathbb{K}}$ and multiplication corresponds to scaling all coefficients by $\hat{r}_i$. $\square$

The above lemma consolidates the many claims in $\mathbb{K}[X]$ into a single evaluation claim in $R_{\mathbb{K}}$ on $\tilde{s}_l \in R_{\mathbb{F}}[X]$ (recall $R_{\mathbb{F}} \subseteq R_{\mathbb{K}}$). However, for efficiency, we would like to reduce to a claim on the smaller ring $\tilde{s}_l \in R_{\mathbb{F}}[X]$ for the following reductions. The $R_{\mathbb{K}}$ -claim over $\tilde{s}_l$ is represented as the following sum-check claim which, when run with verifier challenges in $R_{\mathbb{F}}$, reduces to a claim in $R_{\mathbb{F}}$ on $\tilde{s}_l \in R_{\mathbb{F}}[X]$ at a random point r' in $R_{\mathbb{F}}^{\ell+1+m}$.

$$a_l = \sum_{X \in \{0,1\}^{\ell+1+m}} \tilde{e}q(r, X) \cdot \tilde{s}_l(X). \quad (9)$$

Original polynomial evaluation sum-check claim. Lastly, the original polynomial evaluation claim $\tilde{s}_{\ell,l}(u) = v_l$ must also be checked in the reduction. It can also be represented as a sum-check claim and is batched in with the above (represented as a claim over $R_{\mathbb{K}}$):

$$v_l = \sum_{X \in \{0,1\}^{\ell+1+m}} \tilde{e}q(X_1, 1) \cdot \tilde{e}q(u, X_{[2:]}) \cdot \tilde{s}_l(X). \quad (10)$$

The first input bit $X_1 = 1$ selects for $s_{\ell,l}$, as $s_{\ell,l} \in R_{\mathbb{F}}^{2^{\ell}n\alpha}$ makes up the second half of s_l .

5.1.2 Norm Check Reduction Protocol Description

Protocol 3: Norm check reduction $\Pi_{\text{NC}}^{\text{PE}}$	
Parameters: L commitments, ℓ tree layers per commitment, opening norm bound $\vec{b} = (b_1, \dots, b_L)$	
Input:	$\left(x := \begin{pmatrix} (t_l \in R_{\mathbb{F}}^n)_{l \in [L]}, \\ (u \in R_{\mathbb{F}}^{\ell+1+m}), \\ (v_l \in R_{\mathbb{F}})_{l \in [L]} \end{pmatrix}; w := (s_{i,l} \in R_{\mathbb{F}}^{2^i n \alpha})_{\substack{i \in [\ell], \\ l \in [L]}} \right) \in \text{PE}(L, \ell, \vec{b})$
Output:	$\left(x := \begin{pmatrix} (t_l \in R_{\mathbb{F}}^n)_{l \in [L]}, \\ (r' \in R_{\mathbb{F}}^{\ell+1+m}), \\ (y_l \in R_{\mathbb{F}})_{l \in [L]} \end{pmatrix}; w := (s_{i,l} \in R_{\mathbb{F}}^{2^i n \alpha})_{\substack{i \in [\ell], \\ l \in [L]}} \right) \in \text{TE}(L, \ell, \vec{b})$
1. $\mathcal{P}$ and $\mathcal{V}$ run SC over $\mathbb{K}$ to check the norm of coefficient polynomials $\text{cf}(s_l)_k$, reducing to a polynomial evaluation claim on $r_N \in \mathbb{K}^{\ell+1+m}$, $y_N \in \mathbb{K}$.	

- (a) $\mathcal{V}$ samples $\xi \leftarrow_{\$} \mathbb{K}$, $r \leftarrow_{\$} \mathbb{K}^{\ell+1+m}$ and sends them to $\mathcal{P}$.
- (b) $\mathcal{P}$ and $\mathcal{V}$ define the polynomial $Q_N \in \mathbb{K}[X_{[1:\ell+1+m]}]$, which batches infinity norm-check polynomials for $\text{cf}(s_l)_k$ (see Eq. (8)) across $l \in [L]$, $k \in [0, d-1]$ via challenge ξ :

$$Q_N(X) := \sum_{l \in [L]} \sum_{k \in [0, d-1]} \xi^{(l-1) \cdot d+k} \cdot N_{b_l} \left(\widetilde{\text{cf}(s_l)_k(X)} \right) \in \mathbb{K}[X_{[1:\ell+1+m]}]$$

where $N_b \in \mathbb{F}[X]$ is defined as in Eq. (7) and $\widetilde{\text{cf}(s_l)_k} \in \mathbb{F}[X_{[1:\ell+1+m]}]$ as in Def. 3.

- (c) $\mathcal{P}$ and $\mathcal{V}$ invoke Lem. 4 on Q_N with challenge r to produce a zero-check claim, and then run the sum-check protocol (see Def. 9):

$$(r_N \in \mathbb{K}^{\ell+1+m}, y_N \in \mathbb{K}) \leftarrow \text{SC}(0, \ell+1+m, \tilde{e}q(X, r) \cdot Q_N(X))$$

2. $\mathcal{P}$ and $\mathcal{V}$ invoke Lem. 14 to transform the polynomial evaluation claim on $r_N \in \mathbb{K}^{\ell+1+m}$, $y_N \in \mathbb{K}$ into evaluation claims $\tilde{s}_l(r_N) = a_l \in R_{\mathbb{K}}$.

- (a) $\mathcal{P}$ computes and sends $a_l := \tilde{s}_l(r_N) \in R_{\mathbb{K}}$, which $\mathcal{V}$ interprets via the coefficient map cf (Def. 3) as $(a_{0,l}, \dots, a_{d-1,l}) := \text{cf}(a_l) \in \mathbb{K}^{1 \times d}$.
- (b) $\mathcal{V}$ checks consistency of claims $a_{k,l}$ with y_N (see Lem. 14 with $s_l \in R_{\mathbb{F}}^{2^{\ell+1}n\alpha}$, $r_N \in \mathbb{K}$):

$$y_N \stackrel{?}{=} \tilde{e}q(r_N, r) \cdot \left(\sum_{l \in [L]} \sum_{k \in [0, d-1]} \xi^{(l-1) \cdot d+k} \cdot N_{b_l}(a_{k,l}) \right)$$

3. $\mathcal{P}$ and $\mathcal{V}$ run ShiftSC (Prot. 2) to reduce both the evaluation claims from Step 2 (see Eq. (9)) and the input evaluation claims (see Eq. (10)) to evaluation claims in $R_{\mathbb{F}}$ over the same input point r' .

- (a) For Step 2 evaluation claims $\tilde{s}_l(r_N) = a_l \in R_{\mathbb{K}}$, $\mathcal{P}$ and $\mathcal{V}$ define ShiftSC inputs (see Eq. (9)):

$$\forall l \in [L], C_{\text{sc}, l} : a_l \stackrel{?}{=} \sum_{X \in \{0,1\}^{\ell+1+m}} \tilde{e}q(r_N, X) \cdot \tilde{s}_l(X) \in R_{\mathbb{K}}$$

- (b) For input evaluation claims $\tilde{s}_{\ell, l}(u) = v_l \in R_{\mathbb{F}}$, $\mathcal{P}$ and $\mathcal{V}$ define ShiftSC inputs using the same Q_l (see Eq. (10)):

$$\forall l \in [L], C_{\text{in}, l} : v_l \stackrel{?}{=} \sum_{X \in \{0,1\}^{\ell+1+m}} \tilde{e}q(X_1, 1) \cdot \tilde{e}q(u, X_{[2:]}) \cdot \tilde{s}_l(X) \in R_{\mathbb{F}}$$

- (c) $\mathcal{P}$ and $\mathcal{V}$ invoke ShiftSC on the $2L$ claims from the previous two steps:

$$(r', (y_1, \dots, y_L \in R_{\mathbb{F}})) \leftarrow \text{ShiftSC}((C_{j,l})_{j \in \{\text{sc}, \text{in}\}, l \in [L]})$$

Note that since all $\{Q_j\}_{j \in [L] \cup \{\text{sc}, \text{in}\}}$ are over $R_{\mathbb{F}}$, outputs $\{y_j\}_{j \in [L] \cup \{\text{sc}, \text{in}\}}$ are also in $R_{\mathbb{F}} \subseteq R_{\mathbb{K}}$ (though ShiftSC outputs $R_{\mathbb{K}}$ values in general). Also, the evaluation claims corresponding to the publicly evaluable multilinear on r' can be computed by $\mathcal{V}$.

4. $\mathcal{P}$ and $\mathcal{V}$ output $x := ((t_l)_{l \in [L]}, r', (y_l)_{l \in [L]})$, $w := (s_{i,l})_{l \in [L]}$.

Lemma 28 (Informal). $\Pi_{\text{NC}}^{\text{PE}} : \text{PE}(L, \ell, b) \rightarrow \text{TE}(L, \ell, b)$ is complete with $O(bL \cdot N)$ prover time, $O(Lbd \cdot \log N)$ verifier time, and $O(\log N)$ proof size.

Proof. We defer the proof to Appendix B.3. □

Lemma 29 (Informal). $\Pi_{\text{NC}}^{\text{PE}} : \text{PE}(L, \ell, b) \rightarrow \text{TE}(L, \ell, b)$ ($\text{TE}^*(L, \ell)$) is a strong interactive reduction (Def. 7) with negligible knowledge error.

Proof. We defer the proof to Appendix B.3. □

5.2 Folding Reduction

This section presents the Π_{Fold} reduction which takes as input an ℓ -layer tree evaluation claim and produces (1) a top k -layer tree evaluation claim with respect to the original tree commitment, (2) a folded bottom $(\ell - k)$ -layer tree evaluation claim with respect to a folded commitment, and (3) a polynomial evaluation claim on an additional commitment introduced during the protocol as an optimization. We first present notation for the folded commitment t^* , then arithmetization of the sum-check claims, and then the full reduction in Protocol 4.

Commitment folding notation. As shown in Figure 3, the prover folds 2^k subtrees formed by the layer- k opening for input commitment tree t , $(s_i)_{i \in [\ell]}$. We define the opening for j th subtree for $j \in [2^k]$ by chunking each layer- $(k+i)$ opening into 2^k blocks of size $2^{i-n\alpha}$:

$$\forall i \in [0, \ell - k] \quad s_{k+i} = \mathfrak{s}_{k+i,1} \in R_{\mathbb{F}}^{2^{i-n\alpha}} \parallel \dots \parallel \mathfrak{s}_{k+i,2^k} \in R_{\mathbb{F}}^{2^{i-n\alpha}}$$

Then, recalling Eq. (4) in Section 2.2.2, for verifier challenges $c := c_1, \dots, c_{2^k} \leftarrow_{\S} \mathcal{C}$, the folded bottom tree commitment is t^* , $(s_i^*)_{i \in [\ell-k]}$ where

$$\begin{aligned} t^* &:= \sum_{j \in [2^k]} c_j \cdot (G_n \cdot \mathfrak{s}_{k,j}), \\ s_i^* &:= \sum_{j \in [2^k]} c_j \cdot \mathfrak{s}_{k+i,j} \quad \forall i \in [\ell - k], \end{aligned} \tag{11}$$

and the top tree commitment is t , $(s_i)_{i \in [k]}$.

The output of Π_{Fold} will include TE claims on t , $(s_i)_{i \in [k]}$ and t^* , $(s_i^*)_{i \in [\ell-k]}$, corresponding to polynomial claims on $\tilde{s}_{\text{top}} \in R_{\mathbb{F}}[X_{[1:k+1+m]}]$ and $\tilde{s}_{\text{bot}} \in R_{\mathbb{F}}[X_{[1:\ell-k+1+m]}]$:

$$\begin{aligned} s_{\text{top}} &:= 0^{2n\alpha} \parallel s_1 \parallel \dots \parallel s_k \in R_{\mathbb{F}}^{2^{k+1}n\alpha} \\ s_{\text{bot}} &:= 0^{2n\alpha} \parallel s_1^* \parallel \dots \parallel s_{\ell-k}^* \in R_{\mathbb{F}}^{2^{\ell-k+1}n\alpha} \\ &= \sum_{j \in [2^k]} c_j \cdot \mathfrak{s}_j \\ \text{where } \mathfrak{s}_j &:= 0^{2n\alpha} \parallel \mathfrak{s}_{k+1,j} \parallel \dots \parallel \mathfrak{s}_{\ell,j} \in R_{\mathbb{F}}^{2^{\ell-k+1}n\alpha} \end{aligned}$$

Relating input and output tree evaluations. As an intermediate step to eventually output evaluation claims on $\tilde{s}_{\text{top}}$ and $\tilde{s}_{\text{bot}}$, the prover will relate the input evaluation claim $v \stackrel{?}{=} \tilde{s}(u)$ to claims on $\tilde{s}_{\text{top}}$ and $\mathfrak{s}_j$. Define $\text{bits}_{\eta}(x)$ to be the bit representation of $x \in \mathbb{N}$ in $\eta \geq \log_2(x)$ bits, and expand the multilinear evaluation $\tilde{s}(u)$ as a sum:

$$\begin{aligned} \tilde{s}(u) &= \sum_{j=0}^{2^{\ell+1}n\alpha-1} s^{(j)} \cdot \tilde{e}q(\text{bits}_{\ell+1+m}(j), u) \\ &= \underbrace{\sum_{j=0}^{2^{k+1}n\alpha-1} s^{(j)} \cdot \tilde{e}q(\text{bits}_{\ell+1+m}(j), u)}_{\text{top } k \text{ layers indices}} + \underbrace{\sum_{j=2^{k+1}n\alpha}^{2^{\ell+1}n\alpha-1} s^{(j)} \cdot \tilde{e}q(\text{bits}_{\ell+1+m}(j), u)}_{\text{bottom } \ell - k \text{ layers indices}} \\ &= \tilde{e}q(0^{\ell-k}, u^{(:,\ell-k)}) \cdot \underbrace{\tilde{s}_{\text{top}}(u^{(\ell-k+1,:)})}_{\text{top tree claim}} + \underbrace{\sum_{j=2^{k+1}n\alpha}^{2^{\ell+1}n\alpha-1} s^{(j)} \cdot \tilde{e}q(\text{bits}_{\ell+1+m}(j), u)}_{\text{bottom subtrees claim}} \end{aligned} \tag{12}$$

Note in the third line that the indices (in bits) corresponding to the top k layers begin with $0^{\ell-k}$, and due to the structure of $\tilde{e}q$, we can pull this factor out. The prover sends the top tree claim and bottom subtrees claim to the verifier, which they will prove via sum-check. We describe these sum-checks in the next section.

5.2.1 Folding Sum-Check Claims

Helper powers polynomial. We introduce a polynomial which aids in producing batched claims on a verifier challenge in both the folding and norm decomposition (Sec. 5.3). For a verifier challenge $\eta \leftarrow_{\$} R_{\mathbb{F}}$, we define the powers vector for a power-of-two length 2^ν ,

$$p_{\eta, 2^\nu} := (1, \eta, \dots, \eta^{2^\nu-1}) = \bigotimes_{i=1}^{\nu} (1, \eta^{2^{\nu-i}}) \in R_{\mathbb{F}}^{2^\nu}.$$

with the following efficiently computable multilinear extension

$$\tilde{p}_{\eta, 2^\nu}(Y) = \prod_{i=1}^{\nu} ((1 - Y_i) + Y_i \cdot \eta^{2^{\nu-i}}) = \prod_{i=1}^{\nu} (1 + (\eta^{2^{\nu-i}} - 1) Y_i). \quad (13)$$

First sum-check: Bottom subtrees evaluation claim sum-check claim. $\mathcal{P}$ sends top tree claim $v_{\text{top}} := \tilde{s}_{\text{top}}(u^{(\ell-k+1:\cdot)}) \in R_{\mathbb{F}}$ and bottom subtrees claim $v_{\text{subs}} \in R_{\mathbb{F}}$, and $\mathcal{V}$ will check consistency with input claim v via Eq. (12). Then, $\mathcal{P}$ will prove claims v_{top} and v_{subs} . We begin with v_{subs} here, and later pair v_{top} with another sum-check claim on $\tilde{s}_{\text{top}}$. Returning to the second summand in Eq. (12), we rearrange the sum in terms of subtree concatenated opening vectors $\mathfrak{s}_j$.

$$\begin{aligned} & \sum_{j=2^{k+1}n\alpha}^{2^{\ell+1}n\alpha-1} s^{(j)} \cdot \tilde{e}q(\text{bits}_{\ell+1+m}(j), u) \\ &= \sum_{i=1}^{\ell-k} \sum_{j=1}^{2^k} \sum_{l=0}^{2^i n\alpha-1} s^{((2^{k+i}+(j-1) \cdot 2^i)n\alpha+l)} \cdot \tilde{e}q(\text{bits}_{\ell+1+m}((2^{k+i}+(j-1) \cdot 2^i)n\alpha+l), u) \\ & \quad (\text{index by tree layer, then by subtree, then by element}) \\ &= \sum_{i=1}^{\ell-k} \sum_{j=1}^{2^k} \tilde{e}q(0^{\ell-(k+i)} \| 1 \| \text{bits}_k(j-1), u^{(\cdot:\ell-i+1)}) \cdot \sum_{l=0}^{2^i n\alpha-1} s^{((2^{k+i}+(j-1) \cdot 2^i)n\alpha+l)} \cdot \tilde{e}q(\text{bits}_{i+m}(l), u^{(\ell-i+1:\cdot)}) \\ & \quad (\text{pull out the prefix for each layer and subtree}) \\ &= \sum_{i=1}^{\ell-k} \sum_{j=1}^{2^k} \tilde{e}q(0^{\ell-(k+i)} \| 1 \| \text{bits}_k(j-1), u^{(\cdot:\ell-i+1)}) \cdot \tilde{\mathfrak{s}}_{k+i,j}(u^{(\ell-i+1:\cdot)}) \\ & \quad (\text{rewrite the sum as an MLE}) \\ &= \sum_{j=1}^{2^k} \sum_{i=1}^{\ell-k} \tilde{e}q(0^{\ell-(k+i)} \| 1 \| \text{bits}_k(j-1), u^{(\cdot:\ell-i+1)}) \cdot \tilde{\mathfrak{s}}_{k+i,j}(u^{(\ell-i+1:\cdot)}). \\ & \quad (\text{flip the sum and index by subtree first}) \end{aligned}$$

We rewrite this summation, which is currently expressed as evaluations of subtree layer polynomials $\tilde{\mathfrak{s}}_{k+i,j}$, in terms of subtree concatenated opening polynomials $\mathfrak{s}_j$ for each $j \in [2^k]$ as appears in the TE relation.

$$\sum_{i=1}^{\ell-k} \underbrace{\tilde{e}q(0^{\ell-(k+i)} \| 1 \| \text{bits}_k(j-1), u^{(\cdot:\ell-i+1)})}_{e_{i,j} \in R_{\mathbb{F}}} \cdot \underbrace{\tilde{\mathfrak{s}}_{k+i,j}(u^{(\ell-i+1:\cdot)})}_{\langle f_i, \mathfrak{s}_{k+i,j} \rangle \in R_{\mathbb{F}}} = \langle q_{\text{subs},j}, \mathfrak{s}_j \rangle$$

where $e_{i,j} \in R_{\mathbb{F}}$ is the coefficient on the evaluation for the i th layer and j th subtree, and $f_i \in R_{\mathbb{F}}^{2^{k+i}n\alpha}$ is the vector corresponding to the multilinear evaluation $\tilde{\mathfrak{s}}_{k+i,j}(u^{(\ell-i+1:\cdot)})$.

$$\begin{aligned} e_{i,j} &:= \tilde{e}q(0^{\ell-(k+i)} \| 1 \| \text{bits}_k(j-1), u^{(\cdot:\ell-i+1)}) \in R_{\mathbb{F}} & \forall i \in [\ell-k] \\ \tilde{f}_i(X) &:= \tilde{e}q(u^{(\ell-i+2:\cdot)}, X) \in R_{\mathbb{F}}[X_{[1:i+m]}] & \forall i \in [\ell-k] \\ q_{\text{subs},j} &:= 0^{2n\alpha} \| e_{1,j} \cdot f_1 \| \cdots \| e_{\ell-k,j} \cdot f_{\ell-k} \in R_{\mathbb{F}}^{2^{\ell-k+1}n\alpha}. \end{aligned} \quad (14)$$

Observe that $\mathcal{V}$ can evaluate $\tilde{q}_{\text{subs},j}(X)$ in $O(\ell^2 + \ell m)$ time, as one can write the multilinear evaluation $\tilde{q}_{\text{subs},j}$ in terms of sums of $e_{i,j} \cdot \tilde{f}_i$, $i \in [\ell - k]$ and each $\tilde{f}_i$ takes $O(\ell + m)$ time to evaluate. So $\mathcal{P}$ sends

$$v_{\text{subs}} := \sum_{j=2^{k+1}n\alpha}^{2^{\ell+1}n\alpha-1} s^{(j)} \cdot \tilde{eq}(\text{bits}_{\ell+1+m}(j), u) = \sum_{j=1}^{2^k} \langle q_{\text{subs},j}, \mathfrak{s}_j \rangle \in R_{\mathbb{F}}.$$

and proves the evaluation claim via sum-check claim with respect to t :

$$v_{\text{subs}} = \sum_{X \in \{0,1\}^{\ell-k+1+m}} \left(\sum_{j=1}^{2^k} \tilde{q}_{\text{subs},j}(X) \cdot \tilde{\mathfrak{s}}_j(X) \right) \quad (15)$$

Second sum-check: Finishing the bottom subtrees evaluation sum-check. The first sum-check (Eq. 15) reduces to evaluation claims on $\tilde{\mathfrak{s}}_j$ at a random point $r \in R_{\mathbb{F}}^{\ell-k+1+m}$

$$y_{\text{subs}} \stackrel{?}{=} \sum_{j=1}^{2^k} \tilde{q}_{\text{subs},j}(r) \cdot \tilde{\mathfrak{s}}_j(r).$$

Recall that $\mathcal{V}$ can locally compute $\tilde{q}_{\text{subs},j}(r)$ for all $j \in [2^k]$. To avoid sending all 2^k of $\tilde{\mathfrak{s}}_j(r) \in R_{\mathbb{F}}$, the prover commits to these evaluations⁴:

$$\begin{aligned} f_{\text{subs}} &:= \tilde{\mathfrak{s}}_1(r) \parallel \dots \parallel \tilde{\mathfrak{s}}_{2^k}(r) \parallel 0^{2^k(n-1)} \in R_{\mathbb{F}}^{2^k n} \\ t_{\text{subs}}, (s_{\text{subs},i})_{i \in [k]} &:= \text{Commit}(A, f_{\text{subs}}) \end{aligned} \quad (16)$$

To prove consistency with y_{subs} , $\mathcal{P}$ produces the following sum-check claim using the observation that $g_b^{\mathsf{T}} = p_{b,\alpha}$:

$$y_{\text{subs}} = \sum_{X, X_g \in \{0,1\}^{k+m}} \left(\sum_{j=1}^{2^k} \tilde{q}_{\text{subs},j}(r) \cdot \tilde{eq}(X, 0^{\log n} \parallel \text{bits}_k(j-1)) \cdot \tilde{p}_{b,\alpha}(X_g) \cdot \tilde{\mathfrak{s}}_{\text{subs},k}(X, X_g) \right) \quad (17)$$

The polynomial $\tilde{eq}(X, 0^{\log n} \parallel \text{bits}_k(j-1))$ ignores the zero padding at the end of f_{subs} using prefix $0^{\log n}$ and then selects the j th element of $g_b^{\mathsf{T}} \cdot s_{\text{subs},k}$, which is $\tilde{\mathfrak{s}}_j(r) \in R_{\mathbb{F}}$.

Second sum-check: Well-formedness of t_{subs} . The prover must now prove that t_{subs} in fact commits to the $\tilde{\mathfrak{s}}_j(r)$. To do so, $\mathcal{P}$ batches claimed values for the $\tilde{\mathfrak{s}}_j(r)$ via the random verifier challenges $c := c_1, \dots, c_{2^k} \leftarrow_{\$} \mathcal{C}$ (which will also be used for commitment folding). Define batched claim:

$$v_{\text{bot}} := \sum_{j=1}^{2^k} c_j \cdot \tilde{\mathfrak{s}}_j(r).$$

Observe that this is also an evaluation claim on $v_{\text{bot}} = \tilde{\mathfrak{s}}_{\text{bot}}(r)$, as $s_{\text{bot}} = \sum_{j=1}^{2^k} c_j \cdot \mathfrak{s}_j$. The prover sends v_{bot} and proves via sum-check that it is consistent with the commitment t_{subs} :

$$v_{\text{bot}} = \sum_{X_k, Y, X_g \in \{0,1\}^{k+\log n+\log \alpha}} \tilde{c}(X_k) \cdot \tilde{eq}(Y, 0^{\log n}) \cdot \tilde{p}_{b,\alpha}(X_g) \cdot \tilde{\mathfrak{s}}_{\text{subs},k}(Y, X_k, X_g) \quad (18)$$

Second sum-check: Well-formedness of folded commitment t^* . Recall that the prover also uses verifier challenges $c \leftarrow_{\$} \mathcal{C}^{2^k}$ to produce folded commitment t^* . t^* is of the form in Eq. (11) if and only if

$$\forall i \in [n] \quad t^{*(i)} = \sum_{j \in [2^k]} c_j \cdot (g_b^{\mathsf{T}} \cdot \mathfrak{s}_{k,j})^{(i)} \in R_{\mathbb{F}}$$

⁴We pad f_{subs} to length $2^k n$ so that we can later fold this commitment with the top tree commitment t , $(s_i)_{i \in [k]}$ in the finishing protocol.

To batch these checks across $i \in [n]$ using verifier challenge $\eta \leftarrow_{\$} R_{\mathbb{F}}$, we use the powers polynomial⁵ (see Eq. (13)) assuming n is a power of two to write the batched claim in inner product/tensor form:

$$\sum_{i=1}^n \eta^{i-1} \cdot t^{*(i)} = \sum_{i=1}^n \eta^{i-1} \cdot \left(\sum_{j \in [2^k]} c_j \cdot (g_b^T \cdot \mathbf{s}_{k,j})^{(i)} \right) \iff \langle p_{\eta,n}, t^* \rangle = \langle c \otimes p_{\eta,n} \otimes p_{b,\alpha}, \mathbf{s}_k \rangle$$

This again uses the fact $g_b^T = p_{b,\alpha}$. Rewritten as a sum-check claim in terms of $\tilde{s}_{\text{top}}$,

$$\langle p_{\eta,n}, t^* \rangle = \sum_{X_1, X_c, X_n, X_g \in \{0,1\}^{k+1+m}} \tilde{c}(X_c) \cdot \tilde{p}_{\eta,n}(X_n) \cdot \tilde{p}_{b,\alpha}(X_g) \cdot \tilde{e}q(X_1, 1) \cdot \tilde{s}_{\text{top}}(X_1, X_c, X_n, X_g) \quad (19)$$

where $\tilde{c}$ is the multilinear extension of verifier challenges $c \leftarrow_{\$} \mathcal{C}^{2^k}$, and the bit $X_1 = 1$ selects for s_k within $\mathbf{s}_{\text{top}}$. Note that the verifier can compute $\langle p_{\eta,n}, t^* \rangle$.

Second sum-check: Top tree evaluation sum-check claim. This sum-check proves $v_{\text{top}} := \tilde{s}_{\text{top}}(u^{(\ell-k+1:\cdot)}) \in R_{\mathbb{F}}$. We include this in the second sum-check so that the sum-check claim in Eq. (19) and the following sum-check claim reduce to a unified claim on $\tilde{s}_{\text{top}}$.

$$v_{\text{top}} = \sum_{X \in \{0,1\}^{k+1+m}} \tilde{e}q(X, u^{(\ell-k+1:\cdot)}) \cdot \tilde{s}_{\text{top}}(X) \quad (20)$$

5.2.2 Folding Reduction Protocol Description

Protocol 4: Folding reduction Π_{Fold}

Parameters: ℓ layers, norm bound b , arity 2^k

Input:

$$(x := (t \in R_{\mathbb{F}}^n, u \in R_{\mathbb{F}}^{\ell+1+m}, v \in R_{\mathbb{F}}), w := (s_i \in R_{\mathbb{F}}^{2^i n \alpha})_{i \in [\ell]}) \in \text{TE}(1, \ell, b)$$

Output:

$$\begin{aligned} (x_{\text{top}} := (t, r', y_{\text{top}}), w_{\text{top}} := (s_i)_{i \in [k]}) &\in \text{TE}(1, k, b), \\ (x_{\text{bot}} := (t^*, r, v_{\text{bot}}), w_{\text{bot}} := (s_i^*)_{i \in [\ell-k]}) &\in \text{TE}(1, \ell-k, B), \\ (x_{\text{subs}} := (t_{\text{subs}}, r'^{(\cdot:k+m)}, y_{\text{subs},k}), w_{\text{subs}} := (s_{\text{subs},i})_{i \in [k]}) &\in \text{PE}(1, k, b) \end{aligned}$$

where $B = 2^k \cdot T \cdot b$.

1. $\mathcal{P}$ produces claimed evaluations $v_{\text{top}} \in R_{\mathbb{F}}$ on $\tilde{s}_{\text{top}}$, $v_{\text{subs}} \in R_{\mathbb{F}}$ summed over $\mathbf{s}_j$, which $\mathcal{V}$ checks for consistency with input evaluation claim v (see Eq. (12)).

- (a) $\mathcal{P}$ computes top tree layers evaluation claim v_{top} on $\tilde{s}_{\text{top}}$ and bottom tree layers evaluation claim v_{subs} :

$$\begin{aligned} v_{\text{top}} &:= \tilde{s}_{\text{top}}(u^{(\ell-k+1:\cdot)}) \\ v_{\text{subs}} &:= \sum_{j=2^{k+1}n\alpha}^{2^{\ell+1}n\alpha-1} s^{(j)} \cdot \tilde{e}q(\text{bits}_{\ell+1+m}(j), u) \end{aligned}$$

as in Eq. (12). $\mathcal{P}$ sends $v_{\text{top}} \in R_{\mathbb{F}}$, $v_{\text{subs}} \in R_{\mathbb{F}}$ to $\mathcal{V}$.

- (b) $\mathcal{V}$ checks for consistency of v_{top} , v_{subs} with input evaluation claim v (see Eq. (12)):

$$v \stackrel{?}{=} \tilde{e}q(0^{\ell-k}, u^{(\cdot:\ell-k)}) \cdot v_{\text{top}} + v_{\text{subs}}$$

⁵In this reduction, the verifier could also send n challenges rather than using the powers MLE, but the norm decomposition reduction (Sec. 5.3) will require batching 2^ℓ claims, which would be too large for the verifier.

2. $\mathcal{P}$ and $\mathcal{V}$ run BatchSC (Prot. 1) to check evaluation claim v_{subs} as a sum over $\mathfrak{s}_j$ (see Eq. (15)) which reduces to a claim on $r \in R_{\mathbb{F}}^{\ell-k+1+m}$. Instead of sending evaluations for all subtrees $\tilde{\mathfrak{s}}_j(r)$, $\mathcal{P}$ commits to the evaluations within t_{subs} (see Eq. (16)).

- (a) $\mathcal{P}$ and $\mathcal{V}$ construct BatchSC inputs for the sum-check claim in Eq. (15) to check bottom tree layers claim v_{subs} over $\mathfrak{s}_j, j \in [2^k]$:

$$C_{\text{subs}} : v_{\text{subs}} \stackrel{?}{=} \sum_{X \in \{0,1\}^{\ell-k+1+m}} \left(\sum_{j=1}^{2^k} \tilde{q}_{\text{subs},j}(X) \cdot \tilde{\mathfrak{s}}_j(X) \right) \in R_{\mathbb{F}}$$

where $q_{\text{subs},j}(X) \in R_{\mathbb{F}}^{2^{\ell-k+1}n\alpha}$ are defined in Eq. (14).

- (b) $\mathcal{P}$ and $\mathcal{V}$ invoke BatchSC on the 1 claim from 4(a), up to Step 4 producing $(e_{\mathbb{K}} \in \mathbb{K}^{\ell-k+1+m}, y_{\mathbb{K}} \in \mathbb{K})$ to avoid sending evaluations $\tilde{\mathfrak{s}}_j(r)$:

$$(h_{\mathbb{K}} \in \mathbb{K}^{\ell-k+1+m}, y_{\mathbb{K}} \in \mathbb{K}) \xleftarrow{\text{step 4}} \text{BatchSC}(C_{\text{subs}})$$

- (c) $\mathcal{P}$ and $\mathcal{V}$ compute $r \leftarrow \text{NTT}^{-1}(h_{\mathbb{K}}, \dots, h_{\mathbb{K}}) \in R_{\mathbb{F}}^{\ell-k+1+m}$. $\mathcal{P}$ commits to all subtree evaluations $\tilde{\mathfrak{s}}_j(r), j \in [2^k]$:

$$f_{\text{subs}} := \tilde{\mathfrak{s}}_1(r) \parallel \dots \parallel \tilde{\mathfrak{s}}_{2^k}(r) \parallel 0^{2^k(n-1)} \in R_{\mathbb{F}}^{2^k n}$$

$$t_{\text{subs}} \in R_{\mathbb{F}}^n, (s_{\text{subs},i} \in R_{\mathbb{F}}^{2^i n\alpha})_{i \in [k]} := \text{Commit}(A, f_{\text{subs}}),$$

and evaluates

$$y_{\text{subs}} := \sum_{j=1}^{2^k} \tilde{q}_{\text{subs},j}(r) \cdot \tilde{\mathfrak{s}}_j(r) \in R_{\mathbb{F}}.$$

$\mathcal{P}$ sends $t_{\text{subs}} \in R_{\mathbb{F}}^n, y_{\text{subs}} \in R_{\mathbb{F}}$ to $\mathcal{V}$.

- (d) $\mathcal{V}$ computes $(y^{(1)}, \dots, y^{(d/e)}) \in \mathbb{K}^{d/e} \leftarrow \text{NTT}(y_{\text{subs}})$ and checks $y_{\mathbb{K}} \stackrel{?}{=} \sum_{k=1}^{d/e} \gamma^{k-1} \cdot y^{(k)}$.

3. $\mathcal{P}$ and $\mathcal{V}$ fold together the 2^k subtrees to produce commitment t^* with openings $(s_i^*)_{i \in [\ell-k]}$ (Eq. (11)). $\mathcal{P}$ also computes claim $v_{\text{bot}} \in R_{\mathbb{F}}$ on $\tilde{\mathfrak{s}}_{\text{bot}}$.

- (a) $\mathcal{V}$ samples $c := c_1, \dots, c_{2^k} \leftarrow_{\mathbb{S}} \mathcal{C}^{2^k}$ and sends to $\mathcal{P}$.

- (b) $\mathcal{P}$ computes the folded commitment and openings as in Eq. (11):

$$t^* = \sum_{j=1}^{2^k} c_j \cdot G_n \cdot \mathfrak{s}_{k,j} \in R_{\mathbb{F}}^n, \quad s_i^* = \sum_{j=1}^{2^k} c_j \cdot \mathfrak{s}_{k+i,j} \in R_{\mathbb{F}}^{2^i n\alpha}, \quad \forall i \in [\ell-k],$$

and sends $t^* \in R_{\mathbb{F}}^n$ to $\mathcal{V}$.

- (c) $\mathcal{P}$ computes claim $v_{\text{bot}} \in R_{\mathbb{F}}$ on $\tilde{\mathfrak{s}}_j(r)$ folded along challenges $c \in \mathcal{C}^{2^k}$, which is also a claim on $\tilde{\mathfrak{s}}_{\text{bot}}$:

$$v_{\text{bot}} := \sum_{j=1}^{2^k} c_j \cdot \tilde{\mathfrak{s}}_j(r) \in R_{\mathbb{F}},$$

and sends $v_{\text{bot}} \in R_{\mathbb{F}}$ to $\mathcal{V}$.

4. $\mathcal{P}$ and $\mathcal{V}$ run BatchSC (Prot. 1) to batch together sum-check evaluation claims which check (1) y_{subs} from Step 2 with respect to commitment t_{subs} (see Eq. (17)), (2) the well-formedness of t_{subs} (see Eq. (18)), (3) the well-formedness of t^* with challenges c (see Eq. (19)), and (4) the top layers evaluation claim v_{top} on $\tilde{\mathfrak{s}}_{\text{top}}$ (see. Eq. (20)).

- (a) $\mathcal{P}$ and $\mathcal{V}$ construct BatchSC inputs for the sum-check claim in Eq. (17) to check sum-check evaluation claim y_{subs} from Step 2 with respect to commitment t_{subs} :

$$C_{\text{sc}} : y_{\text{subs}} \stackrel{?}{=} \sum_{X, X_g \in \{0,1\}^{k+m}} \left(\sum_{j=1}^{2^k} \tilde{q}_{\text{subs},j}(r) \cdot \tilde{e}q(X, 0^{\log n} \parallel \text{bits}_k(j-1)) \cdot \tilde{p}_{b,\alpha}(X_g) \cdot \tilde{s}_{\text{subs},k}(X, X_g) \right)$$

where $\tilde{q}_{\text{subs},j}(r)$ is calculated by the verifier.

- (b) $\mathcal{P}$ and $\mathcal{V}$ construct BatchSC inputs for the sum-check claim in Eq. (18) to check the well-formedness of t_{subs} :

$$C_{\text{tsubs}} : v_{\text{bot}} \stackrel{?}{=} \sum_{X_k, Y, X_g \in \{0,1\}^{k+m}} \tilde{c}(X_k) \cdot \tilde{e}q(Y, 0^{\log n}) \cdot \tilde{p}_{b,\alpha}(X_g) \cdot \tilde{s}_{\text{subs},k}(Y, X_k, X_g)$$

- (c) $\mathcal{P}$ and $\mathcal{V}$ construct BatchSC inputs for the sum-check claim in Eq. (19) to check commitment folding correctness on t^*, s_k using challenges c .

- i. $\mathcal{V}$ samples $\eta \leftarrow_{\$} \mathcal{C}$ and sends it to $\mathcal{P}$.
- ii. $\mathcal{P}$ and $\mathcal{V}$ construct BatchSC inputs for Eq. (19) using challenge η to construct $p_{\eta,n}$ (see Eq. (13)):

$$C_{\text{cf}} : \langle p_{\eta,n}, t^* \rangle \stackrel{?}{=} \sum_{\substack{X_1, X_c, X_n, X_g \\ \in \{0,1\}^{k+1+m}}} \tilde{c}(X_c) \cdot \tilde{p}_{\eta,n}(X_n) \cdot \tilde{p}_{b,\alpha}(X_g) \cdot \tilde{e}q(X_1, 1) \cdot \tilde{s}_{\text{top}}(X_1, X_c, X_n, X_g)$$

- (d) $\mathcal{P}$ and $\mathcal{V}$ construct BatchSC inputs for the sum-check claim in Eq. (20) to check evaluation claim v_{top} on $\tilde{s}_{\text{top}}$:

$$C_{\text{top}} : v_{\text{top}} \stackrel{?}{=} \sum_{X \in \{0,1\}^{k+1+m}} \tilde{e}q(X, u^{(\ell-k+1:\cdot)}) \cdot \tilde{s}_{\text{top}}(X)$$

- (e) $\mathcal{P}$ and $\mathcal{V}$ invoke BatchSC on the 4 claims from 4a-4d:

$$(r', (y_{\text{subs},k}, y_{\text{top}} \in R_{\mathbb{F}})) \leftarrow \text{BatchSC}(C_{\text{sc}}, C_{\text{tsubs}}, C_{\text{cf}}, C_{\text{top}})$$

where evaluations of the other multilinear polynomials used in the sum-check claims on r' are calculated by $\mathcal{V}$.

5. $\mathcal{P}$ and $\mathcal{V}$ output

$$\begin{aligned} (x_{\text{top}} &:= (t, r', y_{\text{top}}), w_{\text{top}} := (s_i)_{i \in [k]}) \in \text{TE}(1, k, b), \\ (x_{\text{bot}} &:= (t^*, r, v_{\text{bot}}), w_{\text{bot}} := (s_i^*)_{i \in [\ell-k]}) \in \text{TE}(1, \ell-k, B), \\ (x_{\text{subs}} &:= (t_{\text{subs}}, r'^{(:k+m)}, y_{\text{subs},k}), w_{\text{subs}} := (s_{\text{subs},i})_{i \in [k]}) \in \text{PE}(1, k, b) \end{aligned}$$

where $B = 2^k \cdot T \cdot b$.

Lemma 31 (Informal). *Let $B = 2^k \cdot T \cdot b$. $\Pi_{\text{Fold}} : \text{TE}(1, \ell, b) \rightarrow \text{TE}(1, k, b) \times \text{TE}(1, \ell-k, B) \times \text{PE}(1, k, b)$ is complete with $O(N)$ prover, $O(2^k \cdot \log^2 N)$ verifier time, and $O(\log N)$ proof size.*

Proof. We defer the proof to Appendix B.4. □

Lemma 32 (Informal). *Assuming $\text{MSIS}_{q,n,d,rn\alpha,4 \cdot T_{\text{bck}} \cdot B}$ where T_{bck} is the expansion factor of $\mathcal{C} - \mathcal{C}$ (Def. 15), $\Pi_{\text{Fold}} : \text{TE}(1, \ell, b) \rightarrow \text{TE}^*(1, \ell) \rightarrow \text{TE}(1, k, b) \times \text{TE}(1, \ell-k, B) \times \text{PE}(1, k, b)$ is a weak interactive reduction (Def. 6) with negligible knowledge error.*

Proof. We defer the proof to Appendix B.4. □

5.3 Norm Decomposition Reduction

This section presents the Π_{Dec} reduction which decomposes input commitment opening tree $t, (s_i)_{i \in [\ell]}$ with $\|s_i\|_{\infty} < b^h$ into a new tree commitment of norm bound b . $\mathcal{P}$ proves correct decomposition via sum-check.

As shown in Figure 4, $\mathcal{P}$ commits to the concatenated decomposed openings vector $s_{\text{dec}} \in R_{\mathbb{F}}^{h \cdot 2^{\ell+1} n \alpha}$ with the following claimed structure:

$$\begin{aligned} s_{\text{dec}} &:= s_{\text{dec},1} \parallel \dots \parallel s_{\text{dec},h} \in R_{\mathbb{F}}^{h \cdot 2^{\ell+1} n \alpha}, \\ \forall j \in [h], s_{\text{dec},j} &:= (0^{2n\alpha} \parallel s_{\text{dec},1,j} \parallel s_{\text{dec},2,j} \parallel \dots \parallel s_{\text{dec},\ell,j}) \in R_{\mathbb{F}}^{2^{\ell+1} n \alpha}, \quad \|s_{\text{dec},j}\|_{\infty} < b, \\ \forall i \in [\ell], \underline{s}_i &:= \sum_{j=1}^h b^{j-1} \cdot s_{\text{dec},i,j} \end{aligned} \quad (21)$$

Here, the recomposed $\underline{s}_i \in R_{\mathbb{F}}^{2^i n \alpha}$ are claimed to be the openings of the input commitment t . Further, the fact that $\underline{s} = \sum_{j=1}^h b^{j-1} \cdot s_{\text{dec},j}$ will be useful for converting sum-checks and evaluation claims on $\underline{s}$ into ones over s_{dec} . Querying $\tilde{s}$ at any point r is equivalent to querying each $s_{\text{dec},j}$ at u and taking the same linear combination of the evaluations: $\tilde{s}(u) = \sum_{j=1}^h b^{j-1} \cdot \tilde{s}_{\text{dec},j}(u)$.

