LatticeFold: A Lattice-based Folding Scheme and its Applications to Succinct Proof Systems

Dan Boneh and Binyi Chen

Stanford University

May 27, 2025

Abstract

Folding is a recent technique for building efficient recursive SNARKs. Several elegant folding protocols have been proposed, such as Nova, Supernova, Hypernova, Protostar, and others. However, all of them rely on an additively homomorphic commitment scheme based on discrete log, and are therefore not post-quantum secure and require a large (256-bit) field. In this work we present LatticeFold, the first lattice-based folding protocol based on the Module SIS problem. This folding protocol naturally leads to an efficient recursive lattice-based SNARK and an efficient PCD scheme. LatticeFold supports folding low-degree relations, such as R1CS, as well as high-degree relations, such as CCS. The key challenge is to construct a secure folding protocol that works with the Ajtai commitment scheme. The difficulty is ensuring that extracted witnesses are low norm through many rounds of folding. We present a novel technique using the sumcheck protocol to ensure that extracted witnesses are always low norm no matter how many rounds of folding are used. Since LatticeFold can operate over a small (64-bit) field, our evaluation of the final proof system suggests that it is as performant as Hypernova, while providing plausible post-quantum security. Moreover, LatticeFold operates over the same module structure used by fully homomorphic encryption (FHE) and lattice signatures schemes, and can therefore benefit from software optimizations and custom hardware designed to accelerate these lattice schemes.

Contents

1 Introduction

In recent years we have seen tremendous progress in the design of succinct non-interactive arguments of knowledge (SNARKs). They have become an important enabling technology for scaling blockchains, bridging between chains [Xie+22], authenticating media [NT16; DB22; KHSS22], verifiable delay functions [BBBF18], and much more. Some SNARKs are monolithic and generate the entire proof at once, while others break the task of constructing a proof into small steps and prove each step separately. The latter approach is called incrementally verifiable computation (IVC) [Val08] or proof carrying data (PCD) [CT10]. This approach eliminates the high memory needs of a monolithic SNARK. It can also provide more opportunities for parallelizing the prover.

Historically, IVC and PCD were built from a recursive SNARK [Val08; BCTV14]. However, this requires embedding the SNARK verifier inside the statement being proved at every step, and this introduces a considerable overhead. A new approach called *accumulation* or *folding* was recently introduced in Halo [BGH19] and further developed in [BCMS20; Bün+21; BDFG21] and Nova [KST22]. The idea is to "fold" the SNARK verification work at every step into the SNARK verification of all previous steps. The final folded statement is verified at the end of the computation. The benefit is that now the recursive statement being proved at every step only needs to ensure that folding was performed correctly, which is far simpler than running a full SNARK verifier. Since folding was introduced, many elegant ideas appeared to further optimize this technique [KS22; KS23b; BC23; RZ22; KS23a; KP23; Moh23; NBS23].

To explain folding in more detail we find it convenient to use the language of *reductions of knowledge* introduced by Kothapalli and Parno [KP23] (see Section 2.5 for details). Let $\mathcal{R}_1$ and $\mathcal{R}_2$ be two instance-witness relations. A reduction of knowledge from $\mathcal{R}_1$ to $\mathcal{R}_2$ is a protocol Π between a prover and verifier. The verifier takes as input an instance $\mathbb{x}_1$ for $\mathcal{R}_1$, interacts with the prover, and outputs an instance $\mathbb{x}_2$ for $\mathcal{R}_2$ at the end of the protocol. The key requirement is that if the prover can present a witness $\mathbb{w}_2$ for $\mathbb{x}_2$, then it is possible to extract from the prover a witnesses $\mathbb{w}_1$ for $\mathbb{x}_1$. Hence, knowledge of a valid witness for $\mathbb{x}_2$ proves knowledge of a valid witness for $\mathbb{x}_1$.

A folding scheme is a reduction of knowledge from a product relation $\mathcal{R}_{\mathsf{acc}} \times \mathcal{R}_{\mathsf{comp}}$ to $\mathcal{R}_{\mathsf{acc}}$. That is, two instances $(\mathbb{x}_{\mathsf{acc}}, \mathbb{x}_{\mathsf{comp}})$ are folded to a single instance $\mathbb{x}'_{\mathsf{acc}}$ of $\mathcal{R}_{\mathsf{acc}}$. By repeatedly folding in this way, the prover can accumulate many steps of a computation into a single instance of an accumulation relation $\mathcal{R}_{\mathsf{acc}}$. Eventually, the prover proves knowledge of a witness for the final $\mathcal{R}_{\mathsf{acc}}$ instance, and this proves knowledge of a valid witness for every step of the computation. When $\mathcal{R}_{\mathsf{acc}}$ and $\mathcal{R}_{\mathsf{comp}}$ are different, this type of folding is sometimes called multi-folding [KS23b]. The relation $\mathcal{R}_{\mathsf{acc}}$ is typically a simple extension of $\mathcal{R}_{\mathsf{comp}}$.

The Hypernova system [KS23b], for example, is a folding scheme for proving validity of a multi-step computation where the computation step relation $\mathcal{R}_{\mathsf{comp}}$ is expressed as a customizable constraint system (CCS) [STW23a]. CCS supports high-degree gates and

generalizes the Plonkish, R1CS, and AIR formats for a computation trace. By repeatedly folding, Hypernova enables the prover to accumulate many CCS steps into a single instance of a closely related relation $\mathcal{R}_{\mathsf{acc}}$.

The folding schemes discussed above make use of an additively homomorphic commitment scheme based on discrete log to commit to the various witnesses. The commitments are part of the instances $\mathbb{x}_{\mathsf{acc}}$ and $\mathbb{x}_{\mathsf{comp}}$. Due to the reliance on discrete log, the derived SNARKs are unsound in the presence of a large fault-tolerant quantum computer. Moreover, committing to a long vector with a discrete log commitment scheme, such as Pedersen, leads to significant work for the prover.

Our contributions. We construct LatticeFold, the first lattice-based folding scheme, whose security depends on the Module Short Integer Solution (MSIS) problem [LS15; PR06; LM06]. This problem is believed to be post-quantum secure. A key component of LatticeFold is a new batched proof-of-knowledge protocol for short pre-images of linear maps (See Section 3), which may be of independent interest.

A natural starting point for a lattice-based folding scheme is to replace the discrete-log commitment in existing folding schemes with the Ajtai commitment scheme [Ajt96], which is additively homomorphic. We describe the scheme as it operates in a module $\mathcal{R}^m$ defined over a suitable number ring $\mathcal{R}$. As usual, for a prime q we let $\mathcal{R}_q := \mathcal{R}/q\mathcal{R}$. The Ajtai commitment scheme works as follows (see Section 2.3):

- The public parameters contain a random matrix $\mathbf{A} \in \mathcal{R}_q^{\kappa \times m}$ where $\kappa < m$,
- The commitment to a vector $\vec{\mathbf{x}} \in \mathcal{R}^m$ is $\mathsf{cm} := \mathbf{A}\vec{\mathbf{x}} \in \mathcal{R}_q^{\kappa}$.

If the Module SIS (MSIS) problem is hard, then the commitment is binding for the set of vectors $\vec{\mathbf{x}} \in \mathcal{R}^m$ whose norm $\|\vec{\mathbf{x}}\|_\infty$ is at most some bound B. Throughout the paper we always use the L_∞ norm on $\mathcal{R}^m$, as defined in Section 2.

But one immediately runs into trouble. Folding two witnesses into one is done by taking a random linear combination of the two witnesses, using verifier randomness. Consequently, the norm of the committed vector in the folded instance increases the more times we fold. Eventually the norm exceeds the norm bound B, at which point the commitment scheme is no longer binding. One can try to avoid norm growth by using a folding tree [RZ22], so that the folding depth is logarithmic in the size of the computation. However, long folding chains are required in applications, such as PCD, and Ajtai commitments are simply not compatible with that. The challenge is to use Ajtai commitments while controlling the norm growth as folding takes place.

Our approach to keeping the witness norm below B is to break the folding protocol into three steps: expansion, decomposition, and folding. The first step has to do with the mechanics of folding; it expands the given instance $\mathbb{x}_{\mathsf{comp}}$ of $\mathcal{R}_{\mathsf{comp}}$ to an instance of $\mathcal{R}_{\mathsf{acc}}$. The second, and more important step, decomposes a committed witness $\vec{\mathbf{f}} \in \mathcal{R}^m$ of bounded norm B into a tuple of vectors $\vec{\mathbf{f}}_0, \ldots, \vec{\mathbf{f}}_{k-1} \in \mathcal{R}^m$ of lower norm $b := \lceil B^{1/k} \rceil$. This decomposition works by writing every entry of $\vec{\mathbf{f}}$ in base b, so that the original $\vec{\mathbf{f}}$ satisfies

$\vec{\mathbf{f}} = \vec{\mathbf{f}}_0 + b \cdot \vec{\mathbf{f}}_1 + \ldots + b^{k-1} \cdot \vec{\mathbf{f}}_{k-1}$, and each of the k vectors has norm less than b. When folding two committed witnesses of bounded norm B, this decomposition leaves us with $2k$ vectors of lower bounded norm b. Our third step, called folding, now folds all $2k$ vectors into a single witness for the accumulator relation $\mathcal{R}_{\mathsf{acc}}$. The folding is done by computing a random linear combination of the $2k$ vectors using a random vector of weights $\vec{\rho}$ sampled as $\vec{\rho} \xleftarrow{\$} \mathcal{C}_{\mathsf{small}}^{2k}$. Here $\mathcal{C}_{\mathsf{small}} \subseteq \mathcal{R}_q$ contains only ring elements of low norm so that the final folded witness $\vec{\mathbf{f}}' := \sum_{i=1}^{2k} \rho_i \vec{\mathbf{f}}_i$ has norm at most B. This gives a reduction of knowledge from $\mathcal{R}_{\mathsf{acc}} \times \mathcal{R}_{\mathsf{comp}}$ to $\mathcal{R}_{\mathsf{acc}}$ where the final committed witness satisfies the same norm bound as the original committed witnesses. There is no norm growth.

Unfortunately, this decomposition approach is insufficient: given a witness for the folded instance $\mathbb{x}'_{\mathsf{acc}}$ of $\mathcal{R}_{\mathsf{acc}}$ we cannot extract low-norm witnesses for the two instances $(\mathbb{x}_{\mathsf{acc}}, \mathbb{x}_{\mathsf{comp}})$ that we started with. The problem is that the extractor uses the inverses of elements $c_1 - c_2 \in \mathcal{R}_q$ where $c_1, c_2 \in \mathcal{C}_{\mathsf{small}}$. This forces us to ensure that $\mathcal{C}_{\mathsf{small}}$ is a strong sampling set, meaning that for all $c_1, c_2 \in \mathcal{C}_{\mathsf{small}}$, the difference $c_1 - c_2$ is invertible in $\mathcal{R}_q$. The ring $\mathcal{R}_q$ contains an exponential size strong sampling set (Lemma 2.3), and therefore the challenge space is sufficiently large. However, the norm of $1/(c_1 - c_2)$ in $\mathcal{R}_q$ can be large, and consequently the extractor might end up extracting a high norm witness, which is invalid. One way to solve this problem (e.g., as in [ACK21]) is to ensure that these inverses always have small norm. However, as noted in [AL21], that severely limits the size of the challenge set $\mathcal{C}_{\mathsf{small}}$ and harms the soundness of the folding protocol. In comparison, we highlight that our protocol incurs no slack in witness extraction and avoids using subtractive sets [AL21].

Our core idea is to enhance the folding protocol, and have the prover convince the verifier that it has $2k$ valid witnesses whose norm is less than b. This is sufficient to extract low norm witnesses from the prover. Roughly speaking, the prover can convince the verifier that a vector $\vec{\mathbf{f}} \in \mathcal{R}^m$ has norm less than b, by proving that every component u of $\vec{\mathbf{f}}$ is in the set $[-b, b]$. This is done by proving that $g(u) = 0$, where $g(X)$ is the polynomial $g(X) := X \prod_{i \in [b]} (X - i)(X + i)$. The set of zeroes of this polynomial g is exactly the set $[-b, b]$, and therefore $g(u) = 0$ if and only if u is in $[-b, b]$. By encoding the components of $\vec{\mathbf{f}}$ as the evaluations of a function h on the Boolean hypercube $\{0,1\}^\ell$, the prover can use the sumcheck protocol [LFKN92] on the ℓ-variate polynomial $g(h(\cdot))$ to convince the verifier that $\vec{\mathbf{f}}$ has norm less than b. In other words, the sumcheck protocol is the key tool that enables to prove that $\vec{\mathbf{f}}$ has bounded norm. The complete details are provided in Section 3.

We note that choosing the norm bound b is an interesting optimization problem. On the one hand, a small value of b results in a decomposition of $\vec{\mathbf{f}}$ into many fragments, and this will slow down the folding process because more witnesses need to be folded. On the other hand, choosing a small b reduces the degree of the polynomial $g(X)$ in the norm bound test, making that test faster. The optimal b needs to balance these two effects to minimize the overall running time. We calculate optimal values in our evaluation section.

Finally, we point out that our techniques are generic, and can be used to build folding schemes from any binding commitment that requires norm bounds on the committed vector.

Here we use Ajtai commitments, but other schemes can also be used.

Paper organization. We begin in Section 3 by using the techniques outlined above to construct a folding scheme for the relation $\mathcal{R}_{\mathsf{cm}}^{B}$ that captures the fact that the prover has an opening $\vec{\mathbf{x}} \in \mathcal{R}^m$ to an Ajtai commitment $\mathsf{cm} \in \mathcal{R}_q^\kappa$, where $\|\vec{\mathbf{x}}\|_\infty < B$. This folding scheme leads to a batched proof-of-knowledge protocol for short pre-images of linear maps. It also demonstrates all the essential tools needed to build a folding-based IVC and PCD from the MSIS assumption. However, a relation such as $\mathcal{R}_{\mathsf{cm}}^{B}$ that proves knowledge of a committed value is not enough to implement an IVC or PCD. One would also need to incorporate into $\mathcal{R}_{\mathsf{cm}}^{B}$ a computation checking relation, such as verifying a witness for an R1CS relation. We do so in Section 4 by extending $\mathcal{R}_{\mathsf{cm}}^{B}$ to include such a check.

As an optimization of our folding schemes, we show in Section 3.3 how to adapt the folding scheme for $\mathcal{R}_{\mathsf{cm}}^{B}$ to support relations defined over a small modulus q, say on the order of 2^{64}. This makes arithmetic faster since $\mathbb{Z}_q$ now fits into the native 64-bit registers of a CPU or GPU. Moreover, a small modulus is advantageous for encoding computations that operate on binary values, since a small q reduces the encoding overhead. The problem is that a small q limits the size of the challenge space and harms soundness. We show that with a suitable use of extensions fields we can enlarge the challenge space while supporting relations over a small modulus.

Next, in Section 4 we generalize our basic folding technique to support circuits with high degree gates. In particular we show how to fold a customizable constraint system (CCS) relation [STW23a]. This generalization adds an additional sumcheck step before decomposition to linearize the high degree relation. This is needed to avoid cross terms that would arise if decomposition were applied to a relation involving high degree gates. Hypernova [KS23b] uses a similar approach to avoid cross terms. In Section 4.3, we present an optimized scheme that batch the sumchecks from both the linearization and folding steps into one, further improving efficiency.

Evaluation. In Section 5 we provide a concrete evaluation of the resulting system. A recent implementation by Nethermind [Gar24] suggests that LatticeFold's performance is comparable with Hypernova, a pre-quantum system.

One reason LatticeFold performs well is that all the vectors that it uses lie in a single ring: the domain and range of the Ajtai commitment is the same ring $\mathcal{R}_q$. In contrast, for Pedersen commitments the domain is $\mathbb{Z}_q$ while the range is some other cyclic group. This forces Hypernova to implement elliptic curve scalar multiplications and non-native field arithmetic in the relation, which increases the folding complexity. Furthermore, LatticeFold uses a small 64-bit field, whereas Hypernova uses a 256-bit field due to the use of Pedersen commitments. However, Ajtai commitments adds additional complexity as explained earlier.

Finally, we note that LatticeFold is especially well suited for computations that make

use of operations in the ring $\mathcal{R}_q$. For example, suppose that the RELU function in a deep neural net (DNN) can be replaced by a similar function that can be expressed as a simple circuit using $\mathcal{R}_q$ operations. Then LatticeFold would be especially well suited for proving correct inference using the resulting DNN. The point is that a ring operation is a richer building block than simple arithmetic, and that can simplify some SNARK circuits.

1.1 Additional related work

Hypernova [KS23b] and Protostar [BC23] are two folding schemes that supports CCS relations. In Section 5 we compare the performance of LatticeFold to both schemes. ProtoGalaxy [EG23] is a further optimization of Protostar. The linearization step (from Section 4.2.1) that reduces a high-degree relation to a linear relation is inspired by Hypernova. However, fully adapting the folding techniques from Hypernova to the lattice setting incurs challenges. First, we need to guarantee that witness norms never go out of range after folding and prove that all intermediate witnesses have small norms. This is why we introduce the decomposition and the range proof techniques. Second, we need to adapt everything from a field to a ring in which not all elements are invertible. Finally, with decomposition, the random combination step must fold *more than* two witnesses into one using *independent* and *small-norm* challenges. This makes the security analysis significantly harder.

Several post-quantum SNARKs were constructed from hash-based Merkle commitments. Some examples include Stark [BBHR18b], Ligero [AHIV17], Aurora [Ben+19], Brakedown [Gol+23], BaseFold [ZCF24], and Blaze [Bre+24]. Their proof sizes scale sublinearly with the witness size, but in practice they produce relatively large proofs, and require a significant amount of memory when proving a large statement. In recent years, several elegant lattice-based proof systems with sublinear proof size were constructed [Bau+18a; BLNS20; Alb+22; BCS23]. However, these systems are not competitive with the hash based systems listed above. Other post-quantum proof systems, such as [ENS20; LNP22; Bau+23], perform well for small statements, but their proof size is linear in the size of the witness.

LaBRADOR [BS23] is an elegant succinct lattice-based proof system, with a linear time verifier. LaBRADOR is a recursive proof system based on the MSIS assumption. Thanks to the use of recursion, the resulting proofs are shorter than those obtained from the hash-based systems. LaBRADOR faces many of the same challenges as in this paper, but the proposed solutions are quite different. For example, LaBRADOR uses the method of random projection to prove a norm bound on a committed vector. We explain in Section 6 why this approach would not lead to an efficient folding scheme in our settings. Instead, our approach to proving a norm bound on a committed vector is based on the sumcheck protocol.

Concurrent to LatticeFold, Greyhound [NS24] built upon LaBRADOR and proposed a new polynomial commitment scheme (PCS) with square-root verification time, however, with similar reasons as LaBRADOR, it is not clear how to extend it to folding.

Cini et al. [CMNW24] recently introduced a new lattice-based PCS from Bulletproof/FRI-like techniques [BBHR18a; Bün+18]. Their approach achieves better control over witness norm/slack blowup than previous works. However, their scheme incurs norm blowup/slack at each step, limiting it to supporting only a logarithmic number of folding steps. In contrast, our construction (i) provides folding for general NP statements, and (ii) supports *polynomially* many folding steps for NP statements, as needed for PCD/IVC. It is unclear how to extend the PCS techniques from [CMNW24] to construct PCD/IVC.

Recently Bünz et al. [BMNW24a; BMNW24b] introduced an alternative approach to constructing folding schemes *purely from hashing*. Additionally, they introduced a new compiling technique to build efficient PCD/SNARKs from any folding schemes, which can be applied to LatticeFold as well. Compared to LatticeFold, the complexity of the recursive folding verifier is higher in their scheme because of the need to perform more hash operations. Moreover, in LatticeFold, the time and memory to compute the commitments scales with the number of *non-zero* entries in the committed witness, whereas in the hash-based scheme, it is always proportional to the witness length. This makes LatticeFold advantageous for sparse witnesses.

Subsequent work. A number of subsequent works have built on LatticeFold after it was first posted. LatticeFold+ [BC25] improves the performance of LatticeFold by designing a more efficient range proof. Neo [NS25] suggests a better approach for embedding field elements into the cyclotomic polynomial ring used in LatticeFold. Lova [FKNP24b; FKNP24a] replaces the module-based Ajtai commitments and the ℓ_∞-norm used in LatticeFold, with integer-based Ajtai commitments and the ℓ_2-norm. Finally, Klooß et al. [KLNO24b; KLNO24a] suggest another approach to ℓ_2-norm range proofs that can apply to LatticeFold.

2 Preliminaries

Notation. Let λ denote the security parameter. For $n \in \mathbb{N}$ let $[n]$ be the set $\{1, 2, \ldots, n\}$; for $l, r \in \mathbb{N}$ let $[l, r)$ denote the set $\{l, l+1, \ldots r-1\}$. A function $f(\lambda)$ is $\mathsf{poly}(\lambda)$ if there exists a $c \in \mathbb{N}$ such that $f(\lambda) = O(\lambda^c)$. If $f(\lambda) = o(\lambda^{-c})$ for all $c \in \mathbb{N}$, we say $f(\lambda)$ is in $\mathsf{negl}(\lambda)$ and is **negligible**. A probability that is $1 - \mathsf{negl}(\lambda)$ is **overwhelming**. A vector is always a column vector by default. For vectors $\vec{u}, \vec{v}$ of the same dimension we let $\langle \vec{u}, \vec{v} \rangle$ denote the inner product of $\vec{u}$ and $\vec{v}$. Throughout the paper when we refer to a ring we will always mean a commutative ring. For a ring $\bar{\mathcal{R}}$, we use $\bar{\mathcal{R}}[X_1, \ldots, X_\mu]$ to denote the set of μ-variate polynomials over $\bar{\mathcal{R}}$, and use $\bar{\mathcal{R}}^{\leq d}[X_1, \ldots, X_\mu]$ to denote the set of polynomials where the degree of each variable is at most d.

Modules and module homomorphisms. Let $\bar{\mathcal{R}}$ be an arbitrary ring. An $\bar{\mathcal{R}}$-module M can be understood as a "vector space" over ring $\bar{\mathcal{R}}$, that is, it allows to be scaled by elments in $\bar{\mathcal{R}}$. More precisely, M has an identity element 1 and for all $r, s \in \bar{\mathcal{R}}$ and

$x, y \in M$, we have (i) $r \cdot (x + y) = r \cdot x + r \cdot y$ (ii) $(r + s) \cdot x = r \cdot x + s \cdot x$, (iii) $(rs) \cdot x = r \cdot (s \cdot x)$, and (iv) $1 \cdot x = x$. Moreover, M is commutative, i.e., $r \cdot x = x \cdot r$. An $\bar{\mathcal{R}}$-module homomorphism $\phi : M \to N$ between $\bar{\mathcal{R}}$-modules M and N is a function that preserves additions and scalar multiplications. More precisely, for every $x, y \in M$ and $r \in \bar{\mathcal{R}}$ we have (i) $\phi(x + y) = \phi(x) + \phi(y)$, and (ii) $\phi(r \cdot x) = r \cdot \phi(x)$.

Cyclotomic rings. Let $\mathcal{R} := \mathbb{Z}[X]/(X^d + 1)$ where d is a power of two. Let $t \in \mathbb{N}$ be a divisor of d and q be a prime such that $q \equiv 1 + 2t \pmod{4t}$. Therefore $\mathbb{Z}_q$ has t primitive $2t$-th root of unity $\{\zeta_j\}_{j \in [t]}$ such that $X^d + 1 \equiv \prod_{j=1}^{t} (X^{d/t} - \zeta_j) \pmod{q}$, where $(X^{d/t} - \zeta_j)$ is *irreducible* for all $j \in [t]$. By the Chinese Remainder Theorem, $\mathcal{R}_q := \mathcal{R}/q\mathcal{R} = \mathbb{Z}_q[X]/(X^d + 1)$ can be split to the product of t quotient rings, that is,

$$\mathcal{R}_q \cong \prod_{j=1}^{t} \mathbb{Z}_q[X]/(X^{d/t} - \zeta_j) \cong \mathbb{F}_{q^{d/t}}^t .$$

For a polynomial $f \in \mathcal{R}_q$, the **Number Theoretic Transform** (NTT) of f is defined as

$$\mathsf{NTT}(f) := \left[\hat{f}_1, \ldots, \hat{f}_t\right]^\top \in \mathbb{F}_{q^{d/t}}^t$$

where $\hat{f}_j := f \bmod (X^{d/t} - \zeta_j)$. In the special case where $t = d$, the prime q splits completely in $\mathcal{R}$ and

$$\mathcal{R}_q \cong \prod_{j=1}^{d} \mathbb{Z}_q[X]/(X - \zeta_j) \cong \mathbb{Z}_q^d . \tag{1}$$

Coefficient embedding. For an element $\mathbf{a} = \sum_{i=1}^{d} a_i X^{i-1} \in \mathcal{R}_q$, we use $\mathsf{Coef}(\mathbf{a}) := [a_1, \ldots, a_d]^\top \in \mathbb{Z}_q^d$ to denote the coefficient vector of $\mathbf{a}$ and denote $\mathsf{Coef}_i(\mathbf{a}) := a_i$ for every $i \in [d]$. For a vector $\vec{\mathbf{a}} := [\mathbf{a}_1, \ldots, \mathbf{a}_m]^\top \in \mathcal{R}_q^m$, we use $\mathsf{Coef}(\vec{\mathbf{a}})$ to denote the matrix

$$\mathsf{Coef}(\vec{\mathbf{a}}) := \begin{bmatrix} \mathsf{Coef}_1(\mathbf{a}_1) & \ldots & \mathsf{Coef}_d(\mathbf{a}_1) \\ \vdots & \ddots & \vdots \\ \mathsf{Coef}_1(\mathbf{a}_m) & \ldots & \mathsf{Coef}_d(\mathbf{a}_m) \end{bmatrix} \in \mathbb{Z}_q^{m \times d} \tag{2}$$

and $\mathsf{FCoef}(\vec{\mathbf{a}}) \in \mathbb{Z}_q^{dm}$ denotes the concatenation of $\mathsf{Coef}(\vec{\mathbf{a}})$'s row vectors. For every $i \in [d]$, we define $\mathsf{Coef}_i(\vec{\mathbf{a}}) := [\mathsf{Coef}_i(\mathbf{a}_1), \ldots, \mathsf{Coef}_i(\mathbf{a}_m)]^\top \in \mathbb{Z}_q^m$ as the i-th column of $\mathsf{Coef}(\vec{\mathbf{a}})$. Define $\mathsf{Rot}(\mathbf{a}) := (\mathsf{Coef}(\mathbf{a}), \mathsf{Coef}(X \cdot \mathbf{a}), \ldots, \mathsf{Coef}(X^{d-1} \cdot \mathbf{a})) \in \mathbb{Z}_q^{d \times d}$. We observe that

$$\mathsf{Coef}(\mathbf{a} \cdot \mathbf{b}) = \mathsf{Rot}(\mathbf{a}) \times \mathsf{Coef}(\mathbf{b}) \tag{3}$$

for every $\mathbf{a}, \mathbf{b} \in \mathcal{R}_q$. More generally, for a matrix $\mathbf{A} \in \mathcal{R}_q^{\kappa \times m}$, we define the rotation matrix $\mathsf{Rot}(\mathbf{A})$ as

$$\mathsf{Rot}(\mathbf{A}) := \begin{bmatrix} \mathsf{Rot}(\mathbf{A}_{1,1}) & \dots & \mathsf{Rot}(\mathbf{A}_{1,m}) \\ \vdots & \ddots & \vdots \\ \mathsf{Rot}(\mathbf{A}_{\kappa,1}) & \dots & \mathsf{Rot}(\mathbf{A}_{\kappa,m}) \end{bmatrix} \in \mathbb{Z}_q^{\kappa d \times md} . \tag{4}$$

Note that $\mathsf{FCoef}(\mathbf{A}\vec{\mathbf{f}}) = \mathsf{Rot}(\mathbf{A}) \times \mathsf{FCoef}(\vec{\mathbf{f}})$ for any $\mathbf{A} \in \mathcal{R}_q^{\kappa \times m}$ and $\vec{\mathbf{f}} \in \mathcal{R}_q^m$.

Fix a ring $\mathcal{R}_q \cong \mathbb{F}_{q^\tau}^{d/\tau}$, given $\mathbf{a} \in \mathcal{R}_q$ and the coefficient embeddings of ring elements $\mathbf{b}_1, \dots, \mathbf{b}_\tau$, the following lemma shows that the coefficient embeddings of $\mathbf{a} \cdot \mathbf{b}_1, \dots, \mathbf{a} \cdot \mathbf{b}_\tau$ can be obtained through straightforward linear operations.

Lemma 2.1. *Let $\mathcal{R}_q \cong \mathbb{F}_{q^\tau}^{d/\tau}$ for some $\tau \in \mathbb{N}$ where $\tau \mid d$. Given $\mathbf{a} \in \mathcal{R}_q$ and $\vec{B} := [B_1, \dots, B_d]^\top \in \mathbb{F}_{q^\tau}^d$, we define function $\mathsf{RotSum} : \mathcal{R}_q \times \mathbb{F}_{q^\tau}^d \to \mathbb{F}_{q^\tau}^d$ as*

$$\mathsf{RotSum}(\mathbf{a}, \vec{B}) := \sum_{i=1}^{d} B_i \cdot \mathsf{Coef}(X^{i-1}\mathbf{a}) , \tag{5}$$

where $\cdot : \mathbb{F}_{q^\tau} \times \mathbb{Z}_q^d \to \mathbb{F}_{q^\tau}^d$ denotes scalar multiplication between $\mathbb{F}_{q^\tau}$ and $\mathbb{Z}_q^d$. Then:

1. *For every $a \in \mathbb{Z}_q$ and $\mathbf{b} \in \mathcal{R}_q$, we have that $\mathsf{Coef}(a \cdot \mathbf{b}) = a \cdot \mathsf{Coef}(\mathbf{b})$.*
2. *For every $\mathbf{a}, \mathbf{b} \in \mathcal{R}_q$ (where $\mathsf{Coef}(\mathbf{b}) \in \mathbb{Z}_q^d \subseteq \mathbb{F}_{q^\tau}^d$), we have that*

$$\mathsf{RotSum}(\mathbf{a}, \mathsf{Coef}(\mathbf{b})) = \mathsf{Coef}(\mathbf{a} \cdot \mathbf{b}) .$$

3. *For $\mathbf{a}, \mathbf{b}_1, \dots, \mathbf{b}_\tau \in \mathcal{R}_q$, define*

$$\vec{B} := \sum_{j=1}^{\tau} \mathsf{Coef}(\mathbf{b}_j) \cdot Y^{j-1} = [\vec{B}_1, \dots, \vec{B}_d]^\top \in \mathbb{F}_{q^\tau}^d$$

where $\vec{B}_i := \sum_{j=1}^{\tau} \mathsf{Coef}_i(\mathbf{b}_j) \cdot Y^{j-1} \in \mathbb{F}_{q^\tau}$ for every $i \in [d]$. Then

$$\mathsf{RotSum}(\mathbf{a}, \vec{B}) = \sum_{j=1}^{\tau} \mathsf{Coef}(\mathbf{a} \cdot \mathbf{b}_j) \cdot Y^{j-1} \in \mathbb{F}_{q^\tau}^d .$$

Proof. The 1st claim is clear as $\mathcal{R}_q$ is a $\mathbb{Z}_q$-module. The 2nd claim holds because

$$\mathsf{RotSum}(\mathbf{a}, \mathsf{Coef}(\mathbf{b})) = \mathsf{Rot}(\mathbf{a}) \times \mathsf{Coef}(\mathbf{b}) = \mathsf{Coef}(\mathbf{a} \cdot \mathbf{b})$$

by definition of $\mathsf{Rot}(\mathbf{a})$ and by Eq. (3).

Next, we prove the last claim. For every $i \in [d]$, note that $\vec{B}_i = \sum_{j=1}^{\tau} \mathsf{Coef}_i(\mathbf{b}_j) \cdot Y^{j-1} \in \mathbb{F}_{q^\tau}$ and the i-th column of $\mathsf{Rot}(\mathbf{a})$ is $[\mathsf{Rot}(\mathbf{a})]_i = \mathsf{Coef}(X^{i-1}\mathbf{a}) \in \mathbb{Z}_q^d$. Thus

$$\mathsf{RotSum}(\mathbf{a}, \vec{B}) := \sum_{i=1}^{d} \left(\sum_{j=1}^{\tau} \mathsf{Coef}_i(\mathbf{b}_j) \cdot Y^{j-1} \right) \cdot [\mathsf{Rot}(\mathbf{a})]_i = \sum_{j=1}^{\tau} \left(\sum_{i=1}^{d} \mathsf{Coef}_i(\mathbf{b}_j) \cdot [\mathsf{Rot}(\mathbf{a})]_i \right) \cdot Y^{j-1}$$
$$= \sum_{j=1}^{\tau} \mathsf{RotSum}(\mathbf{a}, \mathsf{Coef}(\mathbf{b}_j)) \cdot Y^{j-1} = \sum_{j=1}^{\tau} \mathsf{Coef}(\mathbf{a} \cdot \mathbf{b}_j) \cdot Y^{j-1} .$$

The 1st and the 3rd equality is by definition of RotSum; the last equality follows by the 2nd claim of the lemma. □

Norms. Let $\mathcal{R} := \mathbb{Z}[X]/(X^d + 1)$. For a polynomial $f := \sum_{i=0}^{d-1} f_i X^i \in \mathcal{R}$, the ℓ_2-norm and ℓ_∞-norm of f are

$$\|f\|_2 := \left(\sum_{i=0}^{d-1} f_i^2 \right)^{\frac{1}{2}} , \qquad \|f\|_\infty := \max_{i=0}^{d-1} (|f_i|) .$$

For a vector of polynomials $\vec{\mathbf{f}} := (f_1, \ldots, f_k) \in \mathcal{R}^k$, its ℓ_2-norm and ℓ_∞-norm are

$$\|\vec{\mathbf{f}}\|_2 := \left(\sum_{i=1}^{k} \|f_i\|_2^2 \right)^{\frac{1}{2}} , \qquad \|\vec{\mathbf{f}}\|_\infty := \max_{i=1}^{k} (\|f_i\|_\infty) .$$

We note that $\|\vec{\mathbf{f}}\|_2 \leq \sqrt{dk} \|\vec{\mathbf{f}}\|_\infty$ for all $\vec{\mathbf{f}} \in \mathcal{R}^k$.

Remark 2.1 (Norms of $\mathcal{R}_q$-elements)**.** *Let $\mathcal{R}_q := \mathcal{R}/q\mathcal{R}$. For a vector $\vec{\mathbf{f}} := (f_1, \ldots, f_k) \in \mathcal{R}_q^k$, we abuse the notation $\|\vec{\mathbf{f}}\|_\infty$ to indicate the norm of $\vec{\mathbf{f}}$ after lifting to $\mathcal{R}^k$. The lifting works by mapping the $\mathbb{Z}_q$-coefficients of $\vec{\mathbf{f}}$ to the interval $(-q/2, q/2] \subseteq \mathbb{Z}$.*

2.1 Sampling Sets

We review the definition of sampling sets from [CCKP19].

Definition 2.1. *For an arbitrary ring $\bar{\mathcal{R}}$, a subset $\mathcal{C}$ of $\bar{\mathcal{R}}$ is a **sampling set** if the difference of any two distinct elements in $\mathcal{C}$ is* not a zero divisor. *$\mathcal{C}$ is further a **strong sampling set** if the difference is also* invertible.

Example: Set $\bar{\mathcal{R}} := \mathcal{R}_q$ where q is a prime. Then $\mathbb{Z}_q \subseteq \mathcal{R}_q$ is a strong sampling set as the difference of any two distinct elements in this set is invertible in $\mathcal{R}_q$.

Sometimes we need a strong sampling set $\mathcal{C}_{\mathsf{small}} \subseteq \mathcal{R}_q$ such that for every $\rho \in \mathcal{C}_{\mathsf{small}}$ and any $\hat{\mathbf{v}} \in \mathcal{R}$, the norm of $\rho\hat{\mathbf{v}}$ will not increase much compared to $\|\hat{\mathbf{v}}\|_\infty$. To quantify this property, we define the **expansion factor** of $\mathcal{C}_{\mathsf{small}} \subseteq \mathcal{R}_q$ as

$$\|\mathcal{C}_{\mathsf{small}}\|_{\mathsf{op}} := \sup_{\rho \in \mathcal{C}_{\mathsf{small}}, \hat{\mathbf{v}} \in \mathcal{R}} \frac{\|\rho \times \hat{\mathbf{v}}\|_\infty}{\|\hat{\mathbf{v}}\|_\infty} . \tag{6}$$

Here, the multiplication $\rho \times \hat{\mathbf{v}}$ is performed over the ring $\mathcal{R}$ where we lift $\rho \in \mathcal{R}_q$ to $\mathcal{R}$ as in Remark 2.1. The lemma below shows that a set with small norm elements has small expansion factors.

Lemma 2.2 (Prop. 2 of [AL21]). *In $\mathcal{R} := \mathbb{Z}[X]/(X^d + 1)$, for all $\mathbf{u}, \hat{\mathbf{v}} \in \mathcal{R}$, we have that*

$$\frac{\|\mathbf{u}\hat{\mathbf{v}}\|_\infty}{\|\hat{\mathbf{v}}\|_\infty} \leq d \cdot \|\mathbf{u}\|_\infty .$$

The following lemma shows that an element in $\mathcal{R}_q$ is invertible if its norm (after lifting to $\mathcal{R}$) is small. Combining with Lemma 2.2, it implies that we can find large strong sampling sets *in* $\mathcal{R}_q$ with small expansion factors. This is because the difference between any two distinct small elements (with norm less than $q/4$) remains small (as there is no modulus overflow) and is therefore invertible.

Lemma 2.3 (Corollary 1.2 of [LS18]). *Let $d \geq t > 1$ be a power-of-two and $q \equiv 1 + 2t \pmod{4t}$ be a prime. Then every $\mathbf{y} \in \mathcal{R}_q := \mathbb{Z}_q[X]/(X^d + 1)$ where $0 < \|\mathbf{y}\|_\infty < \frac{q^{1/t}}{\sqrt{t}}$ is invertible. Here $\|\mathbf{y}\|_\infty$ denotes $\mathbf{y}$'s norm after lifting to $\mathcal{R}$.*

2.2 Module SIS

We recall the Module Short Integer Solution (MSIS) problem [LS15; PR06; LM06].

Definition 2.2 (Module SIS). *Let $\mathcal{R} := \mathbb{Z}[X]/(X^d+1)$ and $\mathcal{R}_q := \mathcal{R}/q\mathcal{R}$. Given a random matrix $\mathbf{A} \leftarrow\!\$\ \mathcal{R}_q^{\kappa \times m}$, the goal of the $\mathsf{MSIS}^q_{\kappa,m,B_{\mathsf{SIS}}}$ problem is to find a non-zero $\vec{\mathbf{x}} \in \mathcal{R}^m$ such that $\|\vec{\mathbf{x}}\|_2 < B_{\mathsf{SIS}}$ and $\mathbf{A}\vec{\mathbf{x}} = \vec{\mathbf{0}}$ over $\mathcal{R}_q$.*

The MSIS-algorithm from Micciancio and Regev [MR09] can output an MSIS solution with ℓ_2-norm $B_{\mathsf{SIS}} \approx \min(q, 2^{2\sqrt{\log_2(\delta) d\kappa \log q}})$ where δ is the root Hermite factor of the lattice reduction algorithm. In practice, setting $\delta \approx 1.0045$ and letting $q/2 > 2^{2\sqrt{\log_2(\delta) d\kappa \log q}}$ is believed to lead to an MSIS problem that has 128 bits of security [Esg+19; APS15]. We will focus on the ℓ_∞-norm. Thus we also review a variant of the MSIS problem that replaces the ℓ_2-norm with ℓ_∞-norm. It is clear that $\mathsf{MSIS}^{\infty,q}_{\kappa,m,B}$ is at least as hard as $\mathsf{MSIS}^q_{\kappa,m,\sqrt{dm}B}$.

Definition 2.3 (Module SIS with ℓ_∞-norms [ACK21]). *Let $\mathcal{R} := \mathbb{Z}[X]/(X^d + 1)$ and $\mathcal{R}_q := \mathcal{R}/q\mathcal{R}$. On input $\mathbf{A} \leftarrow\!\$\ \mathcal{R}_q^{\kappa \times m}$, the goal of the $\mathsf{MSIS}^{\infty,q}_{\kappa,m,B}$ problem is to find a non-zero $\vec{\mathbf{x}} \in \mathcal{R}^m$ such that $\|\vec{\mathbf{x}}\|_\infty < B$ and $\mathbf{A}\vec{\mathbf{x}} = \vec{\mathbf{0}}$ over $\mathcal{R}_q$.*

2.3 The Ajtai Compact Commitment

A commitment scheme CM consists of a setup algorithm Setup that generates a public parameter pp; and a deterministic commit algorithm Commit that takes as input pp, a message $\vec{\mathbf{x}}$ and randomness r, and outputs a commitment cm. We say that CM is **compact** if the commitment cm is shorter than the committed message $\vec{\mathbf{x}}$. We say CM is **binding** if it is hard to find a commitment cm and two different openings $(\vec{\mathbf{x}}_1, r_1)$, $(\vec{\mathbf{x}}_2, r_2)$ such that $\mathsf{cm} = \mathsf{Commit}(\mathsf{pp}, \vec{\mathbf{x}}_1, r_1) = \mathsf{Commit}(\mathsf{pp}, \vec{\mathbf{x}}_2, r_2)$. We say that CM is **hiding** if cm is statistically independent of $\vec{\mathbf{x}}$ over the choice of randomness r.

We review a variant of the Ajtai commitment scheme [Ajt96; PR06; LM06] whereas the messages are ring elements with *small* norms. For brevity, we present the construction (i.e., the Ajtai collision resistant hash function) that achieves only the binding property. It can be extended to support hiding by appending a small random vector to the message.

Construction 2.1 (Ajtai Compact Commitments)**.** *Let $\mathcal{R} := \mathbb{Z}[X]/(X^d + 1)$ and $\mathcal{R}_q := \mathcal{R}/q\mathcal{R}$ where $q \in \mathbb{N}$ is a prime. The commitment $\mathsf{CM}_{\kappa,m,B}$ works as follows:*

- $\mathsf{Setup}(\kappa, m) \to \mathbf{A}$*: sample a random matrix* $\mathbf{A} \leftarrow\!\!\$\, \mathcal{R}_q^{\kappa \times m}$.
- $\mathsf{Commit}(\mathbf{A}, \vec{\mathbf{x}}) \to \mathsf{cm}$*: given $\vec{\mathbf{x}} \in \mathcal{R}^m$ as input, where $\|\vec{\mathbf{x}}\|_\infty < B$, and no randomness, output* $\mathsf{cm} := \mathbf{A}\vec{\mathbf{x}} \bmod q \in \mathcal{R}_q^\kappa$.

It is clear that $\mathsf{CM}_{\kappa,m,B}$ satisfies the binding property for inputs $\|\vec{\mathbf{x}}\|_\infty < B$ assuming that the MSIS problem $\mathsf{MSIS}^{\infty,q}_{\kappa,m,2B}$ is hard. Suppose not, that is, an adversary can open a commitment cm to two different openings $\vec{\mathbf{x}}_1$, $\vec{\mathbf{x}}_2$ (with ℓ_∞-norm less than B), then $\vec{\mathbf{x}}_1 - \vec{\mathbf{x}}_2 \neq 0$ is a solution to the $\mathsf{MSIS}^{\infty,q}_{\kappa,m,2B}$ problem where $\|\vec{\mathbf{x}}_1 - \vec{\mathbf{x}}_2\|_\infty < 2B$.

The analysis of our protocol also needs a relaxed notion of the binding property [ALS20; ACK21]. Let $\mathcal{C} \subset \mathcal{R}_q$ be a strong sampling set with expansion factor T. We say that the pair $(\Delta, \vec{\mathbf{x}}) \in (\mathcal{C} - \mathcal{C}) \times \mathcal{R}^m$ is a B-**weak opening** of cm if $\Delta \cdot \mathsf{cm} = \mathbf{A}\vec{\mathbf{x}} \bmod q$ and $\|\vec{\mathbf{x}}\|_\infty < B$. We say that the commitment scheme is B-**relaxed binding** if it is infeasible to find two B-weak openings $(\Delta_1, \vec{\mathbf{x}}_1)$, $(\Delta_2, \vec{\mathbf{x}}_2)$ for the same commitment cm such that $\Delta_1\vec{\mathbf{x}}_2 \neq \Delta_2\vec{\mathbf{x}}_1$. It is clear that the Ajtai commitment is B-relaxed binding if $\mathsf{MSIS}^{\infty,q}_{\kappa,m,4TB}$ is hard. Suppose not, i.e., for cm, we can find two weak openings $(\Delta_1 := (\rho_1 - \rho_1'), \vec{\mathbf{x}}_1)$, $(\Delta_2 := (\rho_2 - \rho_2'), \vec{\mathbf{x}}_2)$ where $\rho_1, \rho_1', \rho_2, \rho_2' \in \mathcal{C}$, $\Delta_1\vec{\mathbf{x}}_2 \neq \Delta_2\vec{\mathbf{x}}_1$ and $\|\vec{\mathbf{x}}_1\|_\infty, \|\vec{\mathbf{x}}_2\|_\infty < B$. Then $(\rho_2 - \rho_2') \cdot \vec{\mathbf{x}}_1 - (\rho_1 - \rho_1') \cdot \vec{\mathbf{x}}_2 \neq 0$ is a solution to $\mathsf{MSIS}^{\infty,q}_{\kappa,m,4TB}$ with norm at most $4TB$.

2.4 Sum-Checks and Multilinear Extensions over Rings

Generalized Schwartz-Zippel Lemma. We recall a generalization of the Schwartz-Zippel lemma to the commutative ring setting, where each challenge is picked from a *sampling set.*

Lemma 2.4 (Generalized Schwartz-Zippel [BCPS18])**.** *Let $f \in \bar{\mathcal{R}}^{\leq d}[X_1, \ldots, X_\mu]$ be a μ-variate nonzero polynomial over a ring $\bar{\mathcal{R}}$ with per-variable degree at most d. Let $\mathcal{C} \subseteq \bar{\mathcal{R}}$ be a sampling set. Then we have $\Pr_{\vec{\mathbf{r}} \xleftarrow{\$} \mathcal{C}^\mu}[f(\vec{\mathbf{r}}) = 0] \leq \frac{d\mu}{|\mathcal{C}|}$.*

Sum-check over rings. Given Lemma 2.4, the famous sum-check protocol [LFKN92] can be naturally extended to work over a ring $\bar{\mathcal{R}}$ with the modification that the challenges are sampled from a *strong* sampling set.

Lemma 2.5 (Generalized Sum-Check [CCKP19])**.** *Let $f \in \bar{\mathcal{R}}^{\leq d}[X_1, \ldots, X_\mu]$ be a μ-variate polynomial over a ring $\bar{\mathcal{R}}$ with per-variable degree at most d. Let $\mathcal{C} \subseteq \bar{\mathcal{R}}$ be a **strong** sampling set. The protocol below for checking $s = \sum_{\vec{\mathbf{b}} \in \{0,1\}^\mu} f(\vec{\mathbf{b}})$ has soundness error $\frac{\mu d}{|\mathcal{C}|}$.*

1. *In the i-th $(1 \leq i \leq \mu)$ round,*
   - *Upon receiving the challenges $\mathbf{r}_1, \ldots, \mathbf{r}_{i-1}$ from the previous rounds, the prover sends the univariate polynomial*
   $$h_i(X) := \sum_{\vec{\mathbf{b}} \in \{0,1\}^{\mu-i}} f(\mathbf{r}_1, \ldots, \mathbf{r}_{i-1}, X, \vec{\mathbf{b}}) \quad \in \bar{\mathcal{R}}[X]\,.$$
   More specifically, it sends $d+1$ evaluations of h_i at $d+1$ points in $\mathcal{C}$.
   - *Denote $h_0(\mathbf{r}_0) := s$ for notational convenience. The verifier checks that $h_i(0) + h_i(1) \stackrel{?}{=} h_{i-1}(\mathbf{r}_{i-1})$ and sends a random challenge $\mathbf{r}_i \xleftarrow{\$} \mathcal{C}$. (The verifier can do Lagrange interpolation to evaluate $h_{i-1}(\vec{\mathbf{r}}_{i-1})$ given the $d+1$ evaluations sent by the prover, as the differences of distinct evaluation points are invertible.)*
2. *The verifier checks that $h_\mu(\mathbf{r}_i) \stackrel{?}{=} f(\mathbf{r}_1, \ldots, \mathbf{r}_\mu)$.*

Proof. See the proof of Theorem 2 in [CCKP19]. □

Multilinear extensions over rings. We define the multilinear extensions over rings.

Definition 2.4 (Multilinear Extensions over Rings)**.** *Let $\bar{\mathcal{R}}$ be an arbitrary ring with zero 0 and identity 1. Given a function $f : \{0,1\}^\mu \to \bar{\mathcal{R}}$, we define the multilinear extension $\mathsf{mle}\,[f] \in \bar{\mathcal{R}}^{\leq 1}[X_1, \ldots, X_\mu]$ of f as*

$$\mathsf{mle}\,[f]\,(\vec{\mathbf{X}}) := \sum_{\vec{\mathbf{b}} \in \{0,1\}^\mu} f(\vec{\mathbf{b}}) \cdot eq(\vec{\mathbf{b}}, \vec{\mathbf{X}})$$

where $eq(\vec{\mathbf{b}}, \vec{\mathbf{X}})$ is defined as $eq(\vec{\mathbf{b}}, \vec{\mathbf{X}}) := \prod_{i=1}^{\mu} \big[(1 - \vec{\mathbf{b}}_i)(1 - \vec{\mathbf{X}}_i) + \vec{\mathbf{b}}_i \vec{\mathbf{X}}_i\big]$.

2.5 Reduction of Knowledge

Intuitively, a reduction-of-knowledge protocol Π (from $\mathcal{R}_1$ to $\mathcal{R}_2$) allows a prover to convince a verifier on input $\mathbb{x}_1$ to obtain an output $\mathbb{x}_2$, such that from anyone who knows a witness $\mathbb{w}_2$ where $(\mathbb{x}_2, \mathbb{w}_2) \in \mathcal{R}_2$, one can extract a witness $\mathbb{w}_1$ where $(\mathbb{x}_1, \mathbb{w}_1) \in \mathcal{R}_1$. We adapt the definition from [KP23].

Definition 2.5 (Reduction of knowledge [KP23])**.** *Consider ternary relations $\mathcal{R}_1$ and $\mathcal{R}_2$ consisting of public parameters, statement and witness tuples. Let $\langle \mathsf{P}, \mathsf{V} \rangle$ denote an interactive protocol between a prover P and a verifier V. A reduction of knowledge protocol Π from relation $\mathcal{R}_1$ to $\mathcal{R}_2$ consists of the following PPT algorithms/protocols:*

- $\mathsf{Setup}(1^\lambda) \to \mathsf{pp}$*: on input security parameter λ outputs public parameters* pp*.*
- $\langle \mathsf{P}(\mathsf{pp}, \mathbb{x}_1, \mathbb{w}_1), \mathsf{V}(\mathsf{pp}, \mathbb{x}_1) \rangle \to (\mathbb{x}_2, \mathbb{w}_2)$*: on input public parameters* pp *and a shared instance $\mathbb{x}_1$ for $\mathcal{R}_1$, the prover P (which also has a witness $\mathbb{w}_1$ for $\mathcal{R}_1$) and the verifier V run an interactive protocol. At the end of the protocol, the verifier outputs an instance $\mathbb{x}_2$ for $\mathcal{R}_2$ or $\mathbb{x}_2 := \bot$; and the prover additionally outputs a witness $\mathbb{w}_2$ for $\mathcal{R}_2$. We let $(\mathbb{x}_2, \mathbb{w}_2)$ denote the output of the interaction.*

The protocol satisfies the following properties:

Completeness. *For every PPT adversary $\mathcal{A}$ that adaptively chooses an instance-witness pair $(\mathbb{x}_1, \mathbb{w}_1) \leftarrow \mathcal{A}(\mathsf{pp})$ for $\mathcal{R}_1$ after observing the public parameter $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$. If $(\mathsf{pp}, \mathbb{x}_1, \mathbb{w}_1)$ is in $\mathcal{R}_1$, then the output $(\mathbb{x}_2, \mathbb{w}_2)$ of the execution $\langle \mathsf{P}(\mathsf{pp}, \mathbb{x}_1, \mathbb{w}_1), \mathsf{V}(\mathsf{pp}, \mathbb{x}_1) \rangle$ is also in $\mathcal{R}_2$.*

Knowledge soundness. *We say that the protocol is knowledge sound with knowledge error $\kappa(\lambda)$, if there exists an expected polynomial time extractor Ext such that for any expected polynomial time adversary $(\mathcal{A}, \mathsf{P}^*)$, if*

$$\Pr\left[(\mathsf{pp}, \mathbb{x}_2, \mathbb{w}_2) \in \mathcal{R}_2 \;\middle|\; \begin{array}{c} \mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda) \\ (\mathbb{x}_1, \mathsf{st}) \leftarrow \mathcal{A}(\mathsf{pp}) \\ (\mathbb{x}_2, \mathbb{w}_2) \leftarrow \langle \mathsf{P}^*(\mathsf{pp}, \mathbb{x}_1, \mathsf{st}), \mathsf{V}(\mathsf{pp}, \mathbb{x}_1) \rangle \end{array} \right] = \epsilon(\lambda) > \kappa(\lambda)$$

where $\epsilon(\lambda) \geq \frac{1}{\mathsf{poly}(\lambda)}$,[1] *then with probability at least $\epsilon(\lambda) - \kappa(\lambda)$, the extractor $\mathsf{Ext}^{\mathcal{A}, \mathsf{P}^*}$ outputs a witness $\mathbb{w}_1$ such that $(\mathsf{pp}, \mathbb{x}_1, \mathbb{w}_1) \in \mathcal{R}_1$.*

Public reducibility. *There is a deterministic poly-time algorithm f such that for any PPT adversary $\mathcal{A}$ and expected poly-time adversary P^*, given*

$$\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda), \quad (\mathbb{x}_1, \mathsf{st}) \leftarrow \mathcal{A}(\mathsf{pp}), \quad \text{and} \quad (\mathbb{x}_2, \mathbb{w}_2) \leftarrow \langle \mathsf{P}^*(\mathsf{pp}, \mathbb{x}_1, \mathsf{st}), \mathsf{V}(\mathsf{pp}, \mathbb{x}_1) \rangle$$

with transcript tr, we have that $f(\mathsf{pp}, \mathbb{x}_1, \mathsf{tr}) = \mathbb{x}_2$.

Π *is* public-coin *if the verifier only sends uniformly random challenges in each round. Note that a public-coin protocol can be made non-interactive via the Fiat-Shamir transformation.*

As noted by [KP23], the reduction of knowledge protocols can be composed.

[1] In standard definitions, the requirement for $\epsilon(\lambda)$ to be non-negligible is typically omitted. Our constructions also meet the stronger standard definition, though possibly with a slightly weaker bound.

Theorem 2.1 (Sequential Composition, Theorem 5 of [KP23]). *Let $\mathcal{R}_1$, $\mathcal{R}_2$, $\mathcal{R}_3$ be three ternary relations. Given a reduction of knowledge Π_1 from $\mathcal{R}_1$ to $\mathcal{R}_2$ and a reduction of knowledge Π_2 from $\mathcal{R}_2$ to $\mathcal{R}_3$, the composed protocol $\Pi_2 \circ \Pi_1$ is a reduction of knowledge from $\mathcal{R}_1$ to $\mathcal{R}_3$.*

Theorem 2.2 (Parallel Composition, Theorem 6 of [KP23]). *Let $\mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3, \mathcal{R}_4$ be ternary relations. Given a reduction of knowledge Π_1 from $\mathcal{R}_1$ to $\mathcal{R}_2$ and a reduction of knowledge Π_2 from $\mathcal{R}_3$ to $\mathcal{R}_4$, the protocol $\Pi_1 \times \Pi_2$ is a reduction of knowledge from $\mathcal{R}_1 \times \mathcal{R}_3$ to $\mathcal{R}_2 \times \mathcal{R}_4$, where $\Pi_1 \times \Pi_2$ denotes the protocol that runs Π_1 and Π_2 in parallel.*

Remark 2.2. *The knowledge soundness defined in Definition 2.5 should hold for expected polynomial-time adversaries and extractors. This requirements is necessary for proof of composition theorems. Looking ahead, this implies that the MSIS hardness assumption in Definition 2.3 must also hold for expected poly-time adversaries. This is without loss of generality because the MSIS assumption is falsifiable. As shown in [LPS24] (Appendix A), if a falsifiable assumption holds for strict PPT adversaries (i.e., probabilistic adversaries that always run in polynomial time), it also holds for expected poly-time adversaries.*

Remark 2.3 (Folding schemes as reductions of knowledge). *We note that the folding schemes introduced in Hypernova [KS23b] is a special case of reduction of knowledge, where for a computation relation $\mathcal{R}_{\mathsf{comp}}$ and its expanded accumulation relaton $\mathcal{R}_{\mathsf{acc}}$, the goal is to reduce the relation $\mathcal{R}_{\mathsf{acc}} \times \mathcal{R}_{\mathsf{comp}}$ to the relation $\mathcal{R}_{\mathsf{acc}}$.*

3 A Folding Scheme for Ajtai Commitment Openings

In this section, we develop a folding scheme for the Ajtai commitment opening relation. In Section 3.1, we define an algebraic relation $\mathcal{R}^B_{\mathsf{cm}}$ that captures the commitment opening relation, and then extend it to a relation $\mathcal{R}^B_{\mathsf{eval}}$ suitable for folding. In Section 3.2, we construct a reduction of knowledge from $\mathcal{R}^B_{\mathsf{eval}} \times \mathcal{R}^B_{\mathsf{cm}}$ to $\mathcal{R}^B_{\mathsf{eval}}$, leading to a folding scheme for the Ajtai commitment opening relation. In Section 3.3, we describe an optimization that allows the selection of a small prime modulus q for improved efficiency.

Designing a folding scheme for the relation $\mathcal{R}^B_{\mathsf{cm}}$ is the core challenge in constructing a IVC/PCD scheme based on Ajtai commitments. This folding scheme also leads to a batch proof-of-knowledge for short pre-images of linear maps: it allows us to fold k statements (each of size n) into a single statement using a binary folding tree. This reduces the verifier complexity to $\tilde{O}(k+n)$ instead of complexity $\Theta(kn)$ if knowledge of each pre-image was proved on its own.

This folding scheme for $\mathcal{R}^B_{\mathsf{cm}}$, by itself, is insufficient for an IVC/PCD. The relation $\mathcal{R}^B_{\mathsf{cm}}$ needs to be augmented to facilitate the verification of a local computation step, either expressed as a R1CS statement or, more generally, a CCS statement. We come back to this in Section 4 where we build an extended relation $\mathcal{R}^B_{\mathsf{cmccs}}$ in Eq. (33) that is adequate for use in IVC/PCD.

3.1 The Relation for Commitment Openings

In this section, we reformulate the Ajtai commitment opening relation for efficient folding. The core idea, inspired and adapted from [BLS19], is to interpret the norm bound constraint as a Hadamard product over rings.

Recall that $\mathcal{R}_q := \mathcal{R}/q\mathcal{R}$ where $\mathcal{R} := \mathbb{Z}[X]/(X^d + 1)$ and q is a prime. Given $\mathsf{pp} := (\kappa, m, B, \mathbf{A})$ where $\mathbf{A} \in \mathcal{R}_q^{\kappa \times m}$ and $B < q/2$ is the norm bound, the relation $\mathcal{R}^B_{\mathsf{MSIS}\infty}$ for Ajtai commitment openings (Section 2.3) is

$$\mathcal{R}^B_{\mathsf{MSIS}\infty} := \left\{ (\mathsf{pp},\ \mathsf{cm} \in \mathcal{R}_q^{\kappa};\ \vec{\mathbf{x}} \in \mathcal{R}^m) \ :\ (\mathsf{cm} = \mathbf{A}\vec{\mathbf{x}} \bmod q) \wedge \|\vec{\mathbf{x}}\|_\infty < B \right\}.$$

Since $B < q/2$, we can uniquely represent $\vec{\mathbf{x}}$ as a vector in $\mathcal{R}_q^m$ and denote $\|\vec{\mathbf{x}}\|_\infty$ as the norm after lifting $\vec{\mathbf{x}}$ to $\mathcal{R}^m$ (See Remark 2.1). Then we rewrite $\mathcal{R}^B_{\mathsf{MSIS}\infty}$ as

$$\mathcal{R}^B_{\mathsf{MSIS}\infty} := \left\{ (\mathsf{pp},\ \mathsf{cm} \in \mathcal{R}_q^{\kappa};\ \vec{\mathbf{x}} \in \mathcal{R}_q^m) \ :\ (\mathsf{cm} = \mathbf{A}\vec{\mathbf{x}}) \wedge \|\vec{\mathbf{x}}\|_\infty < B \right\}.$$

Let us define an equivalent relation $\mathcal{R}^B_{\mathsf{SIS}\infty}$ over $\mathbb{Z}_q$. Let $\bar{\mathbf{A}} := \mathsf{Rot}(\mathbf{A})$ be the rotation matrix of $\mathbf{A}$ (as in Eq. (4)) and denote by $\vec{x} := \mathsf{FCoef}(\vec{\mathbf{x}}) \in \mathbb{Z}_q^{md}$ the row concatenation of $\mathsf{Coef}(\vec{\mathbf{x}}) \in \mathbb{Z}_q^{m \times d}$. Then the coefficient embedding of $\mathsf{cm} = \mathbf{A}\vec{\mathbf{x}}$ is exactly $\bar{\mathsf{cm}} = \bar{\mathbf{A}}\vec{x}$. Thus we can define the instance-witness relation $\mathcal{R}^B_{\mathsf{SIS}\infty}$ as

$$\mathcal{R}^B_{\mathsf{SIS}\infty} := \left\{ (\mathsf{pp},\ \bar{\mathsf{cm}} \in \mathbb{Z}_q^{\kappa d};\ \vec{x} \in \mathbb{Z}_q^{md}) \ :\ (\bar{\mathsf{cm}} = \bar{\mathbf{A}}\vec{x}) \wedge \|\vec{x}\|_\infty < B \right\}.$$

Equivalently, we write $\|\vec{x}\|_\infty < B < q/2$ as a Hadamard product relation $\mathcal{R}^B_{\mathsf{SISProd}}$ over $\mathbb{Z}_q$:

$$\mathcal{R}^B_{\mathsf{SISProd}} := \left\{ (\mathsf{pp},\ \bar{\mathsf{cm}} \in \mathbb{Z}_q^{\kappa d};\ \vec{x} \in \mathbb{Z}_q^{md}) \ :\ \begin{array}{c} (\bar{\mathsf{cm}} = \bar{\mathbf{A}}\vec{x}) \wedge \\ \left(\vec{x} \circ \left[\bigcirc_{i=1}^{B-1} (\vec{x} - \vec{i}) \circ (\vec{x} + \vec{i})\right] = \vec{0}\right) \end{array} \right\} \tag{7}$$

where $\vec{i} \in \mathbb{Z}_q^{md}$ is i multiplied by the identity vector $I_{md} := (1, \ldots, 1) \in \mathbb{Z}_q^{md}$. Clearly the Hadamard product in (7) is zero if and only if $\|\vec{x}\|_\infty < B$, as required.

The final step involves replacing the Hadamard product relation over $\mathbb{Z}_q$ with a Hadamard product relation over $\mathcal{R}_q$. For brevity, we assume that $q \equiv 1 \bmod 2d$ as in Eq. (1). In Section 3.3 we will explain how to generalize to other primes. Note that $q \equiv 1 \bmod 2d$ implies that $\mathcal{R}_q \cong \mathbb{Z}_q^d$.

There are two ways to interpret $\vec{x} \in \mathbb{Z}_q^{md}$. We alternatively view $\vec{x} \in \mathbb{Z}_q^{m \times d}$ as an m-by-d $\mathbb{Z}_q$-matrix in its natural form. First, $\vec{x}$ can be the coefficient embeddings of some $\vec{\mathbf{f}} \in \mathcal{R}_q^m$, so that $\mathsf{Coef}(\vec{\mathbf{f}}) = \vec{x}$. Alternatively, it can be understood as the NTT representations of some $\hat{\mathbf{f}} \in \mathcal{R}_q^m$, that is, $\mathsf{NTT}(\hat{\mathbf{f}}) = \mathsf{Coef}(\vec{\mathbf{f}}) = \vec{x}$. Moreover, the *Hadamard product* between the NTT slots of two ring elements can map to the *multiplication* of the two ring elements. In other words, $\vec{x} \circ \vec{x} = \mathsf{NTT}(\hat{\mathbf{f}}) \circ \mathsf{NTT}(\hat{\mathbf{f}}) = \mathsf{NTT}(\hat{\mathbf{f}} \circ \hat{\mathbf{f}})$, which maps to $\hat{\mathbf{f}} \circ \hat{\mathbf{f}}$ via the NTT

isomorphism. Here $\vec{x} \circ \vec{x}$ is a Hadamard product over $\mathbb{Z}_q$, while $\hat{\mathbf{f}} \circ \hat{\mathbf{f}}$ is over $\mathcal{R}_q$. Thus, we can rewrite $\mathcal{R}^B_{\mathsf{SISProd}}$ from (7) as the following instance-witness relation $\mathcal{R}^B_{\mathsf{cm}}$ over $\mathcal{R}_q$:

$$\mathcal{R}^B_{\mathsf{cm}} := \left\{ (\mathsf{pp},\ \mathsf{cm} \in \mathcal{R}_q^{\kappa};\ \vec{\mathbf{f}} \in \mathcal{R}_q^m) \ : \ \begin{array}{c} (\mathsf{cm} = \mathbf{A}\vec{\mathbf{f}}) \wedge \\ \left(\hat{\mathbf{f}} \circ \left[\bigcirc_{i=1}^{B-1} (\hat{\mathbf{f}} - \hat{i}) \circ (\hat{\mathbf{f}} + \hat{i})\right] = \hat{0}\right) \end{array} \right\}. \tag{8}$$

Here $\hat{i} \in \mathcal{R}_q^m$ is the ring vector such that $\vec{i} = \mathsf{NTT}(\hat{i})$ where $\vec{i} \in \mathbb{Z}_q^{m \times d}$ is the element $i \in \mathbb{Z}_q$ copied md times. Note that each element in $\hat{i}$ is the constant polynomial $i \in \mathbb{Z}_q$, so that $\hat{i}$ is in $\mathbb{Z}_q^m$.

Proving knowledge of a witness $\vec{\mathbf{f}} \in \mathcal{R}_q^m$ for the $\mathcal{R}^B_{\mathsf{cm}}$ statement $(\mathsf{pp}, \mathsf{cm})$ proves knowledge of a low-norm opening of the Ajtai commitment $\mathsf{cm} \in \mathcal{R}_q^{\kappa}$.

The expanded relation. To construct a folding scheme for $\mathcal{R}^B_{\mathsf{cm}}$, we augment $\mathcal{R}^B_{\mathsf{cm}}$ to a new relation $\mathcal{R}^B_{\mathsf{eval}}$ with an evaluation statement. Looking ahead, in our folding scheme, the verifier runs a sum-check to reduce the norm bound constraint in $\mathcal{R}^B_{\mathsf{cm}}$ to an evaluation statement. Thus, it is necessary to incorporate such an evaluation statement into the accumulated relation. For simplicity, we assume that m is a power of two. The relation $\mathcal{R}^B_{\mathsf{eval}}$ is defined as follows:

$$\mathcal{R}^B_{\mathsf{eval}} := \left\{ (\mathsf{pp},\ \mathbb{x} := (\vec{\mathbf{r}}, \hat{\mathbf{v}}, \mathsf{cm}) \in \mathcal{R}_q^{\log m} \times \mathcal{R}_q \times \mathcal{R}_q^{\kappa};\ \vec{\mathbf{f}} \in \mathcal{R}_q^m) \ : \ \begin{array}{c} (\mathsf{pp}, \mathsf{cm}; \vec{\mathbf{f}}) \in \mathcal{R}^B_{\mathsf{cm}} \\ \wedge \mathsf{mle}\left[\hat{\mathbf{f}}\right](\vec{\mathbf{r}}) = \hat{\mathbf{v}} \end{array} \right\}, \tag{9}$$

where $\mathsf{mle}\left[\hat{\mathbf{f}}\right] \in \mathcal{R}_q^{\leq 1}[X_1, \ldots, X_{\log m}]$ is the multilinear extension (Definition 2.4) of $\hat{\mathbf{f}} \in \mathcal{R}_q^m$. Recall that

$$\mathsf{NTT}(\hat{\mathbf{f}}) = \begin{bmatrix} \mathsf{NTT}(\hat{\mathbf{f}}_1)^\top \\ \vdots \\ \mathsf{NTT}(\hat{\mathbf{f}}_m)^\top \end{bmatrix} = \begin{bmatrix} \mathsf{Coef}(\vec{\mathbf{f}}_1)^\top \\ \vdots \\ \mathsf{Coef}(\vec{\mathbf{f}}_m)^\top \end{bmatrix} = \mathsf{Coef}(\vec{\mathbf{f}}) \in \mathbb{Z}_q^{m \times d}.$$

3.2 A Generic Framework for Folding

In this section, we describe a folding scheme for $\mathcal{R}_{\mathsf{acc}} := \mathcal{R}^B_{\mathsf{eval}}$ and $\mathcal{R}_{\mathsf{comp}} := \mathcal{R}^B_{\mathsf{cm}}$, or equivalently, a reduction of knowledge (Definition 2.5) from $\mathcal{R}^B_{\mathsf{eval}} \times \mathcal{R}^B_{\mathsf{cm}}$ to $\mathcal{R}^B_{\mathsf{eval}}$. This gives us a folding scheme for the Ajtai commitment opening relation. Our construction is highly modular and generic, and consists of three steps.

Step 1: Expansion. First, the relation $\mathcal{R}^B_{\mathsf{eval}} \times \mathcal{R}^B_{\mathsf{cm}}$ is reduced to $\mathcal{R}^B_{\mathsf{eval}} \times \mathcal{R}^B_{\mathsf{eval}}$ via a reduction of knowledge Π_{cm} from $\mathcal{R}^B_{\mathsf{cm}}$ to $\mathcal{R}^B_{\mathsf{eval}}$ shown in Figure 1.

Step 2: Decomposition. Next, using a decomposition protocol Π_{dec} shown in Figure 2, we reduce the relation $\mathcal{R}^B_{\mathsf{eval}} \times \mathcal{R}^B_{\mathsf{eval}}$ to

$$(\mathcal{R}^b_{\mathsf{eval}})^{2k} := \underbrace{\mathcal{R}^b_{\mathsf{eval}} \times \cdots \times \mathcal{R}^b_{\mathsf{eval}}}_{2k}$$

where $b < B$ is a norm bound smaller than B such that exists an integer $k > 1$ for which $b^k = B$. Here b and k are parameters that can be chosen dynamically depending on the relation being proved. We describe an optimization for choosing b and k in Remark 5.1.

Step 3: Folding. Finally, we reduce the relation $(\mathcal{R}^b_{\mathsf{eval}})^{2k}$ back to $\mathcal{R}^B_{\mathsf{eval}}$ using a folding protocol Π_{fold} shown in Figure 3.

By the composition theorems for reductions of knowledge (Theorem 2.1, Theorem 2.2), the composed protocol $\Pi_{\mathsf{mfold}} := \Pi_{\mathsf{fold}} \circ \Pi_{\mathsf{dec}} \circ \Pi_{\mathsf{cm}}$ is a reduction of knowledge from $\mathcal{R}^B_{\mathsf{eval}} \times \mathcal{R}^B_{\mathsf{cm}}$ to $\mathcal{R}^B_{\mathsf{eval}}$ as desired. We state the result in Theorem 3.1.