5.3.1 Decomposition Sum-Check Claims

To prove that $(\underline{s}_i)_{i \in [\ell]}$ are valid openings for input commitment t , we must check the following identities, given only query access to $\tilde{s}_{\text{dec}}$ and an assumption that $\|s_{\text{dec}}\|_{\infty} < b$ (this will be incorporated into the output claim of the reduction):

$$t = A \cdot \underline{s}_1, \quad \text{and} \quad (I_{2^i} \otimes G_n) \cdot \underline{s}_i = (I_{2^i} \otimes A) \cdot \underline{s}_{i+1} \quad \forall i \in [\ell-1]. \quad (22)$$

Consider the following rewrite of Eq. (22), where the layer openings s_i are broken into chunks corresponding to the size of the matrix that they're being multiplied by, i.e. blocks of size $n\alpha$ when multiplied by G_n and $2n\alpha$ when multiplied by A . Denote the $n\alpha$ blocks by $\dot{s}_{i,l} \in R_{\mathbb{F}}^{n\alpha}$ and the $2n\alpha$ blocks by $\ddot{s}_{i+1,l} \in R_{\mathbb{F}}^{2n\alpha}$ for $l \in [2^i]$:

$$\begin{aligned} t &= A \cdot \dot{s}_{1,1} \\ \begin{bmatrix} G_n \cdot \dot{s}_{i,1} \in R_{\mathbb{F}}^{n\alpha} \\ \vdots \\ G_n \cdot \dot{s}_{i,2^i} \in R_{\mathbb{F}}^{n\alpha} \end{bmatrix} &= \begin{bmatrix} A \cdot \ddot{s}_{i+1,1} \in R_{\mathbb{F}}^{2n\alpha} \\ \vdots \\ A \cdot \ddot{s}_{i+1,2^i} \in R_{\mathbb{F}}^{2n\alpha} \end{bmatrix} \quad \forall i \in [\ell-1]. \end{aligned}$$

Let $a_1^{\top}, \dots, a_n^{\top} \in R_{\mathbb{F}}^{1 \times 2n\alpha}$ be the rows of $A \in R_{\mathbb{F}}^{n \times 2n\alpha}$. The above equation holds if and only if

$$\forall r \in [n], \quad t^{(r)} = a_r^{\top} \cdot \dot{s}_{1,1} \quad (23)$$

and

$$\forall r \in [n], i \in [2, \ell], p \in [2^i], \quad g_b^{\top} \cdot \dot{s}_{i,p}^{((r-1)\alpha+1:r\alpha)} = a_r^{\top} \cdot \ddot{s}_{i+1,p}. \quad (24)$$

where the indexing $((r-1)\alpha+1:r\alpha)$ is due to the tensor structure of the gadget matrix $I_n \otimes g^{\top}$.

Eq. (23) and Eq. (24) for $r \in [n], i \in [\ell], p \in [2^i]$ can be batched using verifier challenges $\eta, \xi \leftarrow_{\$} \mathcal{C}$, which combine over opening rows $r \in [n]$ and opening layer blocks $i \in [\ell], p \in [2^i]$, respectively:

$$\begin{aligned} \xi \cdot \sum_{r=1}^n \eta^{r-1} \cdot t^{(r)} &+ \underbrace{\sum_{r=1}^n \eta^{r-1} \cdot \left(\sum_{i=1}^{\ell-1} \sum_{p=1}^{2^i} \xi^{2^i+p-1} \cdot g_b^{\top} \cdot \dot{s}_{i,p}^{((r-1)\alpha+1:r\alpha)} \right)}_{G\text{-matrix constraints sum-check claim}} \\ &= \underbrace{\sum_{r=1}^n \eta^{r-1} \cdot \left(\sum_{i=0}^{\ell-1} \sum_{p=1}^{2^i} \xi^{2^i+p-1} \cdot a_r^{\top} \cdot \ddot{s}_{i+1,p} \right)}_{A\text{-matrix constraints sum-check claim}} \end{aligned} \quad (25)$$

Looking ahead, we begin the powers of ξ with ξ^1 rather than $\xi^0 = 1$ because we will be writing this in terms of the concatenated opening vector $s = 0^{2n\alpha} \parallel s_1 \parallel \dots \parallel s_{\ell}$ and so the ξ^0 power will coincide with the $0^{2n\alpha}$ prefix. The prover will send a claimed sum-check claim on the A matrix constraints and a claimed sum-check claim on the G matrix constraints. The verifier can verify the consistency of these claims with Eq. (25), and then the prover and verifier will conduct a sum-check to check both the A and G matrix constraint claims. We will now rewrite these two sums as sum-check claims.

A-matrix constraints sum-check claim. Define the following vector q_A which combines $a_r \in R_{\mathbb{F}}^{2n\alpha}$, $r \in [n]$ via powers of challenge $\xi \in \mathcal{C}$:

$$\begin{aligned} q_A &:= \left\|_{r=1}^n p_{\xi, 2^\ell} \otimes a_r \right. \\ &= a_1 \|\xi \cdot a_1\| \cdots \|\xi^{2^\ell-1} \cdot a_1\| \cdots \cdots \cdots \|a_n \|\xi \cdot a_n\| \cdots \underbrace{\|\xi^{2^\ell-1} \cdot a_n\|}_{2n\alpha} \in R_{\mathbb{F}}^{n \cdot 2^{\ell+1}n\alpha} \end{aligned}$$

recalling $p_{\eta, 2^\nu} := (1, \eta, \dots, \eta^{2^\nu-1})$ as defined in Eq. (13). Then consider the tensor product

$$\begin{aligned} p_{H, n} \otimes s &= s \|\eta \cdot s\| \cdots \|\eta^{n-1} \cdot s\| \\ &= 0^{2n\alpha} \|\tilde{s}_{1,1}\| \|\tilde{s}_{2,1}\| \cdots \|\tilde{s}_{\ell, 2^\ell}\| 0^{2n\alpha} \|\eta \cdot \tilde{s}_{1,1}\| \cdots \cdots \cdots \underbrace{\|\eta^{n-1} \cdot \tilde{s}_{\ell, 2^\ell}\|}_{2n\alpha} \in R_{\mathbb{F}}^{n \cdot 2^{\ell+1}n\alpha}. \end{aligned}$$

The inner product of these two vectors is the RHS sum in Eq. (25):

$$v_A := \langle q_A, p_{\eta, n} \otimes s \rangle = \sum_{r=1}^n \eta^{r-1} \cdot \left(\sum_{i=0}^{\ell-1} \sum_{p=1}^{2^i} \xi^{2^i+p-1} \cdot a_r^\top \cdot \tilde{s}_{i+1,p} \right).$$

$\mathcal{P}$ will send $v_A \in R_{\mathbb{F}}$ and prove that it is correctly formed via sum-check. We now rewrite the statement above as a sum-check claim, but in terms of $\tilde{s}_{\text{dec}}$, as this is the polynomial that $\mathcal{V}$ gets query access to. Define the following related vector to q_A :

$$q_{\text{dec}, A} := \left\|_{r=1}^n \textcolor{blue}{p}_{b, h} \otimes p_{\xi, 2^\ell} \otimes a_r \in R_{\mathbb{F}}^{n \cdot h \cdot 2^{\ell+1}n\alpha} \quad (26)$$

Observe that since $s_{\text{dec}} = s_{\text{dec}, 1} \|\cdots\| s_{\text{dec}, h} \in R_{\mathbb{F}}^{h \cdot 2^{\ell+1}n\alpha}$,

$$v_A := \langle q_A, p_{\eta, n} \rangle = \langle q_{\text{dec}, A}, p_{\eta, n} \otimes s_{\text{dec}} \rangle$$

as well. Finally, rewrite this statement as a sum-check claim

$$v_A = \sum_{\substack{X_d, X_r, X_c, X_r \\ \in \{0,1\}^{\log h + \ell + 1 + m + \log n}}} \tilde{p}_{b, h}(X_d) \cdot \tilde{A}(X_r, X_c) \cdot \tilde{p}_{\xi, 2^\ell}(X) \cdot \tilde{p}_{\eta, n}(X_r) \cdot \tilde{s}_{\text{dec}}(X_d, X, X_c) \quad (27)$$

where $\tilde{A}$ is the MLE of A viewed as a vector in $R_{\mathbb{F}}^{n \cdot 2n\alpha}$. $\tilde{q}_{\text{dec}, A}(X_d, X_r, X_c, X) := \tilde{p}_{b, h}(X_d) \cdot \tilde{A}(X_r, X_c) \cdot \tilde{p}_{\xi, 2^\ell}(X)$ is evaluable by $\mathcal{V}$ at any point in $O(\log h + n^2\alpha + \ell)$ time.

G-matrix constraints sum-check claim. First observe that $p_{b, \alpha} = g^\top$. We can define a similar vector q_G using verifier challenges ξ, η :

$$\begin{aligned} q_G &:= (p_{\xi, 2^\ell} \otimes p_{\eta, n}) \otimes p_{b, \alpha} \|\ 0^{2^\ell n\alpha} \\ &= 1 \|b\| \cdots \|b^{\alpha-1}\| \|\xi\| \|\xi \cdot b\| \cdots \cdots \cdots \|\eta^{n-1} \cdot \xi^{2^\ell-1} \cdot b^{\alpha-1}\| 0^{2^\ell n\alpha} \in R_{\mathbb{F}}^{2^{\ell+1}n\alpha}, \end{aligned}$$

where the $0^{2^\ell n\alpha}$ suffix is because s_ℓ does not appear on the LHS of Eq. (25). Then it similarly holds that

$$v_G := \langle q_G, s \rangle = \sum_{r=1}^n \eta^{r-1} \cdot \left(\sum_{i=1}^{\ell-1} \sum_{p=1}^{2^i} \xi^{2^i+p-1} \cdot g_b^\top \cdot \dot{s}_{i,p}^{((r-1)\alpha+1:r\alpha)} \right)$$

The structure of q_G is slightly different from that of q_A because $g_b^\top$ is multiplied by every block in every commitment row, as opposed to when defining q_A , where the a_r change across $r \in [n]$. Now we similarly lift to an inner product with s_{dec} by defining related $q_{\text{dec}, G}$:

$$q_{\text{dec}, G} := \textcolor{blue}{p}_{b, h} \otimes \left((p_{\xi, 2^\ell} \otimes p_{\eta, n}) \otimes p_{b, \alpha} \| 0^{2^\ell n\alpha} \right) \in R_{\mathbb{F}}^{h \cdot 2^{\ell+1}n\alpha} \quad (28)$$

Observe it also holds that

$$v_G := \langle q_G, s \rangle = \langle q_{\text{dec},G}, s_{\text{dec}} \rangle.$$

The corresponding sum-check claim is:

$$v_G = \sum_{\substack{X_d, X_0, X, Y, X_g \\ \in \{0,1\}^{\log h + 1 + \ell + m}}} \tilde{p}_{b,h}(X_d) \cdot \tilde{e}q(X_0, 0) \cdot \tilde{p}_{\xi, 2^\ell}(X) \cdot \tilde{p}_{\eta, n}(Y) \cdot \tilde{p}_{b,\alpha}(X_g) \cdot \tilde{s}_{\text{dec}}(X_d, X_0, X, Y, X_g) \quad (29)$$

$\tilde{q}_{\text{dec},G}(X_d, X_0, X, Y, X_g) = \tilde{p}_{b,h}(X_d) \cdot \tilde{e}q(X_0, 0) \cdot \tilde{p}_{\xi, 2^\ell}(X) \cdot \tilde{p}_{\eta, n}(Y) \cdot \tilde{p}_{b,\alpha}(X_g)$ can be evaluated by $\mathcal{V}$ at any point in $O(\log h + \ell + m)$ time.

Evaluation claim sum-check claim. This sum-check claim proves the input evaluation claim $\tilde{s}(u) = v$ while only having query access to $\tilde{s}_{\text{dec}}$. $\tilde{s}(u) = v$ if and only if

$$v = \sum_{j=1}^h b^{j-1} \cdot \tilde{s}_{\text{dec},j}(u).$$

The corresponding sum-check claim is:

$$v = \sum_{X_d, X \in \{0,1\}^{\log h + \ell + 1 + m}} \tilde{p}_{b,h}(X_d) \cdot \tilde{e}q(u, X) \cdot \tilde{s}_{\text{dec}}(X_d, X) \quad (30)$$

where $\tilde{q}_{\text{dec},E}(X_d, X) := \tilde{p}_{b,h}(X_d) \cdot \tilde{e}q(u, X)$ is evaluable by $\mathcal{V}$ at any point in $O(\log h + \ell + m)$ time.

Zero prefix sum-check claim. This sum-check enforces s_{dec} is well-formed in the sense that the recomposed s begins with a $0^{2n\alpha}$ prefix, as is assumed in the construction of all three previous sum-check claims. For verifier challenge $\gamma \leftarrow_{\$} \mathcal{C}$, the sum-check claim is:

$$0 = \sum_{X_d, X_1, X_2 \in \{0,1\}^{\log h + \ell + 1 + m}} \tilde{p}_{b,h}(X_d) \cdot \tilde{e}q(0^\ell, X_1) \cdot \tilde{p}_{\gamma, 2n\alpha}(X_2) \cdot \tilde{s}_{\text{dec}}(X_d, X_1, X_2) \quad (31)$$

$\tilde{e}q(X_1, 0^\ell)$ selects the length $2n\alpha$ prefix, and then $p_{\gamma, 2n\alpha}$ batches the entries of the prefix using verifier challenge γ to check that each entry is 0. $\tilde{p}_{b,h}(X_d) \cdot \tilde{e}q(X_1, 0^\ell) \cdot \tilde{p}_{\gamma, 2n\alpha}(X_2)$ is evaluable by $\mathcal{V}$ at any point in $O(\log h + \ell + m)$ time.

We assume h is a power of two in the protocol below, but could pad s_{dec} and $p_{b,h}$ with zeros at the end if this is not the case.

5.3.2 Norm Decomposition Reduction Protocol Description

Protocol 5: Norm-decomposition reduction Π_{Dec}
Parameters: ℓ layers, input norm bound b^h , decomposition base b , h decomposition factors, $m := \log(n\alpha)$, $\mu := \ell + 1 + \log h$ Input: $(x := (t \in R_{\mathbb{F}}^n, u \in R_{\mathbb{F}}^{\ell+1} \times R_{\mathbb{F}}^m, v \in R_{\mathbb{F}}), w := (s_i \in R_{\mathbb{F}}^{2^i n\alpha})_{i \in [\ell]}) \in \text{TE}(1, \ell, b^h)$ Output: $(x_{\text{dec}} := (t_{\text{dec}}^*, r^{(\cdot; \mu+m)}, y_{\text{dec}}), w_{\text{dec}} := (s_{\text{dec}}^*, i)_{i \in [\mu]}) \in \text{PE}(1, \mu, b)$
1. $\mathcal{P}$ produces commitment $t_{\text{dec}}^* \in R_{\mathbb{F}}^n$ to the vector $s_{\text{dec}} \in R_{\mathbb{F}}^{h \cdot 2^{\ell+1} n\alpha}$, which is the concatenation of base- b decomposition vectors of $s \in R_{\mathbb{F}}^{2^{\ell+1} n\alpha}$ (see Eq. (21)) via the function $\text{split}_b(s)$ (see Def. 1). (a) $\mathcal{P}$ decomposes the concatenated layer opening $s \in R_{\mathbb{F}}^{2^{\ell+1} n\alpha}$ into base- b digits: $(s_{\text{dec},j})_{j \in [h]} := \text{split}_b(s), \quad \text{so } s = \sum_{j=1}^h b^{j-1} \cdot s_{\text{dec},j} \in R_{\mathbb{F}}^{2^i n\alpha}, \quad \ s_{\text{dec},j}\ _{\infty} < b.$ $\mathcal{P}$ computes $(t_{\text{dec}}^*, (s_{\text{dec}}^*, i)_{i \in [\mu]}) := \text{Commit}(A, s_{\text{dec}})$ and sends $t_{\text{dec}}^* \in R_{\mathbb{F}}^n$ to $\mathcal{V}$.

2. $\mathcal{P}$ and $\mathcal{V}$ invoke BatchSC to prove both the linear commitment opening constraints for input $(t, (s_i)_{i \in [\ell]})$ in TE (see Eq. (22), (29), (27)), and the input evaluation claim $v = \tilde{s}(u)$ (see Eq. (30)) with respect to t_{dec}^* .

(a) $\mathcal{P}$ and $\mathcal{V}$ construct BatchSC inputs to prove both the linear commitment opening constraints for expressions involving the gadget matrix in Eq. (29).

- i. $\mathcal{V}$ samples $\eta, \xi, \gamma \leftarrow_{\S} \mathcal{C}$ and sends them to $\mathcal{P}$.
- ii. $\mathcal{P}$ computes sum-check claimed sum v_G :

$$v_G := \langle q_{\text{dec}, G}, s_{\text{dec}} \rangle$$

where $q_{\text{dec}, G}$ is defined using $\mathcal{V}$ challenges η, ξ as in Eq. (28), and sends $v_G \in R_{\mathbb{F}}$ to $\mathcal{V}$.

iii. $\mathcal{P}$ and $\mathcal{V}$ construct BatchSC inputs for the sum-check claim in Eq. (29):

$$C_G : v_G \stackrel{?}{=} \sum_{\substack{X_d, X_0, X, Y, X_g \\ \in \{0,1\}^{\mu+m}}} \tilde{p}_{b,h}(X_d) \cdot \tilde{eq}(X_0, 0) \cdot \tilde{p}_{\xi, 2^\ell}(X) \cdot \tilde{p}_{\eta, n}(Y) \cdot \tilde{p}_{b, \alpha}(X_g) \cdot \tilde{s}_{\text{dec}}(X_d, X_0, X, Y, X_g)$$

(b) $\mathcal{P}$ and $\mathcal{V}$ construct BatchSC inputs to prove both the linear commitment opening constraints for expressions involving the A matrix in Eq. (27).

- i. Using the same $\mathcal{V}$ challenges η, ξ from Step 2(a)i, $\mathcal{P}$ computes sum-check claimed sum v_A :

$$v_A := \langle q_{\text{dec}, A}, p_{\eta, n} \otimes s_{\text{dec}} \rangle$$

where $q_{\text{dec}, A}$ is defined using $\mathcal{V}$ challenges η, ξ as in Eq. (26), and sends $v_A \in R_{\mathbb{F}}$ to $\mathcal{V}$.

ii. $\mathcal{P}$ and $\mathcal{V}$ construct BatchSC inputs for the sum-check claim in Eq. (27):

$$C_A : v_A \stackrel{?}{=} \sum_{\substack{X_d, X, X_c, X_r \\ \in \{0,1\}^{\mu+m+\log n}}} \tilde{p}_{b,h}(X_d) \cdot \tilde{A}(X_r, X_c) \cdot \tilde{p}_{\xi, 2^\ell}(X) \cdot \tilde{p}_{\eta, n}(X_r) \cdot \tilde{s}_{\text{dec}}(X_d, X, X_c)$$

(c) $\mathcal{V}$ checks consistency of sum-check claimed sums v_G and v_A via Eq. (25):

$$\xi \cdot \langle p_{\eta, n}, t \rangle + v_G \stackrel{?}{=} v_A.$$

(d) $\mathcal{P}$ and $\mathcal{V}$ construct BatchSC inputs to prove evaluation claim v as in Eq. (30):

$$C_{\text{ev}} : v \stackrel{?}{=} \sum_{X_d, X \in \{0,1\}^{\mu+m}} \tilde{p}_{b,h}(X_d) \cdot \tilde{eq}(u, X) \cdot \tilde{s}_{\text{dec}}(X_d, X)$$

(e) $\mathcal{P}$ and $\mathcal{V}$ construct BatchSC inputs to prove that s has a 0 prefix as in Eq. (31):

$$C_{\text{pre}} : 0 \stackrel{?}{=} \sum_{X_d, X_1, X_2 \in \{0,1\}^{\mu+m}} \tilde{p}_{b,h}(X_d) \cdot \tilde{eq}(0^\ell, X_1) \cdot \tilde{p}_{\gamma, 2n\alpha}(X_2) \cdot \tilde{s}_{\text{dec}}(X_d, X_1, X_2)$$

(f) $\mathcal{P}$ and $\mathcal{V}$ invoke BatchSC on the 4 claims from Steps 2a-2b, 2d-2e:

$$(r, y_{\text{dec}} \in R_{\mathbb{F}}) \leftarrow \text{BatchSC}(C_G, C_A, C_{\text{ev}}, C_{\text{pre}})$$

where evaluations of public multilinear polynomials in the sum-check claims on r are calculated by $\mathcal{V}$.

3. $\mathcal{P}$ and $\mathcal{V}$ output

$$(x_{\text{dec}} := (t_{\text{dec}}^*, r^{(\cdot: \mu+m)}, y_{\text{dec}}), w_{\text{dec}} := (s_{\text{dec}}^*, i)_{i \in [\mu]}) \in \text{PE}(1, \mu, b)$$

Lemma 33 (Informal). $\Pi_{\text{Dec}} : \text{TE}(1, \ell, b^h) \rightarrow \text{PE}(1, \mu, b)$ is complete, with $O(N \cdot h)$ sum-check prover costs, $O(\log(N \cdot h))$ verifier costs, and $O(\log(N \cdot h))$ proof size.

Proof. We defer the proof to Appendix B.5. □

Lemma 34 (Informal). $\Pi_{\text{Dec}} : \text{TE}(1, \ell, b^h) \rightarrow \text{PE}(1, \mu, b)$ is knowledge sound with negligible knowledge error.

Proof. We defer the proof to Appendix B.5. □

5.4 Greyhound-Based Finishing Protocol

The main cycle of our polynomial commitment scheme consists of ω rounds of $\Pi_{\text{Dec}} \circ \Pi_{\text{Fold}} \circ \Pi_{\text{NC}}^{\text{PE}}$ and leaves remaining claims in $\text{TE}(1, k, b)^\omega \times \text{PE}(1, k, b)^{\omega+1}$. We would like to use Greyhound to resolve these remaining k -layer claims. However, Greyhound does not operate over TCom, but rather a different two-layer commitment scheme from LaBRADOR [BS23]. Our finishing protocol therefore consists of two components: a recommitment via the two-layer commitment scheme to prepare for using Greyhound, and then we run (a slightly modified) Greyhound to finish.

Section 5.4.2 describes how to reduce relations $\text{TE}(1, k, b)^\omega \times \text{PE}(1, k, b)^{\omega+1}$ to a Greyhound-friendly relation $\text{OE}^{(2)}$, which we define with other relevant relations in Section 5.4.1. This happens via two reductions: (1) a reduction of knowledge Π_{Reshape} , which recommits to all the witnesses in $\text{TE}(1, k, b)^\omega \times \text{PE}(1, k, b)^{\omega+1}$ with a single, two-layer commitment scheme compatible with Greyhound (see Def. 23), and (2) a strong interactive reduction $\Pi_{\text{NC}, \text{fin}}^{\text{PE}}$, which is a minor modification to the norm-check reduction (see Sec. 5.1) to output $\text{OE}^{(2)}$.

Lastly, Section 5.4.3 describes a weak interactive reduction GH' , which is a minor modification to the initial three-move subprotocol in Greyhound so that it can support claims of the form $\text{OE}^{(2)}$. $\text{OE}^{(2)}$ includes an evaluation claim on each commitment layer opening rather than just the bottom layer, as in the input relation to Greyhound. Putting these interactive reductions together, the finishing protocol is:

$$\begin{aligned} \Pi_{\text{Fin}} &:= \text{GH}' \circ \Pi_{\text{NC}, \text{fin}}^{\text{PE}} \circ \Pi_{\text{Reshape}} \text{ where } \Pi_{\text{Reshape}} : \text{TE}(1, k, b)^\omega \times \text{PE}(1, k, b)^{\omega+1} \rightarrow \text{PE}^{(2)}(b_1, b) \\ \Pi_{\text{NC}, \text{fin}}^{\text{PE}} &: \text{PE}^{(2)}(b_1, b) \rightarrow \text{OE}^{(2)}(b_1, b) \text{ (OE}^{*(2)}\text{)} \\ \text{GH}' &: \text{OE}^{(2)}(b_1, b) \text{ (OE}^{*(2)}\text{)} \rightarrow (0, 1). \end{aligned}$$

Lemma 15 (Π_{Fin} is a reduction of knowledge). $\Pi_{\text{Fin}} : \text{TE}(1, k, b)^\omega \times \text{PE}(1, k, b)^{\omega+1} \rightarrow (0, 1)$ is a reduction of knowledge (Def. 4) with negligible knowledge error.

Proof. By Cor. 5 (GH' is weak), Cor. 2, 3 ($\Pi_{\text{NC}, \text{fin}}^{\text{PE}}$ is strong), and Thm. 1 (weak-strong composition), $\text{GH}' \circ \Pi_{\text{NC}, \text{fin}}^{\text{PE}}$ is a reduction of knowledge. Then by Lem. 35, 36 (Π_{Reshape} is a reduction of knowledge) and Lem. 2 (RoK sequential composition), Π_{Fin} is a reduction of knowledge. $\square$

5.4.1 Finishing Protocol Relations

In this section, we describe the Greyhound-style two-layer commitment and then define two new relations: one relation analogous to PE, and new relation which includes an evaluation claim on each layer of the Greyhound commitment.

Greyhound commitment. This commitment is similar to the Ajtai tree commitment used throughout the main protocol (see Def. 19), except that it is two-layer commitment which may use different sized commitment keys (B_1, B_2) and norm decomposition bases (b_1, b_2) at each layer. This commitment scheme is described in Def. 23 and is computationally binding. A binding collision on the first or second layer reduces to $\text{MSIS}_{q, n, d, 2^\gamma n \alpha_1, 2b_1}$ or $\text{MSIS}_{q, n, d, 2^\beta \alpha_2, 2b_2}$, respectively.

Definition 23 (Two-layer Ajtai commitments [BS23]). *The commitment scheme is a tuple of algorithms* $(\text{GHSetup}, \text{GHCommit}, \text{GHOpen})$. *Let* $b_1, b_2 \in \mathbb{N}_{>1}$. *Denote* $\alpha_1 := \lceil \log_{b_1} q \rceil$, $\alpha_2 := \lceil \log_{b_2} q \rceil$. γ, β *are size parameters for the commitment keys* $B_1 \in R_{\mathbb{F}}^{n \times 2^\gamma \cdot n \cdot \alpha_1}$, $B_2 \in R_{\mathbb{F}}^{n \times 2^\beta \cdot \alpha_2}$.

- $\text{GHSetup}(1^\lambda, \beta, \gamma) \rightarrow \text{pp}$: *Sample a random matrix* $B_1 \leftarrow_{\$} R_{\mathbb{F}}^{n \times 2^\gamma \cdot n \cdot \alpha_1}$ *and* $B_2 \leftarrow_{\$} R_{\mathbb{F}}^{n \times 2^\beta \cdot \alpha_2}$, *where dimension* n *is chosen appropriately for security parameter* λ . *Return* $\text{pp} \leftarrow \{B_1, B_2\}$.
- $\text{GHCommit}((B_1, B_2), (b_1, b_2), s_2 \in R_{\mathbb{F}}^{2^{\gamma+\beta} \alpha_2}) \rightarrow (t \in R_{\mathbb{F}}^n, \text{st})$:
Input condition: $\|s_2\|_\infty < b_2$. *Compute commitment and openings:*

$$\begin{aligned} s_{1,i} &\leftarrow G_{b_1, n}^{-1}(B_2 \cdot s_{2,i}) \in R_{\mathbb{F}}^{n \cdot \alpha_1} \quad \forall i \in [2^\gamma] \\ t &\leftarrow B_1 \cdot s_1 \in R_{\mathbb{F}}^n. \end{aligned}$$

Return $(t, \text{st} \leftarrow (s_1, s_2))$.

- $\text{GHOpen}((B_1, B_2) \leftarrow pp, (b_1, b_2), t, (s_1, s_2) \leftarrow st) \rightarrow 0/1$: Return 1 if

$$t = B_1 \cdot s_1, \quad G_{b_1, n \cdot 2^\gamma} \cdot s_1 = (I_{2^\gamma} \otimes B_2) \cdot s_2, \quad \|s_1\|_\infty < b_1, \quad \|s_2\|_\infty < b_2.$$

Else, return 0.

Relations. First, recall the principal relation of Greyhound [NS24] with slightly different notation to match our description of the Greyhound commitment in Def. 23:

$$\text{GHPrincipal}(b_1, b_2) := \left\{ \begin{array}{l} \mathbf{x} := \begin{pmatrix} t \in R_{\mathbb{F}}^n, \\ a \in R_{\mathbb{F}}^{2^{\beta} \cdot \alpha_2}, \\ b \in R_{\mathbb{F}}^{2^\gamma}, \\ y_1 \in R_{\mathbb{F}} \end{pmatrix}, \quad \begin{array}{l} t = B_1 \cdot s_1 \\ G_{b_1, n \cdot 2^\gamma} \cdot s_1 = (I_{2^\gamma} \otimes B_2) \cdot s_2 \\ \forall i \in [2], \|s_i\|_\infty < b_i \end{array} \\ \mathbf{w} := \begin{pmatrix} s_1 \in R_{\mathbb{F}}^{2^\gamma \cdot n \cdot \alpha_1}, \\ s_2 \in R_{\mathbb{F}}^{2^{\gamma+\beta} \cdot \alpha_2} \end{pmatrix} \quad a^\top \cdot s_2 \cdot b = y \end{array} \right\}$$

Using this new commitment scheme, we also define an analogous relation to PE. We will show in Section 5.4.3 that $\text{GHPrincipal}(b_1, b_2)$ captures $\text{PE}^{(2)}(b_1, b_2)$.

$$\text{PE}^{(2)}(b_1, b_2) := \left\{ \begin{array}{l} \mathbf{x} := \begin{pmatrix} t \in R_{\mathbb{F}}^n, \\ u \in R_{\mathbb{F}}^{\gamma+\beta+\log \alpha_2}, \\ y \in R_{\mathbb{F}} \end{pmatrix}, \quad \begin{array}{l} t = B_1 \cdot s_1 \\ G_{b_1, n \cdot 2^\gamma} \cdot s_1 = (I_{2^\gamma} \otimes B_2) \cdot s_2 \\ \forall i \in [2], \|s_i\|_\infty < b_i \end{array} \\ \mathbf{w} := \begin{pmatrix} s_1 \in R_{\mathbb{F}}^{2^\gamma \cdot n \cdot \alpha_1}, \\ s_2 \in R_{\mathbb{F}}^{2^{\gamma+\beta} \cdot \alpha_2} \end{pmatrix} \quad \tilde{s}_2(u) = y \end{array} \right\}$$

Greyhound alone is a weak interactive reduction (Def. 6), but we would like Π_{Fin} to be a reduction of knowledge. So rather than directly passing $\text{PE}^{(2)}(b_1, b_2)$ to Greyhound, we first run a norm-check reduction $\Pi_{\text{NC}, \text{fin}}^{\text{PE}}$ which is a strong interactive reduction (Def. 7). Because $\Pi_{\text{NC}, \text{fin}}^{\text{PE}}$ needs to check the norm of both opening layers, it results in an evaluation claim on both $\tilde{s}_1, \tilde{s}_2$. The output of $\Pi_{\text{NC}, \text{fin}}^{\text{PE}}$ is then captured by the following relation, which has an evaluation claim on *each* opening layer of the two-layer commitment in Def. 23.

$$\text{OE}^{(2)}(b_1, b_2) := \left\{ \begin{array}{l} \mathbf{x} := \begin{pmatrix} t \in R_{\mathbb{F}}^n, \\ u_1 \in R_{\mathbb{F}}^{\gamma+\log(n\alpha_1)}, \\ u_2 \in R_{\mathbb{F}}^{\gamma+\beta+\log \alpha_2}, \\ y_1, y_2 \in R_{\mathbb{F}} \end{pmatrix}, \quad \begin{array}{l} t = B_1 \cdot s_1 \\ G_{b_1, n \cdot 2^\gamma} \cdot s_1 = (I_{2^\gamma} \otimes B_2) \cdot s_2 \\ \forall i \in [2], \|s_i\|_\infty < b_i \end{array} \\ \mathbf{w} := \begin{pmatrix} s_1 \in R_{\mathbb{F}}^{2^\gamma \cdot n \cdot \alpha_1}, \\ s_2 \in R_{\mathbb{F}}^{2^{\gamma+\beta} \cdot \alpha_2} \end{pmatrix} \quad \forall i \in [2], \tilde{s}_i(u_i) = y_i \end{array} \right\}$$

To define GH' as a weak interactive reduction (Def. 6), we also introduce the relaxed relation $\text{OE}^{*(2)}$:

$$\text{OE}^{*(2)} := \left\{ \begin{array}{l} \mathbf{x} := \begin{pmatrix} t \in R_{\mathbb{F}}^n, \\ u_1 \in R_{\mathbb{F}}^{\gamma+\log(n\alpha_1)}, \\ u_2 \in R_{\mathbb{F}}^{\gamma+\beta+\log \alpha_2}, \\ y_1, y_2 \in R_{\mathbb{F}} \end{pmatrix}, \quad \begin{array}{l} t = B_1 \cdot s_1 \\ G_{b_1, n \cdot 2^\gamma} \cdot s_1 = (I_{2^\gamma} \otimes B_2) \cdot s_2 \end{array} \\ \mathbf{w} := \begin{pmatrix} s_1 \in R_{\mathbb{F}}^{2^\gamma \cdot n \cdot \alpha_1}, \\ s_2 \in R_{\mathbb{F}}^{2^{\gamma+\beta} \cdot \alpha_2} \end{pmatrix} \quad \forall i \in [2], \tilde{s}_i(u_i) = y_i \end{array} \right\}$$

$\text{OE}^{(2)}(b_1, b_2)$ is not currently captured by $\text{GHPrincipal}(b_1, b_2)$ due to the additional evaluation claim on $\tilde{s}_1$. Hence, we describe how to adapt Greyhound to prove a modified principal Greyhound relation in Sec. 5.4.3:

$$\text{GHPrincipal}'(b_1, b_2) := \left\{ \begin{array}{l} \mathbf{x} := \left(\begin{array}{l} t \in R_{\mathbb{F}}^n, \\ a \in R_{\mathbb{F}}^{2^{\beta} \cdot \alpha_2}, \\ b \in R_{\mathbb{F}}^{2^{\gamma}}, \\ q \in R_{\mathbb{F}}^{2^{\gamma} \cdot n \cdot \alpha_1}, \\ y_1, y_2 \in R_{\mathbb{F}} \end{array} \right), \\ \mathbf{w} := \left(\begin{array}{l} s_1 \in R_{\mathbb{F}}^{2^{\gamma} \cdot n \cdot \alpha_1}, \\ s_2 \in R_{\mathbb{F}}^{2^{\gamma+\beta} \cdot \alpha_2} \end{array} \right) \end{array} \right\}, \quad \left\{ \begin{array}{l} t = B_1 \cdot s_1 \\ G_{b_1, n \cdot 2^{\gamma}} \cdot s_1 = (I_{2^{\gamma}} \otimes B_2) \cdot s_2 \\ \forall i \in [2], \|s_i\|_{\infty} < b_i \\ a^{\top} \cdot s_2 \cdot b = y_2 \\ q^{\top} \cdot s_1 = y_1 \end{array} \right.$$

The relation $\text{GHPrincipal}'(b_1, b_2)$ contains an additional claim $q^{\top} \cdot s_1 = y_1$ to capture the additional evaluation claim $\tilde{s}_1(u_1) = y_1$ in $\text{OE}^{(2)}(b_1, b_2)$. We will show in Section 5.4.3 that $\text{GHPrincipal}'(b_1, b_2)$ captures $\text{OE}^{(2)}(b_1, b_2)$.

5.4.2 Preparing to Compose with Greyhound

In this section, we describe the commitment reshaping reduction (Π_{Reshape}) and modified norm-check reduction ($\Pi_{\text{NC,fin}}^{\text{PE}}$).

Commitment reshape reduction. The Π_{Reshape} reduction has a very similar structure and set of sum-check claims to Π_{Dec} , as we similarly wish to prove commitment opening conditions and evaluation claims on the input. The main difference is that this reduction takes many TE and PE claims as input, and commits to a concatenation of all the openings via the two-layer commitment scheme in Def. 23. In particular, the input has the form

$$\begin{aligned} & \left(\mathbf{x}_{\text{TE},l} := (t_{\text{TE},l} \in R_{\mathbb{F}}^n, u_{\text{TE},l} \in R_{\mathbb{F}}^{k+1} \times R_{\mathbb{F}}^m, v_{\text{TE},l} \in R_{\mathbb{F}}), \mathbf{w}_{\text{TE},l} := (s_{i,\text{TE},l} \in R_{\mathbb{F}}^{2^{i n \alpha}})_{i \in [k]} \right)_{l \in [\omega]}, \\ & \left(\mathbf{x}_{\text{PE},l} := (t_{\text{PE},l} \in R_{\mathbb{F}}^n, u_{\text{PE},l} \in R_{\mathbb{F}}^k \times R_{\mathbb{F}}^m, v_{\text{PE},l} \in R_{\mathbb{F}}), \mathbf{w}_{\text{PE},l} := (s_{i,\text{PE},l} \in R_{\mathbb{F}}^{2^{i n \alpha}})_{i \in [k]} \right)_{l \in [\omega+1]} \\ & \in \text{TE}(1, k, b)^{\omega} \times \text{PE}(1, k, b)^{\omega+1}. \end{aligned}$$

First, let's look at the commitment opening conditions. For all $l \in [\omega]$, we wish to prove

$$\forall r \in [n], \quad t_{\text{TE},l}^{(r)} = a_r^{\top} \cdot \tilde{s}_{1,1,\text{TE},l}$$

and

$$\forall r \in [n], i \in [2, k], p \in [2^i], \quad g_b^{\top} \cdot \dot{s}_{i,p,\text{TE},l}^{((r-1)\alpha+1:r\alpha)} = a_r^{\top} \cdot \tilde{s}_{i+1,p,\text{TE},l}.$$

One can define similar statements for PE, $j \in [\omega + 1]$, and batch all these constraints together via verifier challenge $\zeta \leftarrow_{\$} \mathcal{C}$, analogous to Eq. (25) to show the well-formedness of all input commitments:

$$\begin{aligned}
& \underbrace{\sum_{l \in [\omega]} \zeta^{l-1} \cdot \left(\xi \cdot \sum_{r=1}^n \eta^{r-1} \cdot t_{\text{TE},l}^{(r)} \right)}_{v_t: \text{batched commitment constraints for TE}} + \underbrace{\sum_{l \in [\omega]} \zeta^{l-1} \cdot \left(\sum_{r=1}^n \eta^{r-1} \cdot \left(\sum_{i=1}^{k-1} \sum_{p=1}^{2^i} \xi^{2^i+p-1} \cdot g_b^{\mathbf{T}} \cdot \dot{s}_{i,p,\text{TE},l}^{((r-1)\alpha+1:r\alpha)} \right) \right)}_{v_G: G\text{-matrix constraints sum-check claim for TE}} \\
& \underbrace{\sum_{l \in [\omega+1]} \zeta^{\omega-1+l} \cdot \left(\xi \cdot \sum_{r=1}^n \eta^{r-1} \cdot t_{\text{PE},l}^{(r)} \right)}_{v_t: \text{batched commitment constraints for PE}} + \underbrace{\sum_{l \in [\omega+1]} \zeta^{\omega-1+l} \cdot \left(\sum_{r=1}^n \eta^{r-1} \cdot \left(\sum_{i=1}^{k-1} \sum_{p=1}^{2^i} \xi^{2^i+p-1} \cdot g_b^{\mathbf{T}} \cdot \dot{s}_{i,p,\text{PE},l}^{((r-1)\alpha+1:r\alpha)} \right) \right)}_{v_G: G\text{-matrix constraints sum-check claim for PE}} \\
& = \underbrace{\sum_{l \in [\omega]} \zeta^{l-1} \cdot \sum_{r=1}^n \eta^{r-1} \cdot \left(\sum_{i=0}^{k-1} \sum_{p=1}^{2^i} \xi^{2^i+p-1} \cdot a_r^{\mathbf{T}} \cdot \ddot{s}_{i+1,p,\text{TE},l} \right)}_{v_A: A\text{-matrix constraints sum-check claim for TE}} \\
& + \underbrace{\sum_{l \in [\omega+1]} \zeta^{\omega-1+l} \cdot \sum_{r=1}^n \eta^{r-1} \cdot \left(\sum_{i=0}^{k-1} \sum_{p=1}^{2^i} \xi^{2^i+p-1} \cdot a_r^{\mathbf{T}} \cdot \ddot{s}_{i+1,p,\text{PE},l} \right)}_{v_A: A\text{-matrix constraints sum-check claim for PE}} \quad (32)
\end{aligned}$$

This will incur an additional soundness error of $\frac{2\omega}{|\mathcal{C}|}$ due to a Schwartz-Zippel argument on $\zeta \leftarrow_{\$} \mathcal{C}$. To define the appropriate sum-check claims analogous to Eq. (27), (29), define publicly evaluable polynomials $q_{\text{reshape},A}$, $q_{\text{reshape},G}$:

$$q_{\text{reshape},A} := \left\| \prod_{r=1}^n p_{\zeta,2\omega+1} \otimes p_{\xi,2^k} \otimes a_r \in R_{\mathbb{F}}^{n \cdot (2\omega+1) \cdot 2^{k+1} n\alpha} \right\| \quad (33)$$

$$q_{\text{reshape},G} := p_{\zeta,2\omega+1} \otimes \left((p_{\xi,2^k} \otimes p_{\eta,n}) \otimes p_{b,\alpha} \| 0^{2^k n\alpha} \right) \in R_{\mathbb{F}}^{(2\omega+1) \cdot 2^{k+1} n\alpha}. \quad (34)$$

We also wish to prove input evaluation claims

$$\tilde{s}_{\text{TE},l}(u_{\text{TE},l}) = v_{\text{TE},l} \quad \forall l \in [\omega] \quad (35)$$

$$\tilde{s}_{\text{PE},l}(u_{\text{PE},l}) = v_{\text{PE},l} \quad \forall l \in [\omega + 1]. \quad (36)$$

We can define sum-check claims similar to Eq. (30) for each of the evaluation claims, and batch them together using BatchSC. Similarly, we can define a sum-check claim similar to Eq. (31) to check that each of $s_{\text{TE},l}$, $s_{\text{PE},l}$ in the fresh commitment begin with a zero prefix. See the Π_{Reshape} protocol (Prot. 6) for all the full sum-check claims, which are similar to those in Π_{Dec} .