Theorem 3.1. *Let $\mathcal{R}_q \cong \mathbb{F}_{q^\tau}^{d/\tau}$ for some $\tau \in \mathbb{N}$ where $\tau \mid d$ and $1/q^\tau$ is in $\mathsf{negl}(\lambda)$. Let $\mathcal{C}_{\mathsf{small}} \subseteq \mathcal{R}_q$ be a strong sampling set for which $1/|\mathcal{C}_{\mathsf{small}}|$ is in $\mathsf{negl}(\lambda)$, and the expansion factor $T := \|\mathcal{C}_{\mathsf{small}}\|_{\mathsf{op}} \leq c$ (Definition 6) for some $c \in \mathbb{N}$. Let $\mathcal{C}$ be the strong sampling set as in Eq. (31). Let $\mathsf{pp} := (\kappa, m, \mathbf{A}, B < q/2)$ be public parameters such that $\mathsf{MSIS}^{\infty,q}_{\kappa,m,8TB}$ is hard. Set b, k such that $2kc(b-1) < B$ and $b^k = B$. Let Π_{cm}, Π_{dec}, Π_{fold} be the protocols specified in Figure 1, Figure 2 and Figure 3, respectively. Then the composed protocol $\Pi_{\mathsf{mfold}} := \Pi_{\mathsf{fold}} \circ \Pi_{\mathsf{dec}} \circ \Pi_{\mathsf{cm}}$ is a public-coin reduction of knowledge from $\mathcal{R}^B_{\mathsf{eval}} \times \mathcal{R}^B_{\mathsf{cm}}$ to $\mathcal{R}^B_{\mathsf{eval}}$.*

Proof. The protocol is public-coin as Π_{cm} and Π_{dec} are non-interactive and Π_{fold} is public-coin. For the case where $\mathcal{R}_q \cong \mathbb{Z}_q^d$, the Theorem follows from Lemma 3.1, Lemma 3.3, Theorem 3.2 and the knowledge composition theorems (Theorem 2.1 and Theorem 2.2). The proof for the general case where $\mathcal{R}_q \cong \mathbb{F}_{q^\tau}^{d/\tau}$ naturally follows from the argument in Section 3.3. □

Setup and notation. Before describing the protocols Π_{cm}, Π_{dec}, Π_{fold}, let us recall the common setup. $\mathcal{R}_q$ is the ring $\mathcal{R}_q := \mathcal{R}/q\mathcal{R}$ where $\mathcal{R} := \mathbb{Z}[X]/(X^d + 1)$ and q is a prime. Note that $\mathcal{R}_q \cong \mathbb{F}_{q^\tau}^{d/\tau}$ for some $\tau \in \mathbb{N}$ where $\tau \mid d$. The public parameter is $\mathsf{pp} := (\kappa, m, B, \mathbf{A})$ where $B < q/2$, m is a power-of-two, and $\mathbf{A} \in \mathcal{R}_q^{\kappa \times m}$ is the sampled MSIS matrix. For a vector $\vec{\mathbf{f}} \in \mathcal{R}_q^m$, we use $\hat{\mathbf{f}} := (\hat{\mathbf{f}}_1, \ldots, \hat{\mathbf{f}}_\tau) \in \mathcal{R}_q^{m \times \tau}$ to denote the ring vector such that $\mathsf{NTT}(\hat{\mathbf{f}}) := (\mathsf{NTT}(\hat{\mathbf{f}}_1), \ldots, \mathsf{NTT}(\hat{\mathbf{f}}_\tau)) \in \mathbb{F}_{q^\tau}^{m \times d}$ equals the coefficient embedding matrix of $\vec{\mathbf{f}}$, that is, $\mathsf{Coef}(\vec{\mathbf{f}}) \in \mathbb{Z}_q^{m \times d}$ as in Eq. (2).

In what follows, for ease of exposition, we assume that the prime q satisfies $q \equiv 1 \bmod 2d$ so that $\mathcal{R}_q \cong \mathbb{Z}_q^d$ and $\tau = 1$. In Section 3.3, we generalize to arbitrary prime modulus.

3.2.1 Expansion: the reduction from $\mathcal{R}^B_{\mathsf{eval}} \times \mathcal{R}^B_{\mathsf{cm}}$ to $\mathcal{R}^B_{\mathsf{eval}} \times \mathcal{R}^B_{\mathsf{eval}}$

By the parallel composition theorem (Theorem 2.2), in order to reduce from $\mathcal{R}^B_{\mathsf{eval}} \times \mathcal{R}^B_{\mathsf{cm}}$ to $\mathcal{R}^B_{\mathsf{eval}} \times \mathcal{R}^B_{\mathsf{eval}}$, it suffices to reduce $\mathcal{R}^B_{\mathsf{cm}}$ (Eq. (8)) to $\mathcal{R}^B_{\mathsf{eval}}$ (Eq. (9)). We describe the protocol Π_{cm} in Figure 1.

Input: $(\mathbb{x}; \mathbb{w}) := (\mathsf{cm} \in \mathcal{R}_q^{\kappa};\ \vec{\mathbf{f}} \in \mathcal{R}_q^m)$.
Output: $(\mathbb{x}_o; \mathbb{w}_o) := \left((0^{\log m}, \hat{\mathbf{v}} \in \mathcal{R}_q, \mathsf{cm});\ \vec{\mathbf{f}}\right)$.
The protocol $\langle \mathsf{P}(\mathsf{pp}, \mathbb{x}; \mathbb{w}), \mathsf{V}(\mathsf{pp}, \mathbb{x}) \rangle$**:**

1. $\mathsf{P} \to \mathsf{V}$: P sends V the evaluation $\hat{\mathbf{v}} := \mathsf{mle}\left[\hat{\mathbf{f}}\right](0^{\log m})$.
2. V outputs $\mathbb{x}_o := (0^{\log m}, \hat{\mathbf{v}}, \mathsf{cm})$. P outputs $\mathbb{w}_o := \vec{\mathbf{f}}$.

Figure 1: The protocol Π_{cm} that reduces $\mathcal{R}^B_{\mathsf{cm}}$ to $\mathcal{R}^B_{\mathsf{eval}}$.

Lemma 3.1. *Π_{cm} is a reduction of knowledge from $\mathcal{R}^B_{\mathsf{cm}}$ to $\mathcal{R}^B_{\mathsf{eval}}$ for any $B \in \mathbb{N}$.*

Proof. *Public reducibility:* Given instance $\mathbb{x} = \mathsf{cm}$ and transcript $\hat{\mathbf{v}}$, one can output $\mathbb{x}_o = ((0^{\log m}, \hat{\mathbf{v}}, \mathsf{cm}))$.
Completeness: Given $(\mathsf{pp}, \mathsf{cm}; \vec{\mathbf{f}}) \in \mathcal{R}^B_{\mathsf{cm}}$, the honest prover can compute and send $\hat{\mathbf{v}} := \mathsf{mle}\left[\hat{\mathbf{f}}\right](0^{\log m})$ such that $((0^{\log m}, \hat{\mathbf{v}}, \mathsf{cm}); \vec{\mathbf{f}}) \in \mathcal{R}^B_{\mathsf{eval}}$. The honest verifier will output instance $(0^{\log m}, \hat{\mathbf{v}}, \mathsf{cm})$ and the honest prover will output $\vec{\mathbf{f}}$.
Knowledge soundness: By definition of $\mathcal{R}^B_{\mathsf{eval}}$ (Eq. (9)), given any $((\vec{\mathbf{r}}, \hat{\mathbf{v}}, \mathsf{cm}); \vec{\mathbf{f}}) \in \mathcal{R}^B_{\mathsf{eval}}$, we can extract witness $(\mathsf{cm}; \vec{\mathbf{f}})$ that is in the relation $\mathcal{R}^B_{\mathsf{cm}}$. □

3.2.2 Decomposition: The reduction from $(\mathcal{R}^B_{\mathsf{eval}})^2$ to $(\mathcal{R}^b_{\mathsf{eval}})^{2k}$

Intuitively, the decomposition step splits the two witness vectors, each with a norm less than B, into $2k$ witness vectors with a much smaller norm b. (Typically, b is only 2 or 4 in practice.) This allows them to be folded back later (in the folding step) into a vector with a norm less than B.

By Theorem 2.2, it suffices to construct a protocol Π^*_{dec} that reduces $\mathcal{R}^B_{\mathsf{eval}}$ to $(\mathcal{R}^b_{\mathsf{eval}})^k$, and the reduction of knowledge from $\mathcal{R}^B_{\mathsf{eval}} \times \mathcal{R}^B_{\mathsf{eval}}$ to $(\mathcal{R}^b_{\mathsf{eval}})^{2k}$ is $\Pi_{\mathsf{dec}} := \Pi^*_{\mathsf{dec}} \times \Pi^*_{\mathsf{dec}}$ that runs two instances of Π^*_{dec} in parallel.

More generally, we construct a reduction of knowledge from a relation $\mathcal{R}^B_{\mathsf{hom}}$ to $(\mathcal{R}^b_{\mathsf{hom}})^k$. Let $\mathcal{L}$ be an $\mathcal{R}_q$-module homomorphism from $\mathcal{R}_q^m$ to an $\mathcal{R}_q$-module $\mathcal{Y}$. We treat $\mathcal{L}$ as a part of the public parameter $\mathsf{pp} := (\mathcal{R}_q, m, B < q/2, \mathcal{L})$. Here $\mathcal{R}^B_{\mathsf{hom}}$ is a generalization of

$\mathcal{R}^B_{\mathsf{eval}}$ from (9) defined as

$$\mathcal{R}^B_{\mathsf{hom}} := \left\{ (\mathsf{pp},\ \mathbb{x} := (\vec{\mathbf{r}} \in \mathcal{R}_q^{\log m}, \hat{\mathbf{v}} \in \mathcal{R}_q, y \in \mathcal{Y});\ \vec{\mathbf{f}} \in \mathcal{R}_q^m) \ :\ \begin{array}{c} y = \mathcal{L}(\vec{\mathbf{f}}) \wedge \|\vec{\mathbf{f}}\|_\infty < B \\ \wedge \mathsf{mle}\left[\hat{\mathbf{f}}\right](\vec{\mathbf{r}}) = \hat{\mathbf{v}} \end{array} \right\}, \tag{10}$$

Clearly, $\mathcal{R}^B_{\mathsf{eval}}$ from (9) is a special case of $\mathcal{R}^B_{\mathsf{hom}}$ where $\mathcal{L}(\vec{\mathbf{f}}) := \mathbf{A}\vec{\mathbf{f}}$ and $\mathcal{Y} := \mathcal{R}_q^\kappa$.

For a positive integer $B < q/2$, choose b, k such that $b^k = B$. For notational convenience, for an m-vector $\vec{\mathbf{f}} \in \mathcal{R}_q^m$ where $\|\vec{\mathbf{f}}\|_\infty < B$, we use $\mathsf{split}_{b,k}(\vec{\mathbf{f}})$ to denote the algorithm that decomposes $\vec{\mathbf{f}}$ into an $m \times k$ matrix $\vec{\mathbf{F}} := (\vec{\mathbf{f}}_0, \ldots, \vec{\mathbf{f}}_{k-1}) \in \mathcal{R}_q^{m \times k}$, such that the coefficients of each $\mathcal{R}_q$-element in $\vec{\mathbf{F}}$ has absolute value less than b and

$$\vec{\mathbf{f}} = \vec{\mathbf{F}} \cdot \left[1, b, b^2, \ldots, b^{k-1}\right]^\top = \sum_{i=0}^{k-1} b^i \cdot \vec{\mathbf{f}}_i \,. \tag{11}$$

For example, for $k = 2$ and $m = 1$, assume that $b := \sqrt{B}$ is an integer. Given a polynomial $f = a_0 + a_1 X \in \mathcal{R}_q$ where $|a_0|, |a_1| < B$, we decompose it to $\mathsf{split}_{b,k}(f) = (f_0, f_1) = (c_0 + c_1 X,\ \ d_0 + d_1 X)$, where $c_i := a_i \bmod b$ and $d_i := \lfloor a_i / b \rfloor$ for $i \in \{0, 1\}$. Then $\left|c_i\right| < b$ and $\left|d_i\right| < b$, and $f = f_0 + b f_1$.

Input: $\mathbb{x} := (\vec{\mathbf{r}} \in \mathcal{R}_q^{\log m}, \hat{\mathbf{v}} \in \mathcal{R}_q, y \in \mathcal{Y})$ and $\mathbb{w} := \vec{\mathbf{f}} \in \mathcal{R}_q^m$
Output: $[\mathbb{x}_i = (\vec{\mathbf{r}}, \hat{\mathbf{v}}_i, y_i), \mathbb{w}_i = \vec{\mathbf{f}}_i]_{i=0}^{k-1}$
The protocol $\langle \mathsf{P}(\mathsf{pp}, \mathbb{x}; \mathbb{w}), \mathsf{V}(\mathsf{pp}, \mathbb{x}) \rangle$**:**

1. $\mathsf{P} \to \mathsf{V}$: Let $\vec{\mathbf{F}} := (\vec{\mathbf{f}}_0, \ldots, \vec{\mathbf{f}}_{k-1}) := \mathsf{split}_{b,k}(\vec{\mathbf{f}})$. P sends V the values $[y_i, \hat{\mathbf{v}}_i]_{i=0}^{k-1}$ where

$$y_i := \mathcal{L}(\vec{\mathbf{f}}_i)\,, \qquad \hat{\mathbf{v}}_i := \mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}})$$

2. V checks that $\sum_{i=0}^{k-1} b^i \cdot y_i \stackrel{?}{=} y$, and $\sum_{i=0}^{k-1} b^i \cdot \hat{\mathbf{v}}_i \stackrel{?}{=} \hat{\mathbf{v}}$.
3. V outputs $[\mathbb{x}_i := (\vec{\mathbf{r}}, \hat{\mathbf{v}}_i, y_i)]_{i=0}^{k-1}$. P outputs $[\mathbb{w}_i := \vec{\mathbf{f}}_i]_{i=0}^{k-1}$.

Figure 2: The protocol Π^*_{dec} that reduces $\mathcal{R}^B_{\mathsf{hom}}$ to $(\mathcal{R}^b_{\mathsf{hom}})^k$.

With this notation in place, we describe the protocol Π^*_{dec} in Figure 2. Before proving that Π^*_{dec} is a reduction of knowledge, we state a useful lemma. Informally, it states that a linear combination of instance-witness pairs will be in the relation if every individual instance-witness pair is in the relation. It's important to note that the combiners $[\rho_i]_{i=1}^{\ell}$ can be arbitrary elements in $\mathcal{R}_q$. This generalization extends beyond the decomposition case where the combiners b_i are in $\mathbb{Z}_q \subseteq \mathcal{R}_q$. Looking ahead, this generalization is useful later in the folding protocol (Figure 3).

Lemma 3.2. *Fix a power-of-two $m \in \mathbb{N}$, let $\vec{\mathbf{r}} \in \mathbb{Z}_q^{\log m}$ be a vector and let $\mathcal{L} : \mathcal{R}_q^m \to \mathcal{Y}$ be an $\mathcal{R}_q$-module homomorphism. Fix $\ell \in \mathbb{N}$. For any $[\rho_i]_{i=1}^{\ell} \in \mathcal{R}_q^{\ell}$ and $[\hat{\mathbf{v}}_i, y_i; \vec{\mathbf{f}}_i]_{i=1}^{\ell}$ such that $y_i = \mathcal{L}(\vec{\mathbf{f}}_i)$ and $\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}) = \hat{\mathbf{v}}_i$ for all $i \in [\ell]$. Set $\hat{\mathbf{v}}_o, y_o, \vec{\mathbf{f}}_o$ such that*

$$\mathsf{NTT}(\hat{\mathbf{v}}_o) = \sum_{i=1}^{\ell} \mathsf{RotSum}(\rho_i, \mathsf{NTT}(\hat{\mathbf{v}}_i)), \qquad y_o := \sum_{i=1}^{\ell} \rho_i \cdot y_i, \qquad \vec{\mathbf{f}}_o := \sum_{i=1}^{\ell} \rho_i \cdot \vec{\mathbf{f}}_i,$$

where RotSum *is defined in Lemma 2.1. Then $y_o = \mathcal{L}(\vec{\mathbf{f}}_o)$ and $\mathsf{mle}\left[\hat{\mathbf{f}}_o\right](\vec{\mathbf{r}}) = \hat{\mathbf{v}}_o$.*

Proof. First,

$$\mathcal{L}(\vec{\mathbf{f}}_o) = \mathcal{L}\left(\sum_{i=1}^{\ell} \rho_i \cdot \vec{\mathbf{f}}_i\right) = \sum_{i=1}^{\ell} \rho_i \cdot \mathcal{L}(\vec{\mathbf{f}}_i) = \sum_{i=1}^{\ell} \rho_i \cdot y_i = y_o$$

where the 2nd equality holds by the homomorphic property of $\mathcal{L}$.

For ease of exposition, we define $\bar{\mathbf{v}}_o, \bar{\mathbf{v}}_1, \ldots, \bar{\mathbf{v}}_\ell \in \mathcal{R}_q$ as the values such that $\mathsf{NTT}(\hat{\mathbf{v}}_o) = \mathsf{Coef}(\bar{\mathbf{v}}_o)$ and $\mathsf{NTT}(\hat{\mathbf{v}}_i) = \mathsf{Coef}(\bar{\mathbf{v}}_i)$ for all $i \in [\ell]$. We have that

$$\begin{aligned}
\mathsf{Coef}(\bar{\mathbf{v}}_o) = \mathsf{NTT}(\hat{\mathbf{v}}_o) &= \sum_{i=1}^{\ell} \mathsf{RotSum}(\rho_i, \mathsf{NTT}(\hat{\mathbf{v}}_i)) \\
&= \sum_{i=1}^{\ell} \mathsf{RotSum}(\rho_i, \mathsf{Coef}(\bar{\mathbf{v}}_i)) = \sum_{i=1}^{\ell} \mathsf{RotSum}\left(\rho_i, \left\langle \mathsf{Coef}(\vec{\mathbf{f}}_i), \mathsf{tensor}(\vec{\mathbf{r}}) \right\rangle\right) \\
&= \sum_{i=1}^{\ell} \left\langle \mathsf{RotSum}(\rho_i, \mathsf{Coef}(\vec{\mathbf{f}}_i)), \mathsf{tensor}(\vec{\mathbf{r}}) \right\rangle = \sum_{i=1}^{\ell} \left\langle \mathsf{Coef}(\rho_i \cdot \vec{\mathbf{f}}_i), \mathsf{tensor}(\vec{\mathbf{r}}) \right\rangle \\
&= \left\langle \mathsf{Coef}\left(\sum_{i=1}^{\ell} \rho_i \cdot \vec{\mathbf{f}}_i\right), \mathsf{tensor}(\vec{\mathbf{r}}) \right\rangle = \left\langle \mathsf{Coef}(\vec{\mathbf{f}}_o), \mathsf{tensor}(\vec{\mathbf{r}}) \right\rangle,
\end{aligned}$$

where the 4th equality holds because

$$\mathsf{Coef}(\bar{\mathbf{v}}_i) = \mathsf{NTT}(\hat{\mathbf{v}}_i) = \mathsf{mle}\left[\mathsf{Coef}(\vec{\mathbf{f}}_i)\right][\vec{\mathbf{r}}] = \left\langle \mathsf{Coef}(\vec{\mathbf{f}}_i), \mathsf{tensor}(\vec{\mathbf{r}}) \right\rangle$$

by Lemma A.1 and the facts that $\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}) = \hat{\mathbf{v}}_i$, $\mathsf{NTT}(\hat{\mathbf{f}}_i) = \mathsf{Coef}(\vec{\mathbf{f}}_i)$ and $\mathsf{NTT}(\vec{\mathbf{r}}) = \underbrace{(\vec{\mathbf{r}}, \ldots, \vec{\mathbf{r}})}_{d}$. The 5th equality holds by rearranging the terms and by the property of inner products; the 6th equality holds because $\mathsf{RotSum}(\mathbf{a}, \mathsf{Coef}(\mathbf{b})) = \mathsf{Coef}(\mathbf{a} \cdot \mathbf{b})$ for any $\mathbf{a}, \mathbf{b} \in \mathcal{R}_q$ (2nd claim in Lemma 2.1); the 7th equality is by additivity of inner products and coefficient embedding. Therefore, by Lemma A.1, we have that

$$\hat{\mathbf{v}}_o = \mathsf{NTT}^{-1}(\mathsf{Coef}(\bar{\mathbf{v}}_o)) = \mathsf{NTT}^{-1}\left(\left\langle \mathsf{Coef}(\vec{\mathbf{f}}_o), \mathsf{tensor}(\vec{\mathbf{r}}) \right\rangle\right) = \mathsf{mle}\left[\hat{\mathbf{f}}_o\right](\vec{\mathbf{r}})$$

as required. □

Remark 3.1 (Supporting general $\vec{\mathbf{r}}$.)**.** *Lemma 3.2 requires $\vec{\mathbf{r}}$ to be in $\mathbb{Z}_q^{\log m}$. However, with a minor modification, the proof naturally extends to the case where $\vec{\mathbf{r}}$ is in $\mathcal{C}^{\log m}$ where $\mathcal{C}$ is defined as in Eq. (31). This is useful in the folding protocol (Figure 3) to support small modulus q while preserving negligible sumcheck soundness error. We defer the details to Section 3.3.*

Next we show that Π^*_{dec} is a reduction of knowledge.

Lemma 3.3. *Fix $\mathcal{R}_q \cong \mathbb{Z}_q^d$. For any $B < q/2$ and any b, k such that $b^k = B$, Π^*_{dec} is a reduction of knowledge from $\mathcal{R}^B_{\mathsf{hom}}$ to $(\mathcal{R}^b_{\mathsf{hom}})^k$.*

The proof follows from Lemma 3.4 and Lemma 3.5. Again, we emphasize that the proof naturally extends to the case where $\mathcal{R}_q \cong \mathbb{F}_{q^\tau}^{d/\tau}$. More details are deferred to Section 3.3.

Lemma 3.4. Π^*_{dec} *satisfies public reducibility and completeness.*

Proof. *Public reducibility:* Given instance $\mathbb{x} = (\vec{\mathbf{r}}, \hat{\mathbf{v}}, y)$ and transcript $[y_i, \hat{\mathbf{v}}_i]_{i=0}^{k-1}$, output $[\mathbb{x}_i := (\vec{\mathbf{r}}, \hat{\mathbf{v}}_i, y_i)]_{i=0}^{k-1}$ if the verifier checks pass and $\perp$ otherwise.

Completeness: Let $(\mathbb{x} = (\vec{\mathbf{r}}, \hat{\mathbf{v}}, y); \mathbb{w} := \vec{\mathbf{f}}) \leftarrow \mathcal{A}(\mathsf{pp})$ denote adversary $\mathcal{A}$'s chosen input for $\mathcal{R}_1 := \mathcal{R}^B_{\mathsf{hom}}$ where $\mathsf{pp} := (\mathcal{R}_q, m, B < q/2, \mathcal{L}) \leftarrow \mathsf{Setup}(1^\lambda)$ is the public parameter. WLOG we assume that $(\mathsf{pp}, \mathbb{x}; \mathbb{w})$ is in $\mathcal{R}^B_{\mathsf{hom}}$. The protocol execution $\langle \mathsf{P}(\mathsf{pp}, \mathbb{x}, \mathbb{w}), \mathsf{V}(\mathsf{pp}, \mathbb{x}) \rangle$ proceeds as follows:

1. P computes $\vec{\mathbf{F}} := (\vec{\mathbf{f}}_0, \ldots, \vec{\mathbf{f}}_{k-1}) \leftarrow \mathsf{split}_{b,k}(\vec{\mathbf{f}})$, and sends $y_i = \mathcal{L}(\vec{\mathbf{f}}_i)$, $\hat{\mathbf{v}}_i = \mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}})$ for every $i \in [0, k)$.
2. V checks that $y \stackrel{?}{=} \sum_{i=0}^{k-1} b^i \cdot y_i$ and $\hat{\mathbf{v}} \stackrel{?}{=} \sum_{i=0}^{k-1} b^i \cdot \hat{\mathbf{v}}_i$. It outputs $\perp$ and halts if the check fails.
3. P outputs $[\vec{\mathbf{f}}_i]_{i=0}^{k-1}$. V accepts and outputs $[\vec{\mathbf{r}}, \hat{\mathbf{v}}_i, y_i]_{i=0}^{k-1}$.

We show that V accepts in the honest execution. First,

$$\sum_{i=0}^{k-1} b^i \cdot y_i = \sum_{i=0}^{k-1} b^i \cdot \mathcal{L}(\vec{\mathbf{f}}_i) = \mathcal{L}\left(\sum_{i=0}^{k-1} b^i \cdot \vec{\mathbf{f}}_i\right) = y$$

where the 1st equality is by definition of y_i; the 2nd equality follows from the properties of the $\mathcal{R}_q$-module homomorphism $\mathcal{L}$; the last equality holds because $\vec{\mathbf{f}} = \sum_{i=0}^{k-1} b^i \cdot \vec{\mathbf{f}}_i$ as in Eq. (11) and $\mathcal{L}(\vec{\mathbf{f}}) = y$ by the assumption that $(\mathsf{pp} := (\mathcal{R}_q, m, B < q/2, \mathcal{L}), \mathbb{x}; \mathbb{w}) \in \mathcal{R}^B_{\mathsf{hom}}$.

Similarly, we have that

$$\sum_{i=0}^{k-1} b^i \cdot \hat{\mathbf{v}}_i = \sum_{i=0}^{k-1} b^i \cdot \mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}) = \mathsf{mle}\left[\hat{\mathbf{f}}\right](\vec{\mathbf{r}}) = \hat{\mathbf{v}}\,.$$

The 1st equality is by definition of $\hat{\mathbf{v}}_i$; the 2nd equality holds because (i) the map $g_{\vec{\mathbf{r}}}(f) := f(\vec{\mathbf{r}})$ satisfies that $g_{\vec{\mathbf{r}}}(a \cdot f_1 + b \cdot f_2) = a \cdot g_{\vec{\mathbf{r}}}(f_1) + b \cdot g_{\vec{\mathbf{r}}}(f_2)$ for any multilinear polynomials f_1, f_2 and $a, b \in \mathbb{Z}_q$; and (ii) $\hat{\mathbf{f}} = \sum_{i=0}^{k-1} b^i \cdot \hat{\mathbf{f}}_i$ given that

$$\mathsf{NTT}(\hat{\mathbf{f}}) = \mathsf{Coef}(\vec{\mathbf{f}}) = \mathsf{Coef}\left(\sum_{i=0}^{k-1} b^i \cdot \vec{\mathbf{f}}_i\right) = \sum_{i=0}^{k-1} b^i \cdot \mathsf{Coef}\left(\vec{\mathbf{f}}_i\right) = \sum_{i=0}^{k-1} b^i \cdot \mathsf{NTT}(\hat{\mathbf{f}}_i) = \mathsf{NTT}\left(\sum_{i=0}^{k-1} b^i \cdot \hat{\mathbf{f}}_i\right).$$

The last equality holds because $(\mathsf{pp}, \mathbb{x}; \mathbb{w}) \in \mathcal{R}^B_{\mathsf{hom}}$ by assumption. Thus, V accepts in the honest execution.

Next, we show that $(\mathsf{pp}, [(\vec{\mathbf{r}}, \hat{\mathbf{v}}_i, y_i); \vec{\mathbf{f}}_i]_{i=0}^{k-1})$ is in $\mathcal{R}_2 := (\mathcal{R}^b_{\mathsf{hom}})^k$. For every $i \in [0, k)$, recall that $y_i = \mathcal{L}(\vec{\mathbf{f}}_i)$ and $\hat{\mathbf{v}}_i = \mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}})$. Moreover, by definition of $\mathsf{split}_{b,k}(\vec{\mathbf{f}})$ (Eq. (11)), we have that $\|\vec{\mathbf{f}}_i\|_\infty < b$. Therefore, $(\mathsf{pp}, (\vec{\mathbf{r}}, \hat{\mathbf{v}}_i, y_i); \vec{\mathbf{f}}_i) \in \mathcal{R}^b_{\mathsf{hom}}$ and completeness holds. □

Lemma 3.5. Π^*_{dec} *satisfies knowledge soundness.*

Proof. Let $(\mathbb{x} := (\vec{\mathbf{r}}, \hat{\mathbf{v}}, y); \mathsf{state}) \leftarrow \mathcal{A}(\mathsf{pp})$ denote adversary $\mathcal{A}$'s chosen input instance for $\mathcal{R}_1 := \mathcal{R}^B_{\mathsf{hom}}$, where $\mathsf{pp} := (\mathcal{R}_q, m, B < q/2, \mathcal{L}) \leftarrow \mathsf{Setup}(1^\lambda)$ is the public parameter. The extractor Ext proceeds as follows:

1. Simulate the protocol $\langle \mathsf{P}^*(\mathsf{pp}, \mathbb{x}, \mathsf{state}), \mathsf{V}(\mathsf{pp}, \mathbb{x}) \rangle$ where P^* is the malicious prover.
2. Output $\perp$ if V rejects. Otherwise let $(\mathbb{x}_o, \mathbb{w}_o) := [(\vec{\mathbf{r}}, \hat{\mathbf{v}}_i, y_i); \vec{\mathbf{f}}_i]_{i=0}^{k-1}$ be the protocol output. (Note that $\vec{\mathbf{r}}$ is the same with that in the input instance $\mathbb{x}$ to pass the verification check.) The extractor outputs witness

$$\mathbb{w} := \vec{\mathbf{f}} := \sum_{i=0}^{k-1} b^i \cdot \vec{\mathbf{f}}_i \,. \tag{12}$$

Next, we show that if V accepts and the output satisfies that $(\mathsf{pp}, \mathbb{x}_o, \mathbb{w}_o)$ is in $\mathcal{R}_2 := (\mathcal{R}^b_{\mathsf{hom}})^k$, then the extracted witness $\vec{\mathbf{f}}$ satisfies that $(\vec{\mathbf{r}}, \hat{\mathbf{v}}, y; \vec{\mathbf{f}}) \in \mathcal{R}_1 := \mathcal{R}^B_{\mathsf{hom}}$. Since V accepts, we have that $y = \sum_{i=0}^{2k-1} b^i \cdot y_i$ and $\hat{\mathbf{v}} = \sum_{i=0}^{2k-1} b^i \cdot \hat{\mathbf{v}}_i$. Recall that $y_i = \mathcal{L}(\vec{\mathbf{f}}_i)$ and $\hat{\mathbf{v}}_i = \mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}})$ for all $i \in [0, k)$ by assumption, thus by Lemma 3.2, we have that $y = \mathcal{L}(\vec{\mathbf{f}})$ and $\hat{\mathbf{v}} = \mathsf{mle}\left[\hat{\mathbf{f}}\right](\vec{\mathbf{r}})$. Moreover, note that $\|\vec{\mathbf{f}}_i\|_\infty < b$ for all $i \in [0, k)$ because $(\vec{\mathbf{r}}, \hat{\mathbf{v}}_i, y_i; \vec{\mathbf{f}}_i)$ is in $\mathcal{R}^b_{\mathsf{hom}}$ by assumption. Since $b^k = B < q/2$ and $\vec{\mathbf{f}} = \sum_{i=0}^{k-1} b^i \cdot \vec{\mathbf{f}}_i$, we have

$$\|\vec{\mathbf{f}}\|_\infty = \|\sum_{i=0}^{k-1} b^i \cdot \vec{\mathbf{f}}_i\|_\infty \le \sum_{i=0}^{k-1} b^i \cdot \|\vec{\mathbf{f}}_i\|_\infty \le \sum_{i=0}^{k-1} b^i \cdot (b-1) < b^k = B \,.$$

In summary, $y = \mathcal{L}(\vec{\mathbf{f}})$, $\hat{\mathbf{v}} = \mathsf{mle}\left[\hat{\mathbf{f}}\right](\vec{\mathbf{r}})$ and $\|\vec{\mathbf{f}}\|_\infty < B$ and thus $(\vec{\mathbf{r}}, \hat{\mathbf{v}}, y; \vec{\mathbf{f}}) \in \mathcal{R}^B_{\mathsf{hom}}$. □

3.2.3 Folding: The reduction from $(\mathcal{R}^b_{\mathsf{eval}})^{2k}$ to $\mathcal{R}^B_{\mathsf{eval}}$

We now describe the core protocol Π_{fold} that folds $2k$ instance-witness pairs of $\mathcal{R}^b_{\mathsf{eval}}$ from (9) into a single instance-witness pair in $\mathcal{R}^B_{\mathsf{eval}}$. More generally, the protocol is a reduction of knowledge from $(\mathcal{R}^b_{\mathsf{hom}})^{2k}$ from (10) to $\mathcal{R}^B_{\mathsf{hom}}$ with a further restriction that the public parameter, the *sampled* homomorphism $\mathcal{L}$, is **relaxed binding** as in Section 2.3. This relaxed binding property, defined for bound $2B$ and the challenge space $\mathcal{C}_{\mathsf{small}}$, ensures the hardness of finding two distinct weak openings for a commitment cm. Recall that a $2B$-weak opening $(\rho \in \mathcal{C}_{\mathsf{small}} - \mathcal{C}_{\mathsf{small}}, \vec{\mathbf{x}})$ of cm satisfies that $\rho \cdot \mathsf{cm} = \mathbf{A}\vec{\mathbf{x}} \bmod q$ with the norm constraint $\|\vec{\mathbf{x}}\|_\infty < 2B$. Thus, $\mathcal{R}^B_{\mathsf{eval}}$ from (9) is a special case of $\mathcal{R}^B_{\mathsf{hom}}$ where the sampled homomorphism $\mathcal{L}$ is given by $\mathcal{L}(\vec{\mathbf{f}}) := \mathbf{A}\vec{\mathbf{f}}$, and the $2B$-relaxed binding property follows from the hardness of $\mathsf{MSIS}^{\infty,q}_{\kappa,m,8TB}$ where $T = \|\mathcal{C}_{\mathsf{small}}\|_{\mathsf{op}}$ is the expansion factor of $\mathcal{C}_{\mathsf{small}}$.

Our protocol folds the $2k$ witness vectors (with norm less than b) into a witness vector of norm less than B (where $b < B < q/2$), using *small* random scalars $[\rho_i]_{i=1}^{2k}$ sampled from a strong sampling set $\mathcal{C}_{\mathsf{small}} \subseteq \mathcal{R}_q$. Additionally and crucially, Π_{fold} runs a sum-check protocol to enable extractions of the $2k$ witness vectors with small norms. The sum-check is for a polynomial $g(\vec{\mathbf{x}}) := \sum_{i=1}^{2k}(\alpha_i g_{1,i}(\vec{\mathbf{x}}) + \mu_i g_{2,i}(\vec{\mathbf{x}}))$ with random scalars $[\alpha_i, \mu_i]_{i=1}^{2k}$. Informally, we can understand it as a batch of $4k$ separate sum-checks for polynomials $[g_{1,i}, g_{2,i}]_{i=1}^{2k}$, respectively. The sum-check for $g_{1,i}$ (defined in Eq. (15)) is used to verify that the evaluation statement $\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_i) = \hat{\mathbf{v}}_i$ holds for all $i \in [2k]$. For every $i \in [2k]$, the sum-check for $g_{2,i}$ (defined in Eq. (16)) is used to verify that

$$\prod_{j=-(b-1)}^{b-1} \left(\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{x}}) - j\right) = 0 \quad \text{for all } \vec{\mathbf{x}} \in \{0,1\}^{\log m}$$

This is the same as the Hadamard product check (the norm bound check)

$$\hat{\mathbf{f}}_i \circ \left[\bigcirc_{j=1}^{b-1}(\hat{\mathbf{f}}_i - \hat{j}) \circ (\hat{\mathbf{f}}_i + \hat{j})\right] = \hat{0},$$

in the relation $\mathcal{R}^b_{\mathsf{cm}}$ from (8).

We describe the protocol Π_{fold} in Figure 3. We now see why the relation $\mathcal{R}^B_{\mathsf{cm}}$ from (8) had to be expanded to the related $\mathcal{R}^B_{\mathsf{eval}}$ from (9). Protocol Π_{fold} reduces the claimed properties about the $2k$ input instances to verifying that $\mathsf{mle}\left[\hat{\mathbf{f}}_o\right](\vec{\mathbf{r}}_o) = \hat{\mathbf{v}}_o$, where $(\vec{\mathbf{r}}_o, \hat{\mathbf{v}}_o)$ is part of the output folded instance. Adding this evaluation check to the relation $\mathcal{R}^B_{\mathsf{eval}}$ forces the prover to output a folded statement that satisfies this equality. The verifier cannot check this relation itself as part of Π_{fold} because it does not have access to $\vec{\mathbf{f}}_o$.

The following lemma shows that the protocol Π_{fold} in Figure 3 is a reduction of knowledge from $(\mathcal{R}^b_{\mathsf{hom}})^{2k}$ to $\mathcal{R}^B_{\mathsf{hom}}$ assuming that $\mathcal{L}$ is a relaxed binding.

Parameters: Strong sampling sets $\mathcal{C}, \mathcal{C}_{\mathsf{small}} \subseteq \mathcal{R}_q$ where $\mathcal{C}$ is defined as in Eq. (31) and $\mathcal{C}_{\mathsf{small}}$ has expansion factor $\|\mathcal{C}_{\mathsf{small}}\|_{\mathsf{op}} \leq c \in \mathbb{N}$ as in (6).
Input: $[\mathbb{x}_i := (\vec{\mathbf{r}}_i, \hat{\mathbf{v}}_i, y_i) \in \mathcal{R}_q^{\log m} \times \mathcal{R}_q \times \mathcal{Y}]_{i=1}^{2k}$ and $[\mathbb{w}_i := \vec{\mathbf{f}}_i \in \mathcal{R}_q^m]_{i=1}^{2k}$
Output: $\mathbb{x}_o := (\vec{\mathbf{r}}_o, \hat{\mathbf{v}}_o, y_o)$, $\mathbb{w}_o := \vec{\mathbf{f}}_o$
The protocol $\langle \mathsf{P}(\mathsf{pp}, \mathbb{x}; \mathbb{w}), \mathsf{V}(\mathsf{pp}, \mathbb{x}) \rangle$ where $\mathbb{x} = [\mathbb{x}_i]_{i=1}^{2k}$ and $\mathbb{w} = [\mathbb{w}_i]_{i=1}^{2k}$:

1. $\mathsf{V} \to \mathsf{P}$: V sends P challenges $[\alpha_i, \mu_i]_{i=1}^{2k} \xleftarrow{\$} (\mathcal{C} \times \mathcal{C})^{2k}$ and $\vec{\beta} \xleftarrow{\$} \mathcal{C}^{\log m}$.
2. $\mathsf{V} \leftrightarrow \mathsf{P}$: P and V run a sum-check protocol for the claim

$$\sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log m}} g(\vec{\mathbf{b}}) = \sum_{i=1}^{2k} \alpha_i \hat{\mathbf{v}}_i \,, \tag{13}$$

where the polynomial $g(\vec{\mathbf{x}}) \in \mathcal{R}_q^{\leq 2b}[X_1, \ldots, X_{\log m}]$ is defined as

$$g(\vec{\mathbf{x}}) := g_{\mathsf{eval}}(\vec{\mathbf{x}}) + g_{\mathsf{norm}}(\vec{\mathbf{x}}) \,, \tag{14}$$

$$g_{\mathsf{eval}}(\vec{\mathbf{x}}) := \sum_{i=1}^{2k} \alpha_i \cdot g_{1,i}(\vec{\mathbf{x}}) \quad \text{where} \quad g_{1,i}(\vec{\mathbf{x}}) := eq(\vec{\mathbf{r}}_i, \vec{\mathbf{x}}) \cdot \mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{x}}) \,, \tag{15}$$

$$g_{\mathsf{norm}}(\vec{\mathbf{x}}) := \sum_{i=1}^{2k} \mu_i \cdot g_{2,i}(\vec{\mathbf{x}}) \quad \text{where} \quad g_{2,i}(\vec{\mathbf{x}}) := eq(\vec{\beta}, \vec{\mathbf{x}}) \cdot \prod_{j=-(b-1)}^{b-1} \left(\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{x}}) - j\right) \tag{16}$$

The sumcheck protocol reduces checking (13) to checking the evaluation claim $g(\vec{\mathbf{r}}_o) \stackrel{?}{=} s$, where $s \in \mathcal{R}_q$ and $\vec{\mathbf{r}}_o \xleftarrow{\$} \mathcal{C}^{\log m}$ is a sum-check challenge sampled by V.

3. $\mathsf{P} \to \mathsf{V}$: P sends V values $\left[\theta_i := \mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_o)\right]_{i=1}^{2k}$.
4. V computes $[\mathbf{e}_i := eq(\vec{\mathbf{r}}_i, \vec{\mathbf{r}}_o)]_{i=1}^{2k}$ and $\mathbf{e}^* := eq(\vec{\beta}, \vec{\mathbf{r}}_o)$ and checks that

$$s \stackrel{?}{=} \sum_{i=1}^{2k} \left[\alpha_i \mathbf{e}_i \theta_i + \mu_i \mathbf{e}^* \cdot \prod_{j=1-b}^{b-1} (\theta_i - j)\right] .$$

5. $\mathsf{V} \to \mathsf{P}$: V sends P random challenge $[\rho_i]_{i=1}^{2k} \xleftarrow{\$} \mathcal{C}_{\mathsf{small}}^{2k}$.
6. V outputs $\mathbb{x}_o := (\vec{\mathbf{r}}_o, \hat{\mathbf{v}}_o, y_o)$ where[a]

$$\mathsf{NTT}(\hat{\mathbf{v}}_o) = \sum_{i=1}^{2k} \mathsf{RotSum}(\rho_i, \mathsf{NTT}(\theta_i)) \,, \qquad y_o := \sum_{i=1}^{2k} \rho_i y_i \,.$$

7. P further outputs $\mathbb{w}_o := \vec{\mathbf{f}}_o = \sum_{i=1}^{2k} \rho_i \cdot \vec{\mathbf{f}}_i$.

[a] If $\mathcal{R}_q \cong \mathbb{F}_{q^\tau}^{d/\tau}$ for some $\tau > 1$, the value $\hat{\mathbf{v}}_o$ is replaced with $\hat{V}_o \in \mathcal{R}_q^\tau$ and verifier check is modified as in Eq. (32).

Figure 3: The protocol Π_{fold} that reduces $(\mathcal{R}_{\mathsf{hom}}^b)^{2k}$ to $\mathcal{R}_{\mathsf{hom}}^B$.

Theorem 3.2. *Let $\mathcal{R}_q \cong \mathbb{F}_{q^\tau}^{d/\tau}$ for some $\tau \in \mathbb{N}$ where $\tau \mid d$ and $1/q^\tau$ is in $\mathsf{negl}(\lambda)$. Let $\mathcal{C}_{\mathsf{small}} \subseteq \mathcal{R}_q$ be a strong sampling set for which $1/|\mathcal{C}_{\mathsf{small}}|$ is in $\mathsf{negl}(\lambda)$, and the expansion factor $\|\mathcal{C}_{\mathsf{small}}\|_{\mathsf{op}} \leq c$ (Definition 6) for some $c \in \mathbb{N}$. Let $\mathcal{C}$ be the strong sampling set as in Eq. (31). Let $\mathsf{pp} := (m, B < q/2, \mathcal{L})$ be the public parameters where the sampled $\mathcal{R}_q$-module homomorphism $\mathcal{L} : \mathcal{R}_q^m \to \mathcal{Y}$ is a $2B$-relaxed binding (Section 2.3) for challenge space $\mathcal{C}_{\mathsf{small}}$. For any b, k where $2kc(b-1) < B$, the protocol Π_{fold} is a reduction of knowledge from $(\mathcal{R}_{\mathsf{hom}}^b)^{2k}$ to $\mathcal{R}_{\mathsf{hom}}^B$.*

Remark 3.2. Π_{fold} *(Figure 3) uses two different sampling sets $\mathcal{C}$, $\mathcal{C}_{\mathsf{small}}$ where $\mathcal{C}$ is for sumcheck and $\mathcal{C}_{\mathsf{small}}$ is for folding. We set $\mathcal{C}$ as in Eq. (31) for two reasons:*

1. *Sumcheck efficiency: By Eq. (31), for every $\mathbf{a}_i \in \mathcal{C}$, $\mathsf{NTT}(\mathbf{a}_i) = (i, \ldots, i)$ where $i \in \mathbb{F}_{q^\tau}$ is the embedding of $\mathbf{a}_i$ in $\mathbb{F}_{q^\tau}$. So $\mathcal{C}$ lets us treat the sumcheck over $\mathcal{R}_q$ as d/τ parallel sumchecks over $\mathbb{F}_{q^\tau}$ that share the same sumcheck challenge vector over $\mathbb{F}_{q^\tau}^{\log m}$, and no NTT transformation is needed when running the sumcheck. Note that the removal of the NTT transform also improves the circuit size of the sumcheck verifier.*

2. *Completeness: As discussed in Remark 3.1 and Section 3.3, Lemma 3.2 still holds when $\mathcal{C}$ is set as in Eq. (31). This is essential for the folding protocol to support small modulus q.*

For the special case $\mathcal{R}_q \cong \mathbb{Z}_q^d$, the proof follows from Lemma 3.6 and Theorem 3.3 below. As discussed in Section 3.3, with a minor modification to the protocol, the proof naturally extends to the case where $\mathcal{R}_q \cong \mathbb{F}_{q^\tau}^{d/\tau}$.

Lemma 3.6. Π_{fold} *satisfies public reducibility and completeness.*

Proof. *Public reducibility:* Given input instances $[\vec{\mathbf{r}}_i, \hat{\mathbf{v}}_i, y_i]_{i=1}^{2k}$ and the transcript that includes the challenge $\vec{\mathbf{r}}_o$, evaluations $[\theta_i]_{i=1}^{2k}$ and folding challenges $[\rho_i]_{i=1}^{2k}$. The algorithm outputs $\mathbb{x}_o := (\vec{\mathbf{r}}_o, \hat{\mathbf{v}}_o, y_o := \sum_{i=1}^{2k} \rho_i \cdot y_i)$ where $\mathsf{NTT}(\hat{\mathbf{v}}_o) = \sum_{i=1}^{2k} \mathsf{RotSum}(\rho_i, \mathsf{NTT}(\theta_i))$ if the verification passes. Otherwise, it outputs $\perp$.

Completeness: Let $(\mathbb{x}, \mathbb{w}) := [\mathbb{x}_i = (\vec{\mathbf{r}}_i, \hat{\mathbf{v}}_i, y_i), \mathbb{w}_i = \vec{\mathbf{f}}_i]_{i=1}^{2k} \leftarrow \mathcal{A}(\mathsf{pp})$ denote adversary $\mathcal{A}$'s chosen input for $\mathcal{R}_1 := (\mathcal{R}_{\mathsf{hom}}^b)^{2k}$, where $\mathsf{pp} := (\mathcal{R}_q, m, B < q/2, \mathcal{L}) \leftarrow \mathsf{Setup}(1^\lambda)$ is the public parameter. WLOG we assume that $(\mathsf{pp}, \mathbb{x}_i, \mathbb{w}_i) \in \mathcal{R}_{\mathsf{hom}}^b$ for all $i \in [2k]$. The protocol $\langle \mathsf{P}(\mathsf{pp}, \mathbb{x}, \mathbb{w}), \mathsf{V}(\mathsf{pp}, \mathbb{x}) \rangle$ proceeds as follows:

1. P and V honestly run the sum-check and P sends the correct evaluations $[\theta_i := \mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_o)]_{i=1}^{2k}$.

2. V outputs $\perp$ and halts if the check at Step 4 fails.

3. Otherwise, let $[\rho_i]_{i=1}^{2k}$ be verifier's last folding challenges. P outputs $\mathbb{w}_o := \vec{\mathbf{f}}_o := \sum_{i=1}^{2k} \rho_i \cdot \vec{\mathbf{f}}_i$ and V outputs $\mathbb{x}_o := (\vec{\mathbf{r}}_o, \hat{\mathbf{v}}_o, y_o)$ where $\vec{\mathbf{r}}_o$ is V's sum-check challenges and $(\hat{\mathbf{v}}_o, y_o)$ are defined such that

$$\mathsf{NTT}(\hat{\mathbf{v}}_o) = \sum_{i=1}^{2k} \mathsf{RotSum}(\rho_i, \mathsf{NTT}(\theta_i)) \,, \qquad y_o := \sum_{i=1}^{2k} \rho_i \cdot y_i \,.$$

We first show that V accepts, i.e., the check at Step 4 passes. This follows by definition of the polynomial g (Eq. (14)) and by definition of P's sent evaluations.

It remains to argue that the protocol output $(\mathbb{x}_o, \mathbb{w}_o)$ satisfies that $(\mathsf{pp}, \mathbb{x}_o, \mathbb{w}_o) \in \mathcal{R}_2 := \mathcal{R}_{\mathsf{hom}}^B$ (Eq. (10)). First, because $(\mathsf{pp}, \mathbb{x}_i, \mathbb{w}_i) \in \mathcal{R}_{\mathsf{hom}}^b$ for all $i \in [2k]$, by Lemma 3.2, it holds that $\mathcal{L}(\vec{\mathbf{f}}_o) = y_o$ and $\mathsf{mle}\left[\hat{\mathbf{f}}_o\right](\vec{\mathbf{r}}_o) = \hat{\mathbf{v}}_o$. Moreover,

$$\|\vec{\mathbf{f}}_o\|_\infty = \|\sum_{i=1}^{2k} \rho_i \cdot \vec{\mathbf{f}}_i\|_\infty \leq \sum_{i=1}^{2k} \|\rho_i \cdot \vec{\mathbf{f}}_i\|_\infty \leq \sum_{i=1}^{2k} c \cdot \|\vec{\mathbf{f}}_i\|_\infty \leq \sum_{i=1}^{2k} c \cdot (b-1) < B \,.$$

The first inequality holds because for any $a, b \in \mathcal{R}_q^m$ where $\|a\|_\infty + \|b\|_\infty < B < q/2$, we have that $\|a+b\|_\infty \leq \|a\|_\infty + \|b\|_\infty$. (The norm $\|a\|_\infty$ for an element a in $\mathcal{R}_q$ is defined in Remark 2.1.) The 2nd inequality holds as $\rho_i \in \mathcal{C}_{\mathsf{small}}$ and $\mathcal{C}_{\mathsf{small}}$ has expansion factor at most c; the 3rd inequality holds because $\|\vec{\mathbf{f}}_i\|_\infty < b$ for all $i \in [2k]$ by the assumption that $(\mathsf{pp}, \mathbb{x}_i; \vec{\mathbf{f}}_i) \in \mathcal{R}_{\mathsf{hom}}^b$; the last inequality holds as $2kc(b-1) < B < q/2$ by the premise of Theorem 3.2. Thus $(\mathsf{pp}, \mathbb{x}_o, \mathbb{w}_o)$ is in $\mathcal{R}_{\mathsf{hom}}^B$ from (10) as required. □

Theorem 3.3. *Let* $\mathsf{pp} := (\mathcal{R}_q \cong \mathbb{Z}_q^d, m, B < q/2, \mathcal{L}) \leftarrow \mathsf{Setup}(1^\lambda)$ *denote the public parameters and let* $\mathcal{C}$, $\mathcal{C}_{\mathsf{small}} \subseteq \mathcal{R}_q$ *be the strong sampling sets defined in Theorem 3.2. Assume that the* $\mathcal{R}_q$*-module homomorphism* $\mathcal{L} : \mathcal{R}_q^m \to \mathcal{Y}$ *is* $2B$*-relaxed binding for challenge space* $\mathcal{C}_{\mathsf{small}}$ *with binding error* ϵ_{bind}*. There exists an extractor* Ext *such that for any expected polynomial time adversary* $(\mathcal{A}, \mathsf{P}^*)$ *with success probability* $\epsilon_{\mathsf{fold}}(\mathcal{A}, \mathsf{P}^*) = 1/\mathsf{poly}(\lambda)$*, the extractor* $\mathsf{Ext}^{\mathcal{A}, \mathsf{P}^*}$ *outputs valid witnesses (for input instances) in relation* $(\mathcal{R}_{\mathsf{hom}}^b)^{2k}$ *with probability at least* $\epsilon_{\mathsf{fold}}(\mathcal{A}, \mathsf{P}^*) - \kappa(\lambda)$ *where*

$$\kappa(\lambda) := \frac{2k}{|\mathcal{C}_{\mathsf{small}}|} + \epsilon_{\mathsf{bind}} + \frac{(2b+1)\log m + 4k}{|\mathcal{C}|} \,.$$

The expected running time of $\mathsf{Ext}^{\mathcal{A}, \mathsf{P}^*}$ *is at most*

$$T_{\mathsf{ext}} := \left(1 + \frac{1}{\epsilon_{\mathsf{fold}}(\mathcal{A}, \mathsf{P}^*) - \frac{2k}{|\mathcal{C}_{\mathsf{small}}|}}\right) \cdot (1 + 2k) = \mathsf{poly}(\lambda) \,.$$

Proof. As noted by Remark 1 of [AF22], it is without loss of generality to assume that $(\mathcal{A}, \mathsf{P}^*)$ are deterministic algorithms. For ease of notation, we assume that P^* outputs $\perp$

when it fails the verification or its output is not a valid witness for the output instance. We can always transform a prover to satisfy this requirement without affecting the success probability. To simplify the notation in the proof we introduce the following symbols:

- set $k^* := 2k$,
- the folding challenge space is denoted $\mathcal{S} := \mathcal{C}_{\mathsf{small}}^{k^*}$,
- the remaining challenge space is denoted $\Psi := \mathcal{C}^{2k^*+2\log m}$.

Let us first give some intuition for the extraction strategy. Adversary $\mathcal{A}$ begins by generating some k^* instances $\mathbb{x}_i = (\vec{\mathbf{r}}_i, \hat{\mathbf{v}}_i, \mathbf{y}_i)$ for $i \in [k^*]$. The prover P^* then takes as input a sequence of random challenges from the verifier. These challenges define a folded statement $\mathbb{x}_o = (\vec{\mathbf{r}}_o, \hat{\mathbf{v}}_o, \mathbf{y}_o)$ and P^* outputs a valid witness $\vec{\mathbf{f}}$ for $\mathbb{x}_o$ with non-negligible probability. Our extractor needs to use such a P^* to output valid witnesses $\vec{\mathbf{f}}_i$ for $\mathbb{x}_i = (\vec{\mathbf{r}}_i, \hat{\mathbf{v}}_i, \mathbf{y}_i)$ for all $i \in [k^*]$. Let us see how to extract $\vec{\mathbf{f}}_1$; the other witnesses are extracted similarly. The high level approach to extract $\vec{\mathbf{f}}_1$ is to sample two related folding challenge vectors

$$c_0 := (\rho_1, \rho_2, \ldots, \rho_{k^*}) \quad \text{and} \quad c_1 := (\rho_1', \rho_2, \ldots, \rho_{k^*}) \quad \text{from } \mathcal{S} := \mathcal{C}_{\mathsf{small}}^{k^*}$$

with $\rho_1 \neq \rho_1'$. Then run P^* once on c_0 and once on c_1. All other random challenges from the verifier are the same on both runs. We show that with non-negligible probability, the prover P^* will return two valid witness $\vec{\mathbf{w}}_0$ and $\vec{\mathbf{w}}_1$. If P^* were honest then

$$\vec{\mathbf{w}}_0 = \rho_1 \vec{\mathbf{f}}_1 + \sum_{i=2}^{k^*} \rho_i \vec{\mathbf{f}}_i \qquad \text{and} \qquad \vec{\mathbf{w}}_1 = \rho_1' \vec{\mathbf{f}}_1 + \sum_{i=2}^{k^*} \rho_i \vec{\mathbf{f}}_i$$

from which it follows that $(\vec{\mathbf{w}}_0 - \vec{\mathbf{w}}_1) = (\rho_1 - \rho_1')\vec{\mathbf{f}}_1$. We can now calculate $\vec{\mathbf{f}}_1$ because $\rho_1 - \rho_1'$ is invertible in $\mathcal{R}_q$ (since $\mathcal{C}_{\mathsf{small}}$ is a strong sampling set). However, P^* can be malicious. Fortunately, our analysis will show that a slightly enhanced strategy will either extract a valid witness $\vec{\mathbf{f}}_1$ for $\mathbb{x}_1$ with non-negligible probability, or break the $2B$-relaxed binding property of $\mathcal{L}$. In particular, in Claim 3.7 below we show that the sumcheck in protocol Π_{fold}, combined with the relaxed binding property of $\mathcal{L}$, ensures that the extracted witnesses $\vec{\mathbf{f}}_i$ have norm at most b and that the evaluations of $\mathsf{mle}\left[\hat{\mathbf{f}}_i\right]$ at $\vec{\mathbf{r}}_i$ are correct.

The Extractor. We describe the complete extractor $\mathsf{Ext}^{\mathcal{A},\mathsf{P}^*}$ in Figure 4. The extractor invokes two sub-procedures. The algorithm $\mathsf{IG}^{\mathcal{A}}$ samples the input instances. The algorithm $\mathsf{SubExt}^{\mathsf{P}^*}(\mathsf{inst}, \psi)$, given randomness $\psi := ([\alpha_i, \mu_i]_{i=1}^{k^*}, \vec{\beta}, \vec{\mathbf{r}}_o)$, tries to recover weak openings to the input instances in inst by rewinding P^* multiple times. We emphasize that each run of $\mathsf{SubExt}^{\mathsf{P}^*}(\mathsf{inst}, \psi)$ and $\mathsf{SubExt}^{\mathsf{P}^*}(\mathsf{inst}, \psi')$ uses fresh internal randomness for challenges $(c_0, c_1, \ldots, c_{k^*})$.

Let $[\mathbb{x}_i := (\vec{\mathbf{r}}_i, \hat{\mathbf{v}}_i, \mathbf{y}_i)]_{i=1}^{k^*}$ denote the input instances. For any list of length-m ring

SubProcedure $\mathsf{IG}^{\mathcal{A}}(1^\lambda)$: // sample as instance

1. $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$
2. Return $\mathsf{inst} := ([\mathbb{x}_i]_{i=1}^{k^*}, \mathsf{state}) \leftarrow \mathcal{A}(\mathsf{pp})$

SubProcedure $\mathsf{SubExt}^{\mathsf{P}^*}(\mathsf{inst}, \psi \in \Psi)$: // attempt to extract a witness for inst

1. $c_0 := (\rho_1, \ldots, \rho_{k^*}) \xleftarrow{\$} \mathcal{S}$
2. $\vec{\mathbf{w}}_0 \leftarrow \mathsf{P}^*(\mathsf{inst}, c_0, \psi)$
3. If $\vec{\mathbf{w}}_0 = \perp$ return $(\mathsf{open}, \mathsf{out}) := (\perp, \perp)$
4. For $i = 1, \ldots, k^*$:
   Do: // loop until a good folding randomness ρ_i' is found
   (a) $\rho_i' \xleftarrow{\$} \mathcal{C}_{\mathsf{small}} \setminus \{\rho_i\}$ without replacement
   (b) $c_i := (\rho_1, \ldots, \rho_{i-1}, \rho_i', \rho_{i+1}, \ldots, \rho_{k^*})$
   (c) $\vec{\mathbf{w}}_i \leftarrow \mathsf{P}^*(\mathsf{inst}, c_i, \psi)$
   Repeat until $\vec{\mathbf{w}}_i \neq \perp$; return $(\perp, \perp)$ if all ρ_i' have been tried
5. For $i = 1, \ldots, k^*$: set $\mathsf{open}_i := (\rho_i - \rho_i', \vec{\mathbf{w}}_0 - \vec{\mathbf{w}}_i)$ and $\vec{\mathbf{f}}_i := (\rho_i - \rho_i')^{-1}(\vec{\mathbf{w}}_0 - \vec{\mathbf{w}}_i)$
6. Return $\mathsf{open} := [\mathsf{open}_i]_{i=1}^{k^*}$ and $\mathsf{out} := [\vec{\mathbf{f}}_i]_{i=1}^{k^*}$

The extractor $\mathsf{Ext}^{\mathcal{A}, \mathsf{P}^*}(1^\lambda)$:

1. $\mathsf{inst} \leftarrow \mathsf{IG}^{\mathcal{A}}(1^\lambda)$
2. Do: // loop until a good tuple of challenges ψ is found
   (a) $\psi := ([\alpha_i, \mu_i]_{i=1}^{k^*}, \vec{\beta}, \vec{\mathbf{r}}_o) \xleftarrow{\$} \Psi$ // sample fresh challenges
   (b) $(\mathsf{open}_1, \mathsf{out}_1) \leftarrow \mathsf{SubExt}^{\mathsf{P}^*}(\mathsf{inst}, \psi)$ // attempt to extract a witness using ψ
   Repeat until $\mathsf{out}_1 \neq \perp$
3. $\psi' := ([\alpha_i', \mu_i']_{i=1}^{k^*}, \vec{\beta}', \vec{\mathbf{r}}_o') \xleftarrow{\$} \Psi$ // sample fresh challenges
4. $(\mathsf{open}_2, \mathsf{out}_2) \leftarrow \mathsf{SubExt}^{\mathsf{P}^*}(\mathsf{inst}, \psi')$, abort if $\mathsf{out}_2 = \perp$ or $\mathsf{out}_1 \neq \mathsf{out}_2$
5. Parse $\mathsf{out}_1 = \mathsf{out}_2 = [\vec{\mathbf{f}}_i]_{i=1}^{k^*}$
6. Abort if $\Phi_{\mathsf{valid}}([\mathbb{x}_i, \vec{\mathbf{f}}_i]_{i=1}^{k^*}) = 0$, where $[\mathbb{x}_i]_{i=1}^{k^*}$ are the input instances in inst
7. Return $[\vec{\mathbf{f}}_i]_{i=1}^{k^*}$

Figure 4: The extractor for Π_{fold} using the validity predicate Φ_{valid} from (18).

vectors $[\vec{\mathbf{f}}_i]_{i=1}^{k^*} \in (\mathcal{R}_q^m)^{k^*}$, and $i \in [k^*]$, define the *multilinear* polynomial

$$p_i(\vec{\mathbf{x}}) := \sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log m}} eq(\vec{\mathbf{x}}, \vec{\mathbf{b}}) \cdot \prod_{j=1-b}^{b-1} \left(\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{b}}) - j \right) . \tag{17}$$

Note that $p_i(\vec{\mathbf{x}}) = 0$ implies that $\prod_{j=1-b}^{b-1} \left(\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{b}}) - j \right) = 0$ for every $\vec{\mathbf{b}} \in \{0,1\}^{\log m}$, which in turn implies $\|\vec{\mathbf{f}}_i\|_\infty < b$ from the discussion in Section 3.1. The predicate $\Phi_{\mathsf{valid}}([\mathbb{x}_i, \vec{\mathbf{f}}_i]_{i=1}^{k^*})$ is true if and only if

$$\forall i \in [k^*] \ : \ \left(\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_i) = \hat{\mathbf{v}}_i \right) \ \wedge \ (\mathcal{L}(\vec{\mathbf{f}}_i) = \mathbf{y}_i) \ \wedge \ (p_i(\vec{\mathbf{x}}) = 0) . \tag{18}$$

Hence, $\Phi_{\mathsf{valid}}([\mathbb{x}_i, \vec{\mathbf{f}}_i]_{i=1}^{k^*}) = 1$ if and only if $[\vec{\mathbf{f}}_i]_{i=1}^{k^*}$ are valid witnesses for $[\mathbb{x}_i]_{i=1}^{k^*}$, that is, $[\mathbb{x}_i, \vec{\mathbf{f}}_i]_{i=1}^{k^*} \in (\mathcal{R}_{\mathsf{hom}}^b)^{k^*}$. Thus, given the check at Step 6, the extractor always outputs a valid witness if it does not abort.

Running time. We adapt the proof of Lemma 7.1 in [FMN23] to analyze the expected running time of the extractor. We first analyze the expected running time of each execution of $\mathsf{SubExt}^{\mathsf{P}^*}$. Fix any input (inst, ψ), we denote by $C_0 := (\Sigma_1, \ldots, \Sigma_{k^*})$ the random variable for the folding challenges $c_0 := (\rho_1, \ldots, \rho_{k^*})$. We define event $\Gamma := (\mathsf{P}^*(\mathsf{inst}, C_0, \psi) \neq \perp)$.

Let T be the number of calls to P^* in $\mathsf{SubExt}^{\mathsf{P}^*}(\mathsf{inst}, \psi)$. For $i \in [k^*]$, let T_i be the number of calls to P^* made during the i-th iteration of the loop. We have $\mathbb{E}[T] = 1 + \sum_{i=1}^{k^*} \mathbb{E}[T_i]$ by linearity of expectation.

Define the random variable

$$X_i := \left| \{ x \in \mathcal{C}_{\mathsf{small}} : \mathsf{P}^*(\mathsf{inst}, C(x), \psi) \neq \perp \} \right| \tag{19}$$

where $C(x) := (\Sigma_1, \ldots, \Sigma_{i-1}, x, \Sigma_{i+1}, \ldots, \Sigma_{k^*})$. Let $N := |\mathcal{C}_{\mathsf{small}}|$, we have that

$$\mathbb{E}[T_i] = \sum_{j=0}^{N} \mathbb{E}[T_i \mid X_i = j] \cdot \Pr[X_i = j] .$$

Also note that $T_i = 0$ when $\Gamma = 0$, thus for any $j \geq 0$, we have that

$$\mathbb{E}[T_i | X_i = j] = \Pr[\Gamma = 1 \mid X_i = j] \cdot \mathbb{E}[T_i \mid (\Gamma = 1) \wedge X_i = j]$$

where $\Pr[\Gamma = 1 \mid X_i = j] = j/N$ and $\mathbb{E}[T_i \mid (\Gamma = 1) \wedge X_i = j]$ is the expectation of a negative hypergeometric distribution, that is, challenges ρ_i' are drawn without replacement from a set of size $N - 1$ that contains $j - 1$ correct responses. Hence, $\mathbb{E}[T_i \mid (\Gamma = 1) \wedge X_i = j] \leq N/j$

and therefore $\mathbb{E}[T_i|X_i = j] \leq j/N \cdot (N/j) = 1$. In sum, for all $i \in [k^*]$, we have that $\mathbb{E}[T_i] \leq \sum_{j=0}^{N} 1 \cdot \Pr[X_i = j] = 1$, and therefore

$$\mathbb{E}[T] = 1 + \sum_{i=1}^{k^*} \mathbb{E}[T_i] \leq 1 + k^* .$$

Next, we analyze the success probability of each independent run of $\mathsf{SubExt}^{\mathsf{P}^*}$. Since $\mathcal{A}$, P^* are deterministic, we have that the event $\Gamma := (\mathsf{P}^*(\mathsf{inst}, C_0, \psi) \neq \perp)$ happens with probability $\epsilon_{\mathsf{fold}}(\mathcal{A}, \mathsf{P}^*)$ over the randomness of ψ and C_0. Define E as the event that a fresh call of $\mathsf{SubExt}^{\mathsf{P}^*}(\mathsf{inst}, \psi)$ does not return $\perp$. We have that

$$\begin{aligned}
\Pr[E] &= \Pr[\Gamma = 1 \wedge (\wedge_{i=1}^{k^*} X_i \geq 2)] \qquad (20)\\
&= \Pr[\Gamma = 1] - \Pr[\Gamma = 1 \wedge (\vee_{i=1}^{k^*} X_i = 1)] \\
&\geq \Pr[\Gamma = 1] - \sum_{i=1}^{k^*} \Pr[\Gamma = 1 \wedge X_i = 1] \\
&\geq \Pr[\Gamma = 1] - \frac{k^*}{|\mathcal{C}_{\mathsf{small}}|} \\
&\geq \epsilon_{\mathsf{fold}}(\mathcal{A}, \mathsf{P}^*) - \frac{k^*}{|\mathcal{C}_{\mathsf{small}}|} .
\end{aligned}$$

In sum, the expected number of calls to P^* in the extractor is at most $(\mathbb{E}[T]/\Pr[E]) + E[T] = \left(1 + 1/\left(\epsilon_{\mathsf{fold}}(\mathcal{A}, \mathsf{P}^*) - \frac{2k}{|\mathcal{C}_{\mathsf{small}}|}\right)\right) \cdot (1 + 2k) = \mathsf{poly}(\lambda)$.

Success probability. Towards analyzing the extractor's success probability, we define the following events.

- E_{ext}: the extractor recovers the witnesses $\mathsf{out}_1, \mathsf{out}_2 \neq \perp$, and, $\mathsf{out}_1 = \mathsf{out}_2$.
- E_{valid}: E_{ext} occurs and the extracted witness is valid, i.e., $\Phi_{\mathsf{valid}}([\mathbb{x}_i, \vec{\mathbf{f}}_i]_{i=1}^{k^*}) = 1$ for the input instances $[\mathbb{x}_i]_{i=1}^{k^*}$ and the interpolated witness $[\vec{\mathbf{f}}_i]_{i=1}^{k^*}$.

$E_{\mathsf{ext}} \wedge E_{\mathsf{valid}}$ implies that the extractor returns a valid witness for the input instances. Moreover,

$$\Pr[E_{\mathsf{ext}} \wedge E_{\mathsf{valid}}] = \Pr[E_{\mathsf{ext}}] - \Pr[E_{\mathsf{ext}} \wedge \overline{E_{\mathsf{valid}}}] ,$$

thus it suffices to lower-bound $\Pr[E_{\mathsf{ext}}]$ and upper-bound $\Pr[E_{\mathsf{ext}} \wedge \overline{E_{\mathsf{valid}}}]$.

Claim 1. $\Pr[E_{\mathsf{ext}}] \geq \epsilon_{\mathsf{fold}}(\mathcal{A}, \mathsf{P}^*) - \frac{k^*}{|\mathcal{C}_{\mathsf{small}}|} - \epsilon_{\mathsf{bind}}$.

Proof. We define E'_{ext} as the event that the last call of $\mathsf{SubExt}^{\mathsf{P}^*}$, on input (inst, ψ'), does not return $\perp$. By Eq. (20), we have that

$$\Pr[E'_{\mathsf{ext}}] \geq \epsilon_{\mathsf{fold}}(\mathcal{A}, \mathsf{P}^*) - \frac{k^*}{|\mathcal{C}_{\mathsf{small}}|}$$

because (i) $\mathcal{A}$ is deterministic, and (ii) ψ' and the randomness in $\mathsf{SubExt}^{\mathsf{P}^*}(\mathsf{inst}, \psi')$ are freshly sampled.

Suppose E'_{ext} occurs. Let $(\mathsf{open}_1, \mathsf{out}_1)$, $(\mathsf{open}_2, \mathsf{out}_2)$ be the output of $\mathsf{SubExt}^{\mathsf{P}^*}(\mathsf{inst}, \psi)$ and $\mathsf{SubExt}^{\mathsf{P}^*}(\mathsf{inst}, \psi')$ respectively. We next show that $\mathsf{out}_1 = \mathsf{out}_2$ (and thus E_{ext} holds) with high probability. For every $i \in [k^*]$, let $\mathsf{open}_{1,i} := (\Delta, \vec{\mathbf{w}})$ and $\mathsf{open}_{2,i} := (\Delta', \vec{\mathbf{w}}')$. Recall the assumption that P^* produces an output only when the witness is valid for the folded instance. Therefore, the *interpolated* witnesses $\vec{\mathbf{w}}$, $\vec{\mathbf{w}}'$ output by $\mathsf{SubExt}^{\mathsf{P}^*}$ must satisfy that $\mathcal{L}(\vec{\mathbf{w}}) = \Delta y_i$ and $\mathcal{L}(\vec{\mathbf{w}}') = \Delta' y_i$. Suppose for contradiction that $\mathsf{out}_{1,i} = \Delta^{-1}\vec{\mathbf{w}} \neq (\Delta')^{-1}\vec{\mathbf{w}}' = \mathsf{out}_{2,i}$, then $\Delta'\vec{\mathbf{w}} \neq \Delta\vec{\mathbf{w}}'$, and $(\mathsf{open}_{1,i}, \mathsf{open}_{2,i})$ is a pair of *distinct* $2B$-weak openings that breaks the $2B$-relaxed binding property of $\mathcal{L}$. Specifically, $\mathcal{L}(\vec{\mathbf{w}}) = \Delta y_i$ and $\mathcal{L}(\vec{\mathbf{w}}') = \Delta' y_i$; $\|\vec{\mathbf{w}}\|_\infty, \|\vec{\mathbf{w}}'\|_\infty < 2B$ because $\vec{\mathbf{w}}$ and $\vec{\mathbf{w}}'$ are the subtractions of two vectors with norm less than B; and Δ, Δ' are non-zero differences in the set $\mathcal{C}_{\mathsf{small}}$. Moreover, the extractor is an expected polynomial time algorithm. Thus, by the relaxed binding property of $\mathcal{L}$, we have that

$$\Pr\left[E'_{\mathsf{ext}} \wedge (\mathsf{out}_1 \neq \mathsf{out}_2)\right] \leq \epsilon_{\mathsf{bind}}\,.$$

Therefore, we have that

$$\begin{aligned}
\Pr[E_{\mathsf{ext}}] &= \Pr[E'_{\mathsf{ext}}] - \Pr\left[E'_{\mathsf{ext}} \wedge (\mathsf{out}_1 \neq \mathsf{out}_2)\right] \\
&\geq \Pr[E'_{\mathsf{ext}}] - \epsilon_{\mathsf{bind}} \\
&\geq \epsilon_{\mathsf{fold}}(\mathcal{A}, \mathsf{P}^*) - \frac{k^*}{|\mathcal{C}_{\mathsf{small}}|} - \epsilon_{\mathsf{bind}}\,,
\end{aligned}$$

which completes the proof. □

Next, we upper-bound the probability of $E_{\mathsf{ext}} \wedge \overline{E_{\mathsf{valid}}}$ – the event that extractor recovers witnesses $\mathsf{out}_1 = \mathsf{out}_2$ but the extracted witness is invalid.