Protocol 6 below assumes that $2\omega + 1$ is a power of two. If this is not the case, pad $p_{\zeta,2\omega+1}$ with zeroes to the nearest power of two and set $\mu' := \lceil \log(2\omega + 1) \rceil + k + 1$. In the notation of Def. 23, the commitment in Protocol 6 sets $2^{\gamma+\beta}\alpha_1 = 2^{\mu'}n\alpha$.

Protocol 6: Commitment reshaping reduction Π_{Reshape}
Parameters: k layers, norm bounds b, b_1, b_2 , ω number of rounds in main protocol, $m := \log(n\alpha)$, $\mu' = \lceil \log(2\omega + 1) \rceil + k + 1$, commitment keys B_1, B_2 . Input: $ \begin{aligned} & (x_{\text{TE},l} := (t_{\text{TE},l} \in R_{\mathbb{F}}^n, u_{\text{TE},l} \in R_{\mathbb{F}}^{k+1} \times R_{\mathbb{F}}^m, v_{\text{TE},l} \in R_{\mathbb{F}}), w_{\text{TE},l} := (s_{i,\text{TE},l} \in R_{\mathbb{F}}^{2^i n\alpha})_{i \in [k]})_{l \in [\omega]}, \\ & (x_{\text{PE},l} := (t_{\text{PE},l} \in R_{\mathbb{F}}^n, u_{\text{PE},l} \in R_{\mathbb{F}}^k \times R_{\mathbb{F}}^m, v_{\text{PE},l} \in R_{\mathbb{F}}), w_{\text{PE},l} := (s_{i,\text{PE},l} \in R_{\mathbb{F}}^{2^i n\alpha})_{i \in [k]})_{l \in [\omega+1]} \\ & \in \text{TE}(1, k, b)^{\omega} \times \text{PE}(1, k, b)^{\omega+1} \end{aligned} $

Output:

$$(x_{\text{reshape}} := (t_{\text{reshape}}^*, r^{(\cdot: \mu' + m)}, y_{\text{reshape}}), w_{\text{reshape}} := (s_{\text{reshape}, i})_{i \in [2]}) \in \text{PE}^{(2)}(b_1, b)$$

1. $\mathcal{P}$ produces two-layer Greyhound commitment $t_{\text{reshape}} \in R_{\mathbb{F}}^n$ to the vector $s_{\text{reshape}} \in R_{\mathbb{F}}^{2^{\mu'} n \alpha}$, which is the concatenation of the concatenated opening vectors across all the input commitment openings.

- (a) $\mathcal{P}$ creates a concatenated bottom-layer opening across all the input tree commitment openings:

$$s_{\text{reshape}} := s_{\text{TE}, 1} \parallel \dots \parallel s_{\text{TE}, \omega} \parallel s_{\text{PE}, 1} \parallel \dots \parallel s_{\text{PE}, \omega+1} \parallel 0^{\text{pad}} \in R_{\mathbb{F}}^{2^{\lceil \log(2\omega+1) \rceil + k+1} n \alpha}$$

where $\text{pad} := 2^{\lceil \log(2\omega+1) \rceil + k+1} n \alpha - (2\omega+1) 2^{k+1} n \alpha$. $\mathcal{P}$ computes

$(t_{\text{reshape}}^*, (s_{\text{reshape}, i}^*)_{i \in [2]}) := \text{GHCommit}((B_1, B_2), (b_1, b), s_{\text{reshape}})$ where $s_{\text{reshape}, 2}^* = s_{\text{reshape}}$, and sends $t_{\text{reshape}}^* \in R_{\mathbb{F}}^n$ to $\mathcal{V}$.

2. $\mathcal{P}$ and $\mathcal{V}$ invoke **BatchSC** to prove both the linear commitment opening constraints for inputs $(t_{\text{TE}, l}, (s_{i, \text{TE}, l})_{i \in [k]})_{l \in [\omega]}$, $(t_{\text{PE}, l}, (s_{i, \text{PE}, l})_{i \in [k]})_{l \in [\omega+1]}$ in **TE**, and the input evaluation claims $(v_{\text{TE}, l})_{l \in [\omega]}$, $(v_{\text{PE}, l})_{l \in [\omega+1]}$ with respect to t_{reshape}^* .

- (a) $\mathcal{P}$ and $\mathcal{V}$ construct **BatchSC** inputs to prove both the linear commitment opening constraints for expressions involving the gadget matrix.

- i. $\mathcal{V}$ samples $\eta, \xi, \gamma, \zeta \leftarrow_{\mathcal{G}} \mathcal{C}$ and sends them to $\mathcal{P}$.
- ii. $\mathcal{P}$ computes sum-check claimed sum v_G :

$$v_G := \langle q_{\text{reshape}, G}, s_{\text{reshape}} \rangle$$

where $q_{\text{reshape}, G}$ is defined using $\mathcal{V}$ challenges η, ξ, ζ as in Eq. (34), and sends $v_G \in R_{\mathbb{F}}$ to $\mathcal{V}$.

- iii. $\mathcal{P}$ and $\mathcal{V}$ construct **BatchSC** inputs for the G -matrix sum-check claim:

$$C_G : v_G \stackrel{?}{=} \sum_{\substack{X_{\text{rsh}}, X_0, X, Y, X_g \\ \in \{0, 1\}^{\mu' + m}}} \tilde{p}_{\zeta, 2\omega+1}(X_{\text{rsh}}) \cdot \tilde{e}q(X_0, 0) \cdot \tilde{p}_{\xi, 2^k}(X) \cdot \tilde{p}_{\eta, n}(Y) \\ \cdot \tilde{p}_{b, \alpha}(X_g) \cdot \tilde{s}_{\text{reshape}}(X_{\text{rsh}}, X_0, X, Y, X_g)$$

- (b) $\mathcal{P}$ and $\mathcal{V}$ construct **BatchSC** inputs to prove both the linear commitment opening constraints for expressions involving the A matrix.

- i. Using the same $\mathcal{V}$ challenges η, ξ, ζ from Step 2(a)i, $\mathcal{P}$ computes sum-check claimed sum v_A :

$$v_A := \langle q_{\text{reshape}, A}, p_{\eta, n} \otimes s_{\text{reshape}} \rangle$$

where $q_{\text{reshape}, A}$ is defined using $\mathcal{V}$ challenges ξ, ζ as in Eq. (33), and sends $v_A \in R_{\mathbb{F}}$ to $\mathcal{V}$.

- ii. $\mathcal{P}$ and $\mathcal{V}$ construct **BatchSC** inputs for the A -matrix sum-check claim:

$$C_A : v_A \stackrel{?}{=} \sum_{\substack{X_{\text{rsh}}, X, X_c, X_r \\ \in \{0, 1\}^{\mu' + m + \log n}}} \tilde{p}_{\zeta, 2\omega+1}(X_{\text{rsh}}) \cdot \tilde{A}(X_r, X_c) \cdot \tilde{p}_{\xi, 2^k}(X) \cdot \tilde{p}_{\eta, n}(X_r) \cdot \tilde{s}_{\text{reshape}}(X_{\text{rsh}}, X, X_c)$$

- (c) $\mathcal{V}$ computes the commitment component of the batched constraints $v_t \in R_{\mathbb{F}}$ and checks consistency of sum-check claimed sums v_G and v_A via Eq. (32):

$$v_t := \sum_{l \in [\omega]} \zeta^{l-1} \cdot \xi \cdot \langle p_{\eta, n}, t_{\text{TE}, l} \rangle + \sum_{l \in [\omega+1]} \zeta^{\omega-1+l} \cdot \xi \cdot \langle p_{\eta, n}, t_{\text{PE}, l} \rangle \\ v_t + v_G \stackrel{?}{=} v_A.$$

- (d) $\mathcal{P}$ and $\mathcal{V}$ construct **BatchSC** inputs to prove evaluation claims $(v_{\text{TE}, l})_{l \in [\omega]}$ as in Eq. (35):

$$\forall l \in [\omega], C_{\text{ev}, \text{TE}, l} : v_{\text{TE}, l} \stackrel{?}{=} \sum_{X_{\text{rsh}}, X \in \{0, 1\}^{\mu' + m}} \tilde{e}q(\text{bits}_{\lceil \log(2\omega+1) \rceil}(l-1), X_{\text{rsh}}) \cdot \tilde{e}q(u_{\text{TE}, l}, X) \\ \cdot \tilde{s}_{\text{reshape}}(X_{\text{rsh}}, X)$$

(e) $\mathcal{P}$ and $\mathcal{V}$ construct BatchSC inputs to prove evaluation claims $(v_{\text{PE},l})_{l \in [\omega+1]}$ as in Eq. (36):

$$\forall l \in [\omega+1], C_{\text{ev},\text{PE},l} : v_{\text{PE},l} \stackrel{?}{=} \sum_{X_{\text{rsh}}, X \in \{0,1\}^{\mu'+m}} \tilde{eq}(\text{bits}_{\lceil \log(2\omega+1) \rceil}(\omega-1+l), X_{\text{rsh}}) \cdot \tilde{eq}(1 \| u_{\text{PE},l}, X) \cdot \tilde{s}_{\text{reshape}}(X_{\text{rsh}}, X)$$

(f) $\mathcal{P}$ and $\mathcal{V}$ construct BatchSC inputs to prove that s has a 0 prefix as in Eq. (31):

$$C_{\text{pre}} : 0 \stackrel{?}{=} \sum_{X_{\text{rsh}}, X_1, X_2 \in \{0,1\}^{\mu'+m}} \tilde{p}_{\zeta, 2\omega+1}(X_{\text{rsh}}) \cdot \tilde{eq}(0^k, X_1) \cdot \tilde{p}_{\gamma, 2n\alpha}(X_2) \cdot \tilde{s}_{\text{reshape}}(X_{\text{rsh}}, X_1, X_2)$$

(g) $\mathcal{P}$ and $\mathcal{V}$ invoke BatchSC on the claims from Steps 2a-2b, 2d-2e:

$$(r, y_{\text{reshape}} \in R_{\mathbb{F}}) \leftarrow \text{BatchSC}\left(C_G, C_A, (C_{\text{ev},\text{TE},l})_{l \in [\omega]}, (C_{\text{ev},\text{PE},l})_{l \in [\omega+1]}, C_{\text{pre}}\right)$$

where evaluations of public multilinear polynomials in the sum-check claims on r are calculated by $\mathcal{V}$.

3. $\mathcal{P}$ and $\mathcal{V}$ output

$$(x_{\text{reshape}} := (t_{\text{reshape}}^*, r^{(\cdot: \mu'+m)}, y_{\text{reshape}}), w_{\text{reshape}} := (s_{\text{reshape},i}^*)_{i \in [2]}) \in \text{PE}^{(2)}(b_1, b)$$

Lemma 35 (Informal). $\Pi_{\text{Reshape}} : \text{TE}(1, k, b)^\omega \times \text{PE}(1, k, b)^{\omega+1} \rightarrow \text{PE}^{(2)}(b_1, b)$ is complete, with $O(N \cdot \omega)$ sum-check prover costs, $O(\omega \log(N))$ verifier costs, and $O(\log(N \cdot \omega))$ proof size.

Proof. We defer the proof to Appendix B.6. $\square$

Lemma 36 (Informal). $\Pi_{\text{Reshape}} : \text{TE}(1, k, b)^\omega \times \text{PE}(1, k, b)^{\omega+1} \rightarrow \text{PE}^{(2)}(b_1, b)$ is knowledge sound with negligible knowledge error.

Proof. We defer the proof to Appendix B.6. $\square$

Norm-check reduction. We describe how to modify the norm-check reduction $\Pi_{\text{NC}}^{\text{PE}} : \text{PE}(1, k, b) \rightarrow \text{TE}(1, k, b)$ to instead produce output in $\text{PE}^{(2)}(b_1, b_2)$. Denote this modified reduction by

$$\Pi_{\text{NC}, \text{fin}}^{\text{PE}} : \text{PE}^{(2)}(b_1, b_2) \rightarrow \text{OE}^{(2)}(b_1, b_2).$$

We modify the first sum-check claim in Step 1 to check norm on each layer (s_1, s_2) individually:

$$Q_N(X) := \sum_{i \in [2]} \sum_{k \in [0, d-1]} \xi^{(i-1) \cdot d + k} \cdot N_{b_i}(\widetilde{\text{cf}(s_i)_k}(X)).$$

$\mathcal{P}$ and $\mathcal{V}$ perform a zero-check sum-check on Q_N , which reduces to a claim at verifier challenges $r \in \mathbb{K}^{\gamma+\beta+\log \alpha_2}$. In Step 2, $\mathcal{P}$ similarly sends evaluation claims

$$\begin{aligned} a_1 &:= \tilde{s}_1(r^{(\cdot: \gamma+\log(n\alpha_1))}) \in R_{\mathbb{K}} \\ a_2 &:= \tilde{s}_2(r) \in R_{\mathbb{K}} \end{aligned}$$

which $\mathcal{V}$ checks via Lem. 14. Note that the substring above syntactically requires $n\alpha_1 \leq 2^\beta \alpha_2$. Lastly, in Step 3, $\mathcal{P}$ and $\mathcal{V}$ run ShiftSC on the modified claims for $i \in [1, 2]$

$$\begin{aligned} C_{\text{sc},1} : a_1 &\stackrel{?}{=} \sum_{x \in \{0,1\}^{\gamma+\log(n\alpha_1)}} \tilde{eq}(r^{(\cdot: \gamma+\log(n\alpha_1))}, x) \cdot \tilde{s}_1(x) \in R_{\mathbb{K}} \\ C_{\text{sc},2} : a_2 &\stackrel{?}{=} \sum_{x \in \{0,1\}^{\gamma+\beta+\log \alpha_2}} \tilde{eq}(r, x) \cdot \tilde{s}_2(x) \in R_{\mathbb{K}} \\ C_{\text{in}} : y &\stackrel{?}{=} \sum_{x \in \{0,1\}^{\gamma+\beta+\log \alpha_2}} \tilde{eq}(u, x) \cdot \tilde{s}_2(x) \in R_{\mathbb{F}} \end{aligned}$$

which reduces to evaluation claims $y_1, y_2 \in R_{\mathbb{F}}$ on $\tilde{s}_1, \tilde{s}_2$ on verifier challenges $r' \in R_{\mathbb{F}}^{\gamma+\beta+\log \alpha_2}$.

Corollary 2 (of Lem. 28). $\Pi_{\text{NC,fin}}^{\text{PE}} : \text{PE}^{(2)}(b_1, b_2) \rightarrow \text{OE}^{(2)}(b_1, b_2)$ is complete with $O((b_1 + b_2) \cdot L \cdot N)$ prover time, $O(L(b_1 + b_2)d \cdot \log N)$ verifier time, and $O(\log N)$ proof size.

Corollary 3 (of Lem. 29). $\Pi_{\text{NC,fin}}^{\text{PE}} : \text{PE}^{(2)}(b_1, b_2) \rightarrow \text{OE}^{(2)}(b_1, b_2)$ ($\text{OE}^{*(2)}$) is a strong interactive reduction (Def. 7) with negligible knowledge error.

5.4.3 A Modified Greyhound

We define GH' , which only modifies the initial three-move protocol described in [NS24, Section 3.1] to fit the output relation of our finishing protocol $\text{OE}^{(2)}(b_1, b_2)$. We first describe a non-interactive reduction from $\text{OE}^{(2)}(b_1, b_2)$ to $\text{GHPrincipal}'(b_1, b_2)$. In particular, the multilinear evaluation claims

$$\tilde{s}_1(u_1) = y_1, \quad \tilde{s}_2(u_2) = y_2$$

in $\text{OE}^{(2)}(b_1, b_2)$ can be represented in the form

$$q^\top \cdot s_1 = y_1, \quad a^\top \cdot s_2 \cdot b = y_2$$

in $\text{GHPrincipal}'(b_1, b_2)$. [NS24] proves this for bivariate polynomial evaluations, and we provide an argument to multilinear evaluations here. For $\tilde{s}_1$, by definition of multilinear evaluation:

$$q^\top \cdot s_1 = y_1, \quad \text{where } q^\top = \bigotimes_{j=1}^{\gamma + \log(n\alpha_1)} \begin{bmatrix} 1 - u_{1,j} \\ u_{1,j} \end{bmatrix} \in R_{\mathbb{F}}^{2^{\gamma \cdot n \cdot \alpha_1}}. \quad (37)$$

For $\tilde{s}_2$, we can similarly define a, b , which split the $\tilde{e}q(u_2, \cdot)$ polynomial into two parts.

Lemma 16. Let $\tilde{f} \in R_{\mathbb{F}}[X_1, \dots, X_n]$ be a multilinear polynomial. Then for an evaluation point $(r, u) \in R_{\mathbb{F}}^{\alpha+\beta}$ where $\alpha + \beta = n$, $\tilde{f}(r, u) = a^\top \cdot f \cdot b$ where

$$a = \bigotimes_{j \in [\alpha]} \begin{bmatrix} 1 - r_j \\ r_j \end{bmatrix} \in R_{\mathbb{F}}^{2^\alpha}, \quad b = \bigotimes_{j \in [\beta]} \begin{bmatrix} 1 - u_j \\ u_j \end{bmatrix} \in R_{\mathbb{F}}^{2^\beta}.$$

Proof. Observe that a, b are the evaluation form vectors corresponding to $\tilde{e}q(r, \cdot)$ and $\tilde{e}q(u, \cdot)$, respectively. Then

$$\begin{aligned} a^\top \cdot f \cdot b &= \langle a \otimes b, f \rangle \\ &= \sum_{x \in \{0,1\}^\alpha} \sum_{y \in \{0,1\}^\beta} \tilde{e}q(r, x) \cdot \tilde{f}(x \| y) \cdot \tilde{e}q(u, y) \\ &= \sum_{x \in \{0,1\}^\alpha} \sum_{y \in \{0,1\}^\beta} \tilde{e}q(r \| u, x \| y) \cdot \tilde{f}(x \| y) \\ &= \tilde{f}(r, u). \end{aligned}$$

□

Now the input is in the form of $\text{GHPrincipal}'(b_1, b_2)$. The linear equation which Greyhound proves by using LaBRADOR [BS23] as a sub-protocol (denoted by $\mathbf{P}, \mathbf{z}, \mathbf{h}$ in [NS24, Fig. 4]) can be easily adapted to additionally verify Eq. (37). The equation already checks linear constraints on s_1 , so the constraint $q^\top \cdot s_1 = y_1$ can be incorporated via a random linear combination with verifier challenge $\eta \leftarrow_{\mathbb{S}} \mathcal{C}$. η would need to be sent by the verifier *after* the commitments to $\hat{w}, s_1, z$ are sent by the Greyhound prover in preparation for using LaBRADOR:

$$\underbrace{\begin{bmatrix} D & 0 & 0 \\ 0 & B & 0 \\ b^\top G_{b_1, r} & 0 & 0 \\ c^\top G_{b_1, r} & 0 & -a^\top \\ 0 & c^\top \otimes G_{b_1, n} + \eta \tilde{e}_1 \cdot q^\top & -A \end{bmatrix}}_{\mathbf{P}} \underbrace{\begin{bmatrix} \hat{w} \\ s_1 \\ z \end{bmatrix}}_{\mathbf{z}} = \underbrace{\begin{bmatrix} v \\ u \\ y_2 \\ 0 \\ \eta y_1 \cdot \tilde{e}_1 \end{bmatrix}}_{\mathbf{h}}. \quad (38)$$

	Parameters							Security				Cost	
	$\log_2 q$	d	e	n	b	$\mathcal{C}$	k	MSIS	$\log_2 \mathcal{C} $	$-\log_2(\epsilon_{\text{inv}}(\mathcal{C}))$	Heuristic?	Proof size (KB)	
P1	32	64	8	16	2	$\mathcal{C}_{\text{sp}}(32, 8)$	6	122	124	–	no (Thm. 2)	508	
P2	32	64	4	16	2	$\mathcal{C}_{\text{ter}}$	6	118	101	97	yes (Lem. 7)*	387	
P3	32	64	8	32	2	$\mathcal{C}_{\text{sp}}(32, 8)$	7	240	124	–	no (Thm. 2)	335	
P4	32	64	4	16	2	$\mathcal{C}_{\text{ter}}$	7	110	101	97	yes (Lem. 7)*	224	
P5	32	128	4	16	2	$\mathcal{C}_{\text{ter}}$	7	232	203	123	yes (Lem. 7)	323 [†]	
P6	64	64	4	8	2	$\mathcal{C}_{\text{ter}}$	6	118	101	–	no (Thm. 2)	599 [†]	
P7	64	64	8	8	2	$\mathcal{C}_{\text{sp}}(32, 8)$	6	122	124	–	no (Thm. 2)	863 [†]	
P8	64	64	2	8	2	$\mathcal{C}_{\text{ter}}$	6	118	101	96	yes (Lem. 7)	468 [†]	
P9	64	128	2	8	2	$\mathcal{C}_{\text{ter}}$	6	216	203	122	yes (Lem. 7)	727 [†]	
P10	64	64	4	8	2	$\mathcal{C}_{\text{ter}}$	7	110	101	–	no (Thm. 2)	323 [†]	
P11	64	128	2	8	2	$\mathcal{C}_{\text{ter}}$	7	201	203	122	yes (Lem. 7)	407 [†]	

Figure 5: Candidate parameterizations for Maltese and their security estimates in bits. Ring parameters for $R_{\mathbb{F}} = \mathbb{F}_q[X]/\phi(X)$ are given by field size q and degree of cyclotomic polynomial d that splits into irreducible polynomials of degree e . The Ajtai tree commitment is of size n and uses decomposition base b . Recall, ring sum-checks from Section 4 operate over e -degree extension fields $\mathbb{K}$ of $\mathbb{F}_q$. Folding parameters include choice of challenge set $\mathcal{C}$ and the number of tree layers k to truncate in each folding round (e.g., fold together 2^k elements). MSIS computational security is estimated using the lattice estimator [APS15]. Challenge set size and inversion-failure probability are statistical-security components. Heuristic inversion-failure security is computed using the union bound in the corresponding lemma. Proof sizes are reported in kilobytes. * The heuristic can be validated for a specific 32-bit prime by explicitly computing the summation in Lemma 5. [†] The current implementation of Greyhound does not support parameters $\log_2 q = 64$ or $d = 128$, so we provide rough estimates for Π_{Fin} ’s proof size contribution in these cases.

where $\vec{e}_1$ is the standard basis vector $(1, 0, \dots, 0) \in R_{\mathbb{F}}^n$. GH' takes as input $\text{OE}^{(2)}(b_1, b_2)$, expresses the multilinear evaluation claims on $\tilde{s}_1, \tilde{s}_2$ in the form of $\text{GHPrincipal}(b_1, b_2)$, and runs the modified initial three-move protocol to produce verification equation in Eq. (38).

This modified Greyhound remains a weak interactive reduction. The LaBRADOR extractor is not impacted by this modification to recover $(\hat{w}, s_1, z)$ satisfying Eq. (38), after which a Schwartz-Zippel (Lem. 9) on $\eta \leftarrow_{\$} \mathcal{C}$ (which has degree 1) and binding argument is required to fix (s_1, z) . Then extraction can finish as in [NS24, Lem 3.2]. We now state the following corollary of [NS24, Thm 4.1] for GH' .

Corollary 4 (GH' is complete). *Let $\mu := \lceil \log_2(2\omega + 1) \rceil$. $\text{GH}' : \text{OE}^{(2)}(b_1, b_2) \rightarrow (0, 1)$ is complete, with $O(2^{\mu+k+1}n\alpha)$ prover time, $O(\sqrt{2^{\mu+k+1}n\alpha})$ verifier time, and $O(\text{polylog}(2^{\mu+k+1}n\alpha))$ proof size.*

Corollary 5 (GH' is a weak interactive reduction). *Let norm bound parameters $\bar{\beta}, \bar{\kappa}, \bar{\gamma}$ be as in [NS24, Thm 4.1]. Assuming $\text{MSIS}_{q,n,d,(2\gamma n\alpha_1+2\gamma\alpha_1),\bar{\beta}}$ and $\text{MSIS}_{q,n,d,2\gamma n\alpha+2\beta\alpha_2,\max(4\bar{\kappa}\bar{\beta},2\bar{\gamma})}$, $\text{GH}' : \text{OE}^{(2)}(b_1, b_2) \rightarrow (0, 1)$ is a weak interactive reduction (Def. 6) with negligible knowledge error, where ϕ is the function which projects commitments from the instance $\phi(x_1) = \phi(t, u_1, u_2, y_1, y_2) = t$.*

6 Evaluation

6.1 Parameter Selection

We provide a number of different candidate parameterizations for Maltese in Figure 5. The parameters are chosen to show different options based on desired field size q (32-bit or 64-bit), targeted security level, and whether one is comfortable relying on a heuristic estimate of the inversion-failure probability. The most efficient parameterizations are for 32-bit fields which can achieve greater than 120 bits of security without relying on a heuristic, but can be made even more efficient by accepting a lower security level and relying on a heuristic.

The main benefit of heuristic inversion is that it enables using the minimum 128-bit extension field for the sum-checks. Without the heuristic, the splitting of the cyclotomic polynomial into low degree polynomials for a 128-bit extension field does not yield a large enough challenge set under the deterministic inversion bounds of Theorem 2. Using the ternary coefficient challenge set $\mathcal{C}_{\text{ter}}$ with heuristic inversion from Lemma 7 enables some parameterizations with large enough challenge sets. Note that for a 32-bit prime, this heuristic can be validated by computing the summation in Lemma 5. Even so, for degree 64 cyclotomic polynomials, the ternary challenge set is not large enough to achieve greater than 120 bits of security.

Note that Figure 5 provides bit estimates for components that contribute to the overall concrete security of the scheme: the estimated hardness of the MSIS problem, the challenge set size, and the inversion-failure probability of the

			Accounting	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11
main cycle	NCPE	sum-checks	$SC(2b) + \text{ShiftSC}(2, 1) + 1 R_{\mathbb{K}}$	10.0	5.1	10.0	5.1	6.3	10.5	20.5	5.5	6.9	10.5	6.9
		# commitments	0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Fold	sum-checks	$\text{BatchSC}(2, -) + \text{BatchSC}(2, 2) + 4 R_{\mathbb{F}}$	5.4	3.5	5.4	3.5	4.9	7.0	10.9	5.0	8.1	7.0	8.1
		# commitments	2	8.2	8.2	16.4	8.2	16.4	8.2	8.2	8.2	16.4	8.2	16.4
	Decomp.	sum-checks	$\text{BatchSC}(2, 1) + 2 R_{\mathbb{F}}$	3.8	2.3	3.8	2.2	3.0	4.6	7.7	3.1	4.6	4.5	4.5
		# commitments	1	4.1	4.1	8.2	4.1	8.2	4.1	4.1	4.1	8.2	4.1	8.2
single cycle total				31.5	23.2	43.8	23.1	38.8	34.4	51.4	25.9	44.2	34.3	44.1
$\omega = \# \text{ cycles}$				14	14	6	7	6	15	15	15	14	7	7
finish	$SC(2b) + \text{ShiftSC}(2, 1) + 2 R_{\mathbb{K}}$ + $\text{BatchSC}(2, 1) + 2 R_{\mathbb{F}} + 1 \text{ commitment}$ + $\text{Greyhound}(2^{\mu+k+1}nad)$			~ 68	~ 63	~ 72	~ 62	$\sim 90^\dagger$	$\sim 83^\dagger$	$\sim 92^\dagger$	$\sim 79^\dagger$	$\sim 108^\dagger$	$\sim 83^\dagger$	$\sim 98^\dagger$
	where $\mu := \lceil \log_2(2\omega + 1) \rceil$													
total				508	387	335	224	323	599	863	468	727	323	407

Figure 6: Breakdown of proof size for Maltese with respect to concrete accounting and computed sizes (in KB) for the parameterizations in Fig. 5 for a polynomial of size 2^{30} (i.e., encoding of $2^{30} \mathbb{F}$ elements or $2^{30}/d R_{\mathbb{F}}$ elements where d is the degree of the cyclotomic polynomial). The accounting notation includes costs for the following sum-check subprotocols: $SC(\text{deg.})$, $\text{BatchSC}(\text{max deg.}, \# \text{ MLE claims sent})$, and $\text{ShiftSC}(\text{max deg.}, \# \text{ MLE claims sent})$. $\text{Greyhound}(N)$ denotes the proof size of [NS24] with input size $N \mathbb{F}$ elements. The proof size estimates for $\log_2 q = 32$ are taken from [NS24, Table 1]. † The current implementation of Greyhound does not support parameters $\log_2 q = 64$ or $d = 128$, so we provide rough estimates for Π_{Fin} 's proof size contribution in these cases.

approximate challenge set. We do not report a full security bound for the instantiated scheme of composed reductions. However, from our concrete analysis of the reductions, the concrete knowledge soundness error scales additively with these terms (with small multiplicative factors); see the knowledge-soundness bounds for $\Pi_{\text{NC}}^{\text{PE}}$, Π_{Fold} , Π_{Dec} , and Π_{Reshape} in Equations (48), (52), (67) and (68). Thus, the minimum of the terms given in Figure 5 provides a useful rough estimate of the overall security of the composed protocol (with some small loss).⁶

6.2 Proof Size Accounting

Maltese is made up of a repeating main cycle protocol that shrinks the instance size followed by a finishing protocol. Every run of the main cycle incurs the same proof size, then the cost of the finishing protocol is dependent on the number of iterations of the main cycle. Within the main cycle, the costs for the norm-check, folding, and decomposition reductions can themselves each be broken down into (1) sum-check costs (e.g., claims and sum-check messages), and (2) new commitments. Figure 6 summarizes these costs generically and concretely with respect to specific parameterizations provided in Figure 5.

Acknowledgements

Katarina Cheng was supported by the National Science Foundation Graduate Research Fellowship grant DGE-2140004. While Wilson Nguyen was at New York University, his work was partially funded by Google, DTCC, and Chaincode.

AI Disclosure. Portions of this manuscript were edited with the assistance of an AI, which was used to review protocol correctness and proofs, and improve grammar, wording, and formatting. All technical content, including definitions, theorems, and proofs, were produced and verified by the authors, who take full responsibility for the correctness and originality of the work.

References

- [AC20] Thomas Attema and Ronald Cramer. Compressed Σ -protocol theory and practical application to plug & play secure algorithmics. In *CRYPTO (3)*, volume 12172 of *Lecture Notes in Computer Science*, pages 513–543. Springer, 2020.
- [ACFY24] Gal Armon, Alessandro Chiesa, Giacomo Fenzi, and Eylon Yogev. STIR: reed-solomon proximity testing with fewer queries. In *CRYPTO (10)*, volume 14929 of *Lecture Notes in Computer Science*, pages 380–413. Springer, 2024.

⁶An exception occurs in the knowledge-soundness error of the norm-decomposition protocol Π_{Dec} (Protocol 5), whose bound in Equation (67) contains a Schwartz-Zippel batching term that scales with the size of the tree being decomposed. We do this for ease of exposition, but it is not necessary. One could instead perform a Schwartz-Zippel batching check using a tensor product of independent challenges, which would reduce the contribution to the knowledge soundness error from linear in the tree size to logarithmic in the tree size.

- [ACFY25] Gal Arnon, Alessandro Chiesa, Giacomo Fenzi, and Eylon Yogev. WHIR: reed-solomon proximity testing with super-fast verification. In *EUROCRYPT (4)*, volume 15604 of *Lecture Notes in Computer Science*, pages 214–243. Springer, 2025.
- [Ach03] Dimitris Achlioptas. Database-friendly random projections: Johnson-lindenstrauss with binary coins. *J. Comput. Syst. Sci.*, 66(4):671–687, 2003.
- [ACK21] Thomas Attema, Ronald Cramer, and Lisa Kohl. A compressed Σ -protocol theory for lattices. In *CRYPTO (2)*, volume 12826 of *Lecture Notes in Computer Science*, pages 549–579. Springer, 2021.
- [AFLN24] Martin R. Albrecht, Giacomo Fenzi, Oleksandra Lapiha, and Ngoc Khanh Nguyen. SLAP: succinct lattice-based polynomial commitments from standard assumptions. In *EUROCRYPT (6)*, volume 14656 of *Lecture Notes in Computer Science*, pages 90–119. Springer, 2024.
- [Ajt96] Miklós Ajtai. Generating hard instances of lattice problems (extended abstract). In *STOC*, pages 99–108. ACM, 1996.
- [AL21] Martin R. Albrecht and Russell W. F. Lai. Subtractive sets over cyclotomic rings - limits of schnorr-like arguments over lattices. In *CRYPTO (2)*, volume 12826 of *Lecture Notes in Computer Science*, pages 519–548. Springer, 2021.
- [ALS20] Thomas Attema, Vadim Lyubashevsky, and Gregor Seiler. Practical product proofs for lattice commitments. In *CRYPTO (2)*, volume 12171 of *Lecture Notes in Computer Science*, pages 470–499. Springer, 2020.
- [APS15] Martin R. Albrecht, Rachel Player, and Sam Scott. On the concrete hardness of learning with errors. *J. Math. Cryptol.*, 9(3):169–203, 2015.
- [AST24] Arasu Arun, Srinath T. V. Setty, and Justin Thaler. Jolt: Snarks for virtual machines via lookups. In *EUROCRYPT (6)*, volume 14656 of *Lecture Notes in Computer Science*, pages 3–33. Springer, 2024.
- [BBB⁺18] Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Gregory Maxwell. Bulletproofs: Short proofs for confidential transactions and more. In *IEEE Symposium on Security and Privacy*, pages 315–334. IEEE Computer Society, 2018.
- [BBHR18] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. Fast reed-solomon interactive oracle proofs of proximity. In *ICALP*, volume 107 of *LIPIcs*, pages 14:1–14:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.
- [BBHR19] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. Scalable zero knowledge with no trusted setup. In *CRYPTO (3)*, volume 11694 of *Lecture Notes in Computer Science*, pages 701–732. Springer, 2019.
- [BBL⁺26] Madalina Bolboceanu, Jonathan Bootle, Vadim Lyubashevsky, Antonio Merino-Gallardo, and Gregor Seiler. Orthus: Practical sublinear batch-verification of lattice relations from standard assumptions. *IACR Cryptol. ePrint Arch.*, 2026:398, 2026.
- [BC25a] Dan Boneh and Binyi Chen. Latticefold: A lattice-based folding scheme and its applications to succinct proof systems. In *ASIACRYPT (3)*, volume 16247 of *Lecture Notes in Computer Science*, pages 330–362. Springer, 2025.
- [BC25b] Dan Boneh and Binyi Chen. Latticefold+: Faster, simpler, shorter lattice-based folding for succinct proof systems. In *CRYPTO (7)*, volume 16006 of *Lecture Notes in Computer Science*, pages 327–361. Springer, 2025.
- [BCC⁺16] Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Jens Groth, and Christophe Petit. Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting. In *EUROCRYPT (2)*, volume 9666 of *Lecture Notes in Computer Science*, pages 327–357. Springer, 2016.
- [BCF⁺25] Martijn Brehm, Binyi Chen, Ben Fisch, Nicolas Resch, Ron D. Rothblum, and Hadas Zeilberger. Blaze: Fast snarks from interleaved RAA codes. In *EUROCRYPT (4)*, volume 15604 of *Lecture Notes in Computer Science*, pages 123–152. Springer, 2025.
- [BCG20] Jonathan Bootle, Alessandro Chiesa, and Jens Groth. Linear-time arguments with sublinear verification from tensor codes. In *TCC (2)*, volume 12551 of *Lecture Notes in Computer Science*, pages 19–46. Springer, 2020.
- [BCPS18] Anurag Bishnoi, Pete L. Clark, Aditya Potukuchi, and John R. Schmitt. On zeros of a polynomial in a finite grid. *Comb. Probab. Comput.*, 27(3):310–333, 2018.
- [BCS21] Jonathan Bootle, Alessandro Chiesa, and Katerina Sotiraki. Sumcheck arguments and their applications. In *CRYPTO (1)*, volume 12825 of *Lecture Notes in Computer Science*, pages 742–773. Springer, 2021.
- [BCS23] Jonathan Bootle, Alessandro Chiesa, and Katerina Sotiraki. Lattice-based succinct arguments for NP with polylogarithmic-time verification. In *CRYPTO (2)*, volume 14082 of *Lecture Notes in Computer Science*, pages 227–251. Springer, 2023.
- [BDF⁺25] Dan Boneh, Trisha Datta, Rex Fernando, Kamilla Nazirkhanova, and Alin Tomescu. Dekartproof: Efficient vector range proofs and their applications. *IACR Cryptol. ePrint Arch.*, page 1159, 2025.
- [BFK⁺24] Alexander R. Block, Zhiyong Fang, Jonathan Katz, Justin Thaler, Hendrik Waldner, and Yupeng Zhang. Field-agnostic snarks from expand-accumulate codes. In *CRYPTO (10)*, volume 14929 of *Lecture Notes in Computer Science*, pages 276–307. Springer, 2024.
- [BFRW25] Benedikt Bünz, Giacomo Fenzi, Ron D. Rothblum, and William Wang. Tensorswitch: Nearly optimal polynomial commitments from tensor codes. *IACR Cryptol. ePrint Arch.*, page 2065, 2025.
- [BFS20] Benedikt Bünz, Ben Fisch, and Alan Szepeieniec. Transparent snarks from DARK compilers. In *EUROCRYPT (1)*, volume 12105 of *Lecture Notes in Computer Science*, pages 677–706. Springer, 2020.
- [BGKS20] Eli Ben-Sasson, Lior Goldberg, Swastik Kopparty, and Shubhangi Saraf. DEEP-FRI: sampling outside the box improves soundness. In *ITCS*, volume 151 of *LIPIcs*, pages 5:1–5:32. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.
- [BL17] Carsten Baum and Vadim Lyubashevsky. Simple amortized proofs of shortness for linear relations over polynomial rings. *IACR Cryptol. ePrint Arch.*, page 759, 2017.
- [BL26] Katharina Boudgoust and Oleksandra Lapiha. Leftover hash lemma(s) over cyclotomic rings. In *EUROCRYPT (4)*, volume 16544 of *Lecture Notes in Computer Science*, pages 62–91. Springer, 2026.

- [BLNS20] Jonathan Bootle, Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Gregor Seiler. A non-pcp approach to succinct quantum-safe zero-knowledge. In *CRYPTO (2)*, volume 12171 of *Lecture Notes in Computer Science*, pages 441–469. Springer, 2020.
- [BMM⁺21] Benedikt Bünz, Mary Maller, Pratyush Mishra, Nirvan Tyagi, and Psi Vesely. Proofs for inner pairing products and applications. In *ASIACRYPT (3)*, volume 13092 of *Lecture Notes in Computer Science*, pages 65–97. Springer, 2021.
- [BMNW25] Benedikt Bünz, Pratyush Mishra, Wilson Nguyen, and William Wang. Arc: Accumulation for reed-solomon codes. In *CRYPTO (7)*, volume 16006 of *Lecture Notes in Computer Science*, pages 128–160. Springer, 2025.
- [BS23] Ward Beullens and Gregor Seiler. Labrador: Compact proofs for R1CS from module-sis. In *CRYPTO (5)*, volume 14085 of *Lecture Notes in Computer Science*, pages 518–548. Springer, 2023.
- [CCKP19] Shuo Chen, Jung Hee Cheon, Dongwoo Kim, and Daejun Park. Verifiable computing for approximate computation. *IACR Cryptol. ePrint Arch.*, page 762, 2019.
- [CFF⁺21] Matteo Campanelli, Antonio Faonio, Dario Fiore, Anaïs Querol, and Hadrián Rodríguez. Lunar: A toolbox for more efficient universal and updatable zkSNARKs and commit-and-prove extensions. In *ASIACRYPT (3)*, volume 13092 of *Lecture Notes in Computer Science*, pages 3–33. Springer, 2021.
- [CGKY25] Alessandro Chiesa, Ziyi Guan, Christian Knabenhans, and Zihan Yu. On the fiat-shamir security of succinct arguments from functional commitments. *IACR Cryptol. ePrint Arch.*, page 902, 2025.
- [CHM⁺20] Alessandro Chiesa, Yuncong Hu, Mary Maller, Pratyush Mishra, Psi Vesely, and Nicholas P. Ward. Marlin: Preprocessing zkSNARKs with universal and updatable SRS. In *EUROCRYPT (1)*, volume 12105 of *Lecture Notes in Computer Science*, pages 738–768. Springer, 2020.
- [CLM23] Valerio Cini, Russell W. F. Lai, and Giulio Malavolta. Lattice-based succinct arguments from vanishing polynomials - (extended abstract). In *CRYPTO (2)*, volume 14082 of *Lecture Notes in Computer Science*, pages 72–105. Springer, 2023.
- [CMNW24] Valerio Cini, Giulio Malavolta, Ngoc Khanh Nguyen, and Hoeteck Wee. Polynomial commitments from lattices: Post-quantum security, fast verification and transparent setup. In *CRYPTO (10)*, volume 14929 of *Lecture Notes in Computer Science*, pages 207–242. Springer, 2024.
- [DBK⁺26] Quang Dao, Omid Bodaghi, Amirhossein Khajepour, Giuseppe Vitto, Mohammadtaghi Badakhshan, Markos Georgiades, Fengrun Liu, Jiapeng Zhang, and Justin Thaler. Akita: A high-performance lattice-based polynomial commitment scheme. <https://assets.layrzero.network/pdf/akita.pdf>, 2026.
- [FMN24] Giacomo Fenzi, Hossein Moghaddas, and Ngoc Khanh Nguyen. Lattice-based polynomial commitments: Towards asymptotic and concrete efficiency. *J. Cryptol.*, 37(3):31, 2024.
- [GHL22] Craig Gentry, Shai Halevi, and Vadim Lyubashevsky. Practical non-interactive publicly verifiable secret sharing with thousands of parties. In *EUROCRYPT (1)*, volume 13275 of *Lecture Notes in Computer Science*, pages 458–487. Springer, 2022.
- [GLLO26] Albert Garreta, Helger Lipmaa, Urmas Luhaäär, and Michal Osadnik. Cyclo: Lightweight lattice-based folding via partial range checks. In *EUROCRYPT (7)*, volume 16547 of *Lecture Notes in Computer Science*, pages 33–62. Springer, 2026.
- [GLS⁺23] Alexander Golovnev, Jonathan Lee, Srinath T. V. Setty, Justin Thaler, and Riad S. Wahby. Brakedown: Linear-time and field-agnostic snarks for R1CS. In *CRYPTO (2)*, volume 14082 of *Lecture Notes in Computer Science*, pages 193–226. Springer, 2023.
- [HLSS26] Intak Hwang, Hyeonbum Lee, Jinyeong Seo, and Yongsoo Song. Jindo: Practical lattice-based polynomial commitment for zero-knowledge arguments. *IACR Cryptol. ePrint Arch.*, 2026:44, 2026.
- [HSS24] Intak Hwang, Jinyeong Seo, and Yongsoo Song. Concretely efficient lattice-based polynomial commitment from standard assumptions. In *CRYPTO (10)*, volume 14929 of *Lecture Notes in Computer Science*, pages 414–448. Springer, 2024.
- [JL25] Kalle Jyrkinen and Russell W. F. Lai. Vanishing short integer solution, revisited - reductions, trapdoors, homomorphic signatures for low-degree polynomials. In *PKC (2)*, *Lecture Notes in Computer Science*, pages 273–300. Springer, 2025.
- [KK26] Anders Kallesøe and Hamidreza Khoshakhlagh. Grand danois: Succinct multilinear polynomial commitments over lattices. *IACR Cryptol. ePrint Arch.*, 2026:1196, 2026.
- [KLN⁺26] Michael Kloos, Russell W. F. Lai, Ngoc Khanh Nguyen, Michal Osadnik, and Lorenzo Tucci. RoKoko: Lattice-based succinct arguments, a committed refinement. *IACR Cryptol. ePrint Arch.*, page 575, 2026.
- [KLNO24] Michael Klooß, Russell W. F. Lai, Ngoc Khanh Nguyen, and Michal Osadnik. Rok, paper, scissors toolkit for lattice-based succinct arguments - (extended abstract). In *ASIACRYPT (5)*, volume 15488 of *Lecture Notes in Computer Science*, pages 203–235. Springer, 2024.
- [KLNO25] Michael Klooß, Russell W. F. Lai, Ngoc Khanh Nguyen, and Michal Osadnik. Rok and roll - verifier-efficient random projection for $o(\lambda)$ -size lattice arguments - (extended abstract). In *ASIACRYPT (3)*, volume 16247 of *Lecture Notes in Computer Science*, pages 297–329. Springer, 2025.
- [KLOT25] Shuto Kuriyama, Russell W. F. Lai, Michal Osadnik, and Lorenzo Tucci. SALSAA - sumcheck-aided lattice-based succinct arguments and applications. *IACR Cryptol. ePrint Arch.*, page 2124, 2025.
- [Kot24] Abhiram Kothapalli. *A Theory of Composition for Proofs of Knowledge*. PhD thesis, Carnegie Mellon University, USA, 2024.
- [KP23] Abhiram Kothapalli and Bryan Parno. Algebraic reductions of knowledge. In *CRYPTO (4)*, volume 14084 of *Lecture Notes in Computer Science*, pages 669–701. Springer, 2023.
- [KPV22] Assimakis A. Kattis, Konstantin Panarin, and Alexander Vlasov. Redshift: Transparent snarks from list polynomial commitments. In *CCS*, pages 1725–1737. ACM, 2022.

- [KZG10] Aniket Kate, Gregory M. Zaverucha, and Ian Goldberg. Constant-size commitments to polynomials and their applications. In *ASIACRYPT*, volume 6477 of *Lecture Notes in Computer Science*, pages 177–194. Springer, 2010.
- [Lee21] Jonathan Lee. Dory: Efficient, transparent arguments for generalised inner products and polynomial commitments. In *TCC (2)*, volume 13043 of *Lecture Notes in Computer Science*, pages 1–34. Springer, 2021.
- [LFKN92] Carsten Lund, Lance Fortnow, Howard J. Karloff, and Noam Nisan. Algebraic methods for interactive proof systems. *J. ACM*, 39(4):859–868, 1992.
- [LLNW16] Benoît Libert, San Ling, Khoa Nguyen, and Huaxiong Wang. Zero-knowledge arguments for lattice-based accumulators: Logarithmic-size ring signatures and group signatures without trapdoors. In *EUROCRYPT (2)*, *Lecture Notes in Computer Science*, pages 1–31. Springer, 2016.
- [LM06] Vadim Lyubashevsky and Daniele Micciancio. Generalized compact knapsacks are collision resistant. In *ICALP (2)*, *Lecture Notes in Computer Science*, pages 144–155. Springer, 2006.
- [LNP22] Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Maxime Plançon. Lattice-based zero-knowledge proofs and applications: Shorter, simpler, and more general. In *CRYPTO (2)*, volume 13508 of *Lecture Notes in Computer Science*, pages 71–101. Springer, 2022.
- [LNS21] Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Gregor Seiler. Shorter lattice-based zero-knowledge proofs via one-time commitments. In *Public Key Cryptography (1)*, volume 12710 of *Lecture Notes in Computer Science*, pages 215–241. Springer, 2021.
- [LPS25] Helger Lipmaa, Roberto Parisella, and Janno Siim. On knowledge-soundness of plonk in ROM from falsifiable assumptions. In *CRYPTO (7)*, volume 16006 of *Lecture Notes in Computer Science*, pages 362–395. Springer, 2025.
- [LS15] Adeline Langlois and Damien Stehlé. Worst-case to average-case reductions for module lattices. *Des. Codes Cryptogr.*, 75(3):565–599, 2015.
- [LS18] Vadim Lyubashevsky and Gregor Seiler. Short, invertible elements in partially splitting cyclotomic rings and applications to lattice-based zero-knowledge proofs. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part I*, *Lecture Notes in Computer Science*, pages 204–224. Springer, 2018.
- [NOZ26] Ngoc Khanh Nguyen, George O’Rourke, and Jiapeng Zhang. Hachi: Efficient lattice-based multilinear polynomial commitments over extension fields. *IACR Cryptol. ePrint Arch.*, page 156, 2026.
- [NS24] Ngoc Khanh Nguyen and Gregor Seiler. Greyhound: Fast polynomial commitments from lattices. In *CRYPTO (10)*, volume 14929 of *Lecture Notes in Computer Science*, pages 243–275. Springer, 2024.
- [NS25] Wilson D. Nguyen and Srinath T. V. Setty. Neo: Lattice-based folding scheme for CCS over small fields and pay-per-bit commitments. *IACR Cryptol. ePrint Arch.*, page 294, 2025.
- [PR06] Chris Peikert and Alon Rosen. Efficient collision-resistant hashing from worst-case assumptions on cyclic lattices. In *TCC*, *Lecture Notes in Computer Science*, pages 145–166. Springer, 2006.
- [PSTY13] Charalampos Papamanthou, Elaine Shi, Roberto Tamassia, and Ke Yi. Streaming authenticated data structures. In *EUROCRYPT*, *Lecture Notes in Computer Science*, pages 353–370. Springer, 2013.
- [Sch80] Jacob T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *J. ACM*, 27(4):701–717, 1980.
- [ST25] Srinath T. V. Setty and Justin Thaler. Twist and shout: Faster memory checking arguments via one-hot addressing and increments. *IACR Cryptol. ePrint Arch.*, page 105, 2025.
- [XZS22] Tiancheng Xie, Yupeng Zhang, and Dawn Song. Orion: Zero knowledge proof with linear prover time. In *CRYPTO (4)*, volume 13510 of *Lecture Notes in Computer Science*, pages 299–328. Springer, 2022.
- [ZCF24] Hadas Zeilberger, Binyi Chen, and Ben Fisch. Basefold: Efficient field-agnostic polynomial commitment schemes from foldable codes. In *CRYPTO (10)*, volume 14929 of *Lecture Notes in Computer Science*, pages 138–169. Springer, 2024.

A Applying the PCS to different polynomials

As described in the main body, Maltese takes as input $\text{PE}(1, \ell, b)$, which is an evaluation claim on low norm $\tilde{s}_\ell$ relative to commitment t . This is effectively a polynomial commitment scheme which commits to a low norm $R_{\mathbb{F}}$ vector s_ℓ . In this section, we describe how to upgrade the *first* run of $\Pi_{\text{NC}}^{\text{PE}}$ to support evaluation claims on potentially high norm polynomials in both $R_{\mathbb{F}}[X]$ and $\mathbb{F}[X]$. The rest of the scheme remains the same. Denote the modified polynomial commitment schemes by $\text{Maltese}_{R_{\mathbb{F}}}$ and $\text{Maltese}_{\mathbb{F}}$, respectively.

A.1 Potentially high norm $R_{\mathbb{F}}$ -polynomials

We first describe a modified input relation into $\text{Maltese}_{R_{\mathbb{F}}}$, where the evaluation claim is on a potentially high norm polynomial in $R_{\mathbb{F}}[X]$:

$$\text{PE}_{R_{\mathbb{F}}}(1, \ell, b) := \left\{ \begin{array}{l} \mathbf{x} := \begin{pmatrix} t \in R_{\mathbb{F}}^n, \\ u \in R_{\mathbb{F}}^{\ell + \log n}, \\ v \in R_{\mathbb{F}} \end{pmatrix}, \quad \text{Open}(A, b, t, (s_i)_{i \in [\ell]}) \\ \mathbf{w} := (s_i \in R_{\mathbb{F}}^{2^i n \alpha})_{i \in [\ell]} \end{array} : \begin{array}{l} f := G_{2^{\ell} n} \cdot s_\ell \in R_{\mathbb{F}}^{2^{\ell} n} \\ \tilde{f}(u) = v \end{array} \right\}$$

This scenario requires only a slight change to the *first* run of $\Pi_{\text{NC}}^{\text{PE}}$ in Maltese. We update the claim C_{in} in Step 3b of $\Pi_{\text{NC}}^{\text{PE}}$ to include a claim on the multilinear extension of $f := G_{2^{\ell} n} \cdot s_\ell$ rather than $\tilde{s}_\ell$ by making use of the $\tilde{p}_{b, \alpha}$ polynomial (see Eq. (13)):

$$\begin{aligned} v &= \sum_{X \in \{0,1\}^{\ell+m}} \tilde{e}q(u, X_{[:\ell+\log n+1]}) \cdot \tilde{p}_{b, \alpha}(X_{[\ell+\log n+2:]}) \cdot \tilde{s}_\ell(X) \\ &= \sum_{X \in \{0,1\}^{\ell+1+m}} \tilde{e}q(X_1, 1) \cdot \tilde{e}q(u, X_{[2:\ell+\log n+1]}) \cdot \tilde{p}_{b, \alpha}(X_{[\ell+\log n+2:]}) \cdot \tilde{s}(X). \end{aligned} \quad (39)$$

(written in terms of $\tilde{s}$)

Protocol 7: $\Pi_{\text{NC}}^{\text{PE}, R_{\mathbb{F}}}$: Entry reduction for $R_{\mathbb{F}}$ -polynomials

Parameters: ℓ tree layers per commitment, opening norm bound b

Input: $\left(\mathbf{x} := \begin{pmatrix} t \in R_{\mathbb{F}}^n, \\ u \in R_{\mathbb{F}}^{\ell + \log n}, \\ v \in R_{\mathbb{F}} \end{pmatrix}; \mathbf{w} := (s_i \in R_{\mathbb{F}}^{2^i n \alpha})_{i \in [\ell]} \right) \in \text{PE}_{R_{\mathbb{F}}}(1, \ell, b)$

Output: $\left(\mathbf{x} := \begin{pmatrix} t \in R_{\mathbb{F}}^n, \\ r' \in R_{\mathbb{F}}^{\ell+1+m}, \\ y \in R_{\mathbb{F}} \end{pmatrix}; \mathbf{w} := (s_i \in R_{\mathbb{F}}^{2^i n \alpha})_{i \in [\ell]} \right) \in \text{TE}(1, \ell, b)$

1. $\mathcal{P}$ and $\mathcal{V}$ run SC over $\mathbb{K}$ to check the norm of coefficient polynomials $\text{cf}(s)_k$, reducing to a polynomial evaluation claim on $r_N \in \mathbb{K}^{\ell+1+m}$, $y_N \in \mathbb{K}$.

- (a) $\mathcal{V}$ samples $\xi \leftarrow_{\mathbb{S}} \mathbb{K}$, $r \leftarrow_{\mathbb{S}} \mathbb{K}^{\ell+1+m}$ and sends them to $\mathcal{P}$.
- (b) $\mathcal{P}$ and $\mathcal{V}$ define the polynomial $Q_N \in \mathbb{K}[X_{[1:\ell+1+m]}]$, which batches infinity norm-check polynomials for $\text{cf}(s)_k$ (see Eq. (8)) across $k \in [0, d-1]$ via challenge ξ :

$$Q_N(X) := \sum_{k \in [0, d-1]} \xi^k \cdot N_b(\widetilde{\text{cf}(s)_k}(X)) \in \mathbb{K}[X_{[1:\ell+1+m]}]$$

where $N_b \in \mathbb{F}[X]$ is defined as in Eq. (7) and $\widetilde{\text{cf}(s)_k} \in \mathbb{F}[X_{[1:\ell+1+m]}]$ as in Def. 3.

- (c) $\mathcal{P}$ and $\mathcal{V}$ invoke Lem. 4 on Q_N with challenge r to produce a zero-check claim, and then run the sum-check protocol (see Def. 9):

$$(r_N \in \mathbb{K}^{\ell+1+m}, y_N \in \mathbb{K}) \leftarrow \text{SC}(0, \ell+1+m, \tilde{e}q(X, r) \cdot Q_N(X))$$

2. $\mathcal{P}$ and $\mathcal{V}$ invoke Lem. 14 to transform the polynomial evaluation claim on $r_N \in \mathbb{K}^{\ell+1+m}$, $y_N \in \mathbb{K}$ into evaluation claims $\tilde{s}(r_N) = a \in R_{\mathbb{K}}$.

- (a) $\mathcal{P}$ computes and sends $a := \tilde{s}(r_N) \in R_{\mathbb{K}}$, which $\mathcal{V}$ interprets via the coefficient map cf (Def. 3) as $(a_0, \dots, a_{d-1}) := \text{cf}(a) \in \mathbb{K}^{1 \times d}$.
- (b) $\mathcal{V}$ checks consistency of claims a_k with y_N (see Lem. 14 with $s \in R_{\mathbb{F}}^{2^{\ell+1}n\alpha}$, $r_N \in \mathbb{K}$):

$$y_N \stackrel{?}{=} \tilde{eq}(r_N, r) \cdot \sum_{k \in [0, d-1]} \xi^k \cdot N_b(a_k)$$

3. $\mathcal{P}$ and $\mathcal{V}$ run ShiftSC (Prot. 2) to reduce both the evaluation claims from Step 2 (see Eq. (9)) and the input evaluation claims (see Eq. (10)) to evaluation claims in $R_{\mathbb{F}}$ over the same input point r' .

- (a) For Step 2 evaluation claims $\tilde{s}(r_N) = a \in R_{\mathbb{K}}$, $\mathcal{P}$ and $\mathcal{V}$ define ShiftSC inputs (see Eq. (9)):

$$C_{\text{sc}} : a \stackrel{?}{=} \sum_{X \in \{0,1\}^{\ell+1+m}} \tilde{eq}(r_N, X) \cdot \tilde{s}(X) \in R_{\mathbb{K}}$$

- (b) For input evaluation claims $\tilde{s}_\ell(u) = v \in R_{\mathbb{F}}$, $\mathcal{P}$ and $\mathcal{V}$ define ShiftSC inputs (see Eq. (39)):

$$C_{\text{in}} : v \stackrel{?}{=} \sum_{X \in \{0,1\}^{\ell+1+m}} \tilde{eq}(X_1, 1) \cdot \tilde{eq}(u, X_{[2:\ell+\log n+1]}) \cdot \tilde{p}_{b,\alpha}(X_{[\ell+\log n+2:]}) \cdot \tilde{s}(X) \in R_{\mathbb{F}}$$

- (c) $\mathcal{P}$ and $\mathcal{V}$ invoke ShiftSC on the 2 claims from the previous two steps:

$$(r', y \in R_{\mathbb{F}}) \leftarrow \text{ShiftSC}\left((C_j)_{j \in \{\text{sc}, \text{in}\}}\right)$$

Note that since $\tilde{s}$ is over $R_{\mathbb{F}}$, output y is also in $R_{\mathbb{F}} \subseteq R_{\mathbb{K}}$ (though ShiftSC outputs $R_{\mathbb{K}}$ values in general). Also, the evaluation claims corresponding to the publicly evaluable multilinear on r' can be computed by $\mathcal{V}$.

4. $\mathcal{P}$ and $\mathcal{V}$ output $x := (t, r', y)$, $w := (s_i)_{i \in [\ell]}$.

Lemma 17. $\Pi_{\text{NC}}^{\text{PE}, R_{\mathbb{F}}} : \text{PE}_{R_{\mathbb{F}}}(L, \ell, b) \rightarrow \text{TE}(L, \ell, b)$ ($\text{TE}^*(L, \ell)$) is complete with prover work consisting of an $O(b \cdot 2^{(\ell+1+m)})$ sum-check prover, $O(bd \cdot (\ell + 1 + m))$ verifier time, and proof size consisting of 1 degree $2b$ sum-check, 1 ShiftSC of maximum degree 2 outputting one evaluation claim, and one additional $R_{\mathbb{K}}$ claim.

Proof. This follows by the proof Lem. 28, slightly modified for the correctness of the statement: C_{in} holding if and only if $\tilde{f}(u) = v$. $\square$

Lemma 18. $\Pi_{\text{NC}}^{\text{PE}, R_{\mathbb{F}}} : \text{PE}_{R_{\mathbb{F}}}(L, \ell, b) \rightarrow \text{TE}(L, \ell, b)$ ($\text{TE}^*(L, \ell)$) is a strong interactive reduction (Def. 7) with negligible knowledge error, where ϕ is the function which projects commitments from the instance.

Proof. This follows by the proof Lem. 29, slightly modified for the correctness of the statement: C_{in} holding if and only if $\tilde{f}(u) = v$. The value of ϵ_{NCPe} in Lem. 29 remains the same in this case. $\square$

Theorem 7. $\text{Maltese}_{R_{\mathbb{F}}} : \text{PE}_{R_{\mathbb{F}}}(1, \ell, b) \rightarrow (0, 1)$ is a reduction of knowledge (Def. 4).

Proof. By from Thm. 1 (strong-weak reduction composition), Lem. 17 and 18 ($\Pi_{\text{NC}}^{\text{PE}, R_{\mathbb{F}}}$ is a strong interactive reduction), Lem. 31 and 32 (Π_{Fold} is a weak interactive reduction), $\Pi_{\text{Fold}} \circ \Pi_{\text{NC}}^{\text{PE}, R_{\mathbb{F}}}$ is a reduction of knowledge (Def. 4). By Lem. 2 (reduction of knowledge sequential composition) and Lem. 33 and Lem. 34 (Π_{Dec} is a reduction of knowledge), $\Pi_{\text{Dec}} \circ \Pi_{\text{Fold}} \circ \Pi_{\text{NC}}^{\text{PE}, R_{\mathbb{F}}}$ is a reduction of knowledge. Similarly, additionally due to 28 and 29 ($\Pi_{\text{NC}}^{\text{PE}}$ is a strong interactive reduction), $\Pi_{\text{Dec}} \circ \Pi_{\text{Fold}} \circ \Pi_{\text{NC}}^{\text{PE}}$ is a reduction of knowledge. By Lem. 15, Π_{Fin} is a reduction of knowledge. Thus by Lem. 2, $\text{Maltese}_{R_{\mathbb{F}}}$ is a reduction of knowledge. $\square$

A.2 Potentially high norm $\mathbb{F}$ -polynomials

Evaluation claims on polynomials $\tilde{f} \in \mathbb{F}[X_1, \dots, X_{\ell+\log n+\log d}]$ can also be supported. The input relation will then be the following:

$$\text{PE}_{\mathbb{F}}(1, \ell, b) := \left\{ \begin{array}{l} \text{x} := \left(\begin{array}{l} t \in R_{\mathbb{F}}^n, \\ u \in \mathbb{K}^{\ell+\log n+\log d}, \\ v \in \mathbb{K} \end{array} \right), \quad : \quad \begin{array}{l} \text{Open}(A, b, t, (s_i)_{i \in [\ell]}) \\ f' := G_{2^{\ell}n} \cdot s_{\ell} \in R_{\mathbb{F}}^{2^{\ell}n} \\ f := [\text{cf}(f')_0, \dots, \text{cf}(f')_{d-1}] \in \mathbb{F}^{2^{\ell}nd} \\ \tilde{f}(u) = v \end{array} \end{array} \right\}$$

where $\text{cf}(f')_k$ is defined as in Definition 3. The main idea is to batch this evaluation claim $\tilde{f}(u) = v$ into the field sum-check in step 2 of $\Pi_{\text{NC}}^{\text{PE}}$.

However, $\tilde{f}$ is evaluated on an additional $\log d$ variables. Instead of only directly modifying $\Pi_{\text{NC}}^{\text{PE}}$, we will additionally include a reduction of knowledge $\Pi_{\text{Split}, \mathbb{F}}$ at the prefix of $\text{Maltese}_{\mathbb{F}}$. In this reduction, the prover partially evaluates $\tilde{f}$ on all but the first $\log d$ variables to reduce to a claim on a point in $\mathbb{K}^{\ell+\log n}$. The partial evaluation will result in a vector in $\mathbb{K}^d$. We define the following relation to formalize the output of this reduction

$$\text{PE-Split}(\ell, b) := \left\{ \begin{array}{l} \text{x} := \left(\begin{array}{l} t \in R_{\mathbb{F}}^n, \\ u \in \mathbb{K}^{\ell} \times \mathbb{K}^{\log n}, \\ (v_0, \dots, v_{d-1}) \in \mathbb{K}^d \end{array} \right), \quad : \quad \begin{array}{l} \text{Open}(A, b, t, (s_i)_{i \in [\ell]}) \\ f' := G_{2^{\ell}n} \cdot s_{\ell} \in R_{\mathbb{F}}^{2^{\ell}n} \\ \forall k \in [0, d-1], \\ \widetilde{\text{cf}(f')_k}(u) = v_k \end{array} \end{array} \right\}$$

Then the outputted statement-witness pair in $\text{PE-Split}(\ell, b)$ can be used as input for a modified $\Pi_{\text{NC}}^{\text{PE}}$ (see Prot. 9), which will now take the d evaluation claims on $\ell + 1 + m$ variables.

Protocol 8: $\mathbb{F}$ -polynomial evaluation claim split $\Pi_{\text{Split}, \mathbb{F}}$	
Parameters: ℓ tree layers, opening norm bound b	
Input: $\left(\text{x} := \left(\begin{array}{l} t \in R_{\mathbb{F}}^n, \\ u \in \mathbb{K}^{\log d + \ell + \log n}, \\ v \in \mathbb{K} \end{array} \right), \text{w} := (s_i \in R_{\mathbb{F}}^{2^i n \alpha})_{i \in [\ell]} \right) \in \text{PE}_{\mathbb{F}}(1, \ell, b)$	
Output: $\left(\begin{array}{l} \text{x} := \left(\begin{array}{l} t \in R_{\mathbb{F}}^n, \\ u' \in \mathbb{K}^{\ell} \times \mathbb{K}^{\log n}, \\ v_{\text{split}} \in \mathbb{K}^d \end{array} \right), \\ \text{w} := (s_i \in R_{\mathbb{F}}^{2^i n \alpha})_{i \in [\ell]} \end{array} \right) \in \text{PE-Split}(\ell, b)$	
1. $\mathcal{P}$ and $\mathcal{V}$ create a vector of d evaluation claims $v_{\text{split}} := (v_0, \dots, v_{d-1}) \in \mathbb{K}^d$ on each corresponding $\widetilde{\text{cf}(f')_k}$, $\forall k \in [0, d-1]$. (a) $\mathcal{P}$ computes an evaluation claim on each $\widetilde{\text{cf}(f')_k}$ on the suffix $u^{(\log d : \cdot)} \in R_{\mathbb{F}}^{\ell+\log n}$ for all $k \in [0, d-1]$. $f' := G_{2^{\ell}n} \cdot s_{\ell}$ $v_k := \widetilde{\text{cf}(f')_k}(u^{(\log d + 1 : \cdot)}), \quad \forall k \in [0, d-1]$ and sends $v_{\text{split}} := (v_0, \dots, v_{d-1}) \in \mathbb{K}^d$ to $\mathcal{V}$. (b) $\mathcal{V}$ checks consistency of claimed v_{split} vector with input claim v. $\tilde{v}_{\text{split}}(u^{(\log d : \cdot)}) \stackrel{?}{=} v$	

2. $\mathcal{P}$ and $\mathcal{V}$ output

$$\left(x := \begin{pmatrix} t \in R_{\mathbb{F}}^n, \\ u' := u^{(\log d+1)} \in \mathbb{K}^\ell \times \mathbb{K}^{\log n}, \\ v_{\text{split}} \in \mathbb{K}^d \end{pmatrix}, w := (s_i \in R_{\mathbb{F}}^{2^i n \alpha})_{i \in [\ell]} \right) \in \text{PE-Split}(\ell, b)$$

Now we can similarly use $\tilde{p}_{b,\alpha}$ as in Appendix A.1 and batch the following sum-check claims into the first sum-check (Step 1):

$$v_{\text{split}}^{(k)} \stackrel{?}{=} \sum_{X \in \{0,1\}^{\ell+m+1}} \tilde{eq}(1, X_1) \cdot \tilde{eq}(u', X_{[2:\ell+\log n+1]}) \cdot \tilde{p}_{b,\alpha}(X_{[\ell+\log n+2:]}) \cdot \widetilde{\text{cf}(s)_k}(X) \quad \forall k \in [0, d-1] \quad (40)$$

Protocol 9: $\Pi_{\text{NC}}^{\text{PE}, \mathbb{F}}$: Entry reduction for $\mathbb{F}$ -polynomials

Parameters: ℓ tree layers per commitment, opening norm bound b

Input: $\left(x := \begin{pmatrix} t \in R_{\mathbb{F}}^n, \\ u' \in \mathbb{K}^\ell \times \mathbb{K}^{\log n}, \\ v_{\text{split}} \in \mathbb{K}^d \end{pmatrix}; w := (s_i \in R_{\mathbb{F}}^{2^i n \alpha})_{i \in [\ell]} \right) \in \text{PE-Split}(\ell, b)$

Output: $\left(x := \begin{pmatrix} t \in R_{\mathbb{F}}^n, \\ r' \in R_{\mathbb{F}}^{\ell+1+m}, \\ y \in R_{\mathbb{F}} \end{pmatrix}; w := (s_i \in R_{\mathbb{F}}^{2^i n \alpha})_{i \in [\ell]} \right) \in \text{TE}(1, \ell, b)$

1. $\mathcal{P}$ and $\mathcal{V}$ run SC over $\mathbb{K}$ to check the norm of coefficient polynomials $\text{cf}(s)_k$, reducing to a polynomial evaluation claim on $r_N \in \mathbb{K}^{\ell+1+m}$, $y_N \in \mathbb{K}$.

(a) $\mathcal{V}$ samples $\xi \leftarrow_{\mathbb{S}} \mathbb{K}$, $r \leftarrow_{\mathbb{S}} \mathbb{K}^{\ell+1+m}$ and sends them to $\mathcal{P}$.

(b) $\mathcal{P}$ and $\mathcal{V}$ define the polynomial $Q_N \in \mathbb{K}[X_{[1:\ell+1+m]}]$, which batches infinity norm-check polynomials for $\text{cf}(s)_k$ (see Eq. (8)) across $k \in [0, d-1]$ via challenge ξ :

$$Q_N(X) := \sum_{k \in [0, d-1]} \xi^k \cdot N_b(\widetilde{\text{cf}(s)_k}(X)) \in \mathbb{K}[X_{[1:\ell+1+m]}]$$

where $N_b \in \mathbb{F}[X]$ is defined as in Eq. (7) and $\widetilde{\text{cf}(s)_k} \in \mathbb{F}[X_{[1:\ell+1+m]}]$ as in Def. 3.

(c) $\mathcal{P}$ and $\mathcal{V}$ define polynomial $Q_{\text{in}} \in \mathbb{K}[X_{[1:\ell+1+m]}]$, batching the input $\mathbb{F}$ -polynomial evaluation sum-check claims (see Eq. (40)) over $k \in [0, d-1]$, along with corresponding batched claim $v_{\text{in}} \in \mathbb{K}$:

$$Q_{\text{in}}(X) := \sum_{k \in [0, d-1]} \xi^{d+k} \cdot \tilde{eq}(1, X_1) \cdot \tilde{eq}(u', X_{[2:\ell+\log n+1]}) \cdot \tilde{p}_{b,\alpha}(X_{[\ell+\log n+2:]}) \cdot \widetilde{\text{cf}(s)_k}(X) \in \mathbb{K}[X_{[1:\ell+1+m]}]$$

$$v_{\text{in}} := \sum_{k \in [0, d-1]} \xi^{d+k} \cdot v_{\text{split}}^{(k)} \in \mathbb{K}$$

(d) $\mathcal{P}$ and $\mathcal{V}$ invoke Lem. 4 on Q_N with challenge r to produce a zero-check claim, and then run the sum-check protocol (see Def. 9):

$$(r_N \in \mathbb{K}^{\ell+1+m}, y_N \in \mathbb{K}) \leftarrow \text{SC}(v_{\text{in}}, \ell+1+m, \tilde{eq}(X, r) \cdot Q_N(X) + Q_{\text{in}}(X))$$

2. $\mathcal{P}$ and $\mathcal{V}$ invoke Lem. 14 to transform the polynomial evaluation claim on $r_N \in \mathbb{K}^{\ell+1+m}$, $y_N \in \mathbb{K}$ into evaluation claims $\tilde{s}(r_N) = a \in R_{\mathbb{K}}$.

(a) $\mathcal{P}$ computes and sends $a := \tilde{s}(r_N) \in R_{\mathbb{K}}$, which $\mathcal{V}$ interprets via the coefficient map cf (Def. 3) as $(a_0, \dots, a_{d-1}) := \text{cf}(a) \in \mathbb{K}^{1 \times d}$.

(b) $\mathcal{V}$ checks consistency of claims a_k with y_N (see Lem. 14 with $s \in R_{\mathbb{F}}^{2^{\ell+1}n\alpha}$, $r_N \in \mathbb{K}$):

$$y_N \stackrel{?}{=} \tilde{e}q(r_N, r) \cdot \sum_{k \in [0, d-1]} \xi^k \cdot N_b(a_k) \\ + \sum_{k \in [0, d-1]} \xi^{d+k} \cdot \tilde{e}q(1, r_N^{(1)}) \cdot \tilde{e}q(u', r_N^{(2: \ell + \log n + 1)}) \cdot \tilde{p}_{b, \alpha}(r_N^{(\ell + \log n + 2:)}) \cdot a_k$$

3. $\mathcal{P}$ and $\mathcal{V}$ run ShiftSC (Prot. 2) to reduce both the evaluation claims from Step 2 (see Eq. (9)) and the input evaluation claims (see Eq. (10)) to evaluation claims in $R_{\mathbb{F}}$ over the same input point r' .

(a) For Step 2 evaluation claims $\tilde{s}(r_N) = a \in R_{\mathbb{K}}$, $\mathcal{P}$ and $\mathcal{V}$ define ShiftSC inputs (see Eq. (9)):

$$C_{\text{sc}} : a \stackrel{?}{=} \sum_{X \in \{0,1\}^{\ell+1+m}} \tilde{e}q(r_N, X) \cdot \tilde{s}(X) \in R_{\mathbb{K}}$$

(b) $\mathcal{P}$ and $\mathcal{V}$ invoke ShiftSC on the 1 claim from the previous step:

$$(r', y \in R_{\mathbb{F}}) \leftarrow \text{ShiftSC}(C_{\text{sc}})$$

Note that since $\tilde{s}$ is over $R_{\mathbb{F}}$, output y is also in $R_{\mathbb{F}} \subseteq R_{\mathbb{K}}$ (though ShiftSC outputs $R_{\mathbb{K}}$ values in general). Also, the evaluation claims corresponding to the publicly evaluable multilinear on r' can be computed by $\mathcal{V}$.

4. $\mathcal{P}$ and $\mathcal{V}$ output $x := (t, r', y)$, $w := (s_i)_{i \in [\ell]}$.

Lemma 19. $\Pi_{\text{Split}, \mathbb{F}} : \text{PE}_{\mathbb{F}}(1, \ell, b) \rightarrow \text{PE-Split}(\ell, b)$ is complete, with $O(d \cdot 2^{\ell + \log n})$ prover time, $O(d)$ verifier time, and proof size $d \mathbb{K}$ claims.

Proof. In reduction $\Pi_{\text{Split}, \mathbb{F}}$, if input $(x, w) \in \text{PE}_{\mathbb{F}}(1, \ell, b)$, then for correctly computed v_k , it holds that

$$\begin{aligned} \tilde{v}_{\text{split}}(u^{(: \log d)}) &= \sum_{k \in [0, d-1]} \tilde{e}q(u^{(: \log d)}, \text{bits}_{\log d}(k)) \cdot \tilde{v}_{\text{split}}^{(k)} && \text{(multilinear extension)} \\ &= \sum_{k \in [0, d-1]} \tilde{e}q(u^{(: \log d)}, \text{bits}_{\log d}(k)) \cdot \widetilde{\text{cf}(f')_k}(u^{(\log d + 1:)}) && (v_{\text{split}}^{(k)} \text{ definition}) \\ &= \sum_{k \in [0, d-1]} \tilde{e}q(u^{(: \log d)}, \text{bits}_{\log d}(k)) \cdot \left(\sum_{i \in [2^{\ell} \cdot n]} \tilde{e}q(u^{(\log d + 1:)}, \text{bits}_{\ell + \log n}(i)) \cdot \text{cf}(f')_k^{(i)} \right) && \text{(multilinear extension)} \\ &= \sum_{k \in [0, d-1]} \sum_{i \in [2^{\ell} \cdot n]} \tilde{e}q(u, \text{bits}_{\log d}(k) \parallel \text{bits}_{\ell + \log n}(i)) \cdot \text{cf}(f')_k^{(i)} && \text{(rearrange sum)} \\ &= \tilde{f}(u) = v && (41) \end{aligned}$$

where the last line follows from the definition of the multilinear extension of $f := [\text{cf}(f')_0, \dots, \text{cf}(f')_{d-1}]$. And so the verifier check passes, and the output evaluation claims hold. $\square$

Lemma 20. $\Pi_{\text{Split}, \mathbb{F}} : \text{PE}_{\mathbb{F}}(1, \ell, b) \rightarrow \text{PE-Split}(\ell, b)$ is knowledge sound with knowledge error 0.

Proof. Let $\mathcal{P}^*$ have success probability

$$\epsilon(\mathcal{A}, \mathcal{P}^*) := \Pr \left[(pp, \langle \mathcal{P}^*, \mathcal{V} \rangle(x_1, st)) \in \text{PE-Split}(\ell, b) \mid \begin{array}{l} pp \leftarrow \mathcal{G}(1^\lambda) \\ (x_1, st) \leftarrow \mathcal{A}(pp) \end{array} \right] \geq 1/\text{poly}(\lambda).$$

The extractor is as follows in Fig. 7.

Extractor runtime. The extractor runs the interaction between $\mathcal{P}^*$ and $\mathcal{V}$ once, and then the rest of the operations are clearly polynomial time.

Extractor success probability. If $\mathcal{P}^*$ succeeds in producing w_2 such that $(pp, x_2, w_2) \in \text{PE-Split}(\ell, b)$, then it holds that

$$\forall k \in [0, d-1], \quad \widetilde{\text{cf}(f')}_k u' = v_k$$

which means that for $w_1 := w_2 = (s_i)_{i \in [\ell]}$ and $x_2 := (t, u', (v_0, \dots, v_{d-1}))$, it holds that $\widetilde{\text{cf}(f')}_k(u') = v_k$ for all $k \in [0, d-1]$. Then by construction of x_2 from x_1 in the reduction $\Pi_{\text{Split}, \mathbb{F}}$, $\tilde{f}(u) = v$ where $f := [\text{cf}(f')_0, \dots, \text{cf}(f')_{d-1}]$, as shown by the equalities in Eq. (41). And so

$$\Pr \left[(pp, x_1, w_1) \in \text{PE}_{\mathbb{F}}(1, \ell, b) \mid \begin{array}{l} pp \leftarrow \mathcal{G}(1^\lambda) \\ (x_1, st) \leftarrow \mathcal{A}(pp) \\ w_1 \leftarrow \mathcal{E}(pp, x_1, st) \end{array} \right] \geq \epsilon(\mathcal{A}, \mathcal{P}^*) - 0$$

Extractor $\mathcal{E}(pp, x_1, st) \rightarrow w_1$:
1. Simulate $(x_2, w_2) \leftarrow \langle \mathcal{P}^*, \mathcal{V} \rangle(pp, x_1, st)$ with fresh randomness. 2. If $(pp, x_2, w_2) \notin \text{PE-Split}(\ell, b)$, then output $\perp$. 3. Set $w_1 := w_2$. 4. Output w_1.

Figure 7: Extractor for $\Pi_{\text{Split}, \mathbb{F}}$.

□

Lemma 21. $\Pi_{\text{NC}}^{\text{PE}, \mathbb{F}} : \text{PE-Split}(\ell, b) \rightarrow \text{TE}(1, \ell, b)$ ($\text{TE}^*(1, \ell)$) is complete with prover work consisting of an $O((2b-1) \cdot 2^{(\ell+1+m)})$ sum-check prover, $O(bd \cdot (\ell+1+m))$ verifier time, and proof size consisting of 1 degree $2b$ sum-check, 1 ShiftSC of maximum degree 2 outputting one evaluation claim, and one additional $R_{\mathbb{K}}$ claim.

Proof. This follows by the proof Lem. 28, slightly modified for the correctness of the statement: Eq. (40) holds if and only if $\widetilde{\text{cf}(f')}_k(u') = v_{\text{split}}^{(k)}$ for all $k \in [0, d-1]$. □

Lemma 22. $\Pi_{\text{NC}}^{\text{PE}, \mathbb{F}} : \text{PE-Split}(\ell, b) \rightarrow \text{TE}(1, \ell, b)$ ($\text{TE}^*(1, \ell)$) is a strong interactive reduction (Def. 7) with negligible knowledge error; where ϕ is the function which projects commitments from the instance.

Proof. This follows by the proof Lem. 29, slightly modified for the correctness of the statement: Eq. (40) holds if and only if $\widetilde{\text{cf}(f')}_k(u') = v_{\text{split}}^{(k)}$ for all $k \in [0, d-1]$. The Schwartz-Zippel bound in Claim 2 also becomes $\leq \frac{L(2d-1)+\ell+m}{|\mathbb{K}|}$ to match the degree of ξ in $\Pi_{\text{NC}}^{\text{PE}, \mathbb{F}}$. □

Theorem 8. $\text{Maltese}_{\mathbb{F}} : \text{PE}_{\mathbb{F}}(1, \ell, b) \rightarrow (0, 1)$ is a reduction of knowledge (Def. 4).

Proof. By from Thm. 1 (strong-weak reduction composition), Lem. 21 and 22 ($\Pi_{\text{NC}}^{\text{PE}, \mathbb{F}}$ is a strong interactive reduction), Lem. 31 and 32 (Π_{Fold} is a weak interactive reduction), $\Pi_{\text{Fold}} \circ \Pi_{\text{NC}}^{\text{PE}, \mathbb{F}}$ is a reduction of knowledge (Def. 4). By Lem. 2 (reduction of knowledge sequential composition) and Lem. 33 and Lem. 34 (Π_{Dec} is a reduction of knowledge), $\Pi_{\text{Dec}} \circ \Pi_{\text{Fold}} \circ \Pi_{\text{NC}}^{\text{PE}, \mathbb{F}}$ is a reduction of knowledge. Similarly, additionally due to 28 and 29 ($\Pi_{\text{NC}}^{\text{PE}}$ is a strong interactive reduction), $\Pi_{\text{Dec}} \circ \Pi_{\text{Fold}} \circ \Pi_{\text{NC}}^{\text{PE}}$ is a reduction of knowledge. By Lem. 15, Π_{Fin} is a reduction of knowledge. Lastly, by Lem. 19 and Lem. 20, $\Pi_{\text{Split}, \mathbb{F}}$ is a reduction of knowledge. Thus by Lem. 2, $\text{Maltese}_{\mathbb{F}}$ is a reduction of knowledge. □

B Deferred Proofs

B.1 From Section 3: Schwartz-Zippel for Approximate Strong Sampling Sets

Proof of Lemma 8: Univariate Schwartz-Zippel for approximate strong sampling sets.

Proof. Define

$$\text{Roots}(p) := \{s \in \mathcal{C} : p(s) = 0\},$$

and for each $s \in \mathcal{C}$, define

$$\text{Bad}(s) := \{r \in \mathcal{C} \setminus \{s\} \mid r - s \notin R^\times\}$$

By definition of $\epsilon_{\text{inv}}(\mathcal{C})$,

$$\forall s \in \mathcal{C}, \quad |\text{Bad}(s)| \leq (|\mathcal{C}| - 1) \cdot \epsilon_{\text{inv}}(\mathcal{C}).$$

We prove by induction on the polynomial degree that

$$|\text{Roots}(p)| \leq d \cdot (1 + (|\mathcal{C}| - 1) \cdot \epsilon_{\text{inv}}(\mathcal{C})).$$

Base case. If $d = 0$, then p is a nonzero constant, so $\text{Roots}(p) = \emptyset$ and thus $|\text{Roots}(p)| = 0$.

Inductive step. Suppose now $d \geq 1$. If $\text{Roots}(p) = \emptyset$, then the claim is immediate. Else, choose $s \in \text{Roots}(p)$. Since s is a root of p , we can write

$$p(X) = (X - s) \cdot q(X), \quad \deg(q) = d - 1.$$

Consider any $r \in \text{Roots}(p) \setminus \{s\} \cup \text{Bad}(s)$. Because r is a root of p , it holds that

$$0 = p(r) = (r - s) \cdot q(r).$$

Because $r \notin \text{Bad}(s)$, $(r - s) \in R^\times$, and so we can conclude that $q(r) = 0$. It follows that

$$\text{Roots}(p) \subseteq \{s\} \cup \text{Bad}(s) \cup \text{Roots}(q).$$

By the induction hypothesis on q ,

$$\begin{aligned} |\text{Roots}(p)| &\leq 1 + |\text{Bad}(s)| + |\text{Roots}(q)| \\ &\leq 1 + (|\mathcal{C}| - 1) \cdot \epsilon_{\text{inv}}(\mathcal{C}) \\ &\quad + (d - 1) \cdot (1 + (|\mathcal{C}| - 1) \cdot \epsilon_{\text{inv}}(\mathcal{C})) \\ &= d \cdot (1 + (|\mathcal{C}| - 1) \cdot \epsilon_{\text{inv}}(\mathcal{C})). \end{aligned}$$

Finally, dividing by $|\mathcal{C}|$ gives

$$\begin{aligned} \Pr_{s \leftarrow \mathcal{C}}[p(s) = 0] &= \frac{|\text{Roots}(p)|}{|\mathcal{C}|} \\ &\leq \frac{d \cdot (1 + (|\mathcal{C}| - 1) \cdot \epsilon_{\text{inv}}(\mathcal{C}))}{|\mathcal{C}|} \\ &= d \cdot \left(\frac{1}{|\mathcal{C}|} + \left(1 - \frac{1}{|\mathcal{C}|}\right) \cdot \epsilon_{\text{inv}}(\mathcal{C}) \right) \\ &\leq d \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right). \end{aligned}$$

□

Proof of Lemma 9: Multivariate Schwartz-Zippel for approximate strong sampling sets.