We first reduce $\Pr[E_{\mathsf{ext}} \wedge \overline{E_{\mathsf{valid}}}]$ to the probability of a different event that is easier to analyze. Let inst denote $\mathcal{A}$'s output that includes the input instances $[\mathbb{x}_i := (\vec{\mathbf{r}}_i, \hat{\mathbf{v}}_i, y_i)]_{i=1}^{k^*}$. Let $\psi' := ([\alpha'_i, \mu'_i]_{i=1}^{k^*}, \vec{\beta}', \vec{\mathbf{r}}'_o)$ be the last sampled randomness. We consider the sub-extraction call $\mathsf{SubExt}^{\mathsf{P}^*}(\mathsf{inst}, \psi')$: Let $c_0 := (\rho_1, \ldots, \rho_{k^*})$ denote the initial folding challenge, and let $[\theta'_i]_{i=1}^{k^*}$ denote the claimed evaluations in the transcript of $\mathsf{P}^*(\mathsf{inst}, c_0, \psi')$. Let $[\vec{\mathbf{f}}_i]_{i=1}^{k^*}$ denote the interpolated vectors when E_{ext} occurs, and let $[p_i(\vec{\mathbf{x}})]_{i=1}^{k^*}$ be the corresponding

polynomials specified in Eq. (17). Define events E_{hom}, E_{eval}, E_{bad} as

$$E_{\mathsf{hom}} := E_{\mathsf{ext}} \wedge (\forall i \in [k^*] \ : \ \mathcal{L}(\vec{\mathbf{f}}_i) = y_i)\,, \tag{21}$$

$$E_{\mathsf{eval}} := E_{\mathsf{ext}} \wedge (\forall i \in [k^*] \ : \ \mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}'_o) = \theta'_i)\,, \tag{22}$$

$$E_{\mathsf{bad}} := E_{\mathsf{ext}} \wedge (\exists i \in [k^*] \ : \ \left(\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_i) \neq \hat{\mathbf{v}}_i\right) \vee (p_i(\vec{\mathbf{x}}) \neq 0))\,. \tag{23}$$

Informally, E_{eval} implies that the reduced evaluation claim holds after running the sumcheck; E_{bad} implies that the extracted witness is invalid.

Claim 2. $\Pr[E_{\mathsf{ext}} \wedge \overline{E_{\mathsf{valid}}}] = \Pr[E_{\mathsf{eval}} \wedge E_{\mathsf{bad}}]$.

Proof. Assume that $E_{\mathsf{ext}} \implies (E_{\mathsf{hom}} \wedge E_{\mathsf{eval}})$ holds (which we will prove later), we argue that $E_{\mathsf{ext}} \wedge \overline{E_{\mathsf{valid}}}$ occurs if and only if $E_{\mathsf{eval}} \wedge E_{\mathsf{bad}}$ occurs.

We recall the definitions of E_{ext}, E_{valid}, E_{hom} (Eq. (21)), E_{eval} (Eq. (22)) and E_{bad} (Eq. (23)). Informally, E_{ext} means that the witnesses out_1, $\mathsf{out}_2 \neq \perp$ recovered by the extractor are the same. E_{valid} means that the recovered witness is valid. E_{hom} means that the interpolated witness satisfies the homomorphism relation, i.e., $\mathcal{L}(\vec{\mathbf{f}}_i) = y_i$ for every $i \in [k^*]$. E_{eval} means that the reduced evaluation checks (after sumcheck) pass for the interpolated witness, i.e., $\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}'_o) = \theta'_i$ for every $i \in [k^*]$. Finally, E_{bad} means that the input evaluation claims or the norm constraint do not hold for the interpolated witness.

Suppose $E_{\mathsf{ext}} \wedge \overline{E_{\mathsf{valid}}}$ occurs. By the assumption that $E_{\mathsf{ext}} \implies (E_{\mathsf{hom}} \wedge E_{\mathsf{eval}})$ holds, E_{eval} and E_{hom} must also occur. Moreover, if E_{hom} occurs while E_{valid} does not, it must be the case that E_{bad} occurs. Therefore, $E_{\mathsf{eval}} \wedge E_{\mathsf{bad}}$ will occur.

Conversely, suppose $E_{\mathsf{eval}} \wedge E_{\mathsf{bad}}$ occurs. Then E_{ext} certainly occurs. However, E_{valid} cannot occur because E_{bad} occurs, and thus $E_{\mathsf{ext}} \wedge \overline{E_{\mathsf{valid}}}$ occurs.

Now it suffices to show that $E_{\mathsf{ext}} \implies (E_{\mathsf{hom}} \wedge E_{\mathsf{eval}})$. Suppose E_{ext} occurs. Let $[c_i, \vec{\mathbf{w}}_i]_{i=0}^{k^*}$ denote the tuples collected in $\mathsf{SubExt}^{\mathsf{P}^*}(\mathsf{inst}, \psi')$ where $c_0 := (\rho_1, \ldots, \rho_{k^*})$ and

$$c_i := (\rho_1, \ldots, \rho_{i-1}, \rho'_i, \rho_{i+1}, \ldots, \rho_{k^*})\,.$$

For every $i \in [k^*]$, since $\vec{\mathbf{w}}_0, \vec{\mathbf{w}}_i \neq \perp$ are valid witnesses, we have that

$$\begin{aligned}
\mathcal{L}(\vec{\mathbf{w}}_0) &= \rho_1 y_1 + \cdots + \rho_{i-1} y_{i-1} + \rho_i y_i + \rho_{i+1} y_{i+1} + \cdots + \rho_{k^*} y_{k^*}\,, \\
\mathcal{L}(\vec{\mathbf{w}}_i) &= \rho_1 y_1 + \cdots + \rho_{i-1} y_{i-1} + \rho'_i y_i + \rho_{i+1} y_{i+1} + \cdots + \rho_{k^*} y_{k^*}\,,
\end{aligned}$$

thus

$$\begin{aligned}
\mathcal{L}(\vec{\mathbf{f}}_i) &= \mathcal{L}((\vec{\mathbf{w}}_0 - \vec{\mathbf{w}}_i) \cdot (\rho_i - \rho'_i)^{-1}) \\
&= (\rho_i - \rho'_i)^{-1} \cdot \mathcal{L}(\vec{\mathbf{w}}_0 - \vec{\mathbf{w}}_i) \\
&= (\rho_i - \rho'_i)^{-1} \cdot (\mathcal{L}(\vec{\mathbf{w}}_0) - \mathcal{L}(\vec{\mathbf{w}}_i)) \\
&= (\rho_i - \rho'_i)^{-1} \cdot (\rho_i - \rho'_i) \cdot y_i = y_i\,,
\end{aligned}$$

which implies that E_{hom} occurs.

Similarly, since the prover P^* needs to output $[\theta'_j]_{j=1}^{k^*}$ before receiving the folding challenges, the claimed evaluations $\{\theta'_j\}$ in the executions of $\mathsf{P}^*(\mathsf{inst}, c_0, \psi')$ and $\mathsf{P}^*(\mathsf{inst}, c_i, \psi')$ are the same. Define

$$\hat{\mathbf{v}}_o^{(0)} := \mathsf{mle}\left[\hat{\mathbf{w}}_0\right](\vec{\mathbf{r}}'_o), \qquad \hat{\mathbf{v}}_o^{(i)} := \mathsf{mle}\left[\hat{\mathbf{w}}_i\right](\vec{\mathbf{r}}'_o).$$

Note that

$$\mathsf{NTT}(\hat{\mathbf{v}}_o^{(0)}) := \mathsf{mle}\left[\mathsf{Coef}(\vec{\mathbf{w}}_0)\right](\vec{\mathbf{r}}'_o), \qquad \mathsf{NTT}(\hat{\mathbf{v}}_o^{(i)}) := \mathsf{mle}\left[\mathsf{Coef}(\vec{\mathbf{w}}_i)\right](\vec{\mathbf{r}}'_o).$$

Since P^* outputs valid witnesses for the output instances, we have that

$$\begin{aligned}
\mathsf{NTT}(\hat{\mathbf{v}}_o^{(0)}) &= \sum_{j=1}^{k^*} \mathsf{RotSum}(\rho_j, \mathsf{NTT}(\theta'_j)), \\
\mathsf{NTT}(\hat{\mathbf{v}}_o^{(i)}) &= \sum_{j\in[k^*], j\neq i} \mathsf{RotSum}(\rho_j, \mathsf{NTT}(\theta'_j)) + \mathsf{RotSum}(\rho'_i, \mathsf{NTT}(\theta'_i)),
\end{aligned}$$

which implies that

$$\mathsf{NTT}(\hat{\mathbf{v}}_o^{(0)} - \hat{\mathbf{v}}_o^{(i)}) = \mathsf{RotSum}(\rho_i, \mathsf{NTT}(\theta'_i)) - \mathsf{RotSum}(\rho'_i, \mathsf{NTT}(\theta'_i)). \tag{24}$$

Let Δ_o, $\bar{\theta}'_i$ be the values such that $\mathsf{Coef}(\Delta_o) = \mathsf{NTT}(\hat{\mathbf{v}}_o^{(0)} - \hat{\mathbf{v}}_o^{(i)})$ and $\mathsf{Coef}(\bar{\theta}'_i) = \mathsf{NTT}(\theta'_i)$. By Lemma 2.1, we have that

$$\mathsf{RotSum}(\rho_i, \mathsf{NTT}(\theta'_i)) = \mathsf{Coef}(\rho_i \cdot \bar{\theta}'_i), \qquad \mathsf{RotSum}(\rho'_i, \mathsf{NTT}(\theta'_i)) = \mathsf{Coef}(\rho'_i \cdot \bar{\theta}'_i).$$

Then by Eq. (24), we have that $\mathsf{Coef}(\Delta_o) = \mathsf{Coef}((\rho_i - \rho'_i) \cdot \bar{\theta}'_i)$ and thus $\mathsf{Coef}((\rho_i - \rho'_i)^{-1} \cdot \Delta_o) = \mathsf{Coef}(\bar{\theta}'_i)$. By definition of $\vec{\mathbf{f}}_i$, we have that

$$\begin{aligned}
\mathsf{mle}\left[\mathsf{Coef}(\vec{\mathbf{f}}_i)\right](\vec{\mathbf{r}}'_o) &= \mathsf{mle}\left[\mathsf{Coef}((\vec{\mathbf{w}}_0 - \vec{\mathbf{w}}_i) \cdot (\rho_i - \rho'_i)^{-1})\right](\vec{\mathbf{r}}'_o) \\
&= (\rho_i - \rho'_i)^{-1} \cdot \mathsf{mle}\left[\mathsf{Coef}(\vec{\mathbf{w}}_0 - \vec{\mathbf{w}}_i)\right](\vec{\mathbf{r}}'_o) \\
&= (\rho_i - \rho'_i)^{-1} \cdot (\mathsf{mle}\left[\mathsf{Coef}(\vec{\mathbf{w}}_0)\right](\vec{\mathbf{r}}'_o) - \mathsf{mle}\left[\mathsf{Coef}(\vec{\mathbf{w}}_i)\right](\vec{\mathbf{r}}'_o)) \\
&= (\rho_i - \rho'_i)^{-1} \cdot (\mathsf{NTT}(\hat{\mathbf{v}}_o^{(0)}) - \mathsf{NTT}(\hat{\mathbf{v}}_o^{(i)})) \\
&= (\rho_i - \rho'_i)^{-1} \cdot \mathsf{NTT}(\hat{\mathbf{v}}_o^{(0)} - \hat{\mathbf{v}}_o^{(i)}) \\
&= (\rho_i - \rho'_i)^{-1} \cdot \mathsf{Coef}(\Delta_o) \\
&= \mathsf{Coef}((\rho_i - \rho'_i)^{-1} \cdot \Delta_o) \\
&= \mathsf{Coef}(\bar{\theta}'_i) = \mathsf{NTT}(\theta'_i).
\end{aligned}$$

By Lemma A.1 and because $\mathsf{NTT}(\hat{\mathbf{f}}_i) = \mathsf{Coef}(\vec{\mathbf{f}}_i)$, this implies that $\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}'_o) = \theta'_i$. Therefore, E_{eval} occurs, which finishes the proof. □

Therefore, to analyze $\Pr[E_{\mathsf{ext}} \wedge \overline{E_{\mathsf{valid}}}]$, it suffices to analyze the probability of $E_{\mathsf{eval}} \wedge E_{\mathsf{bad}}$.

Lemma 3.7. $\Pr[E_{\mathsf{eval}} \wedge E_{\mathsf{bad}}] \leq \frac{(2b+1)\log m + 2k^*}{|\mathcal{C}|}$.

Proof. Let $[\varkappa_i := (\vec{\mathbf{r}}_i, \hat{\mathbf{v}}_i, \mathbf{y}_i)]_{i=1}^{k^*}$ and $[\vec{\mathbf{f}}_i]_{i=1}^{k^*}$ denote the input instances and the extracted witness vectors, respectively. For every $i \in [k^*]$, let $p_i(\vec{\mathbf{x}})$ denote the multilinear polynomial specified in Eq. (17), with respect to $\vec{\mathbf{f}}_i$. Let $\psi' := ([\alpha'_i, \mu'_i]_{i=1}^{k^*}, \vec{\beta}', \vec{\mathbf{r}}'_o)$ be the input randomness used in the last call $\mathsf{SubExt}^{\mathsf{P}^*}(\mathsf{inst}, \psi')$, namely on line 4 of the extractor. Define polynomial h as

$$h([X_i, Y_i]_{i=1}^{k^*}) := \sum_{i=1}^{k^*} \left(\hat{\mathbf{v}}_i - \mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_i)\right) \cdot X_i + \sum_{i=1}^{k^*} p_i(\vec{\beta}') \cdot Y_i \,. \tag{25}$$

We define the following events:

$$E_1 := E_{\mathsf{ext}} \wedge (\exists i \in [k^*] \;:\; p_i(\vec{\mathbf{x}}) \neq 0) \wedge \left(p_i(\vec{\beta}') = 0 \forall i \in [k^*]\right)$$
$$E_2 := E_{\mathsf{ext}} \wedge (h([\alpha'_i, \mu'_i]_{i=1}^{k^*}) = 0) \wedge \left(\exists i \in [k^*] \;:\; \left(\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_i) \neq \hat{\mathbf{v}}_i\right) \vee (p_i(\vec{\beta}') \neq 0)\right)$$
$$E_3 := E_{\mathsf{eval}} \wedge (h([\alpha'_i, \mu'_i]_{i=1}^{k^*}) \neq 0)$$

Intuitively, we "split" the event $E_{\mathsf{eval}} \wedge E_{\mathsf{bad}}$ into three parts E_1, E_2, E_3, so that we can reduce $\Pr[E_{\mathsf{eval}} \wedge E_{\mathsf{bad}}]$ to the probability of breaking sumcheck soundness. In particular, E_3 implies that the reduced evaluation claim holds (i.e., E_{eval} holds), while the sumcheck claim is false (i.e., $h([\alpha'_i, \mu'_i]_{i=1}^{k^*}) \neq 0$).

We first show that $(E_{\mathsf{eval}} \wedge E_{\mathsf{bad}}) \implies (E_1 \vee E_2 \vee E_3)$. Note that $E_{\mathsf{eval}} \wedge E_{\mathsf{bad}}$ implies that E_{ext} occurs and $[\vec{\mathbf{f}}_i]_{i=1}^{k^*}$ are well-defined. Moreover, if E_1 doesn't occur, then $\overline{E_1} \wedge E_{\mathsf{bad}}$ occurs, that is,

$$\exists i \in [k^*] \;:\; \left(\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_i) \neq \hat{\mathbf{v}}_i\right) \vee (p_i(\vec{\beta}') \neq 0)\,,$$

which implies that $E_2 \vee E_3$ occurs. Thus, $(E_{\mathsf{eval}} \wedge E_{\mathsf{bad}}) \implies (E_1 \vee E_2 \vee E_3)$ and

$$\Pr[E_{\mathsf{eval}} \wedge E_{\mathsf{bad}}] \leq \Pr[E_1 \vee E_2 \vee E_3] \leq \Pr[E_1] + \Pr[E_2] + \Pr[E_3]\,. \tag{26}$$

Now, it suffices to bound the probabilities $\Pr[E_1]$, $\Pr[E_2]$, $\Pr[E_3]$.

Claim 3. $\Pr[E_1] \leq \log m / |\mathcal{C}|$.

Proof. Consider a mental experiment Exp_1 that simulates $\mathsf{Ext}^{\mathcal{A},\mathsf{P}^*}$ until Step 3. Exp_1 outputs 1 if and only if

$$(\exists i \in [k^*] \;:\; p_i(\vec{\mathbf{x}}) \neq 0) \wedge \left(p_i(\vec{\beta}') = 0 \forall i \in [k^*]\right)$$

where $[p_i(\vec{\mathbf{x}})]_{i=1}^{k^*}$ are derived from $[\vec{\mathbf{f}}_i]_{i=1}^{k^*}$ output by $\mathsf{SubExt}^{\mathsf{P}^*}(\mathsf{inst}, \psi)$. It is clear that $\Pr[\mathsf{Exp}_1 = 1] \geq \Pr[E_1]$. Moreover, let $i^* \in [k^*]$ be the first index such that $p_{i^*}(\vec{\mathbf{x}}) \neq 0$. Since

$[\vec{\mathbf{f}}_i]_{i=1}^{k^*}$ in Exp_1 is independent of ψ', by the Generalized Schwartz Lemma (Lemma 2.4), the probability that $p_{i^*}(\vec{\beta}') = 0$ is at most $\frac{\log m}{|\mathcal{C}|}$ (over the randomness $\vec{\beta}'$). Thus $\Pr[E_1] \leq \Pr[\mathsf{Exp}_1 = 1] \leq \frac{\log m}{|\mathcal{C}|}$. □

Claim 4. $\Pr[E_2] \leq 2k^*/|\mathcal{C}|$.

Proof. Consider a mental experiment Exp_2 that that simulates $\mathsf{Ext}^{\mathcal{A},\mathsf{P}^*}$ until Step 3. Exp_2 outputs 1 if and only if

$$(h([\alpha'_i, \mu'_i]_{i=1}^{k^*}) = 0) \wedge \left(\exists i \in [k^*] \ : \ \left(\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_i) \neq \hat{\mathbf{v}}_i\right) \vee (p_i(\vec{\beta}') \neq 0)\right)$$

where h is specified in Eq. (25), and $[p_i(\vec{\mathbf{x}})]_{i=1}^{k^*}$ are defined given the witness $[\vec{\mathbf{f}}_i]_{i=1}^{k^*}$ derived from $\mathsf{SubExt}^{\mathsf{P}^*}(\mathsf{inst}, \psi)$. It is clear that $\Pr[\mathsf{Exp}_2 = 1] \geq \Pr[E_2]$. Moreover, since $[\vec{\mathbf{f}}_i]_{i=1}^{k^*}$ in Exp_2 is independent of ψ', by the Generalized Schwartz Lemma (Lemma 2.4), the probability that $h([\alpha'_i, \mu'_i]_{i=1}^{k^*}) = 0$ is at most $\frac{2k^*}{|\mathcal{C}|}$ (over the choice of $[\alpha'_i, \mu'_i]_{i=1}^{k^*}$). Thus $\Pr[E_2] \leq \Pr[\mathsf{Exp}_2 = 1] \leq \frac{2k^*}{|\mathcal{C}|}$. □

Claim 5. $\Pr[E_3] \leq 2b \log m/|\mathcal{C}|$.

Proof. Consider a mental experiment Exp_3 that simulates $\mathsf{Ext}^{\mathcal{A},\mathsf{P}^*}$ until Step 3. Additionally, it simulates $\mathsf{P}^*(\mathsf{inst}, \perp, \psi')$ to obtain claimed evaluations $\{\theta'_j\}$ in the partial transcript without providing folding challenges. Exp_3 outputs 1 if and only if the verification at Step 4 passes and

$$(h([\alpha'_i, \mu'_i]_{i=1}^{k^*}) \neq 0) \wedge (\forall i \in [k^*] \ : \ \mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}'_o) = \theta'_i)\,,$$

where h is specified in Eq. (25) and $[\vec{\mathbf{f}}_i]_{i=1}^{k^*}$ are obtained from $\mathsf{SubExt}^{\mathsf{P}^*}(\mathsf{inst}, \psi)$.

We show that $\Pr[\mathsf{Exp}_3 = 1] \geq \Pr[E_3]$. Recall that E_3 implies that E_{ext} occurs, which implies that $[\vec{\mathbf{f}}_i]_{i=1}^{k^*}$ computed from $\mathsf{SubExt}^{\mathsf{P}^*}(\mathsf{inst}, \psi)$ is **identical** to that from $\mathsf{SubExt}^{\mathsf{P}^*}(\mathsf{inst}, \psi')$. Moreover, E_3 implies that E_{eval} occurs, i.e., the evaluation check in the execution $\mathsf{SubExt}^{\mathsf{P}^*}(\mathsf{inst}, \psi')$ passes. Since the witness $[\vec{\mathbf{f}}_i]_{i=1}^{k^*}$ extracted from $\mathsf{SubExt}^{\mathsf{P}^*}(\mathsf{inst}, \psi')$ is the same as that from $\mathsf{SubExt}^{\mathsf{P}^*}(\mathsf{inst}, \psi)$, we have that $\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}'_o) = \theta'_i$ for all $i \in [k^*]$. Therefore, with the same randomness, if E_3 happens, then Exp_3 will output 1. Thus $\Pr[\mathsf{Exp}_3 = 1] \geq \Pr[E_3]$.

Now, it suffices to bound $\Pr[\mathsf{Exp}_3 = 1]$. For every $i \in [k^*]$, we define $p_i(\vec{\mathbf{x}})$ from $\vec{\mathbf{f}}_i$ according to Eq. (17). We can rewrite $p_i(\vec{\beta}')$ as

$$p_i(\vec{\beta}') = \sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log m}} eq(\vec{\beta}', \vec{\mathbf{b}}) \cdot \prod_{j=1-b}^{b-1} \left(\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{b}}) - j\right) = \sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log m}} g_{2,i}(\vec{\mathbf{b}})\,, \tag{27}$$

where $g_{2,i}$ is specified in Eq. (16). Similarly, we can rewrite $\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_i)$ as

$$\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_i) = \sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log m}} eq(\vec{\mathbf{r}}_i, \vec{\mathbf{b}}) \cdot \mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{b}}) = \sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log m}} g_{1,i}(\vec{\mathbf{b}})\,, \tag{28}$$

where $g_{1,i}$ is specified in Eq. (15). Recall that g in Eq. (14) is defined as

$$g(\vec{\mathbf{b}}) := \sum_{i=1}^{k^*} \left[\alpha'_i g_{1,i}(\vec{\mathbf{b}}) + \mu'_i g_{2,i}(\vec{\mathbf{b}}) \right] .$$

By plugging-in Eq. (27) and Eq. (28), we have that

$$\sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log m}} g(\vec{\mathbf{b}}) = \sum_{i=1}^{k^*} \alpha'_i \cdot \mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_i) + \sum_{i=1}^{k^*} \mu'_i \cdot p_i(\vec{\beta}') . \tag{29}$$

Therefore, the sumcheck statement

$$\sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log m}} g(\vec{\mathbf{b}}) = \sum_{i=1}^{k^*} \alpha_i \hat{\mathbf{v}}_i$$

holds if and only if $h([\alpha'_i, \mu'_i]_{i=1}^{k^*}) = 0$. Recall that $\mathsf{Exp}_3 = 1$ implies that $h([\alpha'_i, \mu'_i]_{i=1}^{k^*}) \neq 0$, i.e., the sumcheck statement does not hold. Meanwhile, note that the random evaluation statement for g holds because

$$\forall i \in [k^*] \ : \ \mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}'_o) = \theta'_i$$

and the verification at Step 4 passes. By the Generalized Sum-Check Theorem (Lemma 2.5), $\Pr[E_3] \leq \Pr[\mathsf{Exp}_3] \leq 2b \log m / |\mathcal{C}|$ (over the randomness of $\vec{\mathbf{r}}'_o$). □

In summary, we have that

$$\Pr[E_{\mathsf{eval}} \wedge E_{\mathsf{bad}}] \leq \Pr[E_1] + \Pr[E_2] + \Pr[E_3] \leq \frac{\log m}{|\mathcal{C}|} + \frac{2k^*}{|\mathcal{C}|} + \frac{2b \log m}{|\mathcal{C}|} ,$$

which finishes the proof of Lemma 3.7. □

Thus, the success probability of the extractor is at least

$$\Pr[E_{\mathsf{ext}} \wedge E_{\mathsf{valid}}] \geq \epsilon_{\mathsf{fold}}(\mathcal{A}, \mathsf{P}^*) - \frac{k^*}{|\mathcal{C}_{\mathsf{small}}|} - \epsilon_{\mathsf{bind}} - \frac{(2b+1)\log m + 2k^*}{|\mathcal{C}|} ,$$

which finishes the proof of Theorem 3.3. □

Remark 3.3. *Theorem 3.3 is applicable only when* Π_{fold} *is instantiated as an interactive protocol. In practice,* Π_{fold} *can be converted into a non-interactive protocol using the Fiat-Shamir transform. The knowledge analysis of the Fiat-Shamir transformed version of* Π_{fold} *is left as future work.*

3.3 Supporting Small Prime Modulus

In the protocol Π_{fold} (Figure 3), if $\mathcal{R}_q \cong \mathbb{Z}_q^d$, the size of the strong sampling set $\mathcal{C} := \mathbb{Z}_q$ is only q. This is the best we can hope for: Assume for contradiction that exists $\mathcal{C}$ where $|\mathcal{C}| > q$, by the pigeonhole principle, there exist two elements $\mathbf{a}, \mathbf{b}$ in $\mathcal{C} \subseteq \mathcal{R}_q \cong \mathbb{Z}_q^d$ that share the same value at the 1st slot of their NTT representation. Hence the 1st slot of $\mathsf{NTT}(\mathbf{a}-\mathbf{b})$ is zero, and $\mathbf{a}-\mathbf{b}$ is a zero-divisor as $\mathbf{c}\cdot(\mathbf{a}-\mathbf{b}) = 0$ for the element $\mathbf{c} \neq 0$ whose NTT representation is $(1, 0, \ldots, 0)$. This contradicts with the fact that $\mathcal{C}$ is a sampling set.

To achieve 128-bit security, we need to use at least a 128-bit prime modulus in Π_{fold}. In practice, however, it would be significantly more efficient to use a smaller modulus, say a 32-bit prime, which is a good fit for GPUs that operate on 32-bit data types, or for CPUs that operate on 32 or 64-bit integer types.

In this section, we describe an optimization that extends Π_{fold} to support a small prime modulus q. The key idea is to use q where $\mathcal{R}_q \cong \mathbb{F}_{q^\tau}^t$ for some $\tau > 1$ such that $q^\tau \approx 2^{128}$. Here $\mathbb{F}_{q^\tau}$ is an extension field of $\mathbb{F}_q$. We note, however, that q cannot be too small since we must preserve the hardness of the MSIS problem.

Let $t \in \mathbb{N}$ be a divisor of d and denote $\tau := d/t$. Let q be a prime such that $q \equiv 1 + 2t \pmod{4t}$ and $q^\tau \approx 2^{128}$. Recall from Section 2 that we have $\mathcal{R}_q \cong \mathbb{F}_{q^\tau}^t$ via the NTT isomorphism. Thus we can rewrite the commitment opening relation $\mathcal{R}_{\mathsf{cm}}^B$ (Eq. (8)) as

$$\mathcal{R}_{\mathsf{cm}}^{\tau,B} := \left\{ (\mathsf{pp}, \mathsf{cm} \in \mathcal{R}_q^\kappa; \vec{\mathbf{f}} \in \mathcal{R}_q^m) \ : \ \begin{array}{c} (\mathsf{cm} = \mathbf{A}\vec{\mathbf{f}}) \wedge \\ \forall j \in [\tau] : \\ \left(\hat{\mathbf{f}}_j \circ \left[\bigcirc_{i=1}^{B-1} (\hat{\mathbf{f}}_j - \hat{i}) \circ (\hat{\mathbf{f}}_j + \hat{i})\right] = \hat{0}\right) \end{array} \right\},$$

here $\hat{\mathbf{f}} := (\hat{\mathbf{f}}_1, \ldots, \hat{\mathbf{f}}_\tau) \in \mathcal{R}_q^{m \times \tau}$ is the vector such that

$$\mathsf{NTT}(\hat{\mathbf{f}}) := (\mathsf{NTT}(\hat{\mathbf{f}}_1), \ldots, \mathsf{NTT}(\hat{\mathbf{f}}_\tau)) \in \mathbb{F}_{q^\tau}^{m \times d}$$

equals the coefficient embedding matrix of $\vec{\mathbf{f}}$ (which is in $\mathbb{Z}_q^{m \times d}$), that is, $\mathsf{NTT}(\hat{\mathbf{f}}) = \mathsf{Coef}(\vec{\mathbf{f}})$. Given $\mathcal{R}_{\mathsf{cm}}^{\tau,B}$, we can similarly generalize the expanded commitment opening relation $\mathcal{R}_{\mathsf{eval}}^B$ (Eq. (9)) to $\mathcal{R}_{\mathsf{eval}}^{\tau,B}$ defined as

$$\mathcal{R}_{\mathsf{eval}}^{\tau,B} := \left\{ (\mathsf{pp}, (\vec{\mathbf{r}}, [\hat{\mathbf{v}}_j]_{j=1}^\tau, \mathsf{cm}) \in \mathcal{R}_q^{\log m} \times \mathcal{R}_q^\tau \times \mathcal{R}_q^\kappa; \vec{\mathbf{f}} \in \mathcal{R}_q^m) \ : \ \begin{array}{c} (\mathsf{pp}, \mathsf{cm}; \vec{\mathbf{f}}) \in \mathcal{R}_{\mathsf{cm}}^B \wedge \\ \left(\forall j \in [\tau] : \mathsf{mle}\left[\hat{\mathbf{f}}_j\right](\vec{\mathbf{r}}) = \hat{\mathbf{v}}_j\right) \end{array} \right\}, \tag{30}$$

The reduction of knowledge from $(\mathcal{R}_{\mathsf{eval}}^{\tau,b})^{2k}$ to $\mathcal{R}_{\mathsf{eval}}^{\tau,B}$ is almost identical to Π_{fold} (Figure 3) except for 2 modifications below.

- We define the challenge space $\mathcal{C} \subseteq \mathcal{R}_q$ as the set of elements whose NTT representation equals i multiplying the identity vector $I_t := (1, \ldots, 1) \in \mathbb{F}_{q^\tau}^t$ (where i is enumerated over $\mathbb{F}_{q^\tau}$), that is,

$$\mathcal{C} := \{\mathbf{a}_i \in \mathcal{R}_q \ : \ \mathsf{NTT}(\mathbf{a}_i) = i \cdot I_t\}_{i \in \mathbb{F}_{q^\tau}}. \tag{31}$$

This ensures that $\mathcal{C}$ is a strong sampling set with size $q^\tau \approx 2^{128}$, because the difference of any two distinct elements in $\mathcal{C}$ maps to $a \cdot I_t$ for some a in $\mathbb{F}_{q^\tau}^\times$ through the NTT isomorphism, which has inverse $a^{-1} \cdot I_t$. Thus we can achieve 128-bit security even if q is significantly smaller than 2^{128} (given τ is large enough so that $q^\tau \approx 2^{128}$).

- We argue that all proofs in Section 3.2 will still be valid. Let $[\rho_i]_{i=1}^{2k}$ be the last folding challenges (in Figure 3). For every $i \in [2k]$, let $\Theta_i := [\theta_{i,j}]_{j=1}^{\tau} \in \mathcal{R}_q^\tau$ where $\{\theta_{i,j} \in \mathcal{R}_q\}$ are the claimed evaluations in the protocol execution. We denote by $\mathsf{NTT}(\Theta_i) := (\mathsf{NTT}(\theta_{i,1}), \ldots, \mathsf{NTT}(\theta_{i,\tau})) \in \mathbb{F}_{q^\tau}^d$. The folding verifier computes $\hat{V}_o := [\hat{\mathbf{v}}_{o,j}]_{j=1}^{\tau} \in \mathcal{R}_q^\tau$ such that $\mathsf{NTT}(\hat{V}_o) \in \mathbb{F}_{q^\tau}^d$ satisfies that

$$\mathsf{NTT}(\hat{V}_o) = \sum_{i=1}^{2k} \mathsf{RotSum}(\rho_i, \mathsf{NTT}(\Theta_i)) \,, \tag{32}$$

  where RotSum is defined in Lemma 2.1. Therefore, by Lemma A.1, we can extend Lemma 3.2 to the case where $\mathcal{R}_q \cong \mathbb{F}_{q^\tau}^{d/\tau}$ and the folding verifier can still verify the folding proof. Moreover, by the 3rd claim in Lemma 2.1, we can extend Eq. (24) to a more general setting where $\mathcal{R}_q \cong \mathbb{F}_{q^\tau}^{d/\tau}$. In this setting, the single linear equation (over $\mathcal{R}_q$) in Eq. (24) is extended to τ linear equations. Given this extension, all proofs in Section 3.2 naturally follow.

4 A Lattice-based Folding Scheme for CCS

In this section, we construct a folding scheme for customizable constraint systems (CCS) as introduced in [STW23a]. CCS is a generalization of Rank-1 Constraint Systems (R1CS) that supports high-degree custom gates, enabling better expressiveness and applicability. As discussed at the beginning of Section 3, this folding scheme enables us to build IVC/PCD from Ajtai commitments. Our approach is highly modular and generic. We begin by adapting the definition of customizable constraint systems [STW23a] to the ring setting.

Definition 4.1 (CCS over rings)**.** *Let* $\mathsf{pp} := (n_r, n_c, t, n_s, \mathsf{deg}, \ell_{\mathsf{in}})$ *be the integer public parameters*[2] *and let* $\bar{\mathcal{R}}$ *be an arbitrary ring. Let* $\mathbb{i}$ *be an index that consists of (i)* t *matrices* $M_1, \ldots, M_t \in \bar{\mathcal{R}}^{n_r \times n_c}$ *with* $O(n_r + n_c)$ *non-zero entries; (ii)* n_s *multisets* $S_1, \ldots, S_{n_s} \subseteq [t]$ *such that* $|S_i| \leq \mathsf{deg}$ *for all* $i \in [n_s]$*; and (iii)* n_s *scalars* $c_1, \ldots, c_{n_s} \in \bar{\mathcal{R}}$*.*

We denote by $\mathsf{pp}_{\mathsf{ccs}} := (\mathsf{pp}, \mathbb{i})$ *the index-specific parameters. Given a tuple* $(\mathsf{pp}_{\mathsf{ccs}}, \mathbb{x} \in \bar{\mathcal{R}}^{\ell_{\mathsf{in}}}; \mathbb{w} \in \bar{\mathcal{R}}^{n_c - \ell_{\mathsf{in}} - 1})$ *and let* $\vec{\mathbf{z}} := (\mathbb{x}, 1, \mathbb{w}) \in \bar{\mathcal{R}}^{n_c}$*. We say* $(\mathsf{pp}_{\mathsf{ccs}}, \mathbb{x}; \mathbb{w})$ *is in the relation* $\mathcal{R}_{\mathsf{ccs}}$ *(over ring* $\bar{\mathcal{R}}$*) if and only if*

$$\sum_{i=1}^{n_s} c_i \cdot \bigcirc_{j \in S_i} (M_j \cdot \vec{\mathbf{z}}) = 0^{n_r} \,.$$

[2] Informally, n_r denotes the number of constraints, n_c denotes the extended witness size and deg is the custom gate degree.