Proof. We proceed by induction on the number m of variables.

Base case. When $m = 1$, this is the univariate lemma (Lem. 8).

Inductive step. Now suppose $m \geq 2$. Group together all terms containing the same power of X_m , which has maximum degree $k \leq d$. We can write for

$$p(X_1, \dots, X_m) = \sum_{i=0}^k p_i(X_1, \dots, X_{m-1}) \cdot X_m^i$$

where p_k is nonzero. Since $p_k \cdot X_m^k$ appears in p and the total degree of p is at most d ,

$$\deg(p_k) + k \leq d \implies \deg(p_k) \leq d - k.$$

For independently sampled $s_1, \dots, s_{m-1} \leftarrow_{\mathcal{S}} \mathcal{C}^{m-1}$ and $s_m \leftarrow_{\mathcal{S}} \mathcal{C}$, there are two cases when $p(s_1, \dots, s_m) = 0$.

1. The coefficient $p_k(s_1, \dots, s_{m-1}) = 0$. Since p_k is a nonzero polynomial in $m-1$ variables of degree at most $d-k$, the induction hypothesis gives

$$\Pr[p(s_1, \dots, s_m) = 0 \wedge p_k(s_1, \dots, s_{m-1}) = 0] \leq (d-k) \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right). \quad (42)$$

2. The coefficient $p_k(s_1, \dots, s_{m-1}) \neq 0$. Consider the polynomial in the one remaining variable X_m :

$$q(X_m) := p(s_1, \dots, s_{m-1}, X_m) = \sum_{i=0}^k p_i(s_1, \dots, s_{m-1}) \cdot X_m^i.$$

The coefficient of X_m^k in q is $p_k(s_1, \dots, s_{m-1})$, which is nonzero. Therefore, q is a nonzero univariate polynomial of degree k . By Lemma 8 (univariate Schwartz-Zippel),

$$\Pr_{s_m \leftarrow \mathbb{S}^{\mathcal{C}}} [p(s_1, \dots, s_m) = 0 \wedge p(s_1, \dots, s_{m-1}, s_m) = 0] \leq k \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right). \quad (43)$$

Putting the two cases together,

$$\begin{aligned} \Pr_{(s_1, \dots, s_m) \leftarrow \mathbb{S}^{\mathcal{C}^m}} [p(s) = 0] &\leq \Pr[p(s) = 0 \wedge p_k(s_1, \dots, s_{m-1}) = 0] + \Pr[p(s) = 0 \wedge p_k(s_1, \dots, s_{m-1}) \neq 0] \\ &\leq (d-k) \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right) + k \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right) \\ &= d \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right). \end{aligned}$$

□

B.2 From Section 4: Sum-Checks Over Rings Using Isomorphisms

Lemma 23 (BatchSC completeness). $\text{BatchSC} : \text{BSC}_{\text{in}}(R_{\mathbb{F}}, a, b) \rightarrow \text{BSC}_{\text{out}}(R_{\mathbb{F}}, R_{\mathbb{F}}, b)$ is complete with prover time, logarithmic time verifier, and logarithmic proof size.

Proof. The prover and verifier costs are due to the sum-check in Step 4, NTT costs in Step 2, and evaluations of ring polynomials $Q_j(r_{[1:\ell_j]})$ in Step 5. The proof size is the sum-check proof size in Step 4, which is logarithmic $O(D \cdot \ell)$ along with b oracles, plus the b ring elements sent in Step 5 (these may not all need to be sent depending on how the polynomial oracles are instantiated).

Completeness. If it holds that for all $i \in [a]$, $v_i = \sum_{X \in \{0,1\}^{\ell_i}} G_i(Q_1(X), \dots, Q_b(X))$ then it also holds for $\ell \geq \ell_i$,

$$2^{\ell-\ell_i} \cdot v_i = \sum_{X \in \{0,1\}^{\ell}} G_i(Q_1(X), \dots, Q_b(X))$$

because there are $2^{\ell-\ell_i}$ bitstrings in $\{0,1\}^{\ell}$ that share the same prefix $x' \in \{0,1\}^{\ell_i}$. Then taking a linear combination with powers of η produces

$$\sum_{i=1}^a \eta^{i-1} \cdot 2^{\ell-\ell_i} \cdot v_i = \sum_{X \in \{0,1\}^{\ell}} \left(\sum_{i=1}^a \eta^{i-1} \cdot G_i(Q_1(X), \dots, Q_b(X)) \right)$$

which is exactly the claim $v = \sum_{X \in \{0,1\}^{\ell}} G(X)$ in Step 1. Applying the NTT transform to v and G , the claim

$$v^{(k)} = \sum_{X \in \{0,1\}^{\ell}} G^{(k)}(Q_1^{(k)}(X), \dots, Q_b^{(k)}(X)) \in \mathbb{K} \quad \forall j \in [d/e]$$

also holds because NTT is a ring isomorphism. Taking a linear combination of these equations in $\mathbb{K}$ implies that

$$v^{\star} = \sum_{X \in \{0,1\}^{\ell}} G^{\star}(x)$$

By completeness of the sum-check protocol over $\mathbb{K}$, Step 4 passes and produces claim $y = G^*(h)$. In Step 5, $\mathcal{P}$ honestly sends $y_j = Q_j(r_{[1:\ell_j]})$ for $r \leftarrow \text{NTT}^{-1}(h, \dots, h) \in R_{\mathbb{F}}^{\ell}$. We now show that $\mathcal{V}$'s check in this step passes.

$$\begin{aligned}
y &= G^*(h) \\
&= \sum_{k=1}^{d/e} \gamma^{k-1} \cdot G^{(k)}(Q_1^{(k)}(h), \dots, Q_b^{(k)}(h)) && \text{(def of } G^*) \\
&= \sum_{k=1}^{d/e} \gamma^{k-1} \cdot \left(\sum_{i=1}^a \eta^{i-1} \cdot G_i^{(k)}(Q_1^{(k)}(h), \dots, Q_b^{(k)}(h)) \right) && \text{(def. of } G, \text{ NTT isomorphism)} \\
&= \sum_{k=1}^{d/e} \gamma^{k-1} \cdot \text{NTT} \left(\sum_{i=1}^a \eta^{i-1} \cdot G_i(Q_1(r), \dots, Q_b(r)) \right)^{(k)} && \text{(NTT isomorphism)} \\
&= \sum_{k=1}^{d/e} \gamma^{k-1} \cdot y^{(k)}
\end{aligned}$$

Lastly, because $(y_j)_{j \in [b]}$ are computed honestly,

$$(x := (\ell, r, (y_j, \llbracket Q_j \rrbracket)_{j \in [b]}); w := (Q_j)_{j \in [b]}) \in \text{BSC}_{\text{out}}(R_{\mathbb{F}}, R_{\mathbb{F}}, b).$$

□

Lemma 24 (BatchSC soundness). *Let the map $\text{NTT} : R_{\mathbb{F}} \rightarrow \mathbb{K}^{d/e}$ be defined as in Def. 3.2. Let $\epsilon_{\text{inv}}(\mathcal{C})$ be the local invertibility error of $\mathcal{C}$ (see Def. 10). The protocol $\text{BatchSC} : \text{BSC}_{\text{in}}(R_{\mathbb{F}}, a, b) \rightarrow \text{BSC}_{\text{out}}(R_{\mathbb{F}}, R_{\mathbb{F}}, b)$, which takes as part of its input sum-check claims on $\leq \ell$ variables of degree $\leq D$, is sound with soundness error $\epsilon_{\text{BSC}} \leq (a-1) \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right) + \frac{d/e-1}{|\mathbb{K}|} + \frac{\ell \cdot D}{|\mathbb{K}|}$.*

Proof. Suppose

$$(x := ((v_i, \ell_i, G_i)_{i \in [a]}, (\ell_j, \llbracket Q_j \rrbracket)_{j \in [b]}); w := (Q_j)_{j \in [b]}) \notin \text{BSC}_{\text{in}}(R_{\mathbb{F}}, a, b).$$

Then $\exists i \in [a]$ such that

$$v_i \neq \sum_{X \in \{0,1\}^{\ell_i}} G_i(Q_1(X), \dots, Q_b(X)).$$

Define the following polynomials on indeterminates C, D (which take the place of η and γ , respectively)

$$\begin{aligned}
T_{\mathcal{C}}(C) &:= \sum_{i=1}^a C^{i-1} \cdot 2^{\ell-\ell_i} \cdot v_i, & G(X, C) &:= \sum_{i=1}^a C^{i-1} \cdot G_i(Q_1(X_{[1:\ell_i]}), \dots, Q_b(X_{[1:\ell_i]})) \\
T_{\mathbb{K}}(D) &:= \sum_{k=1}^{d/e} D^{k-1} \cdot v^{(k)}, & G^*(X, D) &:= \sum_{k=1}^{d/e} D^{k-1} \cdot Q^{(k)}(X)
\end{aligned}$$

For $\mathcal{P}^*$ to succeed, at least one of the following must hold:

1. (Step 1) The non-zero polynomial $P(C) := T_{\mathcal{C}}(C) - \sum_{X \in \{0,1\}^{\ell}} G(X, C)$ evaluated to 0 on a random challenge $\eta \leftarrow_{\$} \mathcal{C}$.
2. OR (Step 3) The non-zero polynomial $R(D) := T_{\mathbb{K}}(D) - \sum_{X \in \{0,1\}^{\ell}} G^*(X, D)$ evaluated to 0 on a random challenge $\gamma \leftarrow_{\$} \mathbb{K}$.
3. OR (Step 4) $\mathcal{P}$ succeeded in the sum-check protocol over $\mathbb{K}$, despite $T(D) \neq \sum_{X \in \{0,1\}^{\ell}} G^*(X, D)$.

The probability of (1) occurring is $\leq (a-1) \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right)$ by the Schwartz-Zippel lemma over rings (Lem. 9), where $a-1$ is the degree of C in P . The probability of (2) occurring is $\leq \frac{d/e-1}{|\mathbb{K}|}$ by the Schwartz-Zippel lemma (Lem. 1).

Lastly, (3) occurs with probability $\leq \frac{\ell \cdot \max_{i \in [a]} (\deg(G_i))}{|\mathbb{K}|} = \frac{\ell \cdot D}{|\mathbb{K}|}$ by the soundness of the sum-check protocol (Lem. 3).

By a union bound, the soundness error of BatchSC is $\epsilon_{\text{BSC}} \leq (a-1) \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right) + \frac{d/e-1}{|\mathbb{K}|} + \frac{\ell \cdot D}{|\mathbb{K}|}$. □

Lemma 25 (ShiftSC completeness). $\text{ShiftSC} : \text{BSC}_{\text{in}}(R_{\mathbb{K}}, a, b) \rightarrow \text{BSC}_{\text{out}}(R_{\mathbb{K}}, R_{\mathbb{F}}, b)$ is complete with linear prover time, logarithmic time verifier, and logarithmic proof size.

Proof. The costs for ShiftSC are the cost of $\mathcal{P}$ and $\mathcal{V}$ computing CRT on the a input claims, followed by the costs of BatchSC (see Lem. 23). Completeness follows from the completeness of BatchSC and the output claim holds due to the following:

$$\begin{aligned} y_j &= \text{CRT}^{-1}(y_j^{(1)}, \dots, y_j^{(e)}) \\ &= \text{CRT}^{-1}(Q_j^{(1)}(r), \dots, Q_j^{(e)}(r)) && (\text{sum-check reduces to } y_j^{(k)} = Q_j^{(k)}(r)) \\ &= Q_j(r). && (\text{CRT isomorphism}) \end{aligned}$$

Therefore,

$$(x := (\ell, r, (y_j, \llbracket Q_j \rrbracket)_{j \in [b]}); w := (Q_j)_{j \in [b]}) \in \text{BSC}_{\text{out}}(R_{\mathbb{K}}, R_{\mathbb{F}}, b).$$

□

Lemma 26 (ShiftSC soundness). Let the map $\text{NTT} : R_{\mathbb{F}} \rightarrow \mathbb{K}^{d/e}$ be defined as in Def. 3.2. Let $\epsilon_{\text{inv}}(\mathcal{C})$ be the local invertibility error of $\mathcal{C}$ (see Def. 10). The protocol $\text{ShiftSC} : \text{BSC}_{\text{in}}(R_{\mathbb{K}}, a, b) \rightarrow \text{BSC}_{\text{out}}(R_{\mathbb{K}}, R_{\mathbb{F}}, b)$, which takes as part of its input sum-check claims on $\leq \ell$ variables of degree $\leq D$, is sound with soundness error $\epsilon_{\text{SSC}} \leq (e \cdot a - 1) \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right) + \frac{d/e-1}{|\mathbb{K}|} + \frac{\ell \cdot D}{|\mathbb{K}|}$.

Proof. The soundness error incurred by ShiftSC comes purely from a call to BatchSC on $e \cdot a$ claims. □

The following lemma gives a probability bound that is used in our security proofs for $\Pi_{\text{NC}}^{\text{PE}}$ (Appendix B.3), Π_{Fold} (Appendix B.4), Π_{Dec} (Appendix B.5), and Π_{Reshape} (Appendix B.6). In particular, we apply it to sum-checks with commitments instantiating the polynomial oracles for polynomials in the witness. This probability analysis can be used to leverage the binding of the commitment and soundness error of the sum-check protocol via a virtual experiment on a uniqueness adversary as in Def. 5.

Lemma 27 (Agreement of witnesses generated from the same transcript prefix). Let pp be public parameters, $\mathcal{Z}$ be an output space of polynomial sized strings. Let EPT adversary $\theta \leftarrow \mathcal{A}(pp)$, which outputs a prefix transcript θ . Let $z \leftarrow \mathcal{O}(\theta)$ be an EPT oracle, where $z \in \mathcal{Z} \cup \{\perp\}$. Each call to $\mathcal{O}(\theta)$ restores its state to θ and runs with independent fresh randomness.

For each prefix $\theta \in \text{Supp}(\mathcal{A}(pp))$, define a set of bad events $\text{Bad}(\theta) \subseteq \mathcal{Z}$. For each $\theta \in \text{Supp}(\mathcal{A}(pp))$, there exists a probability $\delta(\theta)$ such that for every fixed $z \in \text{Bad}(\theta)$, $\Pr[\mathcal{O}(\theta) = z] \leq \delta(\theta)$.

There must exist an EPT oracle algorithm $\mathcal{S}^{\mathcal{O}}$, making at most two oracle queries in expectation, such that

$$\Pr_{\substack{\theta \leftarrow \mathcal{A}(pp) \\ z \leftarrow \mathcal{O}(\theta)}} [z \in \text{Bad}(\theta)] \leq \mathbb{E}_{\theta}[\delta(\theta)] + \text{Adv}^{\text{uniq}}(\mathcal{S})$$

where $\text{Adv}^{\text{uniq}}(\mathcal{S})$ is the probability that $\mathcal{S}$ obtains two distinct $z \neq z' \in \mathcal{Z}$ from different queries to $\mathcal{O}(\theta)$.

Proof. We define the following uniqueness adversary, which first queries $\mathcal{O}(\theta)$ once and continues if it obtains a non-aborting value. Then, the adversary queries $\mathcal{O}(\theta)$ repeatedly on the same prefix but with fresh randomness until it obtains a non-aborting value.

Adversary $\mathcal{S}_{\text{uniq}}^{\mathcal{O}(\theta)}(\theta) \rightarrow (z, z')$:
1. Query $z \leftarrow \mathcal{O}(\theta)$ once. If $z = \perp$, output $\perp$. 2. Query $z' \leftarrow \mathcal{O}(\theta)$ with fresh randomness until $z' \neq \perp$. 3. Return (z, z').

For fixed θ , define the following probabilities, where $z \leftarrow \mathcal{O}(\theta)$:

$$\begin{aligned} q(\theta, z) &:= \Pr[\mathcal{O}(\theta) = z] \\ p(\theta) &:= \Pr[\mathcal{O}(\theta) \neq \perp] \\ a(\theta) &:= \sum_{z \in \text{Bad}(\theta)} q(\theta, z) \end{aligned}$$

where observe that $0 \leq a(\theta) \leq p(\theta) \leq 1$. If $p(\theta) = 0$, then the claim follows immediately because the oracle always aborts.

If $p(\theta) > 0$, we partition the event $z \in \text{Bad}(\theta)$ into $z \in \text{Bad}(\theta) \wedge z = z'$ and $z \in \text{Bad}(\theta) \wedge z \neq z'$. The second event can be bounded by the advantage of the uniqueness adversary

$$\Pr[z \in \text{Bad}(\theta) \wedge z \neq z' \mid \theta] \leq \Pr[z \neq z' \mid \theta].$$

Averaging over θ ,

$$\Pr[z \in \text{Bad}(\theta) \wedge z \neq z'] \leq \mathbb{E}_\theta[\Pr[z \neq z' \mid \theta]] \leq \text{Adv}^{\text{uniq}}(\mathcal{S}_{\text{uniq}}). \quad (44)$$

Now we focus on the case where $z = z'$. We bound the distribution of z' , the first non-aborting sample in Step 2. Observe that Step 2 succeeds with probability $p(\theta)$ each time because it is run with fresh randomness, and in the event that $\perp \leftarrow \mathcal{O}(\theta)$, we run the oracle again fresh. At each run, the probability of $z' \leftarrow \mathcal{O}(\theta)$ is $q(\theta, z')$ by definition. So we can write the recurrence:

$$\begin{aligned} \Pr[\mathcal{O}(\theta) = z' \mid \theta, z, \mathcal{O}(\theta) \neq \perp] &= q(\theta, z') + (1 - p(\theta)) \cdot \Pr[\mathcal{O}(\theta) = z' \mid \theta, z, \mathcal{O}(\theta) \neq \perp] \\ \implies \Pr[\mathcal{O}(\theta) = z' \mid \theta, z, \mathcal{O}(\theta) \neq \perp] &= \frac{q(\theta, z')}{p(\theta)} \end{aligned} \quad (45)$$

by rearranging. Then

$$\begin{aligned} \Pr[z \in \text{Bad}(\theta) \wedge z = z' \mid \theta] &= \sum_{z \in \text{Bad}(\theta)} \underbrace{q(\theta, z)}_{\leq \delta(\theta)} \cdot \frac{q(\theta, z)}{p(\theta)} && \text{(Eq. (45), prob. of } z \text{ in Step 2 for fixed } z) \\ &\leq \frac{\delta(\theta)}{p(\theta)} \cdot \sum_{z \in \text{Bad}(\theta)} q(\theta, z) && \text{(def. of } \delta(\theta)) \\ &= \frac{\delta(\theta) \cdot a(\theta)}{p(\theta)} && \text{(def. of } a(\theta)) \\ &\leq \delta(\theta). \end{aligned} \quad (46)$$

where the last inequality holds because $a(\theta)/p(\theta) \leq 1$. Putting the two partitions together,

$$\begin{aligned} \Pr_{\theta, z \leftarrow \mathcal{O}(\theta)}[z \in \text{Bad}(\theta)] &= \Pr_{\theta, z \leftarrow \mathcal{O}(\theta)}[z \in \text{Bad}(\theta) \wedge z = z'] + \Pr_{\theta, z \leftarrow \mathcal{O}(\theta)}[z \in \text{Bad}(\theta) \wedge z \neq z'] \\ &= \mathbb{E}_\theta \left[\Pr_{\theta, z \leftarrow \mathcal{O}(\theta)}[z \in \text{Bad}(\theta) \wedge z = z' \mid \theta] \right] + \Pr_{\theta, z \leftarrow \mathcal{O}(\theta)}[z \in \text{Bad}(\theta) \wedge z \neq z' \mid \theta] \\ &\leq \mathbb{E}_\theta \left[\Pr_{\theta, z \leftarrow \mathcal{O}(\theta)}[z \in \text{Bad}(\theta) \wedge z = z' \mid \theta] \right] + \text{Adv}^{\text{uniq}}(\mathcal{S}_{\text{uniq}}) \quad \text{(linearity of expectation, Eq. (44))} \\ &\leq \mathbb{E}_\theta[\delta(\theta)] + \text{Adv}^{\text{uniq}}(\mathcal{S}_{\text{uniq}}). \end{aligned} \quad \text{(Eq. (46))}$$

Lastly, we argue that $\mathcal{S}_{\text{uniq}}$ makes at most two oracle queries in expectation. For a fixed θ , let T_θ be the expected runtime of a single call to $\mathcal{O}(\theta)$. If $p(\theta) = 0$, then the expected runtime is T_θ because the first call always fails. If $p(\theta) > 0$, Step 1 calls EPT oracle $\mathcal{O}(\theta)$ once. Since each call to $\mathcal{O}(\theta)$ is independent, Step 2 (which occurs with probability $p(\theta)$) has expected runtime $T_\theta/p(\theta)$. Then the total runtime is

$$\underbrace{\leq T_\theta}_{\text{Step 1}} + p(\theta) \cdot \underbrace{\frac{T_\theta}{p(\theta)}}_{\text{Step 2}} = 2T_\theta.$$

□

B.3 From Section 5.1: Norm Check

Lemma 28. $\Pi_{\text{NC}}^{\text{PE}} : \text{PE}(L, \ell, b) \rightarrow \text{TE}(L, \ell, b) \ (\text{TE}^*(L, \ell))$ is complete with prover work consisting of an $O((2b-1) \cdot L \cdot 2^{(\ell+1+m)})$ sum-check prover, $O(Lbd \cdot (\ell+1+m))$ verifier time, and proof size consisting of 1 degree $2b$ sum-check, 1 ShiftSC of maximum degree 2 outputting one evaluation claim, and one additional $R_{\mathbb{K}}$ claim.

Proof. To show completeness, we argue that if input $(pp, x_1, w_1) \in \text{PE}(L, \ell, b)$, the verifier passes in all steps and the output produced by the interaction $\langle \mathcal{P}, \mathcal{V} \rangle$ with an honest prover satisfies $(pp, x_2, w_2) \in \text{TE}(L, \ell, b)$.

In Step 1, because $(s_{i,l})_{i \in [\ell], l \in L} \leftarrow w_1$ satisfy $\|s_l\|_{\infty} < b_l$ according to the definition of $\text{PE}(L, \ell, b)$ the sum-check verifier passes, as described in Sec. 5.1.1. By completeness of the sum-check protocol as a polyIOR, this reduces to the following evaluation claim on r :

$$y_N = \tilde{e}q(r_N, r) \cdot \left(\sum_{l \in L} \sum_{k \in [0, d-1]} \xi^{(l-1) \cdot d + k} \cdot N_{b_l}(\widetilde{\text{cf}(s_l)_k}(r_N)) \right).$$

In Step 2, the evaluation claims $a_l := \tilde{s}_l(r_N)$ are computed honestly by $\mathcal{P}$. By Lem. 14, the verifier check in Step 2b

$$y_N \stackrel{?}{=} \tilde{e}q(r_N, r) \cdot \left(\sum_{l \in [L]} \sum_{k \in [0, d-1]} \xi^{(l-1) \cdot d + k} \cdot N_{b_l}(a_{k,l}) \right)$$

passes. Lastly, the verifier in Step 3 passes because (1) the evaluation claims $a_l := \tilde{s}_l(r_N)$ are computed honestly by $\mathcal{P}$, sum-check claims $C_{\text{sc}, l}$ hold $\forall l \in [L]$, and (2) by definition of $\text{PE}(L, \ell, b)$ $\tilde{s}_l(u) = v_l$ where $((t_l)_{l \in [L]}, u, (v_l)_{l \in [L]}) \leftarrow x_1$, sum-check claims $C'_{\text{in}, l}$ hold (see Sec. 5.1.1). Then by completeness of ShiftSC as a polyIOR (see Lem. 25), this reduces to the claims $y_l = \tilde{s}_l(r'_N)$ for all $l \in [L]$. So the evaluation claims in the conditions of $\text{TE}(L, \ell, b)$ hold on the output (pp, x_2, w_2) . Since x_2 simply uses the same input commitments and openings $t_l, (s_{i,l})_{i \in [\ell]}$ for $l \in [L]$, the commitment opening conditions holds as well. Therefore, $(pp, x_2, w_2) \in \text{TE}(L, \ell, b)$. $\square$

Lemma 29. $\Pi_{\text{NC}}^{\text{PE}} : \text{PE}(L, \ell, b) \rightarrow \text{TE}(L, \ell, b) \ (\text{TE}^*(L, \ell))$ is a strong interactive reduction (Def. 7) with negligible knowledge error, where ϕ is the function which projects commitments from the instance.

Proof. We will prove both properties (i) and (ii) in the strong reduction definition in Def. 7.

(i) ϕ -restricted output. By construction, the verifier sets the commitment in the output instance to be the input commitments $(t_l)_{l \in [L]}$. Then for successful repeated executions of $\Pi_{\text{NC}}^{\text{PE}}$ outputting statements $x_2, x'_2, \phi(x_2) = \phi(x'_2) = (t_l)_{l \in [L]}$.

(ii) Extraction against a limited adversary. Consider an EPT adversary $(\mathcal{A}, \mathcal{P}^*)$ such that $\epsilon'(\mathcal{A}, \mathcal{P}^*) \geq 1/\text{poly}(\lambda)$ and letting $\mathcal{O}_{\text{str}}$ be the oracle from condition (ii) of strong interactive reductions (Def. 7) instantiated with $\mathcal{P}^*$, for any uniqueness adversary $\mathcal{S}_{\text{uniq}}^{\mathcal{O}_{\text{str}}}$ (definition 5),

$$\epsilon_{\text{uniq}} := \Pr \left[\begin{array}{c} w_2, w'_2 \neq \perp \\ \wedge \\ w_2 \neq w'_2 \end{array} \middle| \begin{array}{l} pp \leftarrow \mathcal{G}(1^\lambda) \\ (x_1, \text{st}) \leftarrow \mathcal{A}(pp) \\ \text{in} \leftarrow (pp, x_1, \text{st}) \\ (x_2, w_2), (x'_2, w'_2) \leftarrow \mathcal{S}_{\text{uniq}}^{\mathcal{O}_{\text{str}}(\text{in})}(\text{in}) \end{array} \right] \leq \text{negl}(\lambda) \quad (47)$$

we construct an expected polynomial-time extractor $\mathcal{E}$ in Fig. 8 such that

$$\Pr \left[(pp, x_1, w_1) \in \mathcal{R}_1 \middle| \begin{array}{l} pp \leftarrow \mathcal{G}(1^\lambda) \\ (x_1, \text{st}) \leftarrow \mathcal{A}(pp) \\ w_1 \leftarrow \mathcal{E}(pp, x_1, \text{st}) \end{array} \right] \geq \epsilon'(\mathcal{A}, \mathcal{P}^*) - \text{negl}(\lambda).$$

Extractor runtime. The extractor makes one call to $\mathcal{P}^*$ and otherwise performs polynomial-time operations. So the expected runtime is polynomial.

Extractor success probability. We will prove that that

$$\Pr \left[(pp, x_1, w_1) \in \text{PE}(L, \ell, b) \middle| \begin{array}{l} pp \leftarrow \mathcal{G}(1^\lambda) \\ (x_1, \text{st}) \leftarrow \mathcal{A}(pp) \\ w_1 \leftarrow \mathcal{E}(pp, x_1, \text{st}) \end{array} \right] \geq \epsilon'(\mathcal{A}, \mathcal{P}^*) - (\epsilon_{\text{uniq}} + \epsilon_{\text{NCPE}})$$

Extractor $\mathcal{E}(pp, x_1, st) \rightarrow w_1$:
1. Simulate $(x_2, w_2) \leftarrow \langle \mathcal{P}^*, \mathcal{V} \rangle(pp, x_1, st)$ with fresh randomness. 2. If $(pp, x_2, w_2) \notin \text{TE}^*(L, \ell)$, then output $\perp$. 3. Set $w_1 := w_2$. 4. Output w_1.

Figure 8: Extractor for $\Pi_{\text{NC}}^{\text{PE}}$.

where ϵ_{uniq} is the probability in Eq. (47) and

$$\epsilon_{\text{NCPE}} \leq \underbrace{(2Le - 1) \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right) + \frac{d/e - 1}{|\mathbb{K}|} + \frac{(\ell + 1 + m) \cdot 2}{|\mathbb{K}|}}_{\text{Claim 1}} + \underbrace{\frac{2b \cdot (\ell + 1 + m)}{|\mathbb{K}|} + \frac{Ld + \ell + m}{|\mathbb{K}|}}_{\text{Claim 2}}. \quad (48)$$

We invoke Lemma 27 with the following assignments:

- $\theta = (pp, x_1, st)$.
- The oracle $\mathcal{O}(\theta)$ runs the interaction $(x_2, w_2) \leftarrow \langle \mathcal{P}^*, \mathcal{V} \rangle(\theta)$ with fresh randomness and returns w_2 if $(pp, x_2, w_2) \in \text{TE}^*(L, \ell)$. Else, it returns $\perp$.
- $\text{Bad}(\theta)$ is the set of w_2 such that $(pp, x_1, w_2) \notin \text{PE}(L, \ell, b)$.

Observe that in the extractor in Fig. 8, $w_1 := w_2$ in the non-aborting case, and for $z \leftarrow \mathcal{O}(\theta)$,

$$z \neq \perp \wedge z \notin \text{Bad}(\theta) \implies (pp, x_1, w_1) \in \text{PE}(L, \ell, b).$$

And thus define the event

$$\text{Err}_1 := \{z \neq \perp \wedge z \in \text{Bad}(\theta)\} = \{z \in \text{Bad}(\theta)\}$$

because by definition of Bad , $z \neq \perp$. We can then bound the success probability of the extractor as desired:

$$\begin{aligned} \Pr[(pp, x_1, w_1) \in \text{PE}(L, \ell, b)] &\geq \Pr[z \neq \perp \setminus \text{Err}_1] \\ &= \mathbb{E}_\theta[\Pr[z \neq \perp \mid \theta]] - \mathbb{E}_\theta[\text{Err}_1 \mid \theta] \quad (\text{because } \text{Err}_1 \subseteq z \neq \perp, \text{ linearity of expectation}) \\ &\geq \epsilon'(\mathcal{A}, \mathcal{P}^*) - \mathbb{E}_\theta[\delta(\theta)] - \text{Adv}^{\text{uniq}}(\mathcal{S}_{\text{uniq}}). \end{aligned} \quad (\text{def. of } \epsilon', \text{ Lem. 27})$$

where $\delta, \text{Adv}^{\text{uniq}}(\mathcal{S}_{\text{uniq}})$ are defined as in Lem. 27. It remains to bound $\text{Adv}(\mathcal{S}_{\text{uniq}})$ in Lem. 27 and $\mathbb{E}_\theta[\delta(\theta)]$ to complete use of Lem. 27. By assumption in property (ii) of Def. 7, we have that

$$\text{Adv}(\mathcal{S}_{\text{uniq}}) = \epsilon_{\text{uniq}}.$$

Now to bound $\mathbb{E}_\theta[\delta(\theta)]$, for a fixed θ and w_2 , parse

$$\begin{aligned} &((t_l)_{l \in [L]}, u, (v_l)_{l \in [L]}) \leftarrow x_1 \\ &((t_l)_{l \in [L]}, r, (y_l)_{l \in [L]}), (s_{i,l})_{i \in [\ell], l \in [L]} \leftarrow (x_2, w_2). \end{aligned}$$

By definition of $\text{TE}^*(L, \ell)$, it must hold that

$$\begin{aligned} \forall l \in [L] \quad &\text{Open}(A, |\mathbb{F}|/2, t_l, (s_{i,l})_{i \in [\ell]}) \\ &s'_l := 0^{2n\alpha} \|s_{1,l}\| \cdots \|s_{\ell,l}\| \\ &\tilde{s}_l(r) = y_l \end{aligned}$$

Because $w_2 \in \text{Bad}(\theta)$, it holds that $(pp, x_1, w_1 := w_2) \notin \text{PE}(L, \ell, b)$. The definition $\text{PE}(L, \ell, b)$ implies that either

(E1) There exists $l \in [L], i \in [\ell]$ such that $\|s_{i,l}\|_\infty \geq b_l$.

(E2) OR there exists $l \in [L]$ such that $\tilde{s}_{\ell,l}(u) \neq v_l$.

We will now show that

$$\mathbb{E}_\theta[\delta(\theta)] \leq \epsilon_{\text{NCPE}}$$

is bounded by at least one of these events occurring.

Claim 1 (E2): Suppose the second event occurs, i.e. that there exists $l \in [L]$ such that $\tilde{s}_{\ell,l}(u) \neq v_l$. Recall the sum-check claims from Step 3 that are fed into ShiftSC.

$$\begin{aligned} \forall l \in [L], C_{\text{sc},l} : a_l &\stackrel{?}{=} \sum_{x \in \{0,1\}^{\ell+1+m}} \tilde{e}q(r_N, x) \cdot \tilde{s}_l(x) \in R_{\mathbb{K}} \\ C_{\text{in},l} : v_l &\stackrel{?}{=} \sum_{x \in \{0,1\}^{\ell+1+m}} \tilde{e}q(x_1, 1) \cdot \tilde{e}q(u, x_{[2:]}) \cdot \tilde{s}_l(x) \in R_{\mathbb{F}} \end{aligned}$$

For each $l \in [L]$,

$$\begin{aligned} C_{\text{sc},l} &\text{ if and only if } \tilde{s}_l(r_N) = a_l \\ C_{\text{in},l} &\text{ if and only if } \tilde{s}_{\ell,l}(u) = v_l \end{aligned}$$

as described in Sec. 5.1.1. The sum-check claims $C_{\text{sc},l}, C_{\text{in},l}$ all hold if and only if $\tilde{s}_l(r_N) = a_l$ and $\tilde{s}_{\ell,l}(u) = v_l$ for all $l \in [L]$. If $\mathcal{P}^*$ passes ShiftSC $((C_{j,l})_{j \in \{\text{sc}, \text{in}, l \in [L]\}})$ despite at least one of these sum-check claims not holding (iff corresponding $\tilde{s}_{\ell,l}(u) \neq v_l$ or $\tilde{s}_l(r_N) \neq a_l$), this violates the soundness of ShiftSC (Lem. 26), and thus occurs with probability

$$\leq (2Le - 1) \cdot \left(\frac{1}{|C|} + \epsilon_{\text{inv}}(C) \right) + \frac{d/e - 1}{|\mathbb{K}|} + \frac{(\ell + 1 + m) \cdot 2}{|\mathbb{K}|}$$

for a ShiftSC invocation on $2L$ claims of maximum degree 2 and maximum variable count $\ell + 1 + m$.

Claim 2 (E1): First note that by Claim 1, it is also the case that the claims $\tilde{s}_l(r) = a_l$ sent by $\mathcal{P}^*$ in Step 2 are checked by the ShiftSC call in Step 3, up to soundness error of Claim 1. Now suppose that there exists $l \in [L], i \in [\ell]$ such that $\|s_{i,l}\|_\infty \geq b_l$. Recall the sum-check claim in Step 1:

$$\begin{aligned} 0 &\stackrel{?}{=} \sum_{X \in \{0,1\}^{\ell+1+m}} Q(X, C, D) \\ \text{for } Q(X, C, D) &:= \tilde{e}q(X, D) \cdot Q_N(X, C) \\ Q_N(X) &:= \sum_{l \in [L]} \sum_{k \in [0, d-1]} C^{(l-1) \cdot d + k} \cdot N_{b_l}(\widetilde{\text{cf}(s_l)_k}(X)) \in \mathbb{K}[X_{[1:\ell+1+m]}] \end{aligned}$$

where challenges $\xi \leftarrow_{\$} \mathbb{K}, r \leftarrow_{\$} \mathbb{K}^{\ell+1+m}$ are replaced by indeterminates C, D , respectively. Observe that

$$0 = \sum_{X \in \{0,1\}^{\ell+1+m}} Q(X, C, D) \quad \text{if and only if} \quad \forall l \in [L], \|s_{i,l}\|_\infty < b_l$$

because $\|s_{i,l}\|_\infty < b_l$ if and only if $N_{b_l}(\widetilde{\text{cf}(s_l)_k}(X)) \equiv 0$ on the Boolean hypercube, which is checked by a zero-check sum-check claim, as described in Sec. 5.1.1. The powers of C are linearly independent, and so the sum-check claim holds over indeterminates if and only if the norm condition is satisfied for all $l \in [L]$.

If there exists $l \in [L], i \in [\ell]$ such that $\|s_{i,l}\|_\infty \geq b_l$, as in E1 and the sum-check verifier in Step 1 passed, it must be that either $\mathcal{P}^*$

- succeeded in the sum-check protocol SC, despite $0 \neq \sum_{X \in \{0,1\}^{\ell+1+m}} Q(X, C, D)$
- OR the polynomial $P(X, C, D) = \sum_{X \in \{0,1\}^{\ell+1+m}} Q(X, C, D)$ evaluated to zero on random challenges $\xi \leftarrow_{\$} \mathbb{K}, r \leftarrow_{\$} \mathbb{K}^{\ell+1+m}$.

By the soundness of the sum-check protocol SC (see Lem. 3), the first event occurs with probability

$$\leq \frac{2b \cdot (\ell + 1 + m)}{|\mathbb{K}|}$$

By the Schwartz-Zippel lemma on $\mathbb{K}$ (see Lem. 1), the second event occurs with probability

$$\leq \frac{Ld + \ell + m}{|\mathbb{K}|}$$

where $Ld - 1$ is the degree of ξ in P , and the degree of the entries of r is 1. Union bounding over the two events, we have Claim 2 as in Eq. (48). Invoking Lem. 27, we have shown property (ii). $\square$

B.4 From Section 5.2: Folding

In this section, we first state and prove a modified Thm. 5 which guarantees that for each coordinate $i \in [\ell]$, the difference of challenges $(c_{i,0} - c_{i,i})$ is invertible. Thm. 5 is stated for arbitrary finite challenge sets, but we instantiate our challenge sets with both exact and approximate strong sampling sets (see Def. 10), where the latter does not guarantee invertibility. Second, we prove that Π_{Fold} is complete and a weak interactive reduction.

Lemma 30 (Coordinate-wise extraction for approximate strong sampling sets). *Let $\ell \in \mathbb{N}$, and $\mathcal{C} \subseteq R_{\mathbb{F}}$ be an approximate strong sampling set (see Def. 10) of cardinality N with $\max_{a \in \mathcal{C}} \Pr_{b \leftarrow \mathcal{C} \setminus \{a\}}[a - b \notin R_{\mathbb{F}}^{\times}] = \epsilon_{\text{inv}}$. Take any verification function $V : \mathcal{C}^{\ell} \times \{0,1\}^* \rightarrow \{0,1\}$. Then there exists an oracle algorithm $\mathcal{E}$ with the following properties: the algorithm $\mathcal{E}^{\mathcal{A}}$, given oracle access to a (probabilistic) algorithm $\mathcal{A} : \mathcal{C}^{\ell} \rightarrow \{0,1\}^*$, requires an expected number of at most $\ell + 1$ queries to $\mathcal{A}$ and with probability at least*

$$\epsilon^V(\mathcal{A}) - \ell \cdot \left(\frac{1}{N} + \epsilon_{\text{inv}} \right)$$

outputs $\ell + 1$ pairs $(\vec{c}_0, y_0), (\vec{c}_1, y_1), \dots, (\vec{c}_{\ell}, y_{\ell})$ such that

- $V(c_i, y_i) = 1$ for all $i \in [0, \ell]$,
- and $\{\vec{c}_0, \dots, \vec{c}_{\ell}\}$ satisfy the coordinate-wise property in Def. 16,
- and for all $i \in [\ell]$, $(c_{0,i} - c_{i,i}) \in R_{\mathbb{F}}^{\times}$.

Note that if $\mathcal{C}$ is an exact strong sampling set, this lemma is identical to [FMN24, Lemma 7.1] (Thm. 5).

Proof. We modify the extractor of Lemma 7.1 of [FMN24] in Fig. 9 to abort in Step 4(e) when the difference of challenges is non-invertible.

Extractor runtime. The number of queries made by the extractor in Fig. 9 is upper bounded by the number of queries made by the extractor in Lemma 7.1 of [FMN24], as the only change is an additional abort condition in Step 4(e) which does not increase the number of queries made.

Extractor success probability. We denote by $C_0 := (C_{0,1}, \dots, C_{0,\ell})$ the random variable for the first challenge sampled by $\mathcal{E}$. Also, define the random variable for V 's acceptance on this first challenge to be $\Gamma := V(C_0, \mathcal{A}(C_0))$, and observe that $\Pr[\Gamma = 1] = \epsilon^V(\mathcal{A})$. Define the random variable s_i :

$$S_i := \{x \in \mathcal{C} : V(C_i(x), \mathcal{A}(C_i(x))) = 1\},$$

$$s_i := |S_i|$$

where $C_i(x) := (C_{0,1}, \dots, C_{0,i-1}, x, C_{0,i+1}, \dots, C_{0,\ell})$. Given $\Gamma = 1$, Fail $_i$ to be the event that the extractor fails at coordinate i (if it were to reach coordinate i in the Step 4 loop). This can be split into two cases

- $\Gamma = 1$, but for some $i \in [\ell]$, $s_i \leq 1$,
- $\Gamma = 1$ and $s_i \geq 2$, but the first selected c_i^* in Step 4(e) is such that $(c_{0,i} - c_i^*) \notin R_{\mathbb{F}}^{\times}$.

Extractor $\mathcal{E}^{\mathcal{A}}$
1. Sample $\vec{c}_0 := (c_{0,1}, \dots, c_{0,\ell}) \leftarrow_{\S} \mathcal{C}^\ell$. 2. $y_0 \leftarrow \mathcal{A}(\vec{c}_0)$. 3. If $V(\vec{c}_0, y_0) = 0$, abort. 4. For $i = 1, \dots, \ell$, repeat: (a) Sample $c_i^* \leftarrow \mathcal{C} \setminus \{c_{0,i}\}$ without replacement. (b) $\vec{c}_i := (c_{0,1}, \dots, c_{0,i-1}, c_i^*, c_{0,i+1}, \dots, c_{0,\ell})$. (c) $y_i \leftarrow \mathcal{A}(\vec{c}_i)$. (d) If $V(c_i, y_i) = 0$, go to Step 4(a). (e) Else, if $(c_{0,i} - c_i^*) \notin R_{\mathbb{R}}^\times$, abort. until another successful c_i^* has been found (increment i to continue loop), or until all c_i^* have been tried (in which case, abort). 5. Return the corresponding $(\vec{c}_0, y_0), (\vec{c}_i, y_i)_{i \in [\ell]}$.

Figure 9: Extractor for Lemma 30.

The success probability of $\mathcal{E}^{\mathcal{A}}$ is:

$$\begin{aligned}
\Pr[\Gamma = 1 \wedge (\bigwedge_{i=1}^{\ell} \text{Fail}_i^c)] &= \Pr[\Gamma = 1] - \Pr[\Gamma = 1 \wedge (\bigvee_{i=1}^{\ell} \text{Fail}_i)] && \text{(De Morgan's laws)} \\
&\geq \Pr[\Gamma = 1] - \sum_{i=1}^{\ell} \Pr[\Gamma = 1 \wedge \text{Fail}_i]. && \text{(distributive property, union bound)}
\end{aligned} \tag{49}$$

For a fixed $i \in [\ell]$, we now upper bound $\Pr[\Gamma = 1 \wedge \text{Fail}_i]$ by partitioning the event into the two cases: $s_i \leq 1$ and $s_i \geq 2$.

$s_i \leq 1$: If $s_i = 0$, extraction always fails ($\Gamma = 0$). If $s_i = 1$, the event $\Gamma = 1 \wedge \text{Fail}_i$ occurs when the first challenge sampled is exactly the single challenge in S_i . This happens with probability $1/N$:

$$\begin{aligned}
\Pr[\Gamma = 1 \wedge \text{Fail}_i \mid s_i \leq 1] &= \underbrace{\Pr[\Gamma = 1 \wedge \text{Fail}_i \mid s_i = 0]}_0 \cdot \Pr[s_i = 0 \mid s_i \leq 1] \\
&\quad + \Pr[\Gamma = 1 \wedge \text{Fail}_i \mid s_i = 1] \cdot \underbrace{\Pr[s_i = 1 \mid s_i \leq 1]}_{\leq 1} \\
&\leq \Pr[\Gamma = 1 \wedge \text{Fail}_i \mid s_i = 1] = \frac{1}{N}.
\end{aligned} \tag{50}$$

$s_i \geq 2$: In this case, it is guaranteed that at some point Step 4(d) will succeed and $V(c_i, y_i) = 1$; i.e., the loop will reach some $c_i^* \in S_i$. Then, the event Fail_i occurs exactly when $(c_{0,i} - c_i^*) \notin R_{\mathbb{R}}^\times$. Define the bad set of challenges for a challenge $a \in \mathcal{C}$, which create a non-invertible difference:

$$\text{Bad}_a(\mathcal{C}) = \{b \in \mathcal{C} : a \neq b \wedge a - b \notin R_{\mathbb{R}}^\times\} \subseteq \mathcal{C}.$$

Now we can bound the probability of extraction failure at coordinate i for a fixed S such that $s = |S| \geq 2$,

$$\Pr[\Gamma = 1 \wedge \text{Fail}_i \mid S_i = S]$$

$$\leq \sum_{a \in S} \underbrace{\Pr[C_{0,i} = a \mid S_i = S]}_{1/N} \cdot \Pr_{c_i^* \leftarrow S \setminus \{a\}}[a - c_i^* \notin R_{\mathbb{F}}^\times \mid C_{0,i} = a, S_i = S]$$

summing across S , the set of accepting challenges. Observe that c_i^* is the first element drawn from $S \setminus \{a\}$ in the loop.

$$= \sum_{a \in S} \frac{1}{N} \cdot \frac{|\{b \in S \setminus \{a\} \text{ s.t. } a - b \notin R_{\mathbb{F}}^\times\}|}{s-1}$$

$$= \sum_{a \in S} \frac{\min\{s-1, |\text{Bad}_a(\mathcal{C})|\}}{N \cdot (s-1)}$$

by definition of $\text{Bad}_a(\mathcal{C})$, and because $|S \setminus \{a\}| = s-1$,

$$\leq \frac{s \cdot \min\{s-1, \max_{a \in \mathcal{C}} |\text{Bad}_a(\mathcal{C})|\}}{N \cdot (s-1)} = \frac{s}{N} \cdot \min\left\{1, \frac{\max_{a \in \mathcal{C}} |\text{Bad}_a(\mathcal{C})|}{(s-1)}\right\} \leq \frac{\max_{a \in \mathcal{C}} |\text{Bad}_a(\mathcal{C})| + 1}{N}$$

Where the last inequality is true by the following casework. Consider the two cases: $s-1 \leq \max_{a \in \mathcal{C}} |\text{Bad}(\mathcal{C})|$ which selects the first entry of the min expression, and $s-1 > \max_{a \in \mathcal{C}} |\text{Bad}(\mathcal{C})|$ which selects the second entry.

$$s-1 \leq \max_{a \in \mathcal{C}} |\text{Bad}(\mathcal{C})| \implies \frac{s}{N} \leq \frac{\max_{a \in \mathcal{C}} |\text{Bad}(\mathcal{C})| + 1}{N}$$

$$s-1 > \max_{a \in \mathcal{C}} |\text{Bad}(\mathcal{C})| \implies \frac{s \cdot \max_{a \in \mathcal{C}} |\text{Bad}(\mathcal{C})|}{N \cdot (s-1)} = \frac{\max_{a \in \mathcal{C}} |\text{Bad}(\mathcal{C})|}{N} \cdot \left(1 + \frac{1}{s-1}\right) < \frac{\max_{a \in \mathcal{C}} |\text{Bad}(\mathcal{C})| + 1}{N}$$

where the last inequality holds because $\frac{\max_{a \in \mathcal{C}} |\text{Bad}(\mathcal{C})|}{s-1} < 1$. By definition of local approximate strong sampling set (Def. 10),

$$\max_{a \in \mathcal{C}} |\text{Bad}_a(\mathcal{C})| \leq \epsilon_{\text{inv}} \cdot (N-1).$$

And so averaging over S such that $|S| = s \geq 2$,

$$\Pr[\Gamma = 1 \wedge \text{Fail}_i \mid s_i \geq 2] \leq \frac{\epsilon_{\text{inv}} \cdot (N-1) + 1}{N} \quad (51)$$

Putting all these cases together and union bounding across $i \in [\ell]$, we lower bound the success probability of extractor $\mathcal{E}^{\mathcal{A}}$:

$$\begin{aligned} \Pr[\Gamma = 1 \wedge (\bigwedge_{i=1}^{\ell} \text{Fail}_i^c)] &\geq \Pr[\Gamma = 1] - \sum_{i=1}^{\ell} \Pr[\Gamma = 1 \wedge \text{Fail}_i] && \text{(by Eq. (49))} \\ &= \Pr[\Gamma = 1] - \sum_{i=1}^{\ell} \left(\Pr[\Gamma = 1 \wedge \text{Fail}_i \mid s_i \leq 1] \cdot \Pr[s_i \leq 1] \right. \\ &\quad \left. + \Pr[\Gamma = 1 \wedge \text{Fail}_i \mid s_i \geq 2] \cdot \Pr[s_i \geq 2] \right) \\ &\geq \Pr[\Gamma = 1] - \sum_{i=1}^{\ell} \left(\frac{1}{N} \cdot \Pr[s_i \leq 1] + \frac{\epsilon_{\text{inv}} \cdot (N-1) + 1}{N} \cdot \Pr[s_i \geq 2] \right) && \text{(by Eq. (50), (51))} \\ &\geq \Pr[\Gamma = 1] - \sum_{i=1}^{\ell} \frac{\epsilon_{\text{inv}} \cdot (N-1) + 1}{N} \cdot \underbrace{(\Pr[s_i \leq 1] + \Pr[s_i \geq 2])}_1 \\ &\geq \Pr[\Gamma = 1] - \ell \cdot \frac{\epsilon_{\text{inv}} \cdot (N-1) + 1}{N} \\ &\geq \epsilon^V(\mathcal{A}) - \ell \cdot \left(\frac{1}{N} + \epsilon_{\text{inv}} \right). \end{aligned}$$

□

Lemma 31 (Π_{Fold} completeness). *Let $B = 2^k \cdot T \cdot b$. $\Pi_{\text{Fold}} : \text{TE}(1, \ell, b) \rightarrow \text{TE}(1, k, b) \times \text{TE}(1, \ell - k, B) \times \text{PE}(1, k, b)$ is complete for with a prover time consisting of a $O(2^{\ell-k+1+m})$ sum-check prover and 2 tree commitment generations, $O(2^k \cdot (\ell^2 + \ell m))$ verifier time, and proof size consisting of 2 BatchSC with maximum degree 2 outputting 2 total claims, 4 additional $R_{\mathbb{F}}$ claims, and 2 commitments.*

Proof. Let $x := (t, u, v)$; $w := (s_i)_{i \in [\ell]} \in \text{TE}(1, \ell, b)$. For an honest execution of Π_{Fold} , the verifier in Step 1 passes by the derivation ending in Eq. (12). By the completeness of BatchSC (Prot. 1) and the correctness of NTT^{-1} as a ring isomorphism, the verifier in Step 2 passes and reduces to the corresponding evaluation claim on $r \leftarrow \text{NTT}^{-1}(h_{\mathbb{K}}, \dots, h_{\mathbb{K}})$. Because $t_{\text{subs}}, y_{\text{subs}}$, and t^* are all computed honestly, by the completeness of BatchSC, the verifier in Step 4 passes, and reduces to the corresponding evaluation claims on r' .

$((t, r', y_{\text{top}}); (s_i)_{i \in [k]}) \in \text{TE}(1, k, b)$, because the commitment opening conditions of $\text{TE}(1, k, b)$ are also satisfied in the input relation. The evaluation claim y_{top} on r' holds by the completeness of BatchSC.

$((t_{\text{subs}}, r'^{(\cdot:k+m)}, y_{\text{subs},k}); (s_{\text{subs},i})_{i \in [k]}) \in \text{PE}(1, k, b)$ because t_{subs} is an honest fresh commitment to $(s_{\text{subs},i})_{i \in [k]}$ and the completeness of BatchSC. What remains to show is that

$$((t^*, r, v_{\text{bot}}); (s_i^*)_{i \in [\ell-k]}) \in \text{TE}(1, \ell - k, B)$$

In particular, we will show that

$$\begin{aligned} t^* &= A \cdot s_1^* \\ G_{2^{i_n}} \cdot s_i^* &= (I_{2^i} \otimes A) \cdot s_{i+1}^* \quad \forall i \in [\ell - k - 1] \\ \|s_i^*\|_{\infty} &< B \quad \forall i \in [\ell - k] \\ \tilde{s}_{\text{bot}}(r) &= v_{\text{bot}} \end{aligned}$$

where $B = 2^k \cdot T \cdot b$. Recall that an honest execution of Π_{Fold} ,

$$\begin{aligned} t^* &:= \sum_{j \in [2^k]} c_j \cdot (G_n \cdot \mathfrak{s}_{k,j}), \\ s_i^* &:= \sum_{j \in [2^k]} c_j \cdot \mathfrak{s}_{k+i,j} \quad \forall i \in [\ell - k], \end{aligned}$$

as in Eq. (11). To prove the first condition,

$$t^* = \sum_{j=1}^{2^k} c_j \cdot (G_n \cdot \mathfrak{s}_{k,j}) = \sum_{j=1}^{2^k} c_j \cdot (A \cdot \mathfrak{s}_{k+1,j}) = A \cdot \sum_{j=1}^{2^k} c_j \cdot \mathfrak{s}_{k+1,j} = A \cdot s_1^*$$

where the second equality is due to the tree commitment structure of $(t, (s_i)_{i \in [\ell]})$ and chunking of s_{k+i} into $\mathfrak{s}_{k+i,1} \parallel \dots \parallel \mathfrak{s}_{k+i,2^k}$. For the second condition, we similarly derive

$$\begin{aligned} G_{2^{i_n}} \cdot s_i^* &= G_{2^{i_n}} \cdot \sum_{j=1}^{2^k} c_j \cdot \mathfrak{s}_{k+i,j} && \text{(def. of } s_i^*) \\ &= \sum_{j=1}^{2^k} c_j \cdot (G_{2^{i_n}} \cdot \mathfrak{s}_{k+i,j}) && \text{(linearity)} \\ &= \sum_{j=1}^{2^k} c_j \cdot ((I_{2^i} \otimes A) \cdot \mathfrak{s}_{k+i+1,j}) && \text{(commitment structure of } t) \\ &= (I_{2^i} \otimes A) \cdot \sum_{j=1}^{2^k} c_j \cdot \mathfrak{s}_{k+i+1,j} = (I_{2^i} \otimes A) \cdot s_{i+1}^* && \text{(linearity, def. of } s_{i+1}^*) \end{aligned}$$

For the third condition, $\mathcal{P}$ computes

$$v_{\text{bot}} := \sum_{j=1}^{2^k} 2^k c_j \cdot \tilde{s}_j(r).$$

Recall that by Eq. (11), $s_{\text{bot}} = \sum_{j=1}^{2^k} c_j \cdot \mathfrak{s}_j$. By linearity, $\tilde{s}_{\text{bot}}(r) = v_{\text{bot}}$.
 Finally, for the fourth condition

$$\begin{aligned} \|s_i^*\|_\infty &= \left\| \sum_{j=1}^{2^k} c_j \cdot \mathfrak{s}_{k+i,j} \right\|_\infty \\ &\leq \sum_{j=1}^{2^k} \|c_j \cdot \mathfrak{s}_j\|_\infty && \text{(triangle inequality)} \\ &\leq 2^k \cdot T \cdot b \end{aligned}$$

where the expansion factor (Def 15) of $\mathcal{C}$ is T . □

Lemma 32 (Π_{Fold} is a weak reduction). *Assuming $\text{MSIS}_{q,n,d,rn\alpha,4 \cdot T_{\text{bck}} \cdot B}$ where T_{bck} is the expansion factor of $\mathcal{C} - \mathcal{C}$ (Def. 15), $\Pi_{\text{Fold}} : \text{TE}(1, \ell, b) \text{ (TE}^*(1, \ell)) \rightarrow \text{TE}(1, k, b) \times \text{TE}(1, \ell - k, B) \times \text{PE}(1, k, b)$ is a weak interactive reduction (Def. 6) with negligible knowledge error ϵ_{Fold} , where ϕ is the function which projects commitments from the instance $\phi(x_1) = \phi(t, u, v) = t$.*

Proof. We will prove knowledge properties (i) and (ii) of a weak interactive reduction in Def. 6. For modularity, we split Π_{Fold} into 3 subprotocols and define the corresponding intermediate relations $\Pi_{\text{Fold}} = \Pi_{\text{Fold}}^{(3)} \circ \Pi_{\text{Fold}}^{(2)} \circ \Pi_{\text{Fold}}^{(1)}$.

1. $\Pi_{\text{Fold}}^{(1)} : \text{TE}(1, \ell, b) \rightarrow \mathcal{R}_1$ is the interaction in Step 1-2 (the first partial BatchSC), ending after the verifier check on y_{subs} .

$$\mathcal{R}_1 := \left\{ \begin{array}{l} \mathbf{x} := \left(\begin{array}{l} t \in R_{\mathbb{F}}^n, u \in R_{\mathbb{F}}^{\ell+1+m}, v_{\text{top}} \in R_{\mathbb{F}}, \\ t_{\text{subs}} \in R_{\mathbb{F}}^n, r \in R_{\mathbb{F}}^{\ell-k+1+m}, y_{\text{subs}} \in R_{\mathbb{F}}, \\ (q_{\text{subs},j} \in R_{\mathbb{F}}^{2^{\ell-k+1}n\alpha})_{j \in [2^k]} \end{array} \right), \\ \mathbf{w} := \left(\begin{array}{l} (s_i \in R_{\mathbb{F}}^{2^i n\alpha})_{i \in [\ell]}, \\ (s_{\text{subs},i} \in R_{\mathbb{F}}^{2^i n\alpha})_{i \in [k]} \end{array} \right) \end{array} \right\} : \left\{ \begin{array}{l} \text{Open}(A, b, t, (s_i)_{i \in [\ell]}) = 1 \\ \text{Open}(A, b, t_{\text{subs}}, (s_{\text{subs},i})_{i \in [k]}) = 1 \\ \forall i \in [0, \ell - k], \\ \quad \mathfrak{s}_{k+i,1} \in R_{\mathbb{F}}^{2^i n\alpha} \parallel \dots \parallel \mathfrak{s}_{k+i,2^k} =: s_{k+i} \\ \forall j \in [2^k], \\ \quad \mathfrak{s}_j := 0^{2n\alpha} \parallel \mathfrak{s}_{k+1,j} \parallel \dots \parallel \mathfrak{s}_{\ell,j} \in R_{\mathbb{F}}^{2^{\ell-k+1}n\alpha} \\ s_{\text{top}} := 0^{2n\alpha} \parallel s_1 \parallel \dots \parallel s_k \in R_{\mathbb{F}}^{2^{k+1}n\alpha} \\ f_{\text{subs}} := G_{2^k n} \cdot s_{\text{subs},k} \in R_{\mathbb{F}}^{2^k n} \\ v_{\text{top}} = \tilde{s}_{\text{top}}(u^{(\ell-k+1)}) \\ \forall j \in [2^k], f_{\text{subs}}^{(j)} = \tilde{\mathfrak{s}}_j(r) \\ y_{\text{subs}} = \sum_{j=1}^{2^k} \tilde{q}_{\text{subs},j}(r) \cdot f_{\text{subs}}^{(j)} \end{array} \right\}$$

2. $\Pi_{\text{Fold}}^{(2)} : \mathcal{R}_1 \rightarrow \mathcal{R}_2$ is the interaction in Step 3 (the bottom subtree commitment folding via verifier challenges),

ending after the prover sends v_{bot} .

$$\mathcal{R}_2 := \left\{ \begin{array}{l} \mathbf{x} := \left(\begin{array}{l} t \in R_{\mathbb{F}}^n, u \in R_{\mathbb{F}}^{\ell+1+m}, v_{\text{top}} \in R_{\mathbb{F}}, \\ t_{\text{subs}} \in R_{\mathbb{F}}^n, r \in R_{\mathbb{F}}^{\ell-k+1+m}, y_{\text{subs}} \in R_{\mathbb{F}}, \\ (q_{\text{subs},j} \in R_{\mathbb{F}}^{2^{\ell-k+1}n\alpha})_{j \in [2^k]}, \\ t^* \in R_{\mathbb{F}}^n, v_{\text{bot}} \in R_{\mathbb{F}}, c \in \mathcal{C}^{2^k} \end{array} \right), \\ \mathbf{w} := \left(\begin{array}{l} (s_i \in R_{\mathbb{F}}^{2^i n\alpha})_{i \in [k]}, \\ (s_i^* \in R_{\mathbb{F}}^{2^i n\alpha})_{i \in [\ell-k]}, \\ (s_{\text{subs},i} \in R_{\mathbb{F}}^{2^i n\alpha})_{i \in [k]} \end{array} \right) \end{array} \right\} : \left\{ \begin{array}{l} \text{Open}(A, b, t, (s_i)_{i \in [k]}) = 1 \\ \text{Open}(A, b, t_{\text{subs}}, (s_{\text{subs},i})_{i \in [k]}) = 1 \\ ((t^*, r, v_{\text{bot}}), (s_i^*)_{i \in [\ell-k]}) \in \text{TE}(1, \ell-k, B) \\ s_{\text{top}} := 0^{2n\alpha} \|s_1\| \cdots \|s_k\| \in R_{\mathbb{F}}^{2^{k+1}n\alpha} \\ f_{\text{subs}} := G_{2^k n} \cdot s_{\text{subs},k} \in R_{\mathbb{F}}^{2^k n} \\ \mathfrak{s}_{k,1} \in R_{\mathbb{F}}^{n\alpha} \| \cdots \| \mathfrak{s}_{k,2^k} \in R_{\mathbb{F}}^{n\alpha} =: s_k \\ v_{\text{top}} = \tilde{s}_{\text{top}}(u^{(\ell-k+1)}) \\ y_{\text{subs}} = \sum_{j=1}^{2^k} \tilde{q}_{\text{subs},j}(r) \cdot f_{\text{subs}}^{(j)} \\ v_{\text{bot}} = \sum_{j=1}^{2^k} c_j^{(j)} \cdot f_{\text{subs}}^{(j)} \\ t^* = \sum_{j=1}^{2^k} c_j \cdot (G_n \cdot \mathfrak{s}_{k,j}) \end{array} \right.$$

3. $\Pi_{\text{Fold}}^{(3)} : \mathcal{R}_2 \rightarrow \text{TE}(1, k, b) \times \text{TE}(1, \ell-k, B) \times \text{PE}(1, k, b)$ is the rest of interaction in Step 4-5 (the second BatchSC).

(i) *Relaxed extraction.* Consider an arbitrary expected polynomial time adversary $(\mathcal{A}, \mathcal{P}^*)$ for Π_{Fold} with success probability $\epsilon(\mathcal{A}, \mathcal{P}^*) \geq 1/\text{poly}(\lambda)$. We claim that there exists an EPT extractor $\mathcal{E}$ such that

$$\Pr \left[(pp, x_1, w_1) \in \text{TE}^*(1, \ell) \mid \begin{array}{l} pp \leftarrow \mathcal{G}(1^\lambda) \\ (x_1, \text{st}) \leftarrow \mathcal{A}(pp) \\ w_1 \leftarrow \mathcal{E}(pp, x_1, \text{st}) \end{array} \right] \geq \epsilon(\mathcal{A}, \mathcal{P}^*) - \epsilon_{\text{Fold}}$$

where

$$\begin{aligned} \epsilon_{\text{Fold}} \leq & \underbrace{\left(3 \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right) + \frac{d/e-1}{|\mathbb{K}|} + \frac{(k+1+m) \cdot 2}{|\mathbb{K}|} + (n-1) \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right) \right)}_{\text{Claim 3}} + 2 \cdot \text{Adv}_A^{\text{MSIS}_{q,n,d,2n\alpha,2b}}(\mathcal{B}_1, \lambda) \\ & + \underbrace{2^k \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right) + 2 \cdot \text{Adv}_A^{\text{MSIS}_{q,n,d,2n\alpha,2b}}(\mathcal{B}_1, \lambda)}_{\text{Claim 2}} \\ & + \underbrace{\frac{d/e-1}{|\mathbb{K}|} + \frac{(\ell-k+1+m) \cdot 2}{|\mathbb{K}|} + \text{Adv}_A^{\text{MSIS}_{q,n,d,rn\alpha,4 \cdot T_{\text{bck}} \cdot B}}(\mathcal{B}_2, \lambda) + \text{Adv}_A^{\text{MSIS}_{q,n,d,2n\alpha,2b}}(\mathcal{B}_1, \lambda)}_{\text{Claim 1}} \end{aligned} \quad (52)$$

We make the following claims on $\Pi_{\text{Fold}}^{(1)}$, $\Pi_{\text{Fold}}^{(2)}$, $\Pi_{\text{Fold}}^{(3)}$ and then show to how to compose them to construct $\mathcal{E}$ with knowledge error ϵ_{Fold} . Then we will prove each claim individually.

Claim 1: The following claim about $\Pi_{\text{Fold}}^{(1)}$ is very close to a strong interactive reduction (Def. 7), except that we allow for extraction of a witness for a relaxed relation. And due to relaxed extraction, we also claim the relaxed binding property in Def. 6 property (ii) on $\Pi_{\text{Fold}}^{(1)}$.

Consider the interactive reduction $\Pi_{\text{Fold}}^{(1)} : \text{TE}(1, \ell, b) \times \text{TE}^*(1, \ell) \rightarrow \mathcal{R}_1 \times (\mathcal{R}_1^*(k, b))$ where $\mathcal{R}_1^*(k, b)$ additionally contains slack factors in the witness

$$w := \left(\begin{array}{l} (s_i \in R_{\mathbb{F}}^{2^i n\alpha})_{i \in [\ell]}, \\ (s_{\text{subs},i} \in R_{\mathbb{F}}^{2^i n\alpha})_{i \in [k]}, \\ \Delta \in ((\mathcal{C} - \mathcal{C}) \cap R_{\mathbb{F}}^\times)^{2^k} \end{array} \right)$$

replaces the first two conditions in $\mathcal{R}_1$ with

$$\begin{aligned} \text{Open}(A, |\mathbb{F}|/2, t, (s_i)_{i \in [\ell]}) &= 1 \\ \text{Open}(A, |\mathbb{F}|/2, t_{\text{subs}}, (s_{\text{subs}, i})_{i \in [k]}) &= 1 \\ \forall i \in [k], \quad \|s_i\|_\infty &< b \\ \forall i \in [\ell - k], j \in [2^k], \quad \|\Delta_j \cdot \mathfrak{s}_{k+i, j}\|_\infty &< 2B. \end{aligned}$$

(i) For any EPT adversary $(\mathcal{A}, \mathcal{P}_1^*)$,

$$\Pr \left[\begin{array}{c|c} x_2, x'_2 \neq \perp & pp \leftarrow \mathcal{G}(1^\lambda) \\ \downarrow & (x_1, \text{st}_1) \leftarrow \mathcal{A}(pp) \\ \phi_2(x_2) = \phi_2(x'_2) & (x_2, w_2) \leftarrow \langle \mathcal{P}_1^*, \mathcal{V}_1 \rangle(pp, x_1, \text{st}_1) \\ & (x'_2, w'_2) \leftarrow \langle \mathcal{P}_1^*, \mathcal{V}_1 \rangle(pp, x_1, \text{st}_1) \end{array} \right] = 1$$

where $\phi_2(x_2)$ is the function which projects t from the instance.

(ii) For any EPT adversary $(\mathcal{A}, \mathcal{P}_1^*)$, there exists an EPT extractor $\mathcal{E}_1$ such that if

$$\epsilon(\mathcal{A}, \mathcal{P}_1^*) := \Pr \left[(pp, \langle \mathcal{P}_1^*, \mathcal{V}_1 \rangle(pp, x_1, \text{st})) \in \mathcal{R}_1^*(k, b) \mid \begin{array}{l} pp \leftarrow \mathcal{G}(1^\lambda) \\ (x_1, \text{st}) \leftarrow \mathcal{A}(pp) \end{array} \right]$$

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

$$\Pr \left[(pp, x_1, w_1) \in \text{TE}^*(1, \ell) \mid \begin{array}{l} pp \leftarrow \mathcal{G}(1^\lambda) \\ (x_1, \text{st}) \leftarrow \mathcal{A}(pp) \\ w_1 \leftarrow \mathcal{E}(pp, x_1, \text{st}) \end{array} \right] \geq \epsilon(\mathcal{A}, \mathcal{P}_1^*) - \epsilon_{\text{Fold}, 1} \quad (53)$$

$$\text{where } \epsilon_{\text{Fold}, 1} \leq \left(\frac{d/e - 1}{|\mathbb{K}|} + \frac{(\ell - k + 1 + m) \cdot 2}{|\mathbb{K}|} \right) + \text{Adv}_A^{\text{MSIS}_{q,n,d,rn\alpha,4 \cdot T_{\text{bck}} \cdot B}}(\mathcal{B}_2, \lambda) + \text{Adv}_A^{\text{MSIS}_{q,n,d,2n\alpha,2b}}(\mathcal{B}_1, \lambda).$$

Claim 2: The following claim states that $\Pi_{\text{Fold}}^{(2)}$ is a weak reduction itself, except admitting a binding collision on a projection π applied to the witness in property (ii). Consider the interactive reduction $\Pi_{\text{Fold}}^{(2)} : \mathcal{R}_1(\mathcal{R}_1^*(k, b)) \rightarrow \mathcal{R}_2$.

(i) Then for any EPT adversary $(\mathcal{A}_2, \mathcal{P}_2^*)$, there exists an EPT extractor $\mathcal{E}_2$ such that if the success probability of the adversary $\epsilon(\mathcal{A}_2, \mathcal{P}_2^*) \geq 1/\text{poly}(\lambda)$, then

$$\Pr \left[(pp, x_2, w_2) \in \mathcal{R}_1^*(k, b) \mid \begin{array}{l} pp \leftarrow \mathcal{G}(1^\lambda) \\ (x_2, \text{st}_2) \leftarrow \mathcal{A}(pp) \\ w_2 \leftarrow \mathcal{E}(pp, x_2, \text{st}_2) \end{array} \right] \geq \epsilon(\mathcal{A}_2, \mathcal{P}_2^*) - \epsilon_{\text{Fold}, 2} \quad (54)$$

$$\text{where } \epsilon_{\text{Fold}, 2} \leq 2^k \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right) + 2 \cdot \text{Adv}_A^{\text{MSIS}_{q,n,d,2n\alpha,2b}}(\mathcal{B}_1, \lambda).$$

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

$$\Pr \left[\begin{array}{c|c} x_2, x'_2 \neq \perp & pp \leftarrow \mathcal{G}(1^\lambda) \\ \downarrow & \text{st}^* \leftarrow \mathcal{B}(pp) \\ \phi(x_2) = \phi(x'_2) & (x_2, \text{st}) \leftarrow \mathcal{B}'(\text{st}^*) \\ & (x'_2, \text{st}') \leftarrow \mathcal{B}'(\text{st}^*) \end{array} \right] = 1,$$

where ϕ is the function which projects t from the instance, then the following holds. Define the oracle

$$\begin{aligned} \mathcal{O}_{\text{weak}}(pp, \text{st}^*) &\rightarrow (x_2, w_2) : \\ \text{(a)} \quad &(x_2, \text{st}) \leftarrow \mathcal{B}'(\text{st}^*). \\ \text{(b)} \quad &w_2 \leftarrow \mathcal{E}_2(pp, x_2, \text{st}). \\ \text{(c)} \quad &\text{Output } (x_2, w_2). \end{aligned}$$

For any uniqueness adversary $\mathcal{W}_{\text{uniq}}^{\mathcal{O}_{\text{weak}}}$ (Def. 5), we have

$$\Pr \left[\begin{array}{c} w_2, w'_2 \neq \perp \\ \wedge \pi(w_2) \neq \pi(w'_2) \end{array} \middle| \begin{array}{l} pp \leftarrow \mathcal{G}(1^\lambda) \\ \text{st}^* \leftarrow \mathcal{B}(pp) \\ \text{in} \leftarrow (pp, \text{st}^*) \\ (x_2, w_2), (x'_2, w'_2) \leftarrow \mathcal{W}_{\text{uniq}}^{\mathcal{O}_{\text{weak}}(\text{in})}(\text{in}) \end{array} \right] \\ \leq \text{Adv}_A^{\text{MSIS}_{q,n,d,rn\alpha,4 \cdot T_{\text{bck}} \cdot B}}(\mathcal{B}_2, \lambda) + \text{Adv}_A^{\text{MSIS}_{q,n,d,2n\alpha,2b}}(\mathcal{B}_1, \lambda)$$

where π is the function which projects $(s_i)_{i \in [\ell]}$ from the witness.

Claim 3: Consider the interactive reduction $\Pi_{\text{Fold}}^{(3)} : \mathcal{R}_2 \rightarrow \text{TE}(1, k, b) \times \text{TE}(1, \ell - k, B) \times \text{PE}(1, k, b)$. Then for any EPT adversary $(\mathcal{A}_3, \mathcal{P}_3^*)$, there exists an EPT extractor $\mathcal{E}_3$ such that if the success probability of the adversary $\epsilon(\mathcal{A}_3, \mathcal{P}_3^*) \geq 1/\text{poly}(\lambda)$, then

$$\Pr \left[(pp, x_3, w_3) \in \mathcal{R}_2 \middle| \begin{array}{l} pp \leftarrow \mathcal{G}(1^\lambda) \\ (x_3, \text{st}_3) \leftarrow \mathcal{A}(pp) \\ w_3 \leftarrow \mathcal{E}(pp, x_3, \text{st}_3) \end{array} \right] \geq \epsilon(\mathcal{A}_3, \mathcal{P}_3^*) - \epsilon_{\text{Fold}, 3} \quad (55)$$

where $\epsilon_{\text{Fold}, 3} \leq \left(3 \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right) + \frac{d/e - 1}{|\mathbb{K}|} + \frac{(k+1+m) \cdot 2}{|\mathbb{K}|} + (n-1) \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right) \right) \\ + 2 \cdot \text{Adv}_A^{\text{MSIS}_{q,n,d,2n\alpha,2b}}(\mathcal{B}_1, \lambda).$

Now we describe how to compose Claims 1-3 to achieve $\mathcal{E}$ with knowledge error ϵ_{Fold} . For adversary $(\mathcal{A}, \mathcal{P}^*)$, split $\mathcal{P}^* = (\mathcal{P}_1^*, \mathcal{P}_2^*, \mathcal{P}_3^*)$ where for pp and $(x_1, st) \leftarrow \mathcal{A}(pp)$,

$$\begin{aligned} (x_2, st_2) &\leftarrow \langle \mathcal{P}_1^*, \mathcal{V}_1 \rangle(pp, x_1, st) && \text{for } \mathcal{V}_1 \text{ running reduction } \Pi_{\text{Fold}}^{(1)} \\ (x_3, st_3) &\leftarrow \langle \mathcal{P}_2^*, \mathcal{V}_2 \rangle(pp, x_2, st_2) && \text{for } \mathcal{V}_2 \text{ running reduction } \Pi_{\text{Fold}}^{(2)} \\ (x_4, st_4) &\leftarrow \langle \mathcal{P}_3^*, \mathcal{V}_3 \rangle(pp, x_3, st_3) && \text{for } \mathcal{V}_3 \text{ running reduction } \Pi_{\text{Fold}}^{(3)}. \end{aligned}$$

First, we construct adversaries $\mathcal{A}_3$ against $\Pi_{\text{Fold}}^{(3)}$ and $\mathcal{A}_2$ against $\Pi_{\text{Fold}}^{(2)}$:

Adversary $\mathcal{A}_3(pp) \rightarrow (x_3, st_3)$:
1. $(x_1, st) \leftarrow \mathcal{A}(pp)$. 2. Simulate $(x_2, st_2) \leftarrow \langle \mathcal{P}_1^*, \mathcal{V}_1 \rangle(pp, x_1, st)$. 3. Simulate $(x_3, st_3) \leftarrow \langle \mathcal{P}_2^*, \mathcal{V}_2 \rangle(pp, x_2, st_2)$. 4. Output (x_3, st_3).

The adversary $\mathcal{A}_3$ runs in expected polynomial time because $\mathcal{A}, \mathcal{P}_1^*, \mathcal{P}_2^*$ all do. Observe that by construction $\epsilon(\mathcal{A}_3, \mathcal{P}_3^*) = \epsilon(\mathcal{A}, \mathcal{P}^*)$ for Π_{Fold} . By Claim 3, for adversary $(\mathcal{A}_3, \mathcal{P}_3^*)$, there exists $\mathcal{E}_3$ that extracts a witness for $\mathcal{R}_2$ with the knowledge error given in the claim.

Adversary $\mathcal{A}_2(pp) \rightarrow (x_2, st_2)$:
1. $(x_1, st) \leftarrow \mathcal{A}(pp)$. 2. Simulate $(x_2, st_2) \leftarrow \langle \mathcal{P}_1^*, \mathcal{V}_1 \rangle(pp, x_1, st)$. 3. Output (x_2, st_2).

The adversary $\mathcal{A}_2$ runs in expected polynomial time because $\mathcal{A}$ and $\mathcal{P}_1^*$ do. Next, we construct an adversary $\mathcal{P}_2^{**}$ against $\Pi_{\text{Fold}}^{(2)}$, which runs $\mathcal{P}_2^*$ with $\mathcal{V}_2$ but then uses $\mathcal{E}_3$ to return the witness.

Adversary $\mathcal{P}_2^{**}(pp, x_2, st_2) \rightarrow w_3$:
1. Run $\mathcal{P}_2^*(pp, x_2, st_2)$, relaying each message back-and-forth with $\mathcal{V}_2$. Let tr be the resulting transcript and st_3 the final state of $\mathcal{P}_2^*$. 2. Since $\Pi_{\text{Fold}}^{(2)}$ is public coin, compute the verifier's output x_3 by emulating $\mathcal{V}_2(x_2)$ on tr. 3. $w_3 \leftarrow \mathcal{E}_3(pp, x_3, st_3)$. 4. Output w_3.

The adversary $\mathcal{P}_2^{**}$ runs in expected polynomial time, as $\mathcal{P}_2^*$ and $\mathcal{E}_3$ do, and all remaining computations run in polynomial time. Conditioned on (pp, x_2, st_2) , step 3-4 of $\mathcal{A}_3$ and the interaction between $\langle \mathcal{P}_2^{**}, \mathcal{V}_2 \rangle$ generate (x_3, st_3) according to the same distribution: both run $\langle \mathcal{P}_2^*, \mathcal{V}_2 \rangle$ with the same inputs. Hence, the event in Eq. (55) is exactly the event defining $\epsilon(\mathcal{A}_2, \mathcal{P}_2^{**}) \geq \epsilon(\mathcal{A}_3, \mathcal{P}_3^*) - \epsilon_{\text{Fold}, 3} = \epsilon(\mathcal{A}, \mathcal{P}^*) - \epsilon_{\text{Fold}, 3}$.

Next, by Claim 2, for adversary $(\mathcal{A}_2, \mathcal{P}_2^{**})$ there exists extractor $\mathcal{E}_2$ which extracts w_2 for $\mathcal{R}_1^*(k, b)$ with knowledge error $\epsilon_{\text{Fold}, 2}$. We construct an adversary $\mathcal{P}_1^{**}$ against $\Pi_{\text{Fold}}^{(1)}$, which runs $\mathcal{P}_1^*$ with $\mathcal{V}_1$ but then uses $\mathcal{E}_2$ to return the witness.

Adversary $\mathcal{P}_1^{**}(pp, x_1, st_1) \rightarrow w_2$:
1. Run $\mathcal{P}_1^*(pp, x_1, st_1)$, relaying each message back-and-forth with $\mathcal{V}_1$. Let tr be the resulting transcript and st_2 the final state of $\mathcal{P}_1^*$. 2. Since $\Pi_{\text{Fold}}^{(1)}$ is public coin, compute the verifier's output x_2 by emulating $\mathcal{V}_1(x_1)$ on tr. 3. $w_2 \leftarrow \mathcal{E}_2(pp, x_2, st_2)$. 4. Output w_2.

The adversary $\mathcal{P}_1^{**}$ runs in expected polynomial time, as $\mathcal{P}_1^*$ and $\mathcal{E}_2$ do, and all remaining computations run in polynomial time. Conditioned on (pp, x_1, st_1) , step 2-3 of $\mathcal{A}_2$ and the interaction between $\langle \mathcal{P}_1^{**}, \mathcal{V}_1 \rangle$ generate (x_2, st_2) according to the same distribution: both run $\langle \mathcal{P}_1^*, \mathcal{V}_1 \rangle$ with the same inputs. Hence, the event in Eq. (54) is exactly the event defining $\epsilon(\mathcal{A}, \mathcal{P}_1^{**}) \geq \epsilon(\mathcal{A}_2, \mathcal{P}_2^{**}) - \epsilon_{\text{Fold}, 2} \geq \epsilon(\mathcal{A}, \mathcal{P}^*) - \epsilon_{\text{Fold}, 3} - \epsilon_{\text{Fold}, 2}$.

By Claim 1(ii), for adversary $(\mathcal{A}, \mathcal{P}_1^{**})$ there exists extractor $\mathcal{E}_1$ which extracts w_1 for $\text{TE}^*(1, \ell)$ with knowledge error $\epsilon_{\text{Fold}, 1}$. Then

$$\begin{aligned}
\Pr \left[(pp, x_1, w_1) \in \text{TE}^*(1, \ell) \mid \begin{array}{l} pp \leftarrow \mathcal{G}(1^\lambda) \\ (x_1, \text{st}) \leftarrow \mathcal{A}(pp) \\ w_1 \leftarrow \mathcal{E}_1(pp, x_1, \text{st}) \end{array} \right] &\geq \epsilon(\mathcal{A}, \mathcal{P}_1^{**}) - \epsilon_{\text{Fold}, 1} \\
&\geq \epsilon(\mathcal{A}, \mathcal{P}^*) - \epsilon_{\text{Fold}, 3} - \epsilon_{\text{Fold}, 2} - \epsilon_{\text{Fold}, 1} \\
&\geq \epsilon(\mathcal{A}, \mathcal{P}^*) - \epsilon_{\text{Fold}}
\end{aligned} \tag{Eq. 53}$$

where ϵ_{Fold} is as defined in Eq. (52). 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 $\text{TE}^*(1, \ell)$ with probability at least $\epsilon(\mathcal{A}, \mathcal{P}^*) - \epsilon_{\text{Fold}}$.

(ii) *Relaxed binding.* Let $\mathcal{O}_{\text{weak}}$ be the oracle from condition (ii) of Def. 6 instantiated with $\mathcal{B}'$ and the extractor $\mathcal{E}$ from the proof of property (i):

$$\begin{aligned}
&\mathcal{O}_{\text{weak}}(pp, \text{st}^*) \rightarrow (x_1, w_1) : \\
&\quad 1. (x_1, \text{st}) \leftarrow \mathcal{B}'(\text{st}^*). \\
&\quad 2. $w_1 \leftarrow \mathcal{E}(pp, x_1, \text{st})$. \\
&\quad 3. Output (x_1, w_1) .
\end{aligned}$$

Consider any uniqueness adversary (Def. 5) $\mathcal{W}_{\text{uniq}}^{\mathcal{O}_{\text{weak}}(\text{in})}$. Consider the following adversary $\mathcal{A}_{\text{rlx}}$ against norm b binding or $(2B, \mathcal{C}, k)$ -relaxed binding.

Adversary $\mathcal{A}_{\text{rlx}}(pp)$
1. $\text{st}^* \leftarrow \mathcal{B}(pp)$ and $\text{in} \leftarrow (pp, \text{st}^*)$. 2. Run $\mathcal{W}_{\text{uniq}}^{\mathcal{O}_{\text{weak}}(\text{in})}(\text{in})$. For every query to $\mathcal{O}_{\text{weak}}$: (a) Compute $(x_1, \text{st}) \leftarrow \mathcal{B}'(\text{st}^*)$. (b) Run $\mathcal{E}(pp, x_1, \text{st})$ to obtain w_1, retaining its internal value <code>result</code>. (c) Return (x_1, w_1) and store <code>result</code> for this call. 3. If $\mathcal{W}_{\text{uniq}}$ outputs $\perp$, a $\perp$ witness, or equal witnesses, output $\perp$. 4. Otherwise, let (x_1, w_1) and (x'_1, w'_1) be the selected responses, and retrieve <code>result</code> and <code>result'</code> stored for the two calls that returned them. 5. Parse $(s_i)_{i \in [\ell]} \leftarrow w_1$ and $(s'_i)_{i \in [\ell]} \leftarrow w'_1$. 6. If there is $i \in [k]$ such that $s_i \neq s'_i$, parse t from x_1 and output $(pp, t, \_, \_, (s_i)_{i \in [k]}, (s'_i)_{i \in [k]})$. 7. Else, choose $i \in [k+1, \ell]$, $l \in [K]$ such that $s_{i,l} \neq s'_{i,l}$. 8. Parse <code>result</code> and <code>result'</code> to obtain $\rho_{0,l}, \rho_{l,l}$ and $\rho'_{0,l}, \rho'_{l,l}$. 9. For $l \in [K]$, assign $\Delta_l \leftarrow (\rho_{l,0} - \rho_{l,l})$ $\Delta'_l \leftarrow (\rho'_{l,0} - \rho'_{l,l})$ 10. Parse t from x_1 and output $(pp, t, \Delta, \Delta', (s_i)_{i \in [\ell]}, (s'_i)_{i \in [\ell]})$.