Here $\bigcirc$ denotes the Hadamard product between vectors. And 0 (and 1) is the additive (and multiplicative) identity in $\bar{\mathcal{R}}$ respectively.

Remark 4.1 (Packing multiple CCS field constraints)**.** *If the ring $\bar{\mathcal{R}}$ is isomorphic to $\mathbb{F}^k$ for a field $\mathbb{F}$, we can pack k instance-witness pairs for the CCS relation over $\mathbb{F}$ into a single instance-witness pair for the CCS relation over $\bar{\mathcal{R}}$. More precisely, a set of k tuples $((\mathsf{pp}, \mathbb{i}_i), \mathbb{x}_i, \mathbb{w}_i)_{i=1}^k$, are all in the relation $\mathcal{R}_{\mathsf{ccs}}$ over $\mathbb{F}$ if and only if the transformed tuple $((\mathsf{pp}, \mathbb{i}^*), \mathbb{x}^*, \mathbb{w}^*)$ is in the relation $\mathcal{R}_{\mathsf{ccs}}$ over $\bar{\mathcal{R}}$. Each entry $\mathbf{e} \in \bar{\mathcal{R}}$ in $(\mathbb{i}^*, \mathbb{x}^*, \mathbb{w}^*)$ is set such that $\mathsf{NTT}(\mathbf{e}) = (e_1, \ldots, e_k)$, where $e_i \in \mathbb{F}$ is the corresponding entry in $(\mathbb{i}_i, \mathbb{x}_i, \mathbb{w}_i)$ for $1 \leq i \leq k$.*

4.1 Lattice-based Committed CCS

Next, we introduce the lattice-based committed CCS relation $\mathcal{R}^B_{\mathsf{cmccs}}$ that extends the commitment opening relation $\mathcal{R}^B_{\mathsf{cm}}$ in Eq. (8) to the CCS setting. A folding scheme for $\mathcal{R}^B_{\mathsf{cmccs}}$ would allow us to build an IVC/PCD scheme.

Intuitively, a witness of the committed CCS relation consists of a pair $(\vec{\mathbf{f}}, \vec{\mathbf{w}})$, and the relation checks that (i) $\vec{\mathbf{w}}$ is a valid witness for the CCS instance $\mathbb{x}_{\mathsf{ccs}}$, (ii) $\vec{\mathbf{f}}$ is a low-norm opening of the Ajtai commitment cm, and (iii) $\vec{\mathbf{f}}$ is the gadget decomposition of the witness $\vec{\mathbf{w}}$, meaning $\vec{\mathbf{w}} = \mathbf{G} \times \vec{\mathbf{f}}$ for the gadget $\mathbf{G}$. We formally define the relation below.

Definition 4.2 (Lattice-based committed CCS relation)**.** *Let $\mathcal{R}_q := \mathbb{Z}_q[X]/(X^d+1)$ where q is a prime and d is a power of two. Let $\mathsf{pp} := (\mathsf{pp}_{\mathsf{cm}}, \mathsf{pp}_{\mathsf{ccs}})$ be the public parameters where $\mathsf{pp}_{\mathsf{cm}} = (\kappa, m, B < q/2, \mathbf{A})$ is the public parameter for $\mathcal{R}^B_{\mathsf{cm}}$ (Eq. (8)) and $\mathsf{pp}_{\mathsf{ccs}} = (n_r, n_c, t, n_s, \mathsf{deg}, \ell_{\mathsf{in}}, \mathbb{i})$ (defined in Definition 4.1) is for $\mathcal{R}_{\mathsf{ccs}}$ (over $\mathcal{R}_q$).*

*Set $\ell := m/n_c \in \mathbb{N}$ such that $B^\ell \geq q/2$. Let $\mathbf{G} := \mathbf{I}_{n_c} \otimes [1, B, \ldots, B^{\ell-1}] \in \mathbb{Z}_q^{n_c \times m}$ be the **gadget matrix**. The indexed relation $\mathcal{R}^B_{\mathsf{cmccs}}$ is defined as*

$$\mathcal{R}^B_{\mathsf{cmccs}} := \left\{ \begin{array}{c} \Big(\mathsf{pp}, \mathbb{x} := (\mathsf{cm} \in \mathcal{R}_q^\kappa, \mathbb{x}_{\mathsf{ccs}} \in \mathcal{R}_q^{\ell_{\mathsf{in}}}); \mathbb{w} := (\vec{\mathbf{f}} \in \mathcal{R}_q^m, \mathbb{w}_{\mathsf{ccs}} \in \mathcal{R}_q^{n-\ell_{\mathsf{in}}-1})\Big) \ s.t. \\ (\mathsf{pp}_{\mathsf{cm}}, \mathsf{cm}; \vec{\mathbf{f}}) \in \mathcal{R}^B_{\mathsf{cm}} \wedge (\mathsf{pp}_{\mathsf{ccs}}, \mathbb{x}_{\mathsf{ccs}}; \mathbb{w}_{\mathsf{ccs}}) \in \mathcal{R}_{\mathsf{ccs}} \wedge (\mathbf{z}_{\mathsf{ccs}} = \mathbf{G} \times \vec{\mathbf{f}}) \end{array} \right\}, \tag{33}$$

where $\mathbf{z}_{\mathsf{ccs}} := (\mathbb{x}_{\mathsf{ccs}}, 1, \mathbb{w}_{\mathsf{ccs}}) \in \mathcal{R}_q^{n_c}$.

Remark 4.2. *The constraint $\mathbf{z}_{\mathsf{ccs}} = \mathbf{G} \times \vec{\mathbf{f}}$ is used to capture that $\vec{\mathbf{f}}$ is the "base-B" representation of the original witness $\mathbf{z}_{\mathsf{ccs}}$ in CCS. Crucially, consider a witness $(\vec{\mathbf{f}}, \mathbb{w}_{\mathsf{ccs}})$ for instance $(\mathsf{cm}, \mathbb{x}_{\mathsf{ccs}})$. We know that $\vec{\mathbf{f}}$ is a low-norm opening for the Ajtai binding commitment cm given that $(\mathsf{cm}, \vec{\mathbf{f}})$ is in $\mathcal{R}^B_{\mathsf{cm}}$. This implies that $\mathbb{w}_{\mathsf{ccs}}$ is also bound to cm because the equation $(\mathbb{x}_{\mathsf{ccs}}, 1, \mathbb{w}_{\mathsf{ccs}}) = \mathbf{G} \times \vec{\mathbf{f}}$ holds.*

Remark 4.3. *We set $(\mathbb{x}_{\mathsf{ccs}}, 1, \mathbb{w}_{\mathsf{ccs}}) = \mathbf{G} \times \vec{\mathbf{f}}$ only for ease of exposition. Note that the integrity of $\mathbb{x}_{\mathsf{ccs}}$ is already guaranteed by the verifier checks. Thus it suffices to decompose $\mathbb{w}_{\mathsf{ccs}}$ to $\vec{\mathbf{f}}$ and check the statement $\mathbb{w}_{\mathsf{ccs}} = \mathbf{G} \times \vec{\mathbf{f}}$.*

The expanded relation. Similar to the paradigm in Section 3.1, to construct a folding scheme for $\mathcal{R}_{\mathsf{comp}} := \mathcal{R}^B_{\mathsf{cmccs}}$, we introduce a new relation $\mathcal{R}_{\mathsf{acc}} := \mathcal{R}^B_{\mathsf{evalccs}}$ that augments $\mathcal{R}^B_{\mathsf{cmccs}}$ with a multilinear evaluation statement. Note that in $\mathcal{R}^B_{\mathsf{evalccs}}$, we replace the high-degree custom gate relation $\mathcal{R}^B_{\mathsf{cmccs}}$ with a linearized relation $\mathcal{R}_{\mathsf{lccs}}$. Like the linearization framework from Hypernova [KS23b], this adjustment is necessary because our folding scheme runs sum-checks to reduce the norm constraints and the high-degree custom gate constraints in $\mathcal{R}^B_{\mathsf{cmccs}}$ into linearized statements. Hence, we must modify the accumulated relation accordingly. Notably, $\mathcal{R}^B_{\mathsf{evalccs}}$ extends $\mathcal{R}^B_{\mathsf{eval}}$ from Eq. (9).

Definition 4.3 (Lattice-based linearized CCS relation)**.** *Let* $\mathsf{pp} := (\mathsf{pp}_{\mathsf{cm}}, \mathsf{pp}_{\mathsf{ccs}})$ *be the public parameters in Definition 4.2 where* $\ell := m/n_c \in \mathbb{N}$ *and* $B^\ell \geq q/2$*. Without loss of generality*[3]*, we assume that the number of rows* n_r *in CCS matrices equals to the committed witness length* m*.*

We define the linearized CCS relation $\mathcal{R}_{\mathsf{lccs}}$ *as the set of tuples*

$$(\mathsf{pp}_{\mathsf{ccs}}, \mathbb{x} := (\vec{\mathbf{r}} \in \mathcal{R}_q^{\log m}, [\mathbf{u}_j]_{j=1}^t \in \mathcal{R}_q^t, \mathbb{x}_{\mathsf{ccs}} \in \mathcal{R}_q^{\ell_{\mathsf{in}}}, \mathbf{h} \in \mathcal{R}_q); \mathbb{w} := \mathbb{w}_{\mathsf{ccs}} \in \mathcal{R}_q^{n_c - \ell_{\mathsf{in}} - 1})$$

such that for all $j \in [t]$*:*

$$\mathbf{u}_j = \sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log n_c}} \mathsf{mle}\left[M_j\right](\vec{\mathbf{r}}, \vec{\mathbf{b}}) \cdot \mathsf{mle}\left[\mathbf{z}_{\mathsf{ccs}}\right](\vec{\mathbf{b}})\,. \tag{34}$$

Here $\mathsf{mle}\left[M_j\right] \in \mathcal{R}_q^{\leq 1}[X_1, \ldots, X_{\log n_r + \log n_c}]$ *and* $\mathsf{mle}\left[\mathbf{z}_{\mathsf{ccs}}\right] \in \mathcal{R}_q^{\leq 1}[X_1, \ldots, X_{\log n_c}]$ *are the multilinear extensions of matrix* $M_j \in \mathcal{R}_q^{n_r \times n_c}$ *and* $\mathbf{z}_{\mathsf{ccs}} := (\mathbb{w}_{\mathsf{ccs}}, \mathbf{h}, \mathbb{x}_{\mathsf{ccs}}) \in \mathcal{R}_q^{n_c}$ *respectively.* $\mathcal{R}^B_{\mathsf{evalccs}}$ *is defined as*

$$\mathcal{R}^B_{\mathsf{evalccs}} := \left\{ \begin{array}{c} \Big(\mathsf{pp}, \mathbb{x} := (\vec{\mathbf{r}}, \mathsf{cm}, \hat{\mathbf{v}}, [\mathbf{u}_j]_{j=1}^t, \mathbb{x}_{\mathsf{ccs}}, \mathbf{h}); \\ \mathbb{w} := (\vec{\mathbf{f}} \in \mathcal{R}_q^m, \mathbb{w}_{\mathsf{ccs}})\Big) \;\; s.t. \\ (\mathbf{z}_{\mathsf{ccs}} = \mathbf{G}\vec{\mathbf{f}}) \wedge (\mathsf{pp}_{\mathsf{cm}}, (\mathsf{cm}, \vec{\mathbf{r}}, \hat{\mathbf{v}}); \vec{\mathbf{f}}) \in \mathcal{R}^B_{\mathsf{eval}} \\ \wedge (\mathsf{pp}_{\mathsf{ccs}}, (\vec{\mathbf{r}}, [\mathbf{u}_j]_{j=1}^t, \mathbb{x}_{\mathsf{ccs}}, \mathbf{h}); \mathbb{w}_{\mathsf{ccs}}) \in \mathcal{R}_{\mathsf{lccs}} \end{array} \right\}, \tag{35}$$

where $\mathbf{G} := \mathbf{I}_{n_c} \otimes [1, B, \ldots, B^{\ell-1}] \in \mathbb{Z}_q^{n_c \times m}$ *and* $\mathcal{R}^B_{\mathsf{eval}}$ *is defined in Eq. (9).*

4.2 A Generic Folding Scheme for CCS

In this section, we construct a folding scheme for $\mathcal{R}_{\mathsf{acc}} := \mathcal{R}^B_{\mathsf{evalccs}}$ and $\mathcal{R}_{\mathsf{comp}} := \mathcal{R}^B_{\mathsf{cmccs}}$, or equivalently, a reduction of knowledge (Definition 2.5) from $\mathcal{R}^B_{\mathsf{evalccs}} \times \mathcal{R}^B_{\mathsf{cmccs}}$ to $\mathcal{R}^B_{\mathsf{evalccs}}$. The scheme is presented for modularity and illustration purposes. In Section 4.3, we introduce an optimized version with better efficiency.

Similar to the strategy in Section 3.2, the construction consists of three steps.

[3] We can always pad dummy constraints/witnesses to enforce $n_r = m$.

Step 1: Linearization. First, we reduce the relation $\mathcal{R}^B_{\mathsf{evalccs}} \times \mathcal{R}^B_{\mathsf{cmccs}}$ to $\mathcal{R}^B_{\mathsf{evalccs}} \times \mathcal{R}^B_{\mathsf{evalccs}}$ via a protocol Π_{ccs} (Figure 5) that reduces $\mathcal{R}^B_{\mathsf{cmccs}}$ to $\mathcal{R}^B_{\mathsf{evalccs}}$. The protocol Π_{ccs} runs a sum-check to reduce the high-degree custom gates check to a degree-1 check (e.g., a multilinear evaluation check). Similar to the expansion step in Section 3.2, the linearization step reduces to a linear relation with evaluation-like statements.

Step 2: Decomposition. Next, using a protocol Π_{ccsdec} (Figure 6), we reduce the relation $\mathcal{R}^B_{\mathsf{evalccs}} \times \mathcal{R}^B_{\mathsf{evalccs}}$ to a relation

$$(\mathcal{R}^b_{\mathsf{evalccs}})^{2k} := \underbrace{\mathcal{R}^b_{\mathsf{evalccs}} \times \cdots \times \mathcal{R}^b_{\mathsf{evalccs}}}_{2k}$$

where b, k are chosen such that $b^k = B$. We note that Π_{ccsdec} is an adaptation to Π_{dec} (Figure 2) with similar analysis.

Step 3: Folding. Finally, we reduce $(\mathcal{R}^b_{\mathsf{evalccs}})^{2k}$ back to $\mathcal{R}^B_{\mathsf{evalccs}}$ using a protocol Π_{ccsfold} (Figure 7). Note that Π_{ccsfold} is an adaptation to Π_{fold} (Figure 3) with similar analysis.

By the composition theorems for reductions of knowledge (Theorem 2.1, Theorem 2.2), the composed protocol $\Pi_{\mathsf{mccsfold}} := \Pi_{\mathsf{ccsfold}} \circ \Pi_{\mathsf{ccsdec}} \circ \Pi_{\mathsf{ccs}}$ is a reduction of knowledge from $\mathcal{R}^B_{\mathsf{evalccs}} \times \mathcal{R}^B_{\mathsf{cmccs}}$ to $\mathcal{R}^B_{\mathsf{evalccs}}$ as desired. We formally state the result in Theorem 4.1.

Theorem 4.1. *Let $\mathcal{R}_q \cong \mathbb{F}_{q^\tau}^{d/\tau}$ for some $\tau \in \mathbb{N}$ where $\tau \mid d$ and $1/q^\tau$ is in $\mathsf{negl}(\lambda)$. Let $\mathcal{C}_{\mathsf{small}} \subseteq \mathcal{R}_q$ be a strong sampling set for which $1/|\mathcal{C}_{\mathsf{small}}|$ is in $\mathsf{negl}(\lambda)$, and the expansion factor $T := \|\mathcal{C}_{\mathsf{small}}\|_{\mathsf{op}} \leq c$ (Definition 6) for some $c \in \mathbb{N}$. Let $\mathcal{C}$ be the strong sampling set as in Eq. (31). Let $\mathsf{pp}_{\mathsf{cm}} := (\kappa, m, \mathbf{A}, B < q/2)$ and $\mathsf{pp}_{\mathsf{ccs}} := (n_r := m, n_c, t, n_s, \mathsf{deg}, \ell_{\mathsf{in}}, [M_j]_{j=1}^t, [S_i, c_i]_{i=1}^{n_s})$ be the public parameters such that $B^{m/n_c} \geq q/2$ and $\mathsf{MSIS}^{\infty,q}_{\kappa,m,8TB}$ is hard. Set b, k such that $2kc(b-1) < B$ and $b^k = B$. Let Π_{ccs}, Π_{ccsdec}, Π_{ccsfold} be the protocols in Figure 5, Figure 6 and Figure 7, respectively. The composed protocol $\Pi_{\mathsf{mccsfold}} := \Pi_{\mathsf{ccsfold}} \circ \Pi_{\mathsf{ccsdec}} \circ \Pi_{\mathsf{ccs}}$ is a public-coin reduction of knowledge from relation $\mathcal{R}^B_{\mathsf{evalccs}} \times \mathcal{R}^B_{\mathsf{cmccs}}$ to $\mathcal{R}^B_{\mathsf{evalccs}}$.*

Proof. The protocol is public-coin as the three subprotocols are all public-coin. For the case where $\mathcal{R}_q \cong \mathbb{Z}_q^d$, the Theorem follows from Lemma 4.1, Lemma 4.2, Theorem 4.2 and the knowledge composition theorems (Theorem 2.1 and Theorem 2.2). For the case where $\mathcal{R}_q \cong \mathbb{F}_{q^\tau}^{d/\tau}$, the same optimization in Section 3.3 can be used to extend Theorem 4.1 to support small modulus q. □

4.2.1 Linearization: The reduction from $\mathcal{R}^B_{\mathsf{cmccs}}$ to $\mathcal{R}^B_{\mathsf{evalccs}}$

By Theorem 2.2, to reduce from $\mathcal{R}^B_{\mathsf{evalccs}} \times \mathcal{R}^B_{\mathsf{cmccs}}$ to $\mathcal{R}^B_{\mathsf{evalccs}} \times \mathcal{R}^B_{\mathsf{evalccs}}$, it suffices to construct a protocol that reduces $\mathcal{R}^B_{\mathsf{cmccs}}$ (Eq. (33)) to $\mathcal{R}^B_{\mathsf{evalccs}}$ (Eq. (35)). We describe the

Parameters: A strong sampling set $\mathcal{C} := \mathbb{Z}_q \subseteq \mathcal{R}_q$ (Definition 2.1)
Input: $\mathbb{x} := (\mathsf{cm}, \mathbb{x}_{\mathsf{ccs}}) \in \mathcal{R}_q^{\kappa} \times \mathcal{R}_q^{\ell_{\mathsf{in}}}$ and $\mathbb{w} := (\vec{\mathbf{f}}, \mathbb{w}_{\mathsf{ccs}}) \in \mathcal{R}_q^{m} \times \mathcal{R}_q^{n_c - \ell_{\mathsf{in}} - 1}$
Output: $\mathbb{x}_o := (\vec{\mathbf{r}}_o \in \mathcal{R}_q^{\log m}, \hat{\mathbf{v}} \in \mathcal{R}_q, \mathsf{cm}, [\mathbf{u}_j \in \mathcal{R}_q]_{j=1}^{t}, \mathbb{x}_{\mathsf{ccs}}, 1)$ and $\mathbb{w}_o := (\vec{\mathbf{f}}, \mathbb{w}_{\mathsf{ccs}})$
The protocol $\langle \mathsf{P}(\mathsf{pp}, \mathbb{x}; \mathbb{w}), \mathsf{V}(\mathsf{pp}, \mathbb{x}) \rangle$**:**

1. $\mathsf{V} \to \mathsf{P}$: V sends P a random vector $\vec{\beta} \xleftarrow{\$} \mathcal{C}^{\log m}$.
2. $\mathsf{P} \leftrightarrow \mathsf{V}$: Let $\mathbf{z}_{\mathsf{ccs}} := (\mathbb{x}_{\mathsf{ccs}}, 1, \mathbb{w}_{\mathsf{ccs}})$ and let deg denote the CCS gate degree, define the polynomial $g \in \mathcal{R}_q^{\leq \mathsf{deg}+1}[X_1, \ldots, X_{\log m}]$

$$g(\vec{\mathbf{x}}) := eq(\vec{\beta}, \vec{\mathbf{x}}) \cdot \left(\sum_{i=1}^{n_s} c_i \cdot \left[\prod_{j \in S_i} \left(\sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log n_c}} \mathsf{mle}\left[M_j\right](\vec{\mathbf{x}}, \vec{\mathbf{b}}) \cdot \mathsf{mle}\left[\mathbf{z}_{\mathsf{ccs}}\right](\vec{\mathbf{b}}) \right) \right] \right) .$$

   P and V run a sum-check protocol for the claim $\sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log m}} g(\vec{\mathbf{b}}) = 0$.
   Let $\vec{\mathbf{r}}_o \xleftarrow{\$} \mathcal{C}^{\log m}$ be the sum-check challenge vector. The protocol reduces to a random evaluation check $g(\vec{\mathbf{r}}_o) \stackrel{?}{=} s$ for some $s \in \mathcal{R}_q$.
3. $\mathsf{P} \to \mathsf{V}$: P sends V the values $(\hat{\mathbf{v}}, [\mathbf{u}_j]_{j=1}^{t})$ where $\hat{\mathbf{v}} := \mathsf{mle}\left[\hat{\mathbf{f}}\right](\vec{\mathbf{r}}_o)$ and for every $j \in [t]$, $\mathbf{u}_j$ is computed as

$$\mathbf{u}_j := \sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log n_c}} \mathsf{mle}\left[M_j\right](\vec{\mathbf{r}}_o, \vec{\mathbf{b}}) \cdot \mathsf{mle}\left[\mathbf{z}_{\mathsf{ccs}}\right](\vec{\mathbf{b}}) .$$

4. V computes $\mathbf{e} := eq(\vec{\beta}, \vec{\mathbf{r}}_o)$ and checks that

$$\mathbf{e} \cdot \left(\sum_{i=1}^{n_s} c_i \cdot \prod_{j \in S_i} \mathbf{u}_j \right) \stackrel{?}{=} s .$$

5. V outputs $\mathbb{x}_o := (\vec{\mathbf{r}}_o, \hat{\mathbf{v}}, \mathsf{cm}, [\mathbf{u}_j]_{j=1}^{t}, \mathbb{x}_{\mathsf{ccs}}, 1)$. P outputs $\mathbb{w}_o := (\vec{\mathbf{f}}, \mathbb{w}_{\mathsf{ccs}})$.

Figure 5: The protocol Π_{ccs} that reduces $\mathcal{R}^B_{\mathsf{cmccs}}$ to $\mathcal{R}^B_{\mathsf{evalccs}}$.

protocol Π_{ccs} in Figure 5. The protocol is inspired by the linearization technique from Hypernova [KS23b]. Intuitively, it runs a sum-check to reduce the high-degree CCS relation to a multilinear evaluation relation that has degree 1.

Lemma 4.1. *Π_{ccs} is a reduction of knowledge from $\mathcal{R}^B_{\mathsf{cmccs}}$ to $\mathcal{R}^B_{\mathsf{evalccs}}$ for any bound $B \in \mathbb{N}$.*

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

4.2.2 Decomposition: The reduction from $(\mathcal{R}^B_{\mathsf{evalccs}})^2$ to $(\mathcal{R}^b_{\mathsf{evalccs}})^{2k}$

Next, we describe the decomposition step that splits the witnesses and reduces the norms. By Theorem 2.2, it suffices to construct a protocol Π^*_{ccsdec} that reduces $\mathcal{R}^B_{\mathsf{evalccs}}$ to $(\mathcal{R}^b_{\mathsf{evalccs}})^k$, and the reduction of knowledge from $\mathcal{R}^B_{\mathsf{evalccs}} \times \mathcal{R}^B_{\mathsf{evalccs}}$ to $(\mathcal{R}^b_{\mathsf{evalccs}})^{2k}$ is $\Pi_{\mathsf{ccsdec}} := \Pi^*_{\mathsf{ccsdec}} \times \Pi^*_{\mathsf{ccsdec}}$ that runs two instances of Π^*_{ccsdec} in parallel.

More generally, we construct a reduction of knowledge from a relation $\mathcal{R}^B_{\mathsf{ccshom}}$ to $(\mathcal{R}^b_{\mathsf{ccshom}})^k$. Here $\mathcal{R}^B_{\mathsf{ccshom}}$ is a generalization of $\mathcal{R}^B_{\mathsf{evalccs}}$ (Definition 35) where we generalize Ajtai commitments and gadget matrix multiplications to arbitrary $\mathcal{R}_q$-module homomorphisms. Let $\mathcal{U} := \mathcal{R}_q^t$ for some $t \in \mathbb{N}$. Let $\overline{M} \in \mathcal{U}^{(n+n_{\mathsf{in}})\times m}$ denote a matrix over module $\mathcal{U}$. Let $\mathcal{L} : \mathcal{R}_q^m \to \mathcal{Y}$ and $\mathcal{L}_w : \mathcal{R}_q^m \to \mathcal{R}_q^{n+n_{\mathsf{in}}}$ denote some $\mathcal{R}_q$-module homomorphisms. We define the relation $\mathcal{R}^B_{\mathsf{ccshom}}$ as

$$\mathcal{R}^B_{\mathsf{ccshom}} := \left\{ \begin{array}{c} \mathsf{pp} := (\mathcal{L}, \mathcal{L}_w, \overline{M}), \\ \mathbb{x} := (\vec{\mathbf{r}} \in \mathcal{R}_q^{\log m}, \hat{\mathbf{v}} \in \mathcal{R}_q, \mathbf{u} \in \mathcal{U}, y \in \mathcal{Y}, \mathbb{x}_w \in \mathcal{R}_q^{n_{\mathsf{in}}}); \\ \mathbb{w} := (\vec{\mathbf{f}} \in \mathcal{R}_q^m, \vec{\mathbf{w}} \in \mathcal{R}_q^n) \text{ s.t.} \\ (\mathcal{L}, (\vec{\mathbf{r}}, \hat{\mathbf{v}}, y); \vec{\mathbf{f}}) \in \mathcal{R}^B_{\mathsf{hom}} \wedge \\ (\mathbf{z} = \mathcal{L}_w(\vec{\mathbf{f}})) \wedge (\mathbf{u} = \langle \overline{M} \times \mathsf{tensor}(\vec{\mathbf{r}}), \mathbf{z} \rangle) \end{array} \right\}, \tag{36}$$

where $\mathcal{R}^B_{\mathsf{hom}}$ is defined in Eq. (10), $\mathbf{z} := (\mathbb{x}_w || \vec{\mathbf{w}}) \in \mathcal{R}_q^{n_{\mathsf{in}}+n}$ and

$$\mathsf{tensor}(\vec{\mathbf{r}}) := \bigotimes_{i=1}^{\log m} (\vec{\mathbf{r}}_i, 1 - \vec{\mathbf{r}}_i) \in \mathcal{R}_q^m \tag{37}$$

is the tensor product of $\{(\vec{\mathbf{r}}_i, 1 - \vec{\mathbf{r}}_i)\}_{i=1}^{\log m}$.

Remark 4.4. *$\mathcal{R}^B_{\mathsf{evalccs}}$ is a special case of $\mathcal{R}^B_{\mathsf{ccshom}}$ where $\mathcal{R}^B_{\mathsf{hom}} := \mathcal{R}^B_{\mathsf{eval}}$; $\mathcal{L}_w(\vec{\mathbf{f}}) := \mathbf{G}\vec{\mathbf{f}}$ (where $\mathbf{G}$ is the gadget matrix); $\mathcal{U} := \mathcal{R}_q^t$; $\mathbb{x}_w := (\mathbb{x}_{\mathsf{ccs}}, \mathbf{h})$; and $\overline{M} := (M_1, \ldots, M_t)$, i.e., each entry $(\mathbf{e}_1, \ldots, \mathbf{e}_t) \in \mathcal{U}$ of $\overline{M}$ maps to the entries $(\mathbf{e}_i)_{i=1}^t$ in matrices $M_1, \ldots, M_t$ respectively.*

We describe the protocol Π^*_{ccsdec} in Figure 6. The differences from Π^*_{dec} (Figure 2) are highlighted in red , which are for computing the $\mathbf{u}$-values and the CCS instances.

Lemma 4.2. *Fix $\mathcal{R}_q \cong \mathbb{Z}_q^d$. For any $B < q/2$ and any b, k such that $b^k = B$, Π^*_{ccsdec} is a reduction of knowledge from $\mathcal{R}^B_{\mathsf{ccshom}}$ to $(\mathcal{R}^b_{\mathsf{ccshom}})^k$.*

Proof. The proof is similar to that for Lemma 3.3. We defer the proof to Appendix B.2. □

Input: $\mathbb{x} := (\vec{\mathbf{r}}, \hat{\mathbf{v}}, y, \mathbf{u}, \mathbb{x}_w)$ and $\mathbb{w} := (\vec{\mathbf{f}}, \vec{\mathbf{w}})$
Output: $[\mathbb{x}_i = (\vec{\mathbf{r}}, \hat{\mathbf{v}}_i, y_i, \mathbf{u}_i, \mathbb{x}_{w,i}), \mathbb{w}_i = (\vec{\mathbf{f}}_i, \vec{\mathbf{w}}_i)]_{i=0}^{k-1}$
The protocol $\langle \mathsf{P}(\mathsf{pp}, \mathbb{x}; \mathbb{w}), \mathsf{V}(\mathsf{pp}, \mathbb{x}) \rangle$**:**

1. P computes $(\vec{\mathbf{f}}_0, \ldots, \vec{\mathbf{f}}_{k-1}) \leftarrow \mathsf{split}_{b,k}(\vec{\mathbf{f}})$ (Eq. (11)).
2. $\mathsf{P} \to \mathsf{V} : \mathsf{P}$ sends V the values $[y_i, \hat{\mathbf{v}}_i, \mathbf{u}_i, \mathbb{x}_{w,i}]_{i=0}^{k-1}$ where

$$y_i := \mathcal{L}(\vec{\mathbf{f}}_i)\,, \qquad \hat{\mathbf{v}}_i := \mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}})\,,$$
$$\mathbf{u}_i := \left\langle \overline{M} \times \mathsf{tensor}(\vec{\mathbf{r}}), \mathcal{L}_w(\vec{\mathbf{f}}_i) \right\rangle\,, \qquad \mathbb{x}_{w,i} := \mathcal{L}_w(\vec{\mathbf{f}}_i)[1, n_{\mathsf{in}}]$$

3. V checks that $\sum_{i=0}^{k-1} b^i \cdot [y_i, \hat{\mathbf{v}}_i, \mathbf{u}_i, \mathbb{x}_{w,i}] \stackrel{?}{=} [y, \hat{\mathbf{v}}, \mathbf{u}, \mathbb{x}_w]$.
4. V outputs $[\mathbb{x}_i = (\vec{\mathbf{r}}, \hat{\mathbf{v}}_i, y_i, \mathbf{u}_i, \mathbb{x}_{w,i})]_{i=0}^{k-1}$. P outputs $[\mathbb{w}_i = (\vec{\mathbf{f}}_i, \mathcal{L}_w(\vec{\mathbf{f}}_i))]_{i=0}^{k-1}$.

Figure 6: The protocol Π^*_{ccsdec} that reduces $\mathcal{R}^B_{\mathsf{ccshom}}$ to $(\mathcal{R}^b_{\mathsf{ccshom}})^k$.

4.2.3 Folding: The reduction from $(\mathcal{R}^b_{\mathsf{evalccs}})^{2k}$ to $\mathcal{R}^B_{\mathsf{evalccs}}$

Finally, we describe the core protocol Π_{ccsfold} that folds $2k$ instance-witness pairs of $\mathcal{R}^b_{\mathsf{evalccs}}$ into a single instance-witness pair in $\mathcal{R}_{\mathsf{acc}} := \mathcal{R}^B_{\mathsf{evalccs}}$.

Similar to the treatment in Section 3.2.3, in the following, we assume that the homomorphism $\mathcal{L}$ in the public parameter is $2B$-**relaxed binding** (Section 2.3) for challenge space $\mathcal{C}_{\mathsf{small}}$. For example, $\mathcal{R}^B_{\mathsf{evalccs}}$ is a special case of $\mathcal{R}^B_{\mathsf{ccshom}}$ by Remark 4.4 and the homomorphism $\mathcal{L}(\vec{\mathbf{f}}) := \mathbf{A}\vec{\mathbf{f}}$ is $2B$-relaxed binding given the hardness of $\mathsf{MSIS}^{\infty,q}_{\kappa,m,8TB}$ where $T = \|\mathcal{C}_{\mathsf{small}}\|_{\mathsf{op}}$.

We describe the protocol Π_{ccsfold} in Figure 7, which reduces from $(\mathcal{R}^b_{\mathsf{ccshom}})^{2k}$ to $\mathcal{R}^B_{\mathsf{ccshom}}$. The approach is similar to that in Section 3.2.3, where we fold the witnesses using small random scalars from a strong sampling set, and run sum-check to enable extractions of small-norm witnesses. For brevity, we assume that $\mathcal{U} := \mathcal{R}_q$, hence $t = 1$ and the matrix $\overline{M} = M_1 \in \mathcal{R}_q^{(n+n_{\mathsf{in}}) \times m}$. The protocol naturally extends to the case when $\mathcal{U} := \mathcal{R}_q^t$ for $t > 1$: we set $\{\mathbf{u}_i, \eta_i\}$ to be elements over $\mathcal{R}_q^t$, sample challenges $\{\zeta_i\}$ over $\mathcal{C}^t$ and replace the multiplications between ζ_i and $\mathbf{u}_i$ (and η_i) with inner product operations.