The values from the two selected calls, x_1, x'_1 are such that $t = \phi(x_1) = \phi(x'_1)$ by assumption. Also by assumption, w_1, w'_1 are such that $w_1, w'_1 \neq \perp \wedge w_1 \neq w'_1$. Inside, $\mathcal{E}$ by construction in property (i), it has been verified that corresponding w_2, w'_2 are such that $\pi(w_2) = w_1, \pi(w'_2) = w'_1$ where $(pp, x_2, w_2) \in \mathcal{R}_1^*(k, b)$ and $(pp, x'_2, w'_2) \in \mathcal{R}_1^*(k, b)$ and $\phi(x_2) = \phi(x'_2) = \phi(x_1) = \phi(x'_1) = t$ by construction. Then $\mathcal{A}_{\text{rlx}}$ either outputs a norm b binding collision in Step 6, or a $(2B, \mathcal{C}, k)$ -relaxed binding (Def. 22) collision on t by Eq. (65). By Lem. 11 and Lem. 12,

$$\begin{aligned}
& \Pr \left[\begin{array}{l} w_1, w'_1 \neq \perp \\ \wedge w_1 \neq w'_1 \end{array} \middle| \begin{array}{l} pp \leftarrow \mathcal{G}(1^\lambda) \\ \text{st}^* \leftarrow \mathcal{B}(pp) \\ \text{in} \leftarrow (pp, \text{st}^*) \\ (x_1, w_1), (x'_1, w'_1) \leftarrow \mathcal{W}_{\text{uniq}}^{\mathcal{O}_{\text{weak}}(\text{in})}(\text{in}) \end{array} \right] \\
& \leq \text{Adv}_A^{\text{MSIS}_{q,n,d,rn\alpha,2b}}(\mathcal{B}_1, \lambda) + \text{Adv}_A^{\text{MSIS}_{q,n,d,rn\alpha,4 \cdot T_{\text{bck}} \cdot B}}(\mathcal{B}_2, \lambda)
\end{aligned}$$

which is $\leq \text{negl}(\lambda)$ assuming $\text{MSIS}_{q,n,d,rn\alpha,2b}, \text{MSIS}_{q,n,d,rn\alpha,4 \cdot T_{\text{bck}} \cdot B}$.

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. We will now prove Claims 1-3.

Proof of Claim 1. Property (i) clearly holds by construction. To prove property (ii) we invoke Lemma 27 with the following object assignments

- $\theta = (pp, x_1, st, v_{\text{top}}, v_{\text{subs}}, st')$ where st' is the $\mathcal{P}_1^*$ state after Step 1.
- The oracle $\mathcal{O}(\theta)$ runs the rest of the interaction between $\mathcal{P}_1^*$ and $\mathcal{V}_1$ to produce (x_2, w_2) (including the verifier check in Step 1 that remains). If $w_2 \neq \perp$ and $(pp, x_2, w_2) \in \mathcal{R}_1^*(k, b)$, then the oracle returns the parsed out $(s_i)_{i \in [\ell]}$ from the witness

$$(s_i)_{i \in [\ell]}, (s_{\text{subs},i})_{i \in [k]}, \Delta \in ((\mathcal{C} - \mathcal{C}) \cap R_{\mathbb{F}}^\times)^{2^k} \leftarrow w_2.$$

Else, the oracle returns $\perp$.

- $\text{Bad}(\theta)$ is the set of $(s_i)_{i \in [\ell]}$ such that $(pp, x_1, (s_i)_{i \in [\ell]}) \notin \text{TE}^*(1, \ell)$.

Note the relationship between the oracle and the extractor $\mathcal{E}_1$. The oracle and extractor both run the same $\langle \mathcal{P}_1^*, \mathcal{V}_1 \rangle$ interaction. The oracle is only used in the agreement argument, while the extractor retains the whole witness inside. By construction of $\mathcal{E}_1$, for $z \leftarrow \mathcal{O}(\theta)$,

$$z \neq \perp \wedge z \notin \text{Bad}(\theta) \implies (pp, x_1, w_1) \in \text{TE}^*(1, \ell).$$

And thus define the event

$$\text{Err}_1 := \{z \neq \perp \wedge z \in \text{Bad}(\theta)\} = \{z \in \text{Bad}(\theta)\}$$

because by definition of Bad , $z \neq \perp$. We can then bound the success probability of the extractor as desired:

$$\begin{aligned} \Pr[(pp, x_1, w_1) \in \text{TE}^*(1, \ell)] &\geq \Pr[z \neq \perp \setminus \text{Err}_1] \\ &= \mathbb{E}_\theta[\Pr[z \neq \perp \mid \theta]] - \mathbb{E}_\theta[\text{Err}_1 \mid \theta] \quad (\text{because } \text{Err}_1 \subseteq z \neq \perp, \text{ linearity of expectation}) \\ &\geq \epsilon(\mathcal{A}, \mathcal{P}_1^*) - \mathbb{E}_\theta[\delta(\theta)] - \text{Adv}^{\text{uniq}}(\mathcal{S}_{\text{uniq}}). \end{aligned} \quad (\text{def. of } \epsilon, \text{ Lem. 27})$$

where δ , $\text{Adv}^{\text{uniq}}(\mathcal{S}_{\text{uniq}})$ are defined as in Lem. 27.

Extractor $\mathcal{E}_1(pp, x_1, st_1) \rightarrow w_1$
1. $(x_2, w_2) \leftarrow \langle \mathcal{P}_1^*, \mathcal{V}_1 \rangle(pp, x_1, st_1)$. 2. If $w_2 = \perp$, return $\perp$. 3. Else, parse $((s_i)_{i \in [\ell]}, (s_{\text{subs}, i})_{i \in [k]}, \Delta \in ((\mathcal{C} - \mathcal{C}) \cap R_{\mathbb{F}}^\times)^{2^k}) \leftarrow w_2$. Return $w_1 := (s_i)_{i \in [\ell]}$.

We now compute $\mathbb{E}_\theta[\delta(\theta)]$ and $\text{Adv}(\mathcal{S}_{\text{uniq}})$. We use the uniqueness adversary described in Lem. 27, where outputting w_2, w'_2 such that $\pi(w_2) \neq \pi(w'_2)$ is exactly two different relaxed openings $(s_i)_{i \in [\ell]}, (s'_i)_{i \in [\ell]}$ for the same commitment t in x_1 (see definition of $\mathcal{R}_1^*(k, b)$). This is exactly a binding collision on one of the top k layers or $(2B, \mathcal{C}, k)$ -relaxed binding collision on one of the bottom $\ell - k$ layers, which occur with probability at most

$$\leq \text{Adv}_A^{\text{MSIS}_{q,n,d,rn\alpha,4 \cdot T_{\text{bck}} \cdot B}}(\mathcal{B}_2, \lambda) + \text{Adv}_A^{\text{MSIS}_{q,n,d,2n\alpha,2b}}(\mathcal{B}_1, \lambda).$$

by Lem. 11, Lem. 12. Recall in Lem. 27 that for every fixed θ and fixed $z \in \text{Bad}(\theta)$,

$$\Pr[\mathcal{O}(\theta) = z] \leq \delta(\theta).$$

Because $(x_2, w_2) \in \mathcal{R}_1^*(k, b)$, it holds that $\text{Open}(A, |\mathbb{F}|/2, t, (s_i)_{i \in [\ell]}) = 1$. It remains to show the evaluation claim $\tilde{s}(u) = v$, where $(t, u, v) \leftarrow x_1$.

Let $\mathfrak{s}_j$ be the subtree concatenated opening vector for the extracted witness $w_1 := (s_i)_{i \in [\ell]}$.

$$v_{\text{subs}} = \sum_{j=1}^{2^k} \langle q_{\text{subs}, j}, \mathfrak{s}_j \rangle. \quad (56)$$

holds if and only if the sum-check claim in Step 2 holds:

$$v_{\text{subs}} = \sum_{X \in \{0,1\}^{\ell-k+1+m}} \left(\sum_{j=1}^{2^k} \tilde{q}_{\text{subs}, j}(X) \cdot \tilde{\mathfrak{s}}_j(X) \right)$$

as described in Section 5.2.1. Thus if Eq. (56) does not hold, it must be the case the $\mathcal{P}^*$ passed succeeded in BatchSC in Step 2, despite the sum-check claim not holding. Due to the soundness of BatchSC, this occurs with probability at most

$$\leq (1 - 1) \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right) + \frac{d/e - 1}{|\mathbb{K}|} + \frac{(\ell - k + 1 + m) \cdot 2}{|\mathbb{K}|}$$

where the BatchSC in Step 2 takes in 1 sum-check claim of maximum degree 2 and maximum variable count $\ell - k + 1 + m$. We now proceed assuming Eq. (56) holds, and prove that w_1 satisfies the conditions of $\text{TE}^*(1, \ell)$.

Then given Eq. (56), we show that $\tilde{s}(u) = v$ holds. Further, $(x_2, w_2) \in \mathcal{R}_1^*(k, b)$ also asserts that $f_{\text{subs}}^{(j)} = \tilde{\mathfrak{s}}_j(r)$ for all $j \in [2^k]$ and $y_{\text{subs}} = \sum_{j=1}^{2^k} \tilde{q}_{\text{subs}, j}(r) \cdot f_{\text{subs}}^{(j)}$. Putting these equations together, we have that

$$y_{\text{subs}} = \sum_{j=1}^{2^k} \tilde{q}_{\text{subs}, j}(r) \cdot \tilde{\mathfrak{s}}_j(r).$$

Then, working backwards with a correct y_{subs} , by Eq. (56),

$$v_{\text{subs}} = \sum_{j=1}^{2^k} \langle q_{\text{subs},j}, \mathfrak{s}_j \rangle.$$

Finally, because the verifier check in Step 1 passes,

$$v = \tilde{eq}(0^{\ell-k}, u^{(\cdot:\ell-k)}) \cdot v_{\text{top}} + v_{\text{subs}} \quad (\text{Step 1})$$

$$= \tilde{eq}(0^{\ell-k}, u^{(\cdot:\ell-k)}) \cdot \tilde{s}_{\text{top}}(u^{(\ell-k+1:\cdot)}) + v_{\text{subs}} \quad (\text{Eq. (62)})$$

$$= \tilde{eq}(0^{\ell-k}, u^{(\cdot:\ell-k)}) \cdot \tilde{s}_{\text{top}}(u^{(\ell-k+1:\cdot)}) + \sum_{j=2^{k+1}n\alpha}^{2^{\ell+1}n\alpha-1} s^{(j)} \cdot \tilde{eq}(\text{bits}_{\ell+1+m}(j), u) \quad (\text{Eq. (56), (14)})$$

$$= \tilde{s}(u). \quad (\text{Eq. (12)})$$

And so we can conclude that

$$\mathbb{E}_{\theta}[\delta(\theta)] \leq \frac{d/e-1}{|\mathbb{K}|} + \frac{(\ell-k+1+m) \cdot 2}{|\mathbb{K}|}.$$

Then invoking Lem. 27, we have property (ii) of Claim 1.

Proof of Claim 2. We first prove property (i) by defining the following extractor $\mathcal{E}_2$. To construct $\mathcal{E}_2$ we first construct an adversary and verification function to use Lem. 30,

$A_{(pp, x_2, st_2)}(\vec{c}) :$

1. If $x_2 = \perp$, output $\perp$.
2. Simulate $(x_3, w_3) \leftarrow \langle \mathcal{P}_2^*, \mathcal{V}_2 \rangle(pp, x_2, st_2)$ with verifier randomness $\vec{c}$ and produce transcript tr .
3. Output (w_3, tr) .

$V_{(pp, x_2, st_2)}(\vec{c}, w_3, \text{tr}) \rightarrow \{0, 1\} :$

1. If $x_2 = \perp$, output 0.
2. Simulate $x_3 \leftarrow \mathcal{V}(pp, x_2)$ on tr with verifier randomness $\vec{c}$. Reject if $\mathcal{V}$ rejects.
3. Output 1 if and only if $(x_3, w_3) \in \mathcal{R}_2$.

Let $E_{(pp, x_2, st_2)}$ be the corresponding extractor from Lem. 30, given (pp, x_2, st_2) , run $A_{(pp, x_2, st_2)}$ and $V_{(pp, x_2, st_2)}$ above to generate transcripts satisfying the coordinate-wise property in Def. 16. Now we can define $\mathcal{E}_2$ for $\Pi_{\text{Fold}}^{(2)}$, where let $K := 2^k$.

Extractor $\mathcal{E}_2(pp, x_2, st) \rightarrow w_2 :$

1. $\text{result} \leftarrow E(pp, x_2, st)$.
2. If $x_2 = \perp$ or $\text{result} = \perp$, output $\perp$.
3. Parse $(\vec{c}_0, w'_0), (\vec{c}_1, w'_1), \dots, (\vec{c}_K, w'_K) \leftarrow \text{result}$.
4. Parse $((z_{i,0})_{i \in [k]}, (z_{i,0}^*)_{i \in [\ell-k]}, (z_{\text{subs},i,0})_{i \in [k]}) \leftarrow w'_0$ and $\rho_{1,0}, \dots, \rho_{K,0} \leftarrow \vec{c}_0$.
5. For $l \in [K]$,
 - (a) Parse $((z_{i,l})_{i \in [k]}, (z_{i,l}^*)_{i \in [\ell-k]}, (z_{\text{subs},i,l})_{i \in [k]}) \leftarrow w'_l$ and $\rho_{1,l}, \dots, \rho_{K,l} \leftarrow \vec{c}_l$.
 - (b) Assign $s'_{k+i,l} \leftarrow (\rho_{l,0} - \rho_{l,l})^{-1} \cdot (z_{i,0}^* - z_{i,l}^*)$ for all $i \in [\ell-k]$.
 - (c) Assign $\Delta_l \leftarrow (\rho_{l,0} - \rho_{l,l})$.
6. For $i \in [\ell-k]$, assign $s'_{k+i} \leftarrow s'_{k+i,1} \parallel \dots \parallel s'_{k+i,K}$.

$$7. \text{ Output } w_2 := \left((z_{i,0})_{i \in [k]}, (s'_{k+i})_{i \in [\ell-k]}, (z_{\text{subs},i,0})_{i \in [k]}, (\Delta_l)_{l \in [K]} \right) \text{ if and only if}$$

$$(x_2; w_2) \in \mathcal{R}_1^*(k, b)$$

By Lem. 30, $E(pp, x_2, st_2)$ outputs $(\vec{c}_0, w'_0), (\vec{c}_1, w'_1), \dots, (\vec{c}_K, w'_K)$ such that

- $V_{(pp, x_2, st_2)}(\vec{c}_l, w'_l) = 1$ for all $l \in [0, K]$, and
- $(\vec{c}_0, \vec{c}_1, \dots, \vec{c}_K)$ satisfy the challenge set property described in Def. 16.
- For all $l \in [K]$, $(c_{0,l} - c_{l,l}) \in R_{\mathbb{F}}^\times$.

with probability at least

$$\epsilon^{V_{(pp, x_2, st_2)}}(A_{(pp, x_2, st_2)}(\vec{c})) - K \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right)$$

Observe that

$$\epsilon(\mathcal{A}_2, \mathcal{P}^*) = \mathbb{E}_{(pp, x_2, st_2)}[\epsilon^{V_{(pp, x_2, st_2)}}(A_{(pp, x_2, st_2)}(\vec{c}))].$$

and so averaging over the distribution (pp, x_2, st_2) gives us

$$\Pr[\text{result} \neq \perp] \geq \epsilon(\mathcal{A}_2, \mathcal{P}^*) - K \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right)$$

by linearity of expectation. Now assume $E(pp, x_2, st_2)$ does indeed produce such $(\vec{c}_0, w'_0), (\vec{c}_1, w'_1), \dots, (\vec{c}_K, w'_K)$. We first claim that across all transcripts $l \in [K]$, $(z_{i,0})_{i \in [k]} = (z_{i,l})_{i \in [k]}$. Otherwise, this corresponds to a binding collision on t . Similarly, for all $l \in [K]$, $(z_{\text{subs},i,0})_{i \in [k]} = (z_{\text{subs},i,l})_{i \in [k]}$ or there is a binding collision on t_{subs} . By Lem. 11 and union bound, the probability of at least one of these events occurring is bounded by

$$\leq 2 \cdot \text{Adv}_A^{\text{MSIS}_{q,n,d,2n\alpha,2b}}(\mathcal{B}_1, \lambda).$$

We now proceed assuming all the openings to t and t_{subs} are equal across all transcripts. Since $V_{(pp, x_2, st_2)}(\vec{c}_l, w'_l) = 1$ for all $l \in [0, K]$,

$$\text{Open}(A, b, t, (z_{i,0})_{i \in [k]}) = 1, \quad (57)$$

$$((t_l^*, r, v_{\text{bot},l}); (z_{i,l}^*)_{i \in [\ell-k]}) \in \text{TE}(1, \ell - k, B), \quad (58)$$

$$v_{\text{bot},l} = \sum_{j=1}^{2^k} \rho_{j,l} \cdot f_{\text{subs}}'^{(j)}, \quad (59)$$

$$y_{\text{subs}} = \sum_{j=1}^{2^k} \tilde{q}_{\text{subs},j}(r) \cdot f_{\text{subs}}'^{(j)}, \quad (60)$$

$$t_l^* = \sum_{j=1}^{2^k} \rho_{j,l} \cdot (G_n \cdot \mathfrak{s}'_{k,j}), \quad (61)$$

$$v_{\text{top}} = \tilde{s}_{\text{top}}(u^{(\ell-k+1:\cdot)}) \quad (62)$$

To show that $(x_2, w_2) \in \mathcal{R}_1^*(k, b)$ it remains to show the commitment opening condition on t , and relaxed norm condition on $((z_{i,0})_{i \in [k]}, (s'_{k+i})_{i \in [\ell-k]})$, and the evaluation claim $f_{\text{subs}}^{(j)} = \tilde{s}'_j(r)$.

Commitment opening conditions: We now show that

$$\text{Open}(A, (|\mathbb{F}|/2), t, ((z_{i,0})_{i \in [k]}, (s'_{k+i})_{i \in [\ell-k]})) = 1$$

By combining Eq. (57), Eq. (58), and Eq. (61), we show that the commitment opening conditions are guaranteed. For each $l \in [K]$ we subtract to get Eq. (61).

$$\begin{aligned} t_0^* - t_l^* &= \sum_{j=1}^{2^k} \rho_{j,0} \cdot (G_n \cdot \mathbf{s}'_{k,j}) - \sum_{j=1}^{2^k} \rho_{j,l} \cdot (G_n \cdot \mathbf{s}'_{k,j}) \\ &= \sum_{j=1}^{2^k} (\rho_{j,0} - \rho_{j,l}) \cdot (G_n \cdot \mathbf{s}'_{k,j}) = (\rho_{l,0} - \rho_{l,l}) \cdot (G_n \cdot \mathbf{s}'_{k,l}) \end{aligned}$$

where the last equality follows from the coordinate-wise challenge set property in Def. 16. Using Eq. (58), this is also equal to

$$t_0^* - t_l^* = A \cdot z_{1,0}^* - A \cdot z_{1,l}^* = A \cdot (z_{1,0}^* - z_{1,l}^*) \quad (\text{Eq. (58)})$$

$$= A \cdot (\rho_{l,0} - \rho_{l,l}) \cdot s'_{k+1,l}. \quad (\text{Step 5b})$$

Since $(\rho_{l,0} - \rho_{l,l})$ is invertible, this implies that for all $l \in [K]$,

$$(\rho_{l,0} - \rho_{l,l}) \cdot G_n \cdot \mathbf{s}'_{k,l} = A \cdot (\rho_{l,0} - \rho_{l,l}) \cdot s'_{k+1,l} \implies G_n \cdot \mathbf{s}'_{k,l} = A \cdot s'_{k+1,l}$$

Since $z_{k,0} = \mathbf{s}'_{k,1} \parallel \dots \parallel \mathbf{s}'_{k,K}$ the layer- k opening for the extracted witness, concatenating across all $l \in [K]$ gives us that

$$G_{2^k n} \cdot z_{k,0} = (I_{2^k} \otimes A) \cdot s'_{k+1} \quad (63)$$

Similarly, for intermediate layers of the commitment tree for all $i \in [\ell - k - 1]$, by Eq. (58)

$$\begin{aligned} G_{2^i n} \cdot z_{i,0}^* - G_{2^i n} \cdot z_{i,l}^* &= (I_{2^i} \otimes A) \cdot z_{i+1,0}^* - (I_{2^i} \otimes A) \cdot z_{i+1,l}^* \\ G_{2^i n} \cdot (z_{i,0}^* - z_{i,l}^*) &= (I_{2^i} \otimes A) \cdot (z_{i+1,0}^* - z_{i+1,l}^*) \\ G_{2^i n} \cdot (\rho_{l,0} - \rho_{l,l}) \cdot s'_{k+i,l} &= (I_{2^i} \otimes A) \cdot (\rho_{l,0} - \rho_{l,l}) \cdot s'_{k+i+1,l} \quad (\text{Step 5b}) \\ G_{2^i n} \cdot s'_{k+i,l} &= (I_{2^i} \otimes A) \cdot s'_{k+i+1,l} \quad \forall i \in [\ell - 1] \quad (\text{invertibility of } (\rho_{l,0} - \rho_{l,l})) \end{aligned}$$

And so for all $i \in [\ell - k - 1]$,

$$G_{2^{k+i} n} \cdot s'_{k+i} = (I_{2^{k+i}} \otimes A) \cdot s'_{k+i+1} \quad (64)$$

by concatenation across $l \in [K]$. By Eq. (63), Eq. (64), and the opening relation for the top k layers in Eq. (57),

$$\text{Open}(A, (\|\mathbb{F}\|/2), t, ((z_{i,0})_{i \in [k]}, (s'_{k+i})_{i \in [\ell - k]})) = 1$$

Relaxed norm conditions: For all $i \in [k]$, $\|z_{i,0}\|_\infty < b$ by Eq. (57). And $\forall l \in [K]$,

$$\begin{aligned} \|\Delta_l \cdot s'_{k+i,l}\|_\infty &= \|(\rho_{l,0} - \rho_{l,l}) \cdot s'_{k+i,l}\|_\infty \quad (\text{Step 5c of } \mathcal{E}_2) \\ &= \|z_{i,0}^* - z_{i,l}^*\|_\infty \quad (\text{Step 5b of } \mathcal{E}_2) \\ &\leq \|z_{i,0}^*\|_\infty + \|z_{i,l}^*\|_\infty \quad (\text{triangle inequality}) \\ &< 2B. \quad (65) \end{aligned}$$

Evaluation $f_{\text{subs}}^{(l)} = \tilde{s}'_l(r)$: Then the extracted $\mathbf{s}'_l$ satisfies the evaluation claim at r on the opening of t_{subs} .

$$\begin{aligned} \tilde{s}'_l(r) &= (\rho_{l,0} - \rho_{l,l})^{-1} \cdot (\tilde{z}_0^*(r) - \tilde{z}_l^*(r)) \quad (\text{Step 6, linearity of MLE}) \\ &= (\rho_{l,0} - \rho_{l,l})^{-1} \cdot (v_{\text{bot},0} - v_{\text{bot},l}) \quad (\text{TE}(1, \ell - k, B) \text{ def.}) \\ &= (\rho_{l,0} - \rho_{l,l})^{-1} \cdot \left(\sum_{j \in [2^k]} \rho_{j,0} \cdot f_{\text{subs}}^{(j)} - \sum_{j \in [2^k]} \rho_{j,l} \cdot f_{\text{subs}}^{(j)} \right) \quad (\text{Claim 1 Eq. (59)}) \\ &= f_{\text{subs}}^{(j)}. \quad (\text{coordinate-wise property}) \end{aligned}$$

We now prove property (ii).

(ii) *Relaxed binding:* Let $\mathcal{O}_{\text{weak}}$ be the oracle from condition (ii) of Def. 6 instantiated with B' and $\mathcal{E}$:

- $$\mathcal{O}_{\text{weak}}(pp, \text{st}^*) \rightarrow (u_1, w_1) :$$
1. $(u_1, \text{st}) \leftarrow \mathcal{B}'(\text{st}^*)$.
 2. $w_1 \leftarrow \mathcal{E}(pp, u_1, \text{st})$.
 3. Output (u_1, w_1) .

Consider any uniqueness adversary (Def. 5) $\mathcal{W}_{\text{uniq}}^{\mathcal{O}_{\text{weak}}(\text{in})}$. Consider the following adversary $\mathcal{A}_{\text{rlx}}$ against norm b binding or $(2B, \mathcal{C}, k)$ -relaxed binding.

Adversary $\mathcal{A}_{\text{rlx}}(pp)$
1. $\text{st}^* \leftarrow \mathcal{B}(pp)$ and $\text{in} \leftarrow (pp, \text{st}^*)$. 2. Run $\mathcal{W}_{\text{uniq}}^{\mathcal{O}_{\text{weak}}(\text{in})}(\text{in})$. For every query to $\mathcal{O}_{\text{weak}}$: (a) Compute $(x_2, \text{st}) \leftarrow \mathcal{B}'(\text{st}^*)$. (b) Run $\mathcal{E}(pp, x_2, \text{st})$ to obtain w_2, retaining its internal value result. (c) Return (x_2, w_2) and store result for this call. 3. If $\mathcal{W}_{\text{uniq}}$ outputs $\perp$, a $\perp$ witness, or equal witnesses under π, output $\perp$. 4. Otherwise, let (x_2, w_2) and (x'_2, w'_2) be the selected responses, and retrieve result and result' stored for the two calls that returned them. 5. Parse $\pi(w_2) = (s_i)_{i \in [\ell]}$ and $\pi(w'_2) = (s'_i)_{i \in [\ell]}$. 6. If there is $i \in [k]$ such that $s_i \neq s'_i$, parse t from x_2 and output $(pp, t, \_, \_, (s_i)_{i \in [k]}, (s'_i)_{i \in [k]})$. 7. Else, choose $i \in [k+1, \ell]$, $l \in [K]$ such that $s_{i,l} \neq s'_{i,l}$. 8. Parse result and result' to obtain $\rho_{0,l}, \rho_{l,l}$ and $\rho'_{0,l}, \rho'_{l,l}$. 9. For $l \in [K]$, assign $\Delta_l \leftarrow (\rho_{l,0} - \rho_{l,l})$ $\Delta'_l \leftarrow (\rho'_{l,0} - \rho'_{l,l})$ 10. Parse t from x_2 and output $(pp, t, \Delta, \Delta', (s_i)_{i \in [\ell]}, (s'_i)_{i \in [\ell]})$.

The values from the two selected calls, x_2, x'_2 are such that $t = \phi(x_2) = \phi(x'_2)$ by assumption. Also by assumption, w_2, w'_2 are such that $w_2, w'_2 \neq \perp \wedge \pi(w_2) \neq \pi(w'_2)$, meaning $(x_2, w_2), (x'_2, w'_2) \in \mathcal{R}_1^*(k, b)$. Then $\mathcal{A}_{\text{rlx}}$ either outputs a norm b binding collision in Step 6, or a $(2B, \mathcal{C}, k)$ -relaxed binding (Def. 22) collision on t by Eq. (65). By Lem. 11 and Lem. 12,

$$\Pr \left[\begin{array}{c} w_2, w'_2 \neq \perp \\ \wedge \pi(w_2) \neq \pi(w'_2) \end{array} \middle| \begin{array}{l} pp \leftarrow \mathcal{G}(1^\lambda) \\ \text{st}^* \leftarrow \mathcal{B}(pp) \\ \text{in} \leftarrow (pp, \text{st}^*) \\ (x_2, w_2), (x'_2, w'_2) \leftarrow \mathcal{W}_{\text{uniq}}^{\mathcal{O}_{\text{weak}}(\text{in})}(\text{in}) \end{array} \right] \\ \leq \text{Adv}_A^{\text{MSIS}_{q,n,d,rn\alpha,2b}}(\mathcal{B}_1, \lambda) + \text{Adv}_A^{\text{MSIS}_{q,n,d,rn\alpha,4 \cdot T_{\text{bck}} \cdot B}}(\mathcal{B}_2, \lambda)$$

which is $\leq \text{negl}(\lambda)$ assuming $\text{MSIS}_{q,n,d,rn\alpha,2b}$, $\text{MSIS}_{q,n,d,rn\alpha,4 \cdot T_{\text{bck}} \cdot B}$.

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.

Proof of Claim 3. We invoke Lemma 27 with the following object assignments

- $\theta = (pp, x_3, \text{st}_3)$.
- The oracle $\mathcal{O}(\theta)$ runs the interaction between $\mathcal{P}_3^*$ and $\mathcal{V}_3$ to produce (x_4, w_4) . If $w_4 \neq \perp$ and $(pp, x_4, w_4) \in \text{TE}(1, k, b) \times \text{TE}(1, \ell - k, B) \times \text{PE}(1, k, b)$, then the oracle returns $((s_i)_{i \in [k]}, (s_{\text{subs},i})_{i \in [k]})$ parsed from w_4 . Else, the oracle returns $\perp$.

- $\text{Bad}(\theta)$ is the set of $((s_i)_{i \in [k]}, (s_{\text{subs}, i})_{i \in [k]})$ such that at least one of the following conditions does NOT hold:

$$\begin{aligned}
v_{\text{bot}} &= \sum_{j=1}^{2^k} c^{(j)} \cdot f'_{\text{subs}}(j), \\
y_{\text{subs}} &= \sum_{j=1}^{2^k} \tilde{q}_{\text{subs}, j}(r) \cdot f_{\text{subs}}^{(j)}, \\
t^* &= \sum_{j=1}^{2^k} c^{(j)} \cdot (G_n \cdot \mathfrak{s}_{k, j}), \\
v_{\text{top}} &= \tilde{s}_{\text{top}}(u^{(\ell-k+1:\cdot)}).
\end{aligned} \tag{66}$$

By construction of $\mathcal{E}_3$, for $z \leftarrow \mathcal{O}(\theta)$,

$$z \neq \perp \wedge z \notin \text{Bad}(\theta) \implies (pp, x_3, w_3) \in \mathcal{R}_2.$$

And thus define the event

$$\text{Err}_1 := \{z \neq \perp \wedge z \in \text{Bad}(\theta)\} = \{z \in \text{Bad}(\theta)\}$$

because by definition of Bad , $z \neq \perp$. We can then bound the success probability of the extractor as desired:

$$\begin{aligned}
\Pr[(pp, x_3, w_3) \in \mathcal{R}_2] &\geq \Pr[z \neq \perp \mid \text{Err}_1] \\
&= \mathbb{E}_\theta[\Pr[z \neq \perp \mid \theta]] - \mathbb{E}_\theta[\text{Err}_1 \mid \theta] \quad (\text{because } \text{Err}_1 \subseteq z \neq \perp, \text{ linearity of expectation}) \\
&\geq \epsilon(\mathcal{A}_3, \mathcal{P}_3^*) - \mathbb{E}_\theta[\delta(\theta)] - \text{Adv}^{\text{uniq}}(\mathcal{S}_{\text{uniq}}). \quad (\text{def. of } \epsilon, \text{ Lem. 27})
\end{aligned}$$

where $\delta, \text{Adv}^{\text{uniq}}(\mathcal{S}_{\text{uniq}})$ are defined as in Lem. 27.

Extractor $\mathcal{E}_3(pp, x_3, st_3) \rightarrow w_3$
1. $(x_4, w_4) \leftarrow \langle \mathcal{P}_3^*, \mathcal{V}_3 \rangle(pp, x_3, st_3)$. 2. If $w_4 = \perp$, return $\perp$. 3. Else return $w_3 := w_4$.

We now compute $\mathbb{E}_\theta[\delta(\theta)]$ and $\text{Adv}(\mathcal{S}_{\text{uniq}})$. To compute $\text{Adv}(\mathcal{S}_{\text{uniq}})$, the event of the adversary's success is exactly a binding collision on either t or t_{subs} . And so this occurs with probability at most

$$\leq 2 \cdot \text{Adv}_A^{\text{MSIS}_{q,n,d,rn\alpha,2b}}(\mathcal{B}_1, \lambda).$$

by Lem. 11 applied to each commitment. Recall in Lem. 27 that for every fixed θ and fixed $z \in \text{Bad}(\theta)$,

$$\Pr[\mathcal{O}(\theta) = z] \leq \delta(\theta).$$

Eq. (66) holds if and only if the corresponding sum-check claims with challenge η replaced with indeterminate H hold:

$$\begin{aligned}
v_{\text{bot}} &= \sum_{X_k, Y, X_g \in \{0,1\}^{k+m}} \tilde{c}(X_k) \cdot \tilde{e}q(Y, 0^{\log n}) \cdot \tilde{p}_{b,\alpha}(X_g) \cdot \tilde{s}_{\text{subs}, k}(Y, X_k, X_g) \\
y_{\text{subs}} &= \sum_{X, X_g \in \{0,1\}^{k+m}} \left(\sum_{j=1}^{2^k} \tilde{q}_{\text{subs}, j}(r) \cdot \tilde{e}q(X, 0^{\log n} \parallel \text{bits}_k(j-1)) \cdot \tilde{p}_{b,\alpha}(X_g) \cdot \tilde{s}_{\text{subs}, k}(X, X_g) \right) \\
\langle p_{H,n}, t^* \rangle &= \sum_{X_1, X_c, X_n, X_g \in \{0,1\}^{k+1+m}} \tilde{c}(X_c) \cdot \tilde{p}_{H,n}(X_n) \cdot \tilde{p}_{b,\alpha}(X_g) \cdot \tilde{e}q(X_1, 1) \cdot \tilde{s}_{\text{top}}(X_1, X_c, X_n, X_g) \\
v_{\text{top}} &= \sum_{X \in \{0,1\}^{k+1+m}} \tilde{e}q(X, u^{(\ell-k+1:\cdot)}) \cdot \tilde{s}_{\text{top}}(X)
\end{aligned}$$

as described in Section 5.2.1. Further, the batched claim along indeterminates as constructed in the BatchSC protocol (Prot. 1) must also hold, because the indeterminates used for batching are linearly independent. Thus if any of the statements Eq. (66) do not hold, it must be the case the $\mathcal{P}_3^*$ either

- succeeded in BatchSC in Step 4, despite the batched sum-check claim not holding on indeterminates.
- OR the following non-zero polynomial evaluated to zero at random point $\eta \leftarrow_{\$} \mathcal{C}$:

$$P(H) := \langle p_{H,n}, t^* \rangle - \sum_{X_1, X_c, X_n, X_g \in \{0,1\}^{k+1+m}} \tilde{c}(X_c) \cdot \tilde{p}_{H,n}(X_n) \cdot \tilde{p}_{b,\alpha}(X_g) \cdot \tilde{e}q(X_1, 1) \cdot \tilde{s}_{\text{top}}(X_1, X_c, X_n, X_g).$$

Due to the soundness of BatchSC (see Lem. 24), the first bullet occurs with probability at most

$$\leq (4-1) \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right) + \frac{d/e-1}{|\mathbb{K}|} + \frac{(k+1+m) \cdot 2}{|\mathbb{K}|}$$

where the BatchSC in Step 4 takes in 4 sum-check claims of maximum degree 2 and maximum variable count $k+1+m$. By Schwartz-Zippel lemma and a union bound, the second bullet occurs with probability at most

$$\leq (n-1) \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right)$$

where $n-1$ is the degree of H in P . The probability that $\mathcal{P}_3^*$ cheats is

$$\mathbb{E}_{\theta}[\delta(\theta)] \leq 3 \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right) + \frac{d/e-1}{|\mathbb{K}|} + \frac{(k+1+m) \cdot 2}{|\mathbb{K}|} + (n-1) \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right)$$

by a union bound. Then invoking Lem. 27 on $\mathcal{E}_3$, we have shown Claim 3. $\square$

B.5 From Section 5.3: Norm Decomposition

Lemma 33 (Π_{Dec} completeness). *Let $\mu := \ell + 1 + \log h$. The norm decomposition reduction $\Pi_{\text{Dec}} : \text{TE}(1, \ell, b^h) \rightarrow \text{PE}(1, \mu, b)$ is complete, with prover time consisting of a $O(2^{\ell+1+\log h+m+\log n})$ sum-check prover and 1 tree commitment generation, $O(\log h + \ell + m + n^2\alpha + n)$ verifier time, and proof size consisting of 1 BatchSC with maximum degree 2 outputting 1 evaluation claim, 2 additional $R_{\mathbb{F}}$ claims, and 1 commitment.*

Proof. Let $(x := (t \in R_{\mathbb{F}}^n, u \in R_{\mathbb{F}}^{\ell+1+m}, v \in R_{\mathbb{F}}), w := (s_i \in R_{\mathbb{F}}^{2^{i n \alpha}})_{i \in [\ell]}) \in \text{TE}(1, \ell, b^h)$. Because $(x, w) \in \text{TE}(1, \ell, b^h)$, the sum-check claims

$$\begin{aligned} C_G : v_G &\stackrel{?}{=} \sum_{\substack{X_d, X_0, X, Y, X_g \\ \in \{0,1\}^{\mu+m}}} \tilde{p}_{b,h}(X_d) \cdot \tilde{e}q(X_0, 0) \cdot \tilde{p}_{\eta, 2^{\ell}}(X) \cdot \tilde{p}_{\xi, n}(Y) \cdot \tilde{p}_{b,\alpha}(X_g) \cdot \tilde{s}_{\text{dec}}(X_d, X_0, X, Y, X_g) \\ C_A : v_A &\stackrel{?}{=} \sum_{\substack{X_d, X_r, X, X_c \\ \in \{0,1\}^{\mu+m+\log n}}} \tilde{p}_{b,h}(X_d) \cdot \tilde{A}(X_r, X_c) \cdot \tilde{p}_{\eta, 2^{\ell}}(X) \cdot \tilde{p}_{H,n}(X_r) \cdot \tilde{s}_{\text{dec}}(X_d, X, X_c) \\ C_{\text{ev}} : v &\stackrel{?}{=} \sum_{X_d, X \in \{0,1\}^{\mu+m}} \tilde{p}_{b,h}(X_d) \cdot \tilde{e}q(u, X) \cdot \tilde{s}_{\text{dec}}(X_d, X) \\ C_{\text{pre}} : 0 &\stackrel{?}{=} \sum_{X_d, X_1, X_2 \in \{0,1\}^{\mu+m}} \tilde{p}_{b,h}(X_d) \cdot \tilde{e}q(0^{\ell}, X_1) \cdot \tilde{p}_{\gamma, 2n\alpha}(X_2) \cdot \tilde{s}_{\text{dec}}(X_d, X_1, X_2) \end{aligned}$$

are true on verifier challenges $\eta, \xi, \gamma \leftarrow_{\$} \mathcal{C}$ due to the commitment opening structure on $(t, (s_i)_{i \in [\ell]})$ and evaluation claim $\tilde{s}(u) = v$ from the input as described in Sec. 5.3.1. By the completeness of BatchSC (see Lem. 23), the sum-check verifier in Step 2 will pass along with the check in 2c due to Eq. (25), and reduce to a claim y_{dec} on $\tilde{s}_{\text{dec}}$ at r . Therefore the output $(x_{\text{dec}}, w_{\text{dec}}) \in \text{PE}(1, \mu, b)$. $\square$

Lemma 34 (Π_{Dec} knowledge soundness). *Let $\mu := \ell + 1 + \log h$. Assuming $\text{MSIS}_{q,n,d,2n\alpha,2b}$, the norm decomposition reduction $\Pi_{\text{Dec}} : \text{TE}(1, \ell, b^h) \rightarrow \text{PE}(1, \mu, b)$ is knowledge sound with negligible knowledge error.*

Proof. Let $(\mathcal{A}, \mathcal{P}^*)$ be an EPT adversary for Π_{Dec} with success probability $\epsilon(\mathcal{A}, \mathcal{P}^*) \geq 1/\text{poly}(\lambda)$. We construct extractor $\mathcal{E}^{\mathcal{P}^*}(pp, x_1, w_1)$ for Π_{Dec} in Fig. 10 with knowledge error ϵ_{Dec} , where

$$\begin{aligned} \epsilon_{\text{Dec}} \leq & \underbrace{\text{Adv}_A^{\text{MSIS}_{q,n,d,2n\alpha,2b}}(\mathcal{B}, \lambda)}_{\text{Claim 1}} \\ & + \underbrace{(2^\ell + n + 2n\alpha - 3) \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right)}_{\text{Claim 2}} \\ & + \underbrace{3 \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right) + \frac{d/e - 1}{|\mathbb{K}|} + \frac{(\ell + 1 + \log h + m + \log n) \cdot 2}{|\mathbb{K}|}}_{\text{Claim 3}}. \end{aligned} \quad (67)$$

where $\epsilon_{\text{inv}}(\mathcal{C})$ is the local invertibility error of $\mathcal{C}$ (see Def. 10).

Extractor $\mathcal{E}(pp, x_1, st) \rightarrow w_1$:
1. Simulate $(x_2, w_2) \leftarrow \langle \mathcal{P}^*, \mathcal{V} \rangle(pp, x_1, st)$. 2. If $x_2 = \perp$, or $(x_2, w_2) \notin \text{PE}(1, \mu, b)$, output $\perp$. 3. Parse $(s_{\text{dec},i}^*)_{i \in [\mu]} \leftarrow w_2$. Parse $(s_{\text{dec},1} \parallel \dots \parallel s_{\text{dec},h}) \leftarrow s_{\text{dec},\mu}^*$. 4. For all $j \in [h]$, parse $(g_j \parallel s_{\text{dec},1,j} \parallel \dots \parallel s_{\text{dec},\ell,j}) \leftarrow s_{\text{dec},j}$. 5. For each $i \in [\ell]$, $s_i := \sum_{j=1}^h b^{j-1} \cdot s_{\text{dec},i,j}$. 6. Output $w_1 := (s_i)_{i \in [\ell]}$.

Figure 10: Extractor for Π_{Dec} .

Extractor runtime. The extractor runs in expected polynomial time, since it simulates one execution between $\mathcal{P}^*$ and $\mathcal{V}$.

Extractor success probability. Assume that $\mathcal{P}^*$ succeeds in convincing $\mathcal{V}$, and so the first two steps of the extractor pass and both parties output $(x_2, w_2) \in \text{PE}(1, \mu, b)$. This occurs with probability $\epsilon(\mathcal{A}, \mathcal{P}^*) \geq 1/\text{poly}(\lambda)$. We apply Lem. 27. Assign the relevant objects:

- $\theta = (pp, x_1, st, t_{\text{dec}}^*, st')$ where st' is the prover and verifier state after Step 1.
- The oracle $\mathcal{O}(\theta)$ runs the interaction between $\mathcal{P}^*$ and $\mathcal{V}$ beginning after Step 1 where the t_{dec}^* message has already been sent by the prover. For (x_2, w_2) outputted by the rest of the execution, if $(pp, x_2, w_2) \in \text{PE}(1, \mu, b)$, the oracle returns w_2 . Else, it returns $\perp$.
- $\text{Bad}(\theta)$: Parse $(s_{\text{dec},i}^*)_{i \in [\mu]} \leftarrow w_2$. Then $\text{Bad}(\theta)$ is the set of w_2 such that for $s_{\text{dec},\mu}^*$ as parsed in Fig. 10, $g := \sum_{j=1}^h b^{j-1} \cdot g_j \neq 0^{2n\alpha}$ OR $(pp, x_1, w_1) \notin \text{TE}(1, \ell, b^h)$.

Then for the extractor in Fig. 10, observe that for $z \leftarrow \mathcal{O}(\theta)$,

$$z \neq \perp \wedge z \notin \text{Bad}(\theta) \implies (pp, x_1, w_1) \in \text{TE}(1, \ell, b^h).$$

And thus define the event

$$\text{Err}_1 := \{z \neq \perp \wedge z \in \text{Bad}(\theta)\} = \{z \in \text{Bad}(\theta)\}$$

because by definition of Bad , $z \neq \perp$. We can then bound the success probability of the extractor as desired:

$$\begin{aligned} \Pr[(pp, x_1, w_1) \in \text{TE}(1, \ell, b^h)] & \geq \Pr[z \neq \perp \setminus \text{Err}_1] \\ & = \mathbb{E}_\theta[\Pr[z \neq \perp \mid \theta]] - \mathbb{E}_\theta[\text{Err}_1 \mid \theta] \quad (\text{because } \text{Err}_1 \subseteq z \neq \perp, \text{ linearity of expectation}) \\ & \geq \epsilon(\mathcal{A}, \mathcal{P}^*) - \mathbb{E}_\theta[\delta(\theta)] - \text{Adv}^{\text{uniq}}_{\mathcal{S}_{\text{uniq}}}(\mathcal{S}_{\text{uniq}}). \end{aligned} \quad (\text{def. of } \epsilon, \text{ Lem. 27})$$

where $\delta, \text{Adv}^{\text{uniq}}(\mathcal{S}_{\text{uniq}})$ are defined as in Lem. 27.

Using the language of Lem. 27, we summarize the following claims. Claim 1 bounds the advantage of the uniqueness adversary $\text{Adv}^{\text{uniq}}(\mathcal{S}_{\text{uniq}})$. Claim 2 and Claim 3 bound $\mathbb{E}_\theta[\delta(\theta)]$.

Claim 1: The event of disagreement is exactly two different openings to t_{dec}^* . A binding collision on t_{dec}^* reduces to MSIS (see Lem. 11) and thus we can bound

$$\text{Adv}^{\text{uniq}}(\mathcal{S}_{\text{uniq}}) \leq \text{Adv}_A^{\text{MSIS}_{q,n,d,2n\alpha,2b}}(\mathcal{B}, \lambda).$$

Claim 2: First note that since $(x_2, w_2) \in \text{PE}(1, \mu, b)$, it always holds that $\|w_2\|_\infty < b$ which implies that recomposed $\|w_1\|_\infty < b^h$ due to the summation in Step 5.

Then for fixed w_2 in $\text{Bad}(\theta)$ with parsed out bottom layer $s_{\text{dec},\mu}^*$, it remains to check the commitment opening conditions and evaluation claim of $\text{TE}(1, \ell, |\mathbb{F}|/2)$, and $g = 0^{2n\alpha}$. The commitment opening conditions (i.e., Eq. (23) and Eq. (24)) hold if and only if Eq. (25) with the verifier challenges η, ξ replaced by indeterminates H, Z , respectively, holds. That is, define:

$$\begin{aligned} F_{\text{pre}}(G) &:= \sum_{j=1}^{2n\alpha} G^{j-1} \cdot g^{(j)} \\ F_A(H, Z) &:= \sum_{r=1}^n H^{r-1} \cdot \left(\sum_{i=0}^{\ell-1} \sum_{p=1}^{2^i} Z^{2^i+p-1} \cdot a_r^T \cdot \ddot{s}_{i+1,p} \right) \\ F_G(H, Z) &:= \sum_{r=1}^n H^{r-1} \cdot \left(\sum_{i=1}^{\ell-1} \sum_{p=1}^{2^i} Z^{2^i+p-1} \cdot g_b^T \cdot \dot{s}_{i,p}^{((r-1)\alpha+1:r\alpha)} \right). \end{aligned}$$

For a fixed $s_{\text{dec},\mu}^*$, define the polynomial

$$F(H, Z) := Z \langle p_{H,n}, t \rangle + F_G(H, Z) - F_A(H, Z)$$

corresponding to Eq. (25). Then $g = 0^{2n\alpha}$ if and only if $F_{\text{pre}}(G)$ is the zero polynomial, and (t, w_1) satisfy the commitment opening conditions of $\text{TE}(1, \ell, |\mathbb{F}|/2)$ if and only if $F(H, Z)$ is the zero polynomial. This can occur despite $s_{\text{dec},\mu}^* \in \text{Bad}(\theta)$ in the following ways.

First, if $g \neq 0^{2n\alpha}$, then either $F_{\text{pre}}(G)$ is not the zero polynomial, but the corresponding $F_{\text{pre}}(\gamma) = 0$ on random challenge $\gamma \leftarrow_{\mathcal{S}} \mathcal{C}$, or sum-check claim C_{pre} is false. Second, if $g = 0^{2n\alpha}$, but one of the commitment opening conditions fails, then either $F(H, Z)$ is not the zero polynomial, but the corresponding $F(\eta, \xi) = 0$ on random challenges $\eta, \xi \leftarrow_{\mathcal{S}} \mathcal{C}$, or at least one of sum-check claims C_A, C_G . Third, if $g = 0^{2n\alpha}$ and the commitment opening conditions all hold, then the evaluation claim of $\text{TE}(1, \ell, |\mathbb{F}|/2)$ must not hold. Then sum-check claim C_{ev} is false. We bound the probability of the sum-check claim failures in Claim 3.

The probability that $F_{\text{pre}}(G)$ or $F(H, Z)$ is not the zero polynomial, but evaluates to zero on random challenges is bounded by

$$\leq (2^\ell + n + 2n\alpha - 3) \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right)$$

due to a union bound and the Schwartz-Zippel lemma over rings (see Lem. 9), where $2^\ell - 1$ and $n - 1$ are the degrees of Z, H in $F(H, Z)$, and $2n\alpha - 1$ is the degree of G in $F_{\text{pre}}(G)$.

Claim 3: For the sum-check claim failures, recall that the prover sends claimed $v_A \stackrel{?}{=} F_A(\eta, \xi)$ and $v_G \stackrel{?}{=} F_G(\eta, \xi)$ for verifier challenges η, ξ . Consider the BatchSC in Step 2 where $\theta, s_{\text{dec},\mu}^*, v_A$ and v_G are all fixed. If $\mathcal{P}^*$ passes BatchSC

in Step 2 for these fixed objects, despite at least one of

$$\begin{aligned}
C_G : \quad v_G &\neq \sum_{\substack{X_d, X_0, X, Y, X_g \\ \in \{0,1\}^{\mu+m}}} \tilde{p}_{b,h}(X_d) \cdot \tilde{e}q(X_0, 0) \cdot \tilde{p}_{Z, 2^\ell}(X) \cdot \tilde{p}_{H,n}(Y) \cdot \tilde{p}_{b,\alpha}(X_g) \cdot \tilde{s}_{\text{dec}}(X_d, X_0, X, Y, X_g) \\
C_A : \quad v_A &\neq \sum_{\substack{X_d, X_r, X, X_c \\ \in \{0,1\}^{\mu+m+\log n}}} \tilde{p}_{b,h}(X_d) \cdot \tilde{A}(X_r, X_c) \cdot \tilde{p}_{Z, 2^\ell}(X) \cdot \tilde{p}_{H,n}(X_r) \cdot \tilde{s}_{\text{dec}}(X_d, X, X_c) \\
C_{\text{ev}} : \quad v &\neq \sum_{X_d, X \in \{0,1\}^{\mu+m}} \tilde{p}_{b,h}(X_d) \cdot \tilde{e}q(u, X) \cdot \tilde{s}_{\text{dec}}(X_d, X) \\
C_{\text{pre}} : \quad 0 &\neq \sum_{X_d, X_1, X_2 \in \{0,1\}^{\mu+m}} \tilde{p}_{b,h}(X_d) \cdot \tilde{e}q(0^\ell, X_1) \cdot \tilde{p}_{G, 2n\alpha}(X_2) \cdot \tilde{s}_{\text{dec}}(X_d, X_1, X_2),
\end{aligned}$$

then $\mathcal{P}^*$ has won against the soundness of BatchSC. Recall that $g = 0^{2n\alpha}$ is checked by C_{pre} and the evaluation claim in $\text{TE}(1, \ell, |\mathbb{F}|/2)$ is checked by C_{ev} . The probability of this bad event is

$$\leq (4-1) \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right) + \frac{d/e-1}{|\mathbb{K}|} + \frac{(\ell+1+\log h+m+\log n) \cdot 2}{|\mathbb{K}|}$$

by Lem. 24 over $a = 4$ claims of maximum degree 2 over maximum $(\ell+1+\log h+m+\log n)$ variables.

Putting Claims 1-3 together with Lem. 27,

$$\Pr[(pp, x_1, w_1) \in \text{TE}(1, \ell, b^h)] \geq \epsilon(\mathcal{A}, \mathcal{P}^*) - \epsilon_{\text{Dec}}.$$

where ϵ_{Dec} is defined as in Eq. (67). $\square$

B.6 From Section 5.4: Finishing Protocol

Lemma 35 (Π_{Reshape} is complete). $\Pi_{\text{Reshape}} : \text{TE}(1, k, b)^\omega \times \text{PE}(1, k, b)^{\omega+1} \rightarrow \text{PE}^{(2)}(b_1, b)$ is complete, with $O(\omega \cdot 2^{k+1+m+\log n})$ sum-check prover costs, $O(\omega \cdot (\log(2\omega+1+k+1+m+n^2\alpha+n)))$ verifier costs, and proof size consisting of 1 BatchSC with maximum degree 2 outputting 1 evaluation claim, 2 additional $R_{\mathbb{F}}$ claims, and 1 commitment.

Proof. Let $(x, w) \in \text{TE}(1, k, b)^\omega \times \text{PE}(1, k, b)^{\omega+1}$. Then the sum-check claims given in Step 2 are true on verifier challenges $\eta, \xi, \gamma, \zeta \leftarrow_{\mathcal{C}}$ due to the commitment opening structure on each commitment in (x, w) and evaluation claims $\tilde{s}_{\text{TE}, l}(u_{\text{TE}, l}) = v_{\text{TE}, l}, \forall l \in [\omega], \tilde{s}_{\text{PE}, l}(u_{\text{PE}, l}) = v_{\text{PE}, l}, \forall l \in [\omega+1]$ from the input as described in Sec. 5.4.2. By the completeness of BatchSC (see Lem. 23), the sum-check verifier in Step 2 will pass along with the check in 2c due to Eq. (32), and reduce to a claim y_{reshape} on $s_{\text{reshape}, 2}^*$ at r . Therefore the output $(x_{\text{reshape}}, w_{\text{reshape}}) \in \text{PE}^{(2)}(b_1, b)$. $\square$

Lemma 36 (Π_{Reshape} knowledge soundness). Assuming $\text{MSIS}_{q,n,d,2\gamma n\alpha_1, 2b_1}$, $\text{MSIS}_{q,n,d,2\beta\alpha_2, 2b}$, the commitment reshape reduction $\Pi_{\text{Reshape}} : \text{TE}(1, k, b)^\omega \times \text{PE}(1, k, b)^{\omega+1} \rightarrow \text{PE}^{(2)}(b_1, b)$ is knowledge sound with negligible knowledge error.

Proof. Let $(\mathcal{A}, \mathcal{P}^*)$ be an EPT adversary for Π_{Reshape} with success probability $\epsilon(\mathcal{A}, \mathcal{P}^*) \geq 1/\text{poly}(\lambda)$. We construct extractor $\mathcal{E}^{\mathcal{P}^*}(pp, x_1, w_1)$ for Π_{Reshape} in Fig. 11 with knowledge error $\epsilon_{\text{Reshape}}$, where

$$\begin{aligned}
\epsilon_{\text{Reshape}} &\leq \underbrace{\text{Adv}_A^{\text{MSIS}_{q,n,d,2\gamma n\alpha_1, 2b_1}}(\mathcal{B}, \lambda) + \text{Adv}_A^{\text{MSIS}_{q,n,d,2\beta\alpha_2, 2b}}(\mathcal{B}', \lambda)}_{\text{Claim 1}} \\
&\quad + \underbrace{(2^k + n - 2 + 2^{\lceil \log(2\omega+1) \rceil} - 1) \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right) + (2n\alpha - 1 + 2^{\lceil \log(2\omega+1) \rceil} - 1) \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right)}_{\text{Claim 2}} \\
&\quad + \underbrace{(2\omega+3) \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right) + \frac{d/e-1}{|\mathbb{K}|} + \frac{(k+1+\lceil \log(2\omega+1) \rceil + m + \log n) \cdot 2}{|\mathbb{K}|}}_{\text{Claim 3}}. \tag{68}
\end{aligned}$$

Extractor $\mathcal{E}(pp, x_1, st) \rightarrow w_1$:	
1. Simulate $(x_2, w_2) \leftarrow \langle \mathcal{P}^*, \mathcal{V} \rangle(pp, x_1, st)$.	
2. If $x_2 = \perp$, or $(x_2, w_2) \notin \text{PE}^{(2)}(b_1, b)$, output $\perp$.	
3. Parse $(s_{\text{reshape},1}^*, s_{\text{reshape},2}^*) \leftarrow w_2$.	
4. Parse	$s_{\text{reshape}} := s_{\text{TE},1} \parallel \dots \parallel s_{\text{TE},\omega} \parallel s_{\text{PE},1} \parallel \dots \parallel s_{\text{PE},\omega+1} \parallel g \in R_{\mathbb{F}}^{2^{\lceil \log(2\omega+1) \rceil + k+1} n\alpha} \leftarrow s_{\text{reshape},2}^*$
5. Parse	$g_{\text{TE},l} \parallel s_{\text{TE},1,l} \parallel \dots \parallel s_{\text{TE},k,l} \leftarrow s_{\text{TE},l} \quad \forall l \in [\omega]$ $g_{\text{PE},l} \parallel s_{\text{PE},1,l} \parallel \dots \parallel s_{\text{PE},k,l} \leftarrow s_{\text{PE},l} \quad \forall l \in [\omega+1]$
6. Output	$w_1 := \left(\left((s_{i,\text{TE},l} \in R_{\mathbb{F}}^{2^i n\alpha})_{i \in [k]} \right)_{l \in [\omega]}, \left((s_{i,\text{PE},l} \in R_{\mathbb{F}}^{2^i n\alpha})_{i \in [k]} \right)_{l \in [\omega+1]} \right).$

Figure 11: Extractor for Π_{Reshape} .

where $\epsilon_{\text{inv}}(\mathcal{C})$ is the local invertibility error of $\mathcal{C}$ (see Def. 10).

Extractor runtime. The extractor runs in expected polynomial time, since it simulates one execution between $\mathcal{P}^*$ and $\mathcal{V}$.

Extractor success probability. Assume that $\mathcal{P}^*$ succeeds in convincing $\mathcal{V}$, and so the first two steps of the extractor pass and both parties output $(x_2, w_2) \in \text{PE}^{(2)}(b_1, b)$. This occurs with probability $\epsilon(\mathcal{A}, \mathcal{P}^*) \geq 1/\text{poly}(\lambda)$. We apply Lem. 27. Assign the relevant objects, which reference the extractor in Fig. 11:

- $\theta = (pp, x_1, st, t_{\text{reshape}}^*, st')$ where st' is the prover and verifier state after Step 1.
- The oracle $\mathcal{O}(\theta)$ runs the interaction between $\mathcal{P}^*$ and $\mathcal{V}$ beginning after Step 1 where the t_{reshape}^* message has already been sent by the prover. For (x_2, w_2) outputted by the rest of the execution, if $(pp, x_2, w_2) \in \text{PE}^{(2)}(b_1, b)$, the oracle returns w_2 . Else, it returns $\perp$.
- $\text{Bad}(\theta)$: Parse $(s_{\text{reshape},i}^*)_{i \in [2]} \leftarrow w_2$. $\text{Bad}(\theta)$ is the set of w_2 such that for $s_{\text{reshape},2}^*$ as parsed in Fig. 11, $\exists l \in [\omega]$, $g_{\text{TE},l} \neq 0^{2n\alpha}$, OR $\exists l \in [\omega+1]$, $g_{\text{PE},l} \neq 0^{2n\alpha}$, OR $(pp, x_1, w_1) \notin \text{TE}(1, k, b)^\omega \times \text{PE}(1, k, b)^{\omega+1}$.

Then for the extractor in Fig. 11, observe that for $z \leftarrow \mathcal{O}(\theta)$,

$$z \neq \perp \wedge z \notin \text{Bad}(\theta) \implies \text{TE}(1, k, b)^\omega \times \text{PE}(1, k, b)^{\omega+1}.$$

And thus define the event

$$\text{Err}_1 := \{z \neq \perp \wedge z \in \text{Bad}(\theta)\} = \{z \in \text{Bad}(\theta)\}$$

because by definition of Bad , $z \neq \perp$. We can then bound the success probability of the extractor as desired:

$$\begin{aligned}
\Pr[(pp, x_1, w_1) \in \text{TE}(1, k, b)^\omega \times \text{PE}(1, k, b)^{\omega+1}] &\geq \Pr[z \neq \perp \setminus \text{Err}_1] \\
&= \mathbb{E}_\theta[\Pr[z \neq \perp \mid \theta]] - \mathbb{E}_\theta[\text{Err}_1 \mid \theta] \\
&\quad (\text{because } \text{Err}_1 \subseteq \{z \neq \perp\}, \text{ linearity of expectation}) \\
&\geq \epsilon(\mathcal{A}, \mathcal{P}^*) - \mathbb{E}_\theta[\delta(\theta)] - \text{Adv}_{\mathcal{A}}^{\text{uniq}}(\mathcal{S}_{\text{uniq}}). \quad (\text{def. of } \epsilon, \text{ Lem. 27})
\end{aligned}$$

where $\delta, \text{Adv}_{\mathcal{A}}^{\text{uniq}}(\mathcal{S}_{\text{uniq}})$ are defined as in Lem. 27.

Using the language of Lem. 27, we summarize the following claims. Claim 1 bounds the advantage of the uniqueness adversary $\text{Adv}_{\mathcal{A}}^{\text{uniq}}(\mathcal{S}_{\text{uniq}})$. Claim 2 and Claim 3 bound $\mathbb{E}_\theta[\delta(\theta)]$.

Claim 1: The event of disagreement is exactly two different openings to t_{reshape}^* . A binding collision on t_{reshape}^* (Def. 23) reduces to MSIS and thus we can bound

$$\text{Adv}_{\mathcal{A}}^{\text{uniq}}(\mathcal{S}_{\text{uniq}}) \leq \text{Adv}_A^{\text{MSIS}_{q,n,d,2^{\gamma}n\alpha_1,2b_1}}(\mathcal{B}, \lambda) + \text{Adv}_A^{\text{MSIS}_{q,n,d,2^{\beta}n\alpha_2,2b}}(\mathcal{B}', \lambda).$$

Claim 2: First note that since $(x_2, w_2) \in \text{PE}^{(2)}(b_1, b)$, it always holds that $\|s_{\text{reshape},2}^*\|_\infty < b$ which implies that recomposed $\|w_1\|_\infty < b$.

Then for fixed $w_2 \in \text{Bad}(\theta)$ with parsed out bottom layer $s_{\text{reshape},2}^*$, it remains to check the commitment opening conditions and evaluation claims of $\text{TE}(1, k, |\mathbb{F}|/2)^\omega \times \text{PE}(1, k, |\mathbb{F}|/2)^{\omega+1}$, and $g_{\text{TE},l} = 0^{2n\alpha}$, $g_{\text{PE},l} = 0^{2n\alpha}$. The commitment opening conditions hold if and only if Eq. (32) with the verifier challenges η, ξ, ζ replaced by indeterminates H, Z, R , respectively, holds. That is, define:

$$\begin{aligned} F_t(H, Z, R) &:= \sum_{l \in [\omega]} R^{l-1} \cdot \left(Z \cdot \sum_{r=1}^n H^{r-1} \cdot t_{\text{TE},l}^{(r)} \right) + \sum_{l \in [\omega+1]} R^{\omega-1+l} \cdot \left(Z \cdot \sum_{r=1}^n H^{r-1} \cdot t_{\text{PE},l}^{(r)} \right) \\ F_A(H, Z, R) &:= \sum_{l \in [\omega]} R^{l-1} \cdot \sum_{r=1}^n H^{r-1} \cdot \left(\sum_{i=0}^{k-1} \sum_{p=1}^{2^i} Z^{2^i+p-1} \cdot a_r^\top \cdot \ddot{s}_{i+1,p,\text{TE},l} \right) \\ &\quad + \sum_{l \in [\omega+1]} R^{\omega-1+l} \cdot \sum_{r=1}^n H^{r-1} \cdot \left(\sum_{i=0}^{k-1} \sum_{p=1}^{2^i} Z^{2^i+p-1} \cdot a_r^\top \cdot \ddot{s}_{i+1,p,\text{PE},l} \right) \\ F_G(H, Z, R) &:= \sum_{l \in [\omega]} R^{l-1} \cdot \left(\sum_{r=1}^n H^{r-1} \cdot \left(\sum_{i=1}^{k-1} \sum_{p=1}^{2^i} Z^{2^i+p-1} \cdot g_b^\top \cdot \dot{s}_{i,p,\text{TE},l}^{((r-1)\alpha+1:r\alpha)} \right) \right) \\ &\quad + \sum_{l \in [\omega+1]} R^{\omega-1+l} \cdot \left(\sum_{r=1}^n H^{r-1} \cdot \left(\sum_{i=1}^{k-1} \sum_{p=1}^{2^i} Z^{2^i+p-1} \cdot g_b^\top \cdot \dot{s}_{i,p,\text{PE},l}^{((r-1)\alpha+1:r\alpha)} \right) \right). \end{aligned}$$

(x_1, w_1) satisfy the commitment opening conditions of $\text{TE}(1, k, |\mathbb{F}|/2)^\omega \times \text{PE}(1, k, |\mathbb{F}|/2)^{\omega+1}$ if and only if $F(H, Z, R)$ is the zero polynomial. For a fixed $s_{\text{reshape},2}^*$, define the polynomial

$$F(H, Z, R) := F_t(H, Z, R) + F_G(H, Z, R) - F_A(H, Z, R)$$

corresponding to Eq. (32). Then $g_{\text{TE},l} = 0^{2n\alpha}$, $g_{\text{PE},l} = 0^{2n\alpha}$ hold if and only if $F_{\text{pre}}(G, R)$ is the zero polynomial, where the sum-check challenge γ has been replaced by indeterminate G :

$$F_{\text{pre}}(G, R) := \sum_{l \in [\omega]} R^{l-1} \cdot \sum_{j=1}^{2n\alpha} G^{j-1} \cdot g_{\text{TE},l}^{(j)} + \sum_{l \in [\omega+1]} R^{\omega-1+l} \cdot \sum_{j=1}^{2n\alpha} G^{j-1} \cdot g_{\text{PE},l}^{(j)}$$

This can occur despite $s_{\text{reshape},2}^* \in \text{Bad}(\theta)$ in the following ways.

First, if one of $g_{\text{TE},l} \neq 0^{2n\alpha}$, $g_{\text{PE},l} \neq 0^{2n\alpha}$, then either $F_{\text{pre}}(G, R)$ is not the zero polynomial, but the corresponding $F_{\text{pre}}(\gamma, \zeta) = 0$ on random challenge $\gamma, \zeta \leftarrow_{\mathcal{S}} \mathcal{C}$, or sum-check claim C_{pre} is false. Second, if all $g_{\text{TE},l} = 0^{2n\alpha}$, $g_{\text{PE},l} = 0^{2n\alpha}$ are satisfied, but one of the commitment opening conditions fails, then either $F(H, Z, R)$ is not the zero polynomial but $F(\eta, \xi, \zeta) = 0$ on random challenges $\eta, \xi, \zeta \leftarrow_{\mathcal{S}} \mathcal{C}$, or at least one of sum-check claims C_A, C_G . Third, if all of $g_{\text{TE},l} = 0^{2n\alpha}$, $g_{\text{PE},l} = 0^{2n\alpha}$ and the commitment opening conditions all hold, then one of the evaluation claims in $\text{TE}(1, k, |\mathbb{F}|/2)^\omega \times \text{PE}(1, k, |\mathbb{F}|/2)^{\omega+1}$ must not hold. Then sum-check claim C_{ev} is false. We bound the probability of the sum-check claim failures in Claim 3.

The probability that $F_{\text{pre}}(G, R)$ or $F(H, Z, R)$ is not the zero polynomial, but evaluates to zero on random challenges is bounded by

$$\leq (2^k + n - 2 + 2^{\lceil \log(2\omega+1) \rceil} - 1) \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right) + (2n\alpha - 1 + 2^{\lceil \log(2\omega+1) \rceil} - 1) \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right)$$

due to a union bound and the Schwartz-Zippel lemma over rings (see Lem. 9), where $2^k - 1, n - 1, 2^{\lceil \log(2\omega+1) \rceil} - 1$ are the degrees of Z, H, R in $F(H, Z, R)$, and $2n\alpha - 1, 2\omega$ are the degrees of G, R in $F_{\text{pre}}(G, R)$.

Claim 3: For the sum-check claim failures, recall that the prover sends claimed $v_A \stackrel{?}{=} F_A(\eta, \xi, \zeta)$ and $v_G \stackrel{?}{=} F_G(\eta, \xi, \zeta)$ for verifier challenges η, ξ, ζ . Consider the BatchSC in Step 2 where $\theta, s_{\text{reshape},2}^*, v_A$ and v_G are all fixed. If $\mathcal{P}^*$ passes

BatchSC in Step 2 for these fixed objects, despite at least one of

$$\begin{aligned}
C_G : \quad v_G &\neq \sum_{\substack{X_{\text{rsh}}, X_0, X, Y, X_g \\ \in \{0,1\}^{\mu'+m}}} \tilde{p}_{R,2\omega+1}(X_{\text{rsh}}) \cdot \tilde{e}q(X_0, 0) \cdot \tilde{p}_{Z,2^k}(X) \cdot \tilde{p}_{H,n}(Y) \\
&\quad \cdot \tilde{p}_{b,\alpha}(X_g) \cdot \tilde{s}_{\text{reshape}}(X_{\text{rsh}}, X_0, X, Y, X_g) \\
C_A : \quad v_A &\neq \sum_{\substack{X_{\text{rsh}}, X, X_c, X_r \\ \in \{0,1\}^{\mu'+m+\log n}}} \tilde{p}_{R,2\omega+1}(X_{\text{rsh}}) \cdot \tilde{A}(X_r, X_c) \cdot \tilde{p}_{Z,2^k}(X) \cdot \tilde{p}_{H,n}(X_r) \\
&\quad \cdot \tilde{s}_{\text{reshape}}(X_{\text{rsh}}, X, X_c) \\
\forall l \in [\omega], C_{\text{ev,TE},l} : \quad v_{\text{TE},l} &\neq \sum_{X_{\text{rsh}}, X \in \{0,1\}^{\mu'+m}} \tilde{e}q(\text{bits}_{\lceil \log(2\omega+1) \rceil}(l-1), X_{\text{rsh}}) \cdot \tilde{e}q(u_{\text{TE},l}, X) \cdot \tilde{s}_{\text{reshape}}(X_{\text{rsh}}, X) \\
\forall l \in [\omega+1], C_{\text{ev,PE},l} : \quad v_{\text{PE},l} &\neq \sum_{X_{\text{rsh}}, X \in \{0,1\}^{\mu'+m}} \tilde{e}q(\text{bits}_{\lceil \log(2\omega+1) \rceil}(\omega-1+l), X_{\text{rsh}}) \cdot \tilde{e}q(1 \| u_{\text{PE},l}, X) \cdot \tilde{s}_{\text{reshape}}(X_{\text{rsh}}, X) \\
C_{\text{pre}} : \quad 0 &\neq \sum_{X_{\text{rsh}}, X_1, X_2 \in \{0,1\}^{\mu'+m}} \tilde{p}_{R,2\omega+1}(X_{\text{rsh}}) \cdot \tilde{e}q(0^k, X_1) \cdot \tilde{p}_{G,2n\alpha}(X_2) \cdot \tilde{s}_{\text{reshape}}(X_{\text{rsh}}, X_1, X_2)
\end{aligned}$$

then $\mathcal{P}^*$ has won against the soundness of BatchSC. The probability of this bad event is

$$\leq (2\omega+4-1) \cdot \left(\frac{1}{|\mathcal{C}|} + \epsilon_{\text{inv}}(\mathcal{C}) \right) + \frac{d/e-1}{|\mathbb{K}|} + \frac{(k+1 + \lceil \log(2\omega+1) \rceil + m + \log n) \cdot 2}{|\mathbb{K}|}$$

due to the soundness of BatchSC (see Lem. 24) over $a = 2\omega+4$ claims of maximum degree 2 over maximum $(k+1 + \lceil \log(2\omega+1) \rceil + m + \log n)$ variables.

Putting Claims 1-3 together with Lem. 27,

$$\Pr[(pp, x_1, w_1) \in \text{TE}(1, k, b)^\omega \times \text{PE}(1, k, b)^{\omega+1}] \geq \epsilon(\mathcal{A}, \mathcal{P}^*) - \epsilon_{\text{Reshape}}.$$

where $\epsilon_{\text{Reshape}}$ is defined as in Eq. (68). □

C Average Approximate Strong Sampling Sets

In this section, we look at a weaker, average notion of approximate sampling sets (cf. Def. 10). We provide a corresponding lemma regarding the coordinate-wise extraction error (cf. Def. 30), in which we are only able to prove a square-root rather than linear loss in the invertibility error $\sqrt{\epsilon_{\text{inv}}}$.

Definition 24 (Average approximate sampling sets [GLLO26]). *Let R be a ring and let $\mathcal{C} \subseteq R$ be a finite set with $|\mathcal{C}| \geq 2$. Let $R^\times$ denote the group of units of R (i.e., invertible elements), and define*

$$\epsilon_{\text{inv}}(\mathcal{C}) := \Pr_{\substack{a, b \leftarrow_{\S} \mathcal{C}, \\ a \neq b}} [a - b \notin R^\times].$$

We say that $\mathcal{C}$ is an approximate strong sampling set of R if $\epsilon_{\text{inv}}(\mathcal{C}) \leq \text{negl}(\lambda)$.

Lemma 37 (Coordinate-wise extraction for average approximate strong sampling sets). *Let $\ell \in \mathbb{N}$, and $\mathcal{C} \subseteq R_{\mathbb{F}}$ be an average approximate strong sampling set (see Def. 24) of cardinality N with $\Pr_{a, b \leftarrow_{\S} \mathcal{C}, a \neq b} [a - b \notin R_{\mathbb{F}}^\times] = \epsilon_{\text{inv}}$. Take any verification function $V : \mathcal{C}^\ell \times \{0,1\}^* \rightarrow \{0,1\}$. Then there exists an oracle algorithm $\mathcal{E}$ with the following properties: the algorithm $\mathcal{E}^{\mathcal{A}}$, given oracle access to a (probabilistic) algorithm $\mathcal{A} : \mathcal{C}^\ell \rightarrow \{0,1\}^*$, requires an expected number of at most $\ell+1$ queries to $\mathcal{A}$ and with probability at least*

$$\epsilon^V(\mathcal{A}) - \ell \cdot \left(\frac{1}{N} + \sqrt{\epsilon_{\text{inv}}} \right)$$

outputs $\ell+1$ pairs $(\vec{c}_0, y_0), (\vec{c}_1, y_1), \dots, (\vec{c}_\ell, y_\ell)$ such that

- $V(c_i, y_i) = 1$ for all $i \in [0, \ell]$,
- and $\{\vec{c}_0, \dots, \vec{c}_\ell\}$ satisfy the coordinate-wise property in Def. 16,

- and for all $i \in [\ell]$, $(c_{0,i} - c_{i,i}) \in R_{\mathbb{F}}^{\times}$.

Note that if $\mathcal{C}$ is an exact strong sampling set, this lemma is identical to [FMN24, Lemma 7.1] (Thm. 5).

Proof. We modify the extractor of Lemma 7.1 of [FMN24] in Fig. 12 to abort in Step 4(e) when the difference of challenges in non-invertible.

Extractor $\mathcal{E}^{\mathcal{A}}$
1. Sample $\vec{c}_0 := (c_{0,1}, \dots, c_{0,\ell}) \leftarrow_{\\$} \mathcal{C}^{\ell}$. 2. $y_0 \leftarrow \mathcal{A}(\vec{c}_0)$. 3. If $V(\vec{c}_0, y_0) = 0$, abort. 4. For $i = 1, \dots, \ell$, repeat: (a) Sample $c_i^* \leftarrow \mathcal{C} \setminus \{c_{0,i}\}$ without replacement. (b) $\vec{c}_i := (c_{0,1}, \dots, c_{0,i-1}, c_i^*, c_{0,i+1}, \dots, c_{0,\ell})$. (c) $y_i \leftarrow \mathcal{A}(\vec{c}_i)$. (d) If $V(c_i, y_i) = 0$, go to Step 4(a). (e) Else, if $(c_{0,i} - c_i^*) \notin R_{\mathbb{F}}^{\times}$, abort. until another successful c_i^* has been found (increment i to continue loop), or until all c_i^* have been tried (in which case, abort). 5. Return the corresponding $(\vec{c}_0, y_0), (\vec{c}_i, y_i)_{i \in [\ell]}$.

Figure 12: Extractor for Lemma 37.

Extractor runtime. The number of queries made by the extractor in Fig. 12 is upper bounded by the number of queries made by the extractor in Lemma 7.1 of [FMN24], as the only change is an additional abort condition in Step 4(e) which does not increase the number of queries made.

Extractor success probability. We denote by $C_0 := (C_{0,1}, \dots, C_{0,\ell})$ the random variable for the first challenge sampled by $\mathcal{E}$. Also, define the random variable for V 's acceptance on this first challenge to be $\Gamma := V(C_0, \mathcal{A}(C_0))$, and observe that $\Pr[\Gamma = 1] = \epsilon^V(\mathcal{A})$. Define the random variable s_i :

$$S_i := \{x \in \mathcal{C} : V(C_i(x), \mathcal{A}(C_i(x))) = 1\},$$

$$s_i := |S_i|$$

where $C_i(x) := (C_{0,1}, \dots, C_{0,i-1}, x, C_{0,i+1}, \dots, C_{0,\ell})$. Given $\Gamma = 1$, Fail_i to be the event that extraction at coordinate i fails (if it were to reach coordinate i in the Step 4 loop). This can be split into two cases

- $\Gamma = 1$, but for some $i \in [\ell]$, $s_i \leq 1$,
- $\Gamma = 1$ and $s_i \geq 2$, but the first selected c_i^* in Step 4(e) is such that $(c_{0,i} - c_i^*) \notin R_{\mathbb{F}}^{\times}$.

The success probability of $\mathcal{E}^{\mathcal{A}}$ is

$$\begin{aligned} \Pr[\Gamma = 1 \wedge (\bigwedge_{i=1}^{\ell} \text{Fail}_i^c)] &= \Pr[\Gamma = 1] - \Pr[\Gamma = 1 \wedge (\bigvee_{i=1}^{\ell} \text{Fail}_i)] && \text{(De Morgan's laws)} \\ &\geq \Pr[\Gamma = 1] - \sum_{i=1}^{\ell} \Pr[\Gamma = 1 \wedge \text{Fail}_i]. && \text{(distributive property, union bound)} \end{aligned}$$

(69)

For a fixed $i \in [\ell]$, we now upper bound $\Pr[\Gamma = 1 \wedge \text{Fail}_i]$ by partitioning the event into two cases: $s_i \leq 1$ and $s_i \geq 2$.

$s_i \leq 1$: If $s_i = 0$, extraction always fails ($\Gamma = 0$). If $s_i = 1$, the event $\Gamma = 1 \wedge \text{Fail}_i$ occurs when the first challenge sampled is exactly the single challenge in S_i . This happens with probability $1/N$:

$$\begin{aligned} \Pr[\Gamma = 1 \wedge \text{Fail}_i \mid s_i \leq 1] &= \underbrace{\Pr[\Gamma = 1 \wedge \text{Fail}_i \mid s_i = 0]}_0 \cdot \Pr[s_i = 0 \mid s_i \leq 1] \\ &\quad + \Pr[\Gamma = 1 \wedge \text{Fail}_i \mid s_i = 1] \cdot \underbrace{\Pr[s_i = 1 \mid s_i \leq 1]}_{\leq 1} \\ &\leq \Pr[\Gamma = 1 \wedge \text{Fail}_i \mid s_i = 1] = \frac{1}{N} \end{aligned} \quad (70)$$

$s_i \geq 2$: In this case, it is guaranteed that at some point Step 4(d) will succeed and $V(c_i, y_i) = 1$; i.e., the loop will reach some $c_i^* \in S_i$. Then, the event Fail_i occurs exactly when $(c_{0,i} - c_i^*) \notin R_{\mathbb{F}}$. Define the bad set of non-invertible differences of challenges:

$$\text{Bad}(\mathcal{C}) = \{a, b \in \mathcal{C} : a \neq b \wedge a - b \notin R_{\mathbb{F}}^\times\} \subseteq \mathcal{C} \times \mathcal{C}.$$

Now we can bound the probability of extraction failure at coordinate i for a fixed S such that $s = |S| \geq 2$,

$$\begin{aligned} \Pr[\Gamma = 1 \wedge \text{Fail}_i \mid S_i = S] &\leq \sum_{a \in S} \underbrace{\Pr[C_{0,i} = a \mid S_i = S]}_{1/N} \cdot \Pr_{c_i^* \leftarrow S \setminus \{a\}}[a - c_i^* \notin R_{\mathbb{F}}^\times \mid C_{0,i} = a, S_i = S] \end{aligned}$$

summing across S , the set of accepting challenges. Observe that c_i^* is the first element drawn from $S \setminus \{a\}$ in the loop.

$$\begin{aligned} &= \sum_{a \in S} \frac{1}{N} \cdot \frac{|\{b \in S \setminus \{a\} \text{ s.t. } a - b \notin R_{\mathbb{F}}^\times\}|}{s-1} \\ &= \frac{|\{(a, b) \in S \times S \mid a \neq b\} \cap \text{Bad}(\mathcal{C})|}{N \cdot (s-1)} \end{aligned}$$

using the definition of $\text{Bad}(\mathcal{C})$.

$$\leq \frac{\min\{s \cdot (s-1), |\text{Bad}(\mathcal{C})|\}}{N \cdot (s-1)}$$

because $|\{(a, b) \in S \times S \mid a \neq b\}| = s \cdot (s-1)$ for any S of size s .

$$= \min\left\{\frac{s}{N}, \frac{|\text{Bad}(\mathcal{C})|}{N \cdot (s-1)}\right\}$$

We now consider two cases: $s-1 \leq \sqrt{|\text{Bad}(\mathcal{C})|}$ and $s-1 > \sqrt{|\text{Bad}(\mathcal{C})|}$.

$$\begin{aligned} s-1 \leq \sqrt{|\text{Bad}(\mathcal{C})|} &\implies \frac{s}{N} \leq \frac{\sqrt{|\text{Bad}(\mathcal{C})|} + 1}{N} \\ s-1 > \sqrt{|\text{Bad}(\mathcal{C})|} &\implies \frac{|\text{Bad}(\mathcal{C})|}{N \cdot (s-1)} \leq \frac{|\text{Bad}(\mathcal{C})|}{N \cdot \sqrt{|\text{Bad}(\mathcal{C})|}} = \frac{\sqrt{|\text{Bad}(\mathcal{C})|}}{N}. \end{aligned}$$

In both cases, $\min\left\{\frac{s}{N}, \frac{|\text{Bad}(\mathcal{C})|}{N \cdot (s-1)}\right\} \leq \frac{\sqrt{|\text{Bad}(\mathcal{C})|} + 1}{N}$. What remains is to express $|\text{Bad}(\mathcal{C})|$ in terms of ϵ_{inv} .

$$|\text{Bad}(\mathcal{C})| \leq \epsilon_{\text{inv}} \cdot (N^2 - N),$$

because there are $N(N-1)$ ordered pairs (a, b) such that $a \neq b$. Then as given in the definition of strong sampling set (Def. 24), the fraction of these that are bad is ϵ_{inv} . And so averaging over S such that $|S| = s \geq 2$,

$$\Pr[\Gamma = 1 \wedge \text{Fail}_i \mid s_i \geq 2] \leq \frac{\sqrt{\epsilon_{\text{inv}} \cdot (N^2 - N)} + 1}{N} \quad (71)$$

Putting all these cases together, we lower bound the success probability of extractor $\mathcal{E}^{\mathcal{A}}$:

$$\begin{aligned}
\Pr[\Gamma = 1 \wedge (\wedge_{i=1}^{\ell} \text{Fail}_i^c)] &\geq \Pr[\Gamma = 1] - \sum_{i=1}^{\ell} \Pr[\Gamma = 1 \wedge \text{Fail}_i] && \text{(by Eq. (69))} \\
&= \Pr[\Gamma = 1] - \sum_{i=1}^{\ell} \left(\Pr[\Gamma = 1 \wedge \text{Fail}_i \mid s_i \leq 1] \cdot \Pr[s_i \leq 1] \right. \\
&\quad \left. + \Pr[\Gamma = 1 \wedge \text{Fail}_i \mid s_i \geq 2] \cdot \Pr[s_i \geq 2] \right) \\
&\geq \Pr[\Gamma = 1] - \sum_{i=1}^{\ell} \left(\frac{1}{N} \cdot \Pr[s_i \leq 1] + \frac{1 + \sqrt{\epsilon_{\text{inv}} \cdot (N^2 - N)}}{N} \cdot \Pr[s_i \geq 2] \right) \\
&\quad \text{(by Eq. (70), (71))} \\
&\geq \Pr[\Gamma = 1] - \sum_{i=1}^{\ell} \frac{1 + \sqrt{\epsilon_{\text{inv}} \cdot (N^2 - N)}}{N} \cdot \underbrace{(\Pr[s_i \leq 1] + \Pr[s_i \geq 2])}_1 \\
&\geq \Pr[\Gamma = 1] - \ell \cdot \frac{1 + \sqrt{\epsilon_{\text{inv}} \cdot (N^2 - N)}}{N} \\
&\geq \epsilon^V(\mathcal{A}) - \ell \cdot \left(\frac{1}{N} + \sqrt{\epsilon_{\text{inv}}} \right).
\end{aligned}$$

□