Theorem 4.2. *Let $\mathcal{R}_q \cong \mathbb{Z}_q^d$. Let $\mathcal{C}$, $\mathcal{C}_{\mathsf{small}}$ be strong sampling sets where $1/|\mathcal{C}|, 1/|\mathcal{C}_{\mathsf{small}}| = \mathsf{negl}(\lambda)$ and $\mathcal{C}_{\mathsf{small}}$ has expansion factor at most c (Definition 6). Let $\mathsf{pp} := (m, n, n_{\mathsf{in}}, B, \mathcal{L}, \mathcal{L}_w)$ be the public parameter where the homomorphism $\mathcal{L}$ is $2B$-relaxed binding (Section 2.3) for challenge space $\mathcal{C}_{\mathsf{small}}$. For any b, k such that $2kc(b-1) < B$, Π_{ccsfold} is a reduction of knowledge from $(\mathcal{R}^b_{\mathsf{ccshom}})^{2k}$ to $\mathcal{R}^B_{\mathsf{ccshom}}$.*

Proof. The proof is similar to that for Theorem 3.2. We defer the proof to Appendix B.3. □

Parameters: $c \in \mathbb{N}$, $\mathcal{C} := \mathbb{Z}_q \subseteq \mathcal{R}_q$ and a strong sampling set $\mathcal{C}_{\mathsf{small}}$ with $\|\mathcal{C}_{\mathsf{small}}\|_{\mathsf{op}} \leq c$
Input: $\mathbb{x} := [\mathbb{x}_i := (\vec{\mathbf{r}}_i, \hat{\mathbf{v}}_i, y_i, \mathbf{u}_i, \mathbb{x}_{w,i})]_{i=1}^{2k}$ and $\mathbb{w} := [\mathbb{w}_i := (\vec{\mathbf{f}}_i, \vec{\mathbf{w}}_i)]_{i=1}^{2k}$
Output: $\mathbb{x}_o := (\vec{\mathbf{r}}_o, \hat{\mathbf{v}}_o, y_o, \mathbf{u}_o, \mathbb{x}_{w,o})$, $\mathbb{w}_o := (\vec{\mathbf{f}}_o, \vec{\mathbf{w}}_o)$
The protocol $\langle \mathsf{P}(\mathsf{pp}, \mathbb{x}; \mathbb{w}), \mathsf{V}(\mathsf{pp}, \mathbb{x}) \rangle$**:**

1. $\mathsf{V} \to \mathsf{P}$: V sends P $[\alpha_i, \mu_i, \zeta_i]_{i=1}^{2k} \xleftarrow{\$} (\mathcal{C} \times \mathcal{C} \times \mathcal{C})^{2k}$ and $\vec{\beta} \xleftarrow{\$} \mathcal{C}^{\log m}$.
2. $\mathsf{V} \leftrightarrow \mathsf{P}$: P and V run a sum-check protocol for the claim

$$\sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log m}} g(\vec{\mathbf{b}}) = \sum_{i=1}^{2k} (\alpha_i \hat{\mathbf{v}}_i + \zeta_i \mathbf{u}_i) \,.$$

Here the polynomial $g(\vec{\mathbf{x}}) \in \mathcal{R}_q^{\leq 2b}[X_1, \ldots, X_{\log m}]$ is defined as

$$g(\vec{\mathbf{x}}) := \sum_{i=1}^{2k} [\alpha_i g_{1,i}(\vec{\mathbf{x}}) + \mu_i g_{2,i}(\vec{\mathbf{x}}) + \zeta_i g_{3,i}(\vec{\mathbf{x}})] \,, \tag{38}$$

where for all $i \in [2k]$,

$$g_{1,i}(\vec{\mathbf{x}}) := eq(\vec{\mathbf{r}}_i, \vec{\mathbf{x}}) \cdot \mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{x}}) \,, \quad g_{2,i}(\vec{\mathbf{x}}) := eq(\vec{\beta}, \vec{\mathbf{x}}) \cdot \prod_{j=-(b-1)}^{b-1} \left(\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{x}}) - j\right) \,,$$

$$g_{3,i}(\vec{\mathbf{x}}) := eq(\vec{\mathbf{r}}_i, \vec{\mathbf{x}}) \cdot \left(\sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log(n+n_{\mathsf{in}})}} \mathsf{mle}\left[M_1\right](\vec{\mathbf{x}}, \vec{\mathbf{b}}) \cdot \mathsf{mle}\left[\mathbf{z}_i\right](\vec{\mathbf{b}}) \right) \,.$$

Here $\mathbf{z}_i := (\mathbb{x}_{w,i} || \vec{\mathbf{w}}_i)$ for all $i \in [2k]$. The protocol reduces to check the evaluation claim $g(\vec{\mathbf{r}}_o) \stackrel{?}{=} s$ where $\vec{\mathbf{r}}_o \xleftarrow{\$} \mathcal{C}^{\log m}$ is the sum-check challenge sampled by V.

3. $\mathsf{P} \to \mathsf{V}$: P sends V values $\left[\theta_i := \mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_o), \eta_i\right]_{i=1}^{2k}$, where for all $i \in [2k]$,

$$\eta_i := \sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log(n+n_{\mathsf{in}})}} \mathsf{mle}\left[M_1\right](\vec{\mathbf{r}}_o, \vec{\mathbf{b}}) \cdot \mathsf{mle}\left[\mathbf{z}_i\right](\vec{\mathbf{b}}) \,.$$

4. V computes $[\mathbf{e}_i := eq(\vec{\mathbf{r}}_i, \vec{\mathbf{r}}_o)]_{i=1}^{2k}$ and $\mathbf{e}^* := eq(\vec{\beta}, \vec{\mathbf{r}}_o)$ and checks that

$$s \stackrel{?}{=} \sum_{i=1}^{2k} \left[\alpha_i \mathbf{e}_i \theta_i + \mu_i \mathbf{e}^* \cdot \prod_{j=1-b}^{b-1} (\theta_i - j) + \zeta_i \mathbf{e}_i \eta_i \right] \,.$$

5. $\mathsf{V} \to \mathsf{P}$: V sends P random challenges $[\rho_i]_{i=1}^{2k} \xleftarrow{\$} \mathcal{C}_{\mathsf{small}}^{2k}$.
6. V output $\mathbb{x}_o := (\vec{\mathbf{r}}_o, \hat{\mathbf{v}}_o, y_o, \mathbf{u}_o, \mathbb{x}_{w,o})$ where $\mathsf{NTT}(\hat{\mathbf{v}}_o) = \sum_{i=1}^{2k} \mathsf{RotSum}(\rho_i, \mathsf{NTT}(\theta_i))$ and $[y_o, \mathbf{u}_o, \mathbb{x}_{w,o}] := \sum_{i=1}^{2k} \rho_i \cdot [y_i, \eta_i, \mathbb{x}_{w,i}]$.
7. P outputs $\vec{\mathbf{f}}_o = \sum_{i=1}^{2k} \rho_i \cdot \vec{\mathbf{f}}_i$ and $\vec{\mathbf{w}}_o := \mathcal{L}_w(\vec{\mathbf{f}}_o)[n_{\mathsf{in}} + 1, n_{\mathsf{in}} + n]$.

Figure 7: The protocol Π_{ccsfold} that reduces $(\mathcal{R}_{\mathsf{ccshom}}^b)^{2k}$ to $\mathcal{R}_{\mathsf{ccshom}}^B$. The $\mathcal{R}_q$-module $\mathcal{U}$ is set to $\mathcal{U} := \mathcal{R}_q$ for brevity.

4.3 An Optimized Folding Scheme for CCS

Recall that the folding scheme in Section 4.2 requires two sequential sumcheck executions. The first sumcheck is in the linearization protocol Π_{ccs} (Figure 5) that reduces $\mathcal{R}^{B}_{\mathsf{cmccs}}$ to $\mathcal{R}^{B}_{\mathsf{evalccs}}$; the second sumcheck is in the folding protocol Π_{ccsfold} (Figure 7) that reduces $(\mathcal{R}^{b}_{\mathsf{evalccs}})^{2k}$ to $\mathcal{R}^{B}_{\mathsf{evalccs}}$. Note that a decomposition protocol (Figure 6) for witness norm deduction is executed in the middle, thus it is unclear how to batch the two sumchecks into one given that the witnesses of the two sumchecks are quite different.

Fortunately, with a simple trick, we build a folding scheme for CCS that executes sumcheck only once. If the CCS gate degree deg and the range parameter b are set such that $\mathsf{deg} \approx 2b$, both the prover time and verifier complexity can be improved by a factor of two. Moreover, the prover saves the computation of an Ajtai commitment to the witness.

The core observation is that we can decompose the witness *before* running the linearization protocol. Recall that in the committed CCS relation $\mathcal{R}^{B}_{\mathsf{cmccs}}$ in Definition 4.2, the instance consists of a CCS public input $\mathbb{x}_{\mathbb{w}}$ and a commitment cm, and the witness is a pair of vectors $(\vec{\mathbf{f}}, \vec{\mathbf{w}})$ such that

- $(\mathbb{x}_{\mathsf{ccs}}, \vec{\mathbf{w}})$ is in the CCS relation,
- $\vec{\mathbf{f}}$ is an opening of cm with norm less than B, and
- $\vec{\mathbf{w}} = \mathbf{G} \times \vec{\mathbf{f}}$ where $\mathbf{G}$ is the gadget matrix.

Instead of transforming the CCS relation to $\mathcal{R}^{B}_{\mathsf{cmccs}}$, we consider a variant of $\mathcal{R}^{B}_{\mathsf{cmccs}}$ called *splitted committed CCS relations*. Set parameter $b, k \in \mathbb{N}$ such that $b^k = B$. The instance now consists of k vectors $[\mathbb{x}_{\mathbb{w},i}]_{i=1}^{k}$ and k commitments $[\mathsf{cm}_i]_{i=1}^{k}$, the witness is $([\vec{\mathbf{f}}_i]_{i=1}^{k}, \vec{\mathbf{w}})$ such that

- $(\mathbb{x}_{\mathsf{ccs}} := \sum_{i=1}^{k} b^{i-1} \cdot \mathbb{x}_{\mathbb{w},i}, \vec{\mathbf{w}})$ is in the CCS relation,
- For every $i \in [k]$, $\vec{\mathbf{f}}_i$ is an opening of cm with norm less than b, and
- $\vec{\mathbf{w}} = \mathbf{G} \times \vec{\mathbf{f}}$ where $\mathbf{G}$ is the gadget matrix and $\vec{\mathbf{f}} := \sum_{i=1}^{k} b^{i-1} \vec{\mathbf{f}}_i$.

We provide the formal definition below.

Definition 4.4 (Splitted committed CCS relation)**.** *Let* $\mathcal{R}_q := \mathbb{Z}_q[X]/(X^d + 1)$, $\mathsf{pp} := (\mathsf{pp}_{\mathsf{cm}}, \mathsf{pp}_{\mathsf{ccs}})$ *be the parameters defined in Definition 4.2 where* $\mathsf{pp}_{\mathsf{cm}} = (\kappa, m, B < q/2, \mathbf{A})$ *and* $\mathsf{pp}_{\mathsf{ccs}} = (n_r, n_c, t, n_s, \mathsf{deg}, \ell_{\mathsf{in}}, \mathbb{i})$*. Set* $\ell := m/n_c \in \mathbb{N}$ *where* $B^{\ell} \geq q/2$*. Let* $\mathbf{G} := \mathbf{I}_{n_c} \otimes [1, B, \ldots, B^{\ell-1}] \in \mathbb{Z}_q^{n_c \times m}$ *and set* $b, k \in \mathbb{N}$ *such that* $b^k = B$*. The indexed relation* $\mathcal{R}^{b,k}_{\mathsf{splitccs}}$ *is defined as*

$$\mathcal{R}^{b,k}_{\mathsf{splitccs}} := \left\{ \begin{array}{c} \left(\mathsf{pp}, \mathbb{x} := [\mathsf{cm}_i, \mathbb{x}_{\mathsf{ccs},i}]_{i=1}^{k}; \mathbb{w} := (\mathbb{w}_{\mathsf{ccs}}, [\vec{\mathbf{f}}_i]_{i=1}^{k})\right) \ s.t. \\ \forall i \in [k] \ : \ (\mathsf{pp}_{\mathsf{cm}}, \mathsf{cm}_i; \vec{\mathbf{f}}_i) \in \mathcal{R}^{b}_{\mathsf{cm}} \wedge \\ \mathbf{z}_{\mathsf{ccs}} := (\mathbb{x}_{\mathsf{ccs}} := \sum_{i=1}^{k} b^{i-1} \cdot \mathbb{x}_{\mathsf{ccs},i}, 1, \mathbb{w}_{\mathsf{ccs}}) = \mathbf{G} \times (\sum_{i=1}^{k} b^{i-1} \cdot \vec{\mathbf{f}}_i) \\ \wedge (\mathsf{pp}_{\mathsf{ccs}}, \mathbb{x}_{\mathsf{ccs}}; \mathbb{w}_{\mathsf{ccs}}) \in \mathcal{R}_{\mathsf{ccs}} \end{array} \right\}. \quad (39)$$

Set $b, k \in \mathbb{N}$ such that $b^k = B$. It is clear that if $([\mathbb{x}_{\mathbb{w},i}, \mathsf{cm}_i]_{i=1}^{k}, ([\vec{\mathbf{f}}_i]_{i=1}^{k}, \vec{\mathbf{w}}))$ is in the splitted committed CCS relation $\mathcal{R}^{b,k}_{\mathsf{splitccs}}$, then $([\mathbb{x}_{\mathbb{w}}, \mathsf{cm}], (\vec{\mathbf{f}}, \vec{\mathbf{w}}))$ is in relation $\mathcal{R}^{B}_{\mathsf{cmccs}}$, where

$[\mathbb{x}_{\mathsf{ccs}}, \mathsf{cm}] := \sum_{i=1}^{k} b^{i-1} \cdot [\mathbb{x}_{\mathbb{w},i}, \mathsf{cm}_i]$ and $\vec{\mathbf{f}} := \sum_{i=1}^{k} b^{i-1}\vec{\mathbf{f}}_i$. Hence, there is a straightforward reduction from $\mathcal{R}^{B}_{\mathsf{cmccs}}$ to $\mathcal{R}^{b,k}_{\mathsf{splitccs}}$. Therefore, to build a folding scheme for CCS, it suffices to set $\mathcal{R}_{\mathsf{comp}} := \mathcal{R}^{b,k}_{\mathsf{splitccs}}$ and $\mathcal{R}_{\mathsf{acc}} := \mathcal{R}^{B}_{\mathsf{evalccs}}$ and construct a reduction of knowledge from $\mathcal{R}_{\mathsf{comp}} \times \mathcal{R}_{\mathsf{acc}}$ to $\mathcal{R}_{\mathsf{acc}}$. The construction consists of two steps.

Step 1: Decomposition. Run protocol $\Pi^{*}_{\mathsf{ccsdec}}$ (Figure 6), to reduce $\mathcal{R}_{\mathsf{acc}} := \mathcal{R}^{B}_{\mathsf{evalccs}}$ to

$$(\mathcal{R}^{b}_{\mathsf{evalccs}})^{k} := \underbrace{\mathcal{R}^{b}_{\mathsf{evalccs}} \times \cdots \times \mathcal{R}^{b}_{\mathsf{evalccs}}}_{k} .$$

Step 2: Batch Folding. Reduce $\mathcal{R}^{b,k}_{\mathsf{splitccs}} \times (\mathcal{R}^{b}_{\mathsf{evalccs}})^{k}$ back to $\mathcal{R}^{B}_{\mathsf{evalccs}}$ by running the protocol Π_{batch} below.

4.3.1 Batch Folding: The reduction from $\mathcal{R}^{b,k}_{\mathsf{splitccs}} \times (\mathcal{R}^{b}_{\mathsf{evalccs}})^{k}$ to $\mathcal{R}^{B}_{\mathsf{evalccs}}$

Using the techniques from previous sections, we can perceive all of the following statements (underlying $\mathcal{R}^{b,k}_{\mathsf{splitccs}}$ and $\mathcal{R}^{b}_{\mathsf{evalccs}}$) as sumcheck statements:

- The multilinear evaluation statements underlying $(\mathcal{R}^{b}_{\mathsf{evalccs}})^{k}$;
- The linearized CCS statements (i.e. $\mathcal{R}_{\mathsf{lccs}}$ from Definition 4.3) underlying $(\mathcal{R}^{b}_{\mathsf{evalccs}})^{k}$;
- The range proof statements (with norm b) underlying $\mathcal{R}^{b,k}_{\mathsf{splitccs}}$ and $(\mathcal{R}^{b}_{\mathsf{evalccs}})^{k}$;
- The high-degree CCS gate-check (i.e. Definition 4.1) underlying $\mathcal{R}^{b,k}_{\mathsf{splitccs}}$.

Intuitively, the protocol Π_{batch} runs a sumcheck protocol to reduce all statements above into a folded statement in $\mathcal{R}^{B}_{\mathsf{evalccs}}$. We formally describe the protocol below. To make the notation consistent with the folding protocol Π_{ccsfold} (Figure 7), we denote $\mathcal{L}$ and $\mathcal{L}_w$ as the module homomorphisms $\mathcal{L}(\vec{\mathbf{f}}) := \mathbf{A}\vec{\mathbf{f}}$ and $\mathcal{L}_w(\vec{\mathbf{f}}) := \mathbf{G}\vec{\mathbf{f}}$ (where $\mathbf{G}$ is the gadget matrix), and we set $n_{\mathsf{in}} := \ell_{\mathsf{in}} + 1$ and $n := n_c$.

The protocol Π_{batch} that reduces $\mathcal{R}^{b,k}_{\mathsf{splitccs}} \times (\mathcal{R}^{b}_{\mathsf{evalccs}})^{k}$ to $\mathcal{R}^{B}_{\mathsf{evalccs}}$:

Parameters: $c \in \mathbb{N}$, strong sampling sets[4] $\mathcal{C} := \mathbb{Z}_q \subseteq \mathcal{R}_q$ and $\mathcal{C}_{\mathsf{small}}$ with $\|\mathcal{C}_{\mathsf{small}}\|_{\mathsf{op}} \leq c$

Input:

- $\mathbb{x} := \left([\mathbb{x}_i := (\vec{\mathbf{r}}_i, \hat{\mathbf{v}}_i, y_i, [\mathbf{u}_i^j]_{j=1}^{t}, \mathbb{x}_{w,i})]_{i=1}^{k}, \mathbb{x}' := [\mathbb{x}_{\mathsf{ccs},i}, y'_i]_{i=1}^{k}\right)$ and
- $\mathbb{w} := \left([\mathbb{w}_i := (\vec{\mathbf{f}}_i, \vec{\mathbf{w}}_i)]_{i=1}^{k}, \mathbb{w}' := (\mathbb{w}_{\mathsf{ccs}}, [\vec{\mathbf{f}}'_i]_{i=1}^{k})\right)$

Output: $\mathbb{x}_o := (\vec{\mathbf{r}}_o, \hat{\mathbf{v}}_o, y_o, [\mathbf{u}_o^j]_{j=1}^{t}, \mathbb{x}_{w,o})$, $\mathbb{w}_o := (\vec{\mathbf{f}}_o, \vec{\mathbf{w}}_o)$

The protocol $\langle \mathsf{P}(\mathsf{pp}, \mathbb{x}; \mathbb{w}), \mathsf{V}(\mathsf{pp}, \mathbb{x})\rangle$**:**

1. $\mathsf{V} \to \mathsf{P}$: V sends P

$$\vec{\beta} \xleftarrow{\$} \mathcal{C}^{\log m}, \quad \gamma \xleftarrow{\$} \mathcal{C}, \quad [\alpha_i, \mu_i, \mu'_i]_{i=1}^{k} \xleftarrow{\$} (\mathcal{C} \times \mathcal{C} \times \mathcal{C})^{k}, \quad [\zeta_i^j \xleftarrow{\$} \mathcal{C}]_{i\in[k], j\in[t]} .$$

[4] $\mathcal{C}$ can be any strong sampling sets, we set $\mathcal{C} := \mathbb{Z}_q$ for efficiency reasons.

2. $\mathsf{V} \leftrightarrow \mathsf{P} : \mathsf{P}$ and V run a sum-check protocol for the claim

$$\sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log m}} g(\vec{\mathbf{b}}) = \sum_{i=1}^{k} \alpha_i \hat{\mathbf{v}}_i + \sum_{i=1}^{k} \sum_{j=1}^{t} \zeta_i^j \mathbf{u}_i^j \,.$$

Let $d := \max(2b, \mathsf{deg} + 1)$. The polynomial $g(\vec{\mathbf{x}}) \in \mathcal{R}_q^{\leq d}[X_1, \ldots, X_{\log m}]$ is defined as

$$g(\vec{\mathbf{x}}) := \gamma g'_{\mathsf{ccs}}(\vec{\mathbf{x}}) + \sum_{i=1}^{k} \left[\alpha_i g^i_{\mathsf{eval}}(\vec{\mathbf{x}}) + \mu_i g^i_{\mathsf{rg}}(\vec{\mathbf{x}})\right] + \sum_{i=1}^{k} \sum_{j=1}^{t} \zeta_i^j g^{i,j}_{\mathsf{lccs}}(\vec{\mathbf{x}}) + \sum_{i=1}^{k} \mu'_i g'^{i}_{\mathsf{rg}}(\vec{\mathbf{x}}) \,. \tag{40}$$

Set $\mathbb{x}_{\mathsf{ccs}} := \sum_{i=1}^{k} b^{i-1} \mathbb{x}_{\mathsf{ccs},i}$ and let $\mathbf{z}_{\mathsf{ccs}} := (\mathbb{x}_{\mathsf{ccs}}, 1, \mathbb{w}_{\mathsf{ccs}})$. The polynomial g'_{ccs} is

$$g'_{\mathsf{ccs}}(\vec{\mathbf{x}}) := eq(\vec{\beta}, \vec{\mathbf{x}}) \cdot \left(\sum_{i=1}^{n_s} c_i \cdot \left[\prod_{j \in S_i} \left(\sum_{\vec{\mathbf{y}} \in \{0,1\}^{\log n + n_{\mathsf{in}}}} \mathsf{mle}\left[M_j\right](\vec{\mathbf{x}}, \vec{\mathbf{y}}) \cdot \mathsf{mle}\left[\mathbf{z}_{\mathsf{ccs}}\right](\vec{\mathbf{y}}) \right) \right] \right) \,. \tag{41}$$

For all $i \in [k]$,

$$\begin{aligned}
g^i_{\mathsf{eval}}(\vec{\mathbf{x}}) &:= eq(\vec{\mathbf{r}}_i, \vec{\mathbf{x}}) \cdot \mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{x}}) \,, \\
g^i_{\mathsf{rg}}(\vec{\mathbf{x}}) &:= eq(\vec{\beta}, \vec{\mathbf{x}}) \cdot \prod_{j=-(b-1)}^{b-1} \left(\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{x}}) - j \right) \,, \\
g'^{i}_{\mathsf{rg}}(\vec{\mathbf{x}}) &:= eq(\vec{\beta}, \vec{\mathbf{x}}) \cdot \prod_{j=-(b-1)}^{b-1} \left(\mathsf{mle}\left[\hat{\mathbf{f}}'_i\right](\vec{\mathbf{x}}) - j \right) \,.
\end{aligned}$$

For all $i \in [k], j \in [t]$, denote $\mathbf{z}_i := (\mathbb{x}_{w,i} || \vec{\mathbf{w}}_i) = \mathcal{L}_w(\vec{\mathbf{f}}_i)$,

$$g^{i,j}_{\mathsf{lccs}}(\vec{\mathbf{x}}) := eq(\vec{\mathbf{r}}_i, \vec{\mathbf{x}}) \cdot \left(\sum_{\vec{\mathbf{y}} \in \{0,1\}^{\log (n + n_{\mathsf{in}})}} \mathsf{mle}\left[M_j\right](\vec{\mathbf{x}}, \vec{\mathbf{y}}) \cdot \mathsf{mle}\left[\mathbf{z}_i\right](\vec{\mathbf{y}}) \right) \,.$$

The protocol reduces to check the evaluation claim $g(\vec{\mathbf{r}}_o) \stackrel{?}{=} s$ where $\vec{\mathbf{r}}_o \xleftarrow{\$} \mathcal{C}^{\log m}$ is the sum-check challenge sampled by V.

3. $\mathsf{P} \rightarrow \mathsf{V} : \mathsf{P}$ sends V values

$$\left[\theta_i := \mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_o);\ \theta'_i := \mathsf{mle}\left[\hat{\mathbf{f}}'_i\right](\vec{\mathbf{r}}_o)\right]_{i=1}^{k} \,;$$

For all $i \in [k]$, $j \in [t]$, denote by $\mathbf{z}_i := \mathcal{L}_w(\vec{\mathbf{f}}_i)$ and $\mathbf{z}'_i := \mathcal{L}_w(\vec{\mathbf{f}}'_i)$. P send V

$$\eta^{i,j} := \sum_{\vec{\mathbf{y}} \in \{0,1\}^{\log (n + n_{\mathsf{in}})}} \mathsf{mle}\left[M_j\right](\vec{\mathbf{r}}_o, \vec{\mathbf{y}}) \cdot \mathsf{mle}\left[\mathbf{z}_i\right](\vec{\mathbf{y}}) \,,$$

and

$$\eta_*^{i,j} := \sum_{\vec{\mathbf{y}} \in \{0,1\}^{\log(n+n_{\mathsf{in}})}} \mathsf{mle}\left[M_j\right](\vec{\mathbf{r}}_o, \vec{\mathbf{y}}) \cdot \mathsf{mle}\left[\mathbf{z}'_i\right](\vec{\mathbf{y}}).$$

4. V computes $[\mathbf{e}_i := eq(\vec{\mathbf{r}}_i, \vec{\mathbf{r}}_o)]_{i=1}^k$ and $\mathbf{e}^* := eq(\vec{\beta}, \vec{\mathbf{r}}_o)$ and checks that

$$\begin{aligned} s \stackrel{?}{=} & \sum_{i=1}^{k} \left[\alpha_i \mathbf{e}_i \theta_i + \mu_i \mathbf{e}^* \prod_{j=1-b}^{b-1} (\theta_i - j) \right] + \sum_{i=1}^{k} \sum_{j=1}^{t} \zeta_i^j \mathbf{e}_i \eta^{i,j} \\ & + \sum_{i=1}^{k} \mu'_i \mathbf{e}^* \prod_{j=1-b}^{b-1} (\theta'_i - j) + \gamma \mathbf{e}^* \left(\sum_{i=1}^{n_s} c_i \cdot \prod_{j \in S_i} \left(\sum_{\ell=1}^{k} \eta_*^{\ell,j} b^{\ell-1} \right) \right) \end{aligned}$$

5. $\mathsf{V} \to \mathsf{P}$: V sends P random challenges $[\rho_i]_{i=1}^k \xleftarrow{\$} \mathcal{C}_{\mathsf{small}}^k$ and $[\rho'_i]_{i=1}^k \xleftarrow{\$} \mathcal{C}_{\mathsf{small}}^k$.
6. V output $\mathbb{x}_o := (\vec{\mathbf{r}}_o, \hat{\mathbf{v}}_o, y_o, [\mathbf{u}_o^j]_{j=1}^t, \mathbb{x}_{w,o})$ where

$$\begin{aligned} \mathsf{NTT}(\hat{\mathbf{v}}_o) &= \sum_{i=1}^{k} \mathsf{RotSum}(\rho_i, \mathsf{NTT}(\theta_i)) + \sum_{i=1}^{k} \mathsf{RotSum}(\rho'_i, \mathsf{NTT}(\theta'_i)) \\ y_o &= \sum_{i=1}^{k} \rho_i \cdot y_i + \sum_{i=1}^{k} \rho'_i \cdot y'_i \\ \forall j \in [t] \ : \ u_o^j &= \sum_{i=1}^{k} \rho_i \eta^{i,j} + \sum_{i=1}^{k} \rho'_i \eta_*^{i,j} \\ \mathbb{x}_{w,o} &= \sum_{i=1}^{k} \rho_i \mathbb{x}_{w,i} + \rho'_1 \cdot [\mathbb{x}_{\mathsf{ccs},1}, 1] + \sum_{i=2}^{k} \rho'_i \cdot [\mathbb{x}_{\mathsf{ccs},i}, 0]. \end{aligned}$$

7. P outputs $\vec{\mathbf{f}}_o = \sum_{i=1}^k \rho_i \cdot \vec{\mathbf{f}}_i + \sum_{i=1}^k \rho'_i \cdot \vec{\mathbf{f}}'_i$ and $\vec{\mathbf{w}}_o := \mathcal{L}_w(\vec{\mathbf{f}}_o)[n_{\mathsf{in}} + 1, n_{\mathsf{in}} + n]$.

Lemma 4.3. *Let $\mathcal{R}_q := \mathbb{Z}_q[X]/(X^d+1) \cong \mathbb{Z}_q^d$, $\mathsf{pp} := (\mathsf{pp}_{\mathsf{cm}}, \mathsf{pp}_{\mathsf{ccs}})$ be the parameters defined in Definition 4.2 where $\mathsf{pp}_{\mathsf{cm}} = (\kappa, m, B < q/2, \mathbf{A})$ and $\mathsf{pp}_{\mathsf{ccs}} = (n_r = m, n_c, t, n_s, \mathsf{deg}, \ell_{\mathsf{in}}, \mathbb{i})$ such that $B^{m/n_c} \geq q/2$. Let $\mathcal{C}$, $\mathcal{C}_{\mathsf{small}}$ be strong sampling sets where $1/|\mathcal{C}|, 1/|\mathcal{C}_{\mathsf{small}}| = \mathsf{negl}(\lambda)$ and $\mathcal{C}_{\mathsf{small}}$ has expansion factor $T := \|\mathcal{C}_{\mathsf{small}}\|_{\mathsf{op}} \leq c$ (Definition 6). Assume that $\mathsf{MSIS}_{\kappa,m,8TB}^{\infty,q}$ is hard. For any b, k such that $2kc(b-1) < B$ and $b^k = B$, Π_{batch} is a reduction of knowledge from $\mathcal{R}_{\mathsf{splitccs}}^{b,k} \times (\mathcal{R}_{\mathsf{evalccs}}^b)^k$ to $\mathcal{R}_{\mathsf{evalccs}}^B$.*

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

By Lemma 4.3 and Lemma 4.2 and the knowledge composition theorem (Theorem 2.1), we obtain the theorem below.

Theorem 4.3. *Let* $\mathcal{R}_q$, $\mathcal{C}$, $\mathcal{C}_{\mathsf{small}}$, $\mathsf{pp}_{\mathsf{cm}} = (\kappa, m, B, \mathbf{A})$, $\mathsf{pp}_{\mathsf{ccs}} = (n_r, n_c, t, n_s, \mathsf{deg}, \ell_{\mathsf{in}}, \mathbb{i})$, *and* b, k *be defined as in Lemma 4.3.* $\Pi_{\mathsf{batch}} \circ \Pi_{\mathsf{ccsdec}}$ *is a reduction of knowledge from* $\mathcal{R}^{b,k}_{\mathsf{splitccs}} \times \mathcal{R}^{B}_{\mathsf{evalccs}}$ *to* $\mathcal{R}^{B}_{\mathsf{evalccs}}$.

Remark 4.5 (Supporting small prime modulus)**.** *The optimization in Section 3.3 can be used to extend Theorem 4.3 to support small modulus* q, *i.e., Theorem 4.3 still holds when* $\mathcal{R}_q \cong \mathbb{F}_{q^\tau}^{d/\tau}$ *for any* $\tau \in \mathbb{N}$ *that divides* d.

IVC proof compression. In the final step of IVC/PCD, the final IVC verifier needs to check witnesses for two statements, one in $\mathcal{R}_{\mathsf{comp}}$ and one in $\mathcal{R}_{\mathsf{acc}}$. A naive approach is letting the prover send the statement witnesses, and the verifier checks the statements in the clear. We can further improve the verifier complexity by generating another SNARK (e.g., Stark or LaBRADOR) that proves the correctness of the two final statements. Then the verifier only needs to check the SNARK proof.

However, in the optimized folding scheme for CCS, the statement in $\mathcal{R}_{\mathsf{comp}} := \mathcal{R}^{b,k}_{\mathsf{splitccs}}$ is more expensive to prove, as it involves checking the openings of k (rather than 1) commitments. Fortunately, we observe that in the last IVC step, there is no need to translate the online IVC statement into a *committed* CCS relation statement, because the *committed* CCS relation is only helpful when you need to *fold* the statement further. Instead, it is sufficient to translate the IVC statement as a CCS relation statement. The SNARK only needs to prove the IVC statement (plus the statement in the accumulated relation $\mathcal{R}_{\mathsf{acc}}$) without checking additional commitment openings inside the SNARK circuit.

Alternatively, one can also fold the last two statements, one in $\mathcal{R}_{\mathsf{comp}}$ and one in $\mathcal{R}_{\mathsf{acc}}$, into a statement in $\mathcal{R}_{\mathsf{acc}}$, so the SNARK only needs to prove a single statement in $\mathcal{R}_{\mathsf{acc}}$ (plus the folding verification logic). There is no need to prove any logic related to $\mathcal{R}_{\mathsf{comp}}$. Additionally, we can make the final SNARK proof zero knowledge, thereby hiding secret information.

5 Performance Estimates

We specify the complexity of the folding schemes in Table 1. For CCS relations, we consider the optimized folding scheme in Section 4.3. Recall that τ denote the extension field degree such that $\mathcal{R}_q \cong \mathbb{F}_{q^\tau}$, κ is the rank of the MSIS matrix. $(n_r, n_c, t, n_s, \mathsf{deg}, \ell_{\mathsf{in}})$ is the parameter where n_r, n_c are the number of rows and columns of the CCS matrices, t is the number of matrices in the CCS relation, n_s is the number of sub-gates per CCS constraint, and ℓ_{in} is the public input length.

Each instance consists of $\tau + \kappa + t + \ell_{\mathsf{in}} + 1$ $\mathcal{R}_q$-elements for the values $[\hat{\mathbf{v}}_i]_{i=1}^{\tau}$, $[\mathbf{u}_j]_{j=1}^{t}$, commitment $\mathsf{cm} \in \mathcal{R}_q^{\kappa}$ and public input $\mathbb{x}_w$; additionally it takes $\log m$ field elements for the challenge vector $\vec{\mathbf{r}}$ in the strong sampling set $\mathcal{C} \cong \mathbb{F}_{q^\tau}$.

Recall that b, k are the parameters such that b^k equals the norm bound B. The prover takes $O(mk(\kappa + t))$ $\mathcal{R}_q$ multiplications to compute the commitments and $\mathbf{u}$-values for the

decomposed witness (Figure 6); and it takes $O(mD\log^2 D)$ $\mathcal{R}_q$ multiplications to run the sum-check in protocol Π_{batch} (Section 4.3.1), where $D := \max(2b, \mathsf{deg})$.

We split the verifier into three phases:

- It takes $k \cdot (\kappa + \tau + t + \ell_{\mathsf{in}} + 1)$ $\mathcal{R}_q$ multiplications to check the correctness of decomposition in Figure 6;
- It takes $n_s \deg + k(\tau + t + 4b\tau)$ $\mathcal{R}_q$ multiplications to check the high-degree random evaluation claims and takes $2D\log m$ $\mathcal{R}_q$ multiplications[5] to run the sumcheck verifiers in Figure 5. Additionally, it takes $2(k+1)\log m$ field multiplications to compute the $k+1$ of eq values.[6]
- It $2k \cdot (\kappa + \tau + t + \ell_{\mathsf{in}} + 1)$ $\mathcal{R}_q$ multiplications to fold the decomposed instances in Π_{batch} (Section 4.3.1).
- Besides, the Fiat-Shamir transform takes $\approx 2k\kappa$ hashes to absorb the inputs and the decomposed commitments, and it takes $\approx 2k\log m$ hashes to compute the sumcheck verifier challenges.

To highlight its practicality, we consider the following example instantiation.

Relation	Ajtai	CCS
Instance	$\mathcal{R}_q$: $\tau + \kappa$ $\mathbb{F}_{q^\tau}$: $\log m$	$\mathcal{R}_q$: $\tau + \kappa + t + \ell_{\mathsf{in}} + 1$ $\mathbb{F}_{q^\tau}$: $\log m$
Prover	$\mathcal{R}_q$-mul : $O(mk\kappa) +$ $O(mb\log^2(b))$	$\mathcal{R}_q$-mul : $O(mk(\kappa + t)) +$ $O(mD\log^2(D))$
Verifier	$\mathcal{R}_q$-mul : $4k \cdot (\kappa + \tau)$ $4kb\tau + 4b\log m$ $\mathbb{F}$-mul : $(4k+2)\log m$ H : $\approx 2k\kappa + 2b\log m$	$\mathcal{R}_q$-mul : $3k \cdot (\kappa + \tau + t + \ell_{\mathsf{in}} + 1) +$ $n_s \deg + k(\tau + t + 4b\tau) +$ $2D\log m$ $\mathbb{F}$-mul : $(2k+2)\log m$ H : $\approx 2k\kappa + D\log m$

Table 1: The complexity of our folding schemes. $\mathcal{R}_q \cong \mathbb{F}_{q^\tau}^{d/\tau}$ for some $\tau, d \in \mathbb{N}$ where $\tau \mid d$. κ is the number of $\mathcal{R}_q$-elements in the Ajtai commitment; $(t, n_s, \ell_{\mathsf{in}}, \mathsf{deg})$ are the CCS parameters in Definition 4.1. Let B be the norm bound of the committed witness and b, k are the parameters such that $b^k = B$ and $D := \max(2b, \mathsf{deg})$. $\mathcal{R}_q$-mul and $\mathbb{F}$-mul denote multiplications over $\mathcal{R}_q \cong \mathbb{F}_{q^\tau}^{d/\tau}$ and $\mathbb{F}_{q^\tau}$ respectively; H denotes a hash from $\mathcal{R}_q^2$ to $\mathcal{R}_q$.

[5]Note that each univariate random evaluation takes $\approx 2D$ multiplication in $\mathcal{R}_q$, where D is the degree of the univariate polynomial.

[6]The sum-check over $\mathcal{R}_q$ can be understood as batching d/τ sum-checks over field $\mathbb{F}_{q^\tau}$; thus the verifier can simulate all sum-check operations over $\mathbb{F}_{q^\tau}$ without any NTT inversions.

Instantiation. We set $\kappa := 16$, $d := 64$ and use a 64-bit prime q such that $\mathcal{R}_q \cong \mathbb{F}_{q^4}^{d/4}$. Assign the last challenge set $\mathcal{C}_{\mathsf{small}}$ as the set of elements in $\mathcal{R}_q$ with coefficients -1, 0, 1 or 2. Since $d = 64$, we have that $|\mathcal{C}_{\mathsf{small}}| = 4^{64} = 2^{128}$. Moreover, $\mathcal{C}_{\mathsf{small}}$ is a strong sampling set, because the difference of any two distinct elements has infinite norm at most $3 < \frac{q^{1/16}}{\sqrt{16}} = 4$ and thus is invertible by Lemma 2.3. By Lemma 2.2, $\mathcal{C}_{\mathsf{small}}$ has expansion factor at most $2d = 128$. By the MSIS hardness bound in Section 2.2, we can achieve 128-bit security if

$$0.5 \cdot (\log m + \log d) + \log(8\|\mathcal{C}_{\mathsf{small}}\|_{\mathsf{op}} \cdot B) \approx 2\sqrt{\log_2(1.0045) d\kappa \log q}\,,$$

which holds if $\log B \leq 29 - 0.5 \cdot \log m$.

For simplicity, we set the CCS parameters $(t, n_s, \ell_{\mathsf{in}})$ where $\ell_{\mathsf{in}} = 0$, $t = n_s = 1$. Assume that the number of CCS constraints (over $\mathcal{R}_q$) in the IVC/PCD recursive circuit is $m \leq 2^{26}$. We set $B := 2^{16}$ so that $\log B \leq 29 - 0.5 \cdot \log m \leq 16$. We emphasize that by Remark 4.1, m constraints over $\mathcal{R}_q$ can be used to pack $md/4 = 16m$ constraints over $\mathbb{F}_{q^4}$, so an upper bound $m \leq 2^{26}$ leads to an upper bound 2^{30} on the number of constraints over $\mathbb{F}_{q^4}$, which is more than enough.

Let's see the concrete instance size, prover cost and verifier circuit size. We set $b := 2$ and $k := 16$ so that $b^k = 2^{16} = B$ and $2kc(b-1) = 2 \cdot 16 \cdot 128 \cdot 1 < B = 2^{16}$ as required in Theorem 4.2.

The instance size is $\tau + \kappa + t + \ell_{\mathsf{in}} + 1 = 22$ $\mathcal{R}_q$-elements and $\log m$ $\mathbb{F}_{q^\tau}$-elements. E.g., for $m := 2^{26}$, this is $\approx$ 12KB. The recursive folding verifier takes

$$|\mathsf{V}| \approx (1648 + (2D+2) \cdot \log m) \cdot |\mathcal{R}_q| + (2k\kappa + D \log m) \cdot |\mathsf{H}|$$

CCS constraints. Here $|\mathcal{R}_q|$ denotes the number of constraints for a single $\mathcal{R}_q$ multiplication, $|\mathsf{H}|$ denotes the number of constraints for simulating a two-to-one hash. Note that $|\mathcal{R}_q| \leq 1$ as it takes at most one contraint to simulate an $\mathcal{R}_q$ multiplication; by [Bou+23], we can set $|\mathsf{H}| \leq 100$. The number of constraints is dominated by the Fiat-Shamir circuit, which takes $\approx$ 50K constraints.

Remark 5.1. *After fixing the parameter b, instead of setting k such that $b^k = B$, we can set k^* as the minimal integer such that $b^{k^*} \geq 2k^*(b-1)\|\mathcal{C}_{\mathsf{small}}\|_{\mathsf{op}}$ where $\|\mathcal{C}_{\mathsf{small}}\|_{\mathsf{op}}$ is the expansion factor of $\mathcal{C}_{\mathsf{small}}$. In each folding step, we decompose the folded witness to $k^* \leq k$ parts, leading to an efficiency improvement. This works because after each folding step, the norm of the folded witness is always less than $2k^*(b-1)\|\mathcal{C}_{\mathsf{small}}\|_{\mathsf{op}} \leq b^{k^*}$. This also indicates that we can use different decomposition factors k^*, k for relations $\mathcal{R}_{\mathsf{acc}}$ and $\mathcal{R}_{\mathsf{comp}}$.*

Remark 5.2 (Supporting non-power-of-two cyclotomic rings)**.** *Our current analysis requires $\mathcal{R} := \mathbb{Z}[X]/(X^d + 1)$ to be a cyclotomic ring where d is a power-of-two. We conjecture that the scheme remains secure when using a non-power-of-two cyclotomic ring and leave the formal analysis for future work.*

6 Discussion of an Alternative Approach

In this section we discuss another technique for proving a norm bound on a committed vector, called *random projection*. We explain why this approach, which may at first seem appealing, does not seem to work in the context of folding.

Suppose we aim to fold $2k$ instance-witness pairs $[\mathbb{x}_i, \mathbb{w}_i]_{i=1}^{2k}$ of an Ajtai commitment-based relation (e.g., $\mathcal{R}_{\mathsf{eval}}^{b}$ from Eq. (9)) to a single instance-witness pair $(\mathbb{x}_o, \mathbb{w}_o)$ in a related relation (e.g., $\mathcal{R}_{\mathsf{eval}}^{B}$ where $B > b$). To extract knowledge of $[\mathbb{w}_i]_{i=1}^{2k}$ (with $\|\mathbb{w}_i\|_\infty < b$ for all $i \in [2k]$) from a folding prover P^* that outputs correct witness in $\mathcal{R}_{\mathsf{eval}}^{B}$, the most naive approach is to have the prover directly transmit $[\mathbb{w}_i]_{i=1}^{2k}$ and let the verifier check that $[\mathbb{w}_i]_{i=1}^{2k}$ have small norms and are consistent with the folded instance. This certainly does not work as the verifier has complexity linear to the witness size while constructing a IVC/PCD requires folding verifiers to be sublinear. Alternatively, the prover could generate a proof demonstrating that each element of $[\mathbb{w}_i]_{i=1}^{2k}$ has a small norm, but this method is excessively costly due to the requirement for $\Theta(m)$ range-check circuits, where m is the witness length.

To circumvent these challenges, a natural idea is to leverage the random projection technique from LaBRADOR [BS23]: The verifier samples and sends a random matrix $\Pi \in \mathbb{Z}_q^{\lambda d \times md}$ with small norms, where λ is the security parameter and m is the size of $\vec{\mathbf{w}} := [\mathbb{w}_i]_{i=1}^{2k}$. Subsequently, the prover sends $\vec{\mathbf{v}} := \Pi\vec{\mathbf{w}} \in \mathbb{Z}_q^{\lambda d}$ and the verifier checks that $\|\vec{\mathbf{v}}\|_\infty$ is small and $\vec{\mathbf{v}}$ is computed honestly. Notably, the size of $\vec{\mathbf{v}}$ is independent of the witness size. Additionally, if $\|\vec{\mathbf{v}}\|_\infty$ has a small norm, by the Johnson-Lindenstrauss Lemma [WL84; GHL22; BS23], the original witnesses also have small norms with high probability (over the random choice of Π). Nonetheless, several challenges arise in the context of folding schemes.

First, the size of the matrix Π is large, making it impractical for the verifier to generate Π itself. A potential solution involves having the verifier generate a short random seed s, which the prover then uses to generate Π and subsequently proves the correctness of Π's generation. However, this approach introduces significant complexity in terms of circuit size, as simulating PRG computations in circuits for a large output is prohibitively expensive.

Second, how does the verifier check that $\vec{\mathbf{v}}$ was computed honestly? It's impractical for the verifier to directly receive $\vec{\mathbf{w}}$ and verify its correctness, as this would result in a linear-sized verifier. An alternative approach could be to have the prover generate another instance-witness pair $(\mathbb{x}', \mathbb{w}')$ for proving $\vec{\mathbf{v}} = \Pi\vec{\mathbf{w}}$ and then fold it together with the original instances. However, this leads to a chicken-and-egg problem: how do we check that the committed witness in $\mathbb{w}'$ has a small norm? The most viable solution appears to involve the prover computing a post-quantum secure SNARK π for proving that $\vec{\mathbf{v}} = \Pi\vec{\mathbf{w}}$ and $\Pi = \mathsf{PRG}(s)$ (where s is the short random seed), with the verifier subsequently verifying the correctness of π. Concretely, this solution is inefficient due to the high complexity of the SNARK verifier circuit. Furthermore, since we can already construct IVC/PCD directly

from SNARKs [BCCT13; BCTV14], employing folding in conjunction with SNARKs serves little purpose.

Even if we manage to overcome the aforementioned challenges, we still encounter an inherent obstacle: the random projection idea cannot guarantee *perfect completeness*: Even if the prover is honest and provides input witnesses with genuinely small norms, there remains a small probability that the random projection $\vec{\mathbf{v}} = \Pi\vec{\mathbf{w}}$ would yield large norms, resulting in rejection by the verifier. We note that perfect completeness is essential for constructing an IVC/PCD [Bün+21], and it appears inherently difficult to construct a folding scheme that achieves perfect completeness using the random projection approach.

7 Conclusion, open problems, and future work

We presented LatticeFold, the first lattice folding scheme based on the Module SIS problem. Our folding protocol ensures that the witnesses extracted from a folded statement always satisfy the required norm bounds. This is done by requiring the prover to prove that its starting witnesses are all low norm. This proof is done efficiently using the sumcheck protocol.

There are many directions for future work. First, it is not difficult to extend the scheme to support the Lasso [STW23b; ST23] lookup argument. This is because the sumcheck used by Lasso is compatible with the sumchecks in LatticeFold. Second, it remains to implement LatticeFold and experiment with its real-world performance. We estimate that LatticeFold is competitive with the best pre-quantum folding schemes. It is likely to be the most performant folding system for computations using high-degree CCS. LatticeFold could be an example where (plausible) post-quantum security leads to better performance.

Finally, it would be interesting to explore the performance of LatticeFold using other lattice-based additively homomorphic commitments schemes, for example, ones based on SIS rather MSIS. In addition, recall that much of the work in the design of LatticeFold is due to the fact that the Ajtai commitment scheme [Ajt96] is binding only when the committed vector is low norm. If we had a lattice-based linearly homomorphic commitment that was binding for arbitrary vectors, irrespective of their norm, then one could more directly use that commitment scheme in the Hypernova [KS23b] or Protostar [BC23] systems. Such commitment schemes exist (e.g., [Bau+18b]), however they are not succinct: the commitment string is quite long. Using them would result in a SNARK with poor prover performance and long proofs. Designing a *succinct* lattice-based linearly homomorphic commitment scheme for committing to vectors of arbitrary norm is an interesting area for further research.

Acknowledgments. We thank the anonymous CRYPTO reviewers and Wilson Nguyen for their valuable feedback, and Srinath Setty for bringing up the question of batching all Sumchecks. This work was funded by NSF, DARPA, the Simons Foundation, UBRI, and

NTT Research. Opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of DARPA.

References

[ACK21] Thomas Attema, Ronald Cramer, and Lisa Kohl. "A Compressed Σ-Protocol Theory for Lattices". In: *CRYPTO 2021, Part II*. Ed. by Tal Malkin and Chris Peikert. Vol. 12826. LNCS. Virtual Event: Springer, Cham, Aug. 2021, pp. 549–579. DOI: 10.1007/978-3-030-84245-1_19.

[AF22] Thomas Attema and Serge Fehr. "Parallel Repetition of $(k_1, \ldots, k_\mu)$-Special-Sound Multi-round Interactive Proofs". In: *CRYPTO 2022, Part I*. Ed. by Yevgeniy Dodis and Thomas Shrimpton. Vol. 13507. LNCS. Springer, Cham, Aug. 2022, pp. 415–443. DOI: 10.1007/978-3-031-15802-5_15.

[AHIV17] Scott Ames, Carmit Hazay, Yuval Ishai, and Muthuramakrishnan Venkitasubramaniam. "Ligero: Lightweight Sublinear Arguments Without a Trusted Setup". In: *ACM CCS 2017*. Ed. by Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu. ACM Press, 2017, pp. 2087–2104. DOI: 10.1145/3133956.3134104.

[Ajt96] Miklós Ajtai. "Generating Hard Instances of Lattice Problems (Extended Abstract)". In: *28th ACM STOC*. ACM Press, May 1996, pp. 99–108. DOI: 10.1145/237814.237838.

[AL21] Martin R. Albrecht and Russell W. F. Lai. "Subtractive Sets over Cyclotomic Rings - Limits of Schnorr-Like Arguments over Lattices". In: *CRYPTO 2021, Part II*. Ed. by Tal Malkin and Chris Peikert. Vol. 12826. LNCS. Virtual Event: Springer, Cham, Aug. 2021, pp. 519–548. DOI: 10.1007/978-3-030-84245-1_18.

[Alb+22] Martin R. Albrecht, Valerio Cini, Russell W. F. Lai, Giulio Malavolta, and Sri Aravinda Krishnan Thyagarajan. "Lattice-Based SNARKs: Publicly Verifiable, Preprocessing, and Recursively Composable - (Extended Abstract)". In: *CRYPTO 2022, Part II*. Ed. by Yevgeniy Dodis and Thomas Shrimpton. Vol. 13508. LNCS. Springer, Cham, Aug. 2022, pp. 102–132. DOI: 10.1007/978-3-031-15979-4_4.

[ALS20] Thomas Attema, Vadim Lyubashevsky, and Gregor Seiler. "Practical Product Proofs for Lattice Commitments". In: *CRYPTO 2020, Part II*. Ed. by Daniele Micciancio and Thomas Ristenpart. Vol. 12171. LNCS. Springer, Cham, Aug. 2020, pp. 470–499. DOI: 10.1007/978-3-030-56880-1_17.

[APS15] Martin R Albrecht, Rachel Player, and Sam Scott. "On the concrete hardness of learning with errors". In: *Journal of Mathematical Cryptology* 9.3 (2015), pp. 169–203.

[Bau+18a] Carsten Baum, Jonathan Bootle, Andrea Cerulli, Rafaël del Pino, Jens Groth, and Vadim Lyubashevsky. "Sub-linear Lattice-Based Zero-Knowledge Arguments for Arithmetic Circuits". In: *CRYPTO 2018, Part II*. Ed. by Hovav Shacham and Alexandra Boldyreva. Vol. 10992. LNCS. Springer, Cham, Aug. 2018, pp. 669–699. DOI: 10.1007/978-3-319-96881-0_23.

[Bau+18b] Carsten Baum, Ivan Damgård, Vadim Lyubashevsky, Sabine Oechsner, and Chris Peikert. "More Efficient Commitments from Structured Lattice Assumptions". In: *SCN 18*. Ed. by Dario Catalano and Roberto De Prisco. Vol. 11035. LNCS. Springer, Cham, Sept. 2018, pp. 368–385. DOI: 10.1007/978-3-319-98113-0_20.

[Bau+23] Carsten Baum, Lennart Braun, Cyprien Delpech de Saint Guilhem, Michael Klooß, Emmanuela Orsini, Lawrence Roy, and Peter Scholl. "Publicly Verifiable Zero-Knowledge and Post-Quantum Signatures from VOLE-in-the-Head". In: *CRYPTO 2023, Part V*. Ed. by Helena Handschuh and Anna Lysyanskaya. Vol. 14085. LNCS. Springer, Cham, Aug. 2023, pp. 581–615. DOI: 10.1007/978-3-031-38554-4_19.

[BBBF18] Dan Boneh, Joseph Bonneau, Benedikt Bünz, and Ben Fisch. "Verifiable Delay Functions". In: *CRYPTO 2018, Part I*. Ed. by Hovav Shacham and Alexandra Boldyreva. Vol. 10991. LNCS. Springer, Cham, Aug. 2018, pp. 757–788. DOI: 10.1007/978-3-319-96884-1_25.

[BBHR18a] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. "Fast Reed-Solomon Interactive Oracle Proofs of Proximity". In: *ICALP 2018*. Ed. by Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella. Vol. 107. LIPIcs. Schloss Dagstuhl, July 2018, 14:1–14:17. DOI: 10.4230/LIPIcs.ICALP.2018.14.

[BBHR18b] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. *Scalable, transparent, and post-quantum secure computational integrity*. Cryptology ePrint Archive, Report 2018/046. 2018. URL: https://eprint.iacr.org/2018/046.

[BC23] Benedikt Bünz and Binyi Chen. "Protostar: Generic efficient accumulation/folding for special sound protocols". In: *Cryptology ePrint Archive* (2023).

[BC25] Dan Boneh and Binyi Chen. *LatticeFold+: Faster, Simpler, Shorter Lattice-Based Folding for Succinct Proof Systems*. Cryptology ePrint Archive, Report 2025/247. 2025. URL: https://eprint.iacr.org/2025/247.

[BCCT13] Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. "Recursive composition and bootstrapping for SNARKS and proof-carrying data". In: *45th ACM STOC*. Ed. by Dan Boneh, Tim Roughgarden, and Joan Feigenbaum. ACM Press, June 2013, pp. 111–120. DOI: 10.1145/2488608.2488623.

[BCMS20] Benedikt Bünz, Alessandro Chiesa, Pratyush Mishra, and Nicholas Spooner. "Recursive Proof Composition from Accumulation Schemes". In: *TCC 2020, Part II*. Ed. by Rafael Pass and Krzysztof Pietrzak. Vol. 12551. LNCS. Springer, Cham, Nov. 2020, pp. 1–18. DOI: 10.1007/978-3-030-64378-2_1.

[BCPS18] Anurag Bishnoi, Pete L Clark, Aditya Potukuchi, and John R Schmitt. "On zeros of a polynomial in a finite grid". In: *Combinatorics, Probability and Computing* 27.3 (2018), pp. 310–333.

[BCS23] Jonathan Bootle, Alessandro Chiesa, and Katerina Sotiraki. "Lattice-Based Succinct Arguments for NP with Polylogarithmic-Time Verification". In: *CRYPTO 2023, Part II*. Ed. by Helena Handschuh and Anna Lysyanskaya. Vol. 14082. LNCS. Springer, Cham, Aug. 2023, pp. 227–251. DOI: 10.1007/978-3-031-38545-2_8.

[BCTV14] Eli Ben-Sasson, Alessandro Chiesa, Eran Tromer, and Madars Virza. "Scalable Zero Knowledge via Cycles of Elliptic Curves". In: *CRYPTO 2014, Part II*. Ed. by Juan A. Garay and Rosario Gennaro. Vol. 8617. LNCS. Springer, Berlin, Heidelberg, Aug. 2014, pp. 276–294. DOI: 10.1007/978-3-662-44381-1_16.

[BDFG21] Dan Boneh, Justin Drake, Ben Fisch, and Ariel Gabizon. "Halo Infinite: Proof-Carrying Data from Additive Polynomial Commitments". In: *CRYPTO 2021, Part I*. Ed. by Tal Malkin and Chris Peikert. Vol. 12825. LNCS. Virtual Event: Springer, Cham, Aug. 2021, pp. 649–680. DOI: 10.1007/978-3-030-84242-0_23.

[Ben+19] Eli Ben-Sasson, Alessandro Chiesa, Michael Riabzev, Nicholas Spooner, Madars Virza, and Nicholas P. Ward. "Aurora: Transparent Succinct Arguments for R1CS". In: *EUROCRYPT 2019, Part I*. Ed. by Yuval Ishai and Vincent Rijmen. Vol. 11476. LNCS. Springer, Cham, May 2019, pp. 103–128. DOI: 10.1007/978-3-030-17653-2_4.

[BGH19] Sean Bowe, Jack Grigg, and Daira Hopwood. *Halo: Recursive Proof Composition without a Trusted Setup*. Cryptology ePrint Archive, Report 2019/1021. 2019. URL: https://eprint.iacr.org/2019/1021.

[BLNS20] Jonathan Bootle, Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Gregor Seiler. "A Non-PCP Approach to Succinct Quantum-Safe Zero-Knowledge". In: *CRYPTO 2020, Part II*. Ed. by Daniele Micciancio and Thomas Ristenpart. Vol. 12171. LNCS. Springer, Cham, Aug. 2020, pp. 441–469. DOI: 10.1007/978-3-030-56880-1_16.

[BLS19] Jonathan Bootle, Vadim Lyubashevsky, and Gregor Seiler. "Algebraic Techniques for Short(er) Exact Lattice-Based Zero-Knowledge Proofs". In: *CRYPTO 2019, Part I*. Ed. by Alexandra Boldyreva and Daniele Micciancio. Vol. 11692. LNCS. Springer, Cham, Aug. 2019, pp. 176–202. DOI: 10.1007/978-3-030-26948-7_7.

[BMNW24a] Benedikt Bünz, Pratyush Mishra, Wilson Nguyen, and William Wang. *Accumulation without Homomorphism*. Cryptology ePrint Archive, Report 2024/474. 2024. URL: https://eprint.iacr.org/2024/474.

[BMNW24b] Benedikt Bünz, Pratyush Mishra, Wilson Nguyen, and William Wang. *Arc: Accumulation for Reed–Solomon Codes*. Cryptology ePrint Archive, Report 2024/1731. 2024. URL: https://eprint.iacr.org/2024/1731.

[Bou+23] Clémence Bouvier, Pierre Briaud, Pyrros Chaidos, Léo Perrin, Robin Salen, Vesselin Velichkov, and Danny Willems. "New Design Techniques for Efficient Arithmetization-Oriented Hash Functions: Anemoi Permutations and Jive Compression Mode". In: *CRYPTO 2023, Part III*. Ed. by Helena Handschuh and Anna Lysyanskaya. Vol. 14083. LNCS. Springer, Cham, Aug. 2023, pp. 507–539. DOI: 10.1007/978-3-031-38548-3_17.

[Bre+24] Martijn Brehm, Binyi Chen, Ben Fisch, Nicolas Resch, Ron D. Rothblum, and Hadas Zeilberger. *Blaze: Fast SNARKs from Interleaved RAA Codes*. Cryptology ePrint Archive, Report 2024/1609. 2024. URL: https://eprint.iacr.org/2024/1609.

[BS23] Ward Beullens and Gregor Seiler. "LaBRADOR: Compact Proofs for R1CS from Module-SIS". In: *CRYPTO 2023, Part V*. Ed. by Helena Handschuh and Anna Lysyanskaya. Vol. 14085. LNCS. Springer, Cham, Aug. 2023, pp. 518–548. DOI: 10.1007/978-3-031-38554-4_17.

[Bün+18] Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Greg Maxwell. "Bulletproofs: Short Proofs for Confidential Transactions and More". In: *2018 IEEE Symposium on Security and Privacy*. IEEE Computer Society Press, May 2018, pp. 315–334. DOI: 10.1109/SP.2018.00020.

[Bün+21] Benedikt Bünz, Alessandro Chiesa, William Lin, Pratyush Mishra, and Nicholas Spooner. "Proof-Carrying Data Without Succinct Arguments". In: *CRYPTO 2021, Part I*. Ed. by Tal Malkin and Chris Peikert. Vol. 12825. LNCS. Virtual Event: Springer, Cham, Aug. 2021, pp. 681–710. DOI: 10.1007/978-3-030-84242-0_24.

[CCKP19] Shuo Chen, Jung Hee Cheon, Dongwoo Kim, and Daejun Park. *Verifiable Computing for Approximate Computation*. Cryptology ePrint Archive, Report 2019/762. 2019. URL: https://eprint.iacr.org/2019/762.

[CMNW24] Valerio Cini, Giulio Malavolta, Ngoc Khanh Nguyen, and Hoeteck Wee. *Polynomial Commitments from Lattices: Post-Quantum Security, Fast Verification and Transparent Setup*. Cryptology ePrint Archive, Paper 2024/281. https://eprint.iacr.org/2024/281. 2024. URL: https://eprint.iacr.org/2024/281.

[CT10] Alessandro Chiesa and Eran Tromer. "Proof-Carrying Data and Hearsay Arguments from Signature Cards". In: *ICS 2010*. Ed. by Andrew Chi-Chih Yao. Tsinghua University Press, Jan. 2010, pp. 310–331.

[DB22] Trisha Datta and Dan Boneh. *Using ZK Proofs to Fight Disinformation*. link. 2022.

[EG23] Liam Eagen and Ariel Gabizon. *ProtoGalaxy: Efficient ProtoStar-style folding of multiple instances*. Cryptology ePrint Archive, Paper 2023/1106. https://eprint.iacr.org/2023/1106. 2023. URL: https://eprint.iacr.org/2023/1106.

[ENS20] Muhammed F. Esgin, Ngoc Khanh Nguyen, and Gregor Seiler. "Practical Exact Proofs from Lattices: New Techniques to Exploit Fully-Splitting Rings". In: *ASIACRYPT 2020, Part II*. Ed. by Shiho Moriai and Huaxiong Wang. Vol. 12492. LNCS. Springer, Cham, Dec. 2020, pp. 259–288. DOI: 10.1007/978-3-030-64834-3_9.

[Esg+19] Muhammed F. Esgin, Ron Steinfeld, Amin Sakzad, Joseph K. Liu, and Dongxi Liu. "Short Lattice-Based One-out-of-Many Proofs and Applications to Ring Signatures". In: *ACNS 19International Conference on Applied Cryptography and Network Security*. Ed. by Robert H. Deng, Valérie Gauthier-Umaña, Martín Ochoa, and Moti Yung. Vol. 11464. LNCS. Springer, Cham, June 2019, pp. 67–88. DOI: 10.1007/978-3-030-21568-2_4.

[FKNP24a] Giacomo Fenzi, Christian Knabenhans, Ngoc Khanh Nguyen, and Duc Tu Pham. *Lova: Lattice-Based Folding Scheme from Unstructured Lattices*. Cryptology ePrint Archive, Report 2024/1964. 2024. URL: https://eprint.iacr.org/2024/1964.

[FKNP24b] Giacomo Fenzi, Christian Knabenhans, Ngoc Khanh Nguyen, and Duc Tu Pham. "Lova: Lattice-Based Folding Scheme from Unstructured Lattices". In: *ASIACRYPT 2024, Part IV*. Ed. by Kai-Min Chung and Yu Sasaki. Vol. 15487. LNCS. Springer, Singapore, Dec. 2024, pp. 303–326. DOI: 10.1007/978-981-96-0894-2_10.

[FMN23] Giacomo Fenzi, Hossein Moghaddas, and Ngoc Khanh Nguyen. *Lattice-Based Polynomial Commitments: Towards Asymptotic and Concrete Efficiency*. Cryptology ePrint Archive, Paper 2023/846. https://eprint.iacr.org/2023/846. 2023. URL: https://eprint.iacr.org/2023/846.

[Gar24] Albert Garreta. *Implementing LatticeFold: Advancing Post-Quantum Folding*. ZKProof 7. link. 2024.

[GHL22] Craig Gentry, Shai Halevi, and Vadim Lyubashevsky. "Practical Non-interactive Publicly Verifiable Secret Sharing with Thousands of Parties". In: *EUROCRYPT 2022, Part I*. Ed. by Orr Dunkelman and Stefan Dziembowski. Vol. 13275. LNCS. Springer, Cham, 2022, pp. 458–487. DOI: 10.1007/978-3-031-06944-4_16.

[Gol+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 2023, Part II*. Ed. by Helena Handschuh and Anna Lysyanskaya. Vol. 14082. LNCS. Springer, Cham, Aug. 2023, pp. 193–226. DOI: 10.1007/978-3-031-38545-2_7.

[KHSS22] Daniel Kang, Tatsunori Hashimoto, Ion Stoica, and Yi Sun. *ZK-IMG: Attested Images via Zero-Knowledge Proofs to Fight Disinformation*. 2022. eprint: 2211.04775.

[KLNO24a] Michael Klooß, Russell W. F. Lai, Ngoc Khanh Nguyen, and Michał Osadnik. *RoK, Paper, SISsors – Toolkit for Lattice-based Succinct Arguments*. Cryptology ePrint Archive, Report 2024/1972. 2024. URL: https://eprint.iacr.org/2024/1972.

[KLNO24b] Michael Klooß, Russell W. F. Lai, Ngoc Khanh Nguyen, and Michal Osadnik. "RoK, Paper, SISsors Toolkit for Lattice-Based Succinct Arguments - (Extended Abstract)". In: *ASIACRYPT 2024, Part V*. Ed. by Kai-Min Chung and Yu Sasaki. Vol. 15488. LNCS. Springer, Singapore, Dec. 2024, pp. 203–235. DOI: 10.1007/978-981-96-0935-2_7.

[KP23] Abhiram Kothapalli and Bryan Parno. "Algebraic Reductions of Knowledge". In: *CRYPTO 2023, Part IV*. Ed. by Helena Handschuh and Anna Lysyanskaya. Vol. 14084. LNCS. Springer, Cham, Aug. 2023, pp. 669–701. DOI: 10.1007/978-3-031-38551-3_21.

[KS22] Abhiram Kothapalli and Srinath Setty. *SuperNova: Proving universal machine executions without universal circuits*. Cryptology ePrint Archive, Report 2022/1758. 2022. URL: https://eprint.iacr.org/2022/1758.

[KS23a] Abhiram Kothapalli and Srinath Setty. "CycleFold: Folding-scheme-based recursive arguments over a cycle of elliptic curves". In: *Cryptology ePrint Archive* (2023).

[KS23b] Abhiram Kothapalli and Srinath Setty. "HyperNova: Recursive arguments for customizable constraint systems". In: *Cryptology ePrint Archive* (2023).

[KST22] Abhiram Kothapalli, Srinath Setty, and Ioanna Tzialla. "Nova: Recursive Zero-Knowledge Arguments from Folding Schemes". In: *CRYPTO 2022, Part IV*. Ed. by Yevgeniy Dodis and Thomas Shrimpton. Vol. 13510. LNCS. Springer, Cham, Aug. 2022, pp. 359–388. DOI: 10.1007/978-3-031-15985-5_13.

[LFKN92] Carsten Lund, Lance Fortnow, Howard Karloff, and Noam Nisan. "Algebraic methods for interactive proof systems". In: *Journal of the ACM (JACM)* 39.4 (1992), pp. 859–868.

[LM06] Vadim Lyubashevsky and Daniele Micciancio. "Generalized Compact Knapsacks Are Collision Resistant". In: *ICALP 2006, Part II*. Ed. by Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener. Vol. 4052. LNCS. Springer, Berlin, Heidelberg, July 2006, pp. 144–155. DOI: 10.1007/11787006_13.

[LNP22] Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Maxime Plançon. "Lattice-Based Zero-Knowledge Proofs and Applications: Shorter, Simpler, and More General". In: *CRYPTO 2022, Part II*. Ed. by Yevgeniy Dodis and Thomas Shrimpton. Vol. 13508. LNCS. Springer, Cham, Aug. 2022, pp. 71–101. DOI: 10.1007/978-3-031-15979-4_3.

[LPS24] Helger Lipmaa, Roberto Parisella, and Janno Siim. "Constant-Size zk-SNARKs in ROM from Falsifiable Assumptions". In: *Cryptology ePrint Archive* (2024).

[LS15] Adeline Langlois and Damien Stehlé. "Worst-case to average-case reductions for module lattices". In: *DCC* 75.3 (2015), pp. 565–599. DOI: 10.1007/s10623-014-9938-4.

[LS18] Vadim Lyubashevsky and Gregor Seiler. "Short, Invertible Elements in Partially Splitting Cyclotomic Rings and Applications to Lattice-Based Zero-Knowledge Proofs". In: *EUROCRYPT 2018, Part I*. Ed. by Jesper Buus Nielsen and Vincent Rijmen. Vol. 10820. LNCS. Springer, Cham, 2018, pp. 204–224. DOI: 10.1007/978-3-319-78381-9_8.

[Moh23] Nicolas Mohnblatt. *Sangria: a folding scheme for PLONK*. link. 2023.

[MR09] Daniele Micciancio and Oded Regev. "Lattice-based cryptography". In: *Post-quantum cryptography*. Springer, 2009, pp. 147–191.

[NBS23] Wilson Nguyen, Dan Boneh, and Srinath Setty. *Revisiting the Nova Proof System on a Cycle of Curves*. Cryptology ePrint Archive, Paper 2023/969. https://eprint.iacr.org/2023/969. 2023. URL: https://eprint.iacr.org/2023/969.

[NS24] Ngoc Khanh Nguyen and Gregor Seiler. "Greyhound: Fast Polynomial Commitments from Lattices". In: *CRYPTO 2024, Part X*. Ed. by Leonid Reyzin and Douglas Stebila. Vol. 14929. LNCS. Springer, Cham, Aug. 2024, pp. 243–275. DOI: 10.1007/978-3-031-68403-6_8.

[NS25] Wilson Nguyen and Srinath Setty. *Neo: Lattice-based folding scheme for CCS over small fields and pay-per-bit commitments*. Cryptology ePrint Archive, Report 2025/294. 2025. URL: https://eprint.iacr.org/2025/294.

[NT16] Assa Naveh and Eran Tromer. "PhotoProof: Cryptographic Image Authentication for Any Set of Permissible Transformations". In: *2016 IEEE Symposium on Security and Privacy*. IEEE Computer Society Press, May 2016, pp. 255–271. DOI: 10.1109/SP.2016.23.

[PR06] Chris Peikert and Alon Rosen. "Efficient Collision-Resistant Hashing from Worst-Case Assumptions on Cyclic Lattices". In: *TCC 2006*. Ed. by Shai Halevi and Tal Rabin. Vol. 3876. LNCS. Springer, Berlin, Heidelberg, Mar. 2006, pp. 145–166. DOI: 10.1007/11681878_8.

[RZ22] Carla Ràfols and Alexandros Zacharakis. *Folding Schemes with Selective Verification*. Cryptology ePrint Archive, Report 2022/1576. 2022. URL: https://eprint.iacr.org/2022/1576.

[ST23] Srinath Setty and Justin Thaler. *BabySpartan: Lasso-based SNARK for non-uniform computation*. Cryptology ePrint Archive, Paper 2023/1799. https://eprint.iacr.org/2023/1799. 2023. URL: https://eprint.iacr.org/2023/1799.

[STW23a] Srinath Setty, Justin Thaler, and Riad Wahby. "Customizable constraint systems for succinct arguments". In: *Cryptology ePrint Archive* (2023).

[STW23b] Srinath Setty, Justin Thaler, and Riad Wahby. *Unlocking the lookup singularity with Lasso*. Cryptology ePrint Archive, Paper 2023/1216. https://eprint.iacr.org/2023/1216. 2023. URL: https://eprint.iacr.org/2023/1216.

[Val08] Paul Valiant. "Incrementally Verifiable Computation or Proofs of Knowledge Imply Time/Space Efficiency". In: *TCC 2008*. Ed. by Ran Canetti. Vol. 4948. LNCS. Springer, Berlin, Heidelberg, Mar. 2008, pp. 1–18. DOI: 10.1007/978-3-540-78524-8_1.

[WL84] B Johnson William and Joram Lindenstrauss. "Extensions of Lipschitz mapping into Hilbert space". In: *Contemporary mathematics* 26.189-206 (1984), p. 323.

[Xie+22] Tiancheng Xie, Jiaheng Zhang, Zerui Cheng, Fan Zhang, Yupeng Zhang, Yongzheng Jia, Dan Boneh, and Dawn Song. "zkBridge: Trustless Cross-chain Bridges Made Practical". In: *ACM CCS 2022*. Ed. by Heng Yin, Angelos Stavrou, Cas Cremers, and Elaine Shi. ACM Press, Nov. 2022, pp. 3003–3017. DOI: 10.1145/3548606.3560652.

[ZCF24] Hadas Zeilberger, Binyi Chen, and Ben Fisch. "BaseFold: Efficient Field-Agnostic Polynomial Commitment Schemes from Foldable Codes". In: *CRYPTO 2024, Part X*. Ed. by Leonid Reyzin and Douglas Stebila. Vol. 14929. LNCS. Springer, Cham, Aug. 2024, pp. 138–169. DOI: 10.1007/978-3-031-68403-6_5.

A Multilinear Evaluation Mapping Lemma

We derive a lemma from the fact that every ring homomorphism $\phi : R \to S$ induces a natural homomorphism $\phi' : R[X] \to S[X]$ over the polynomial rings. In the special case where $\mathcal{R}_q \cong \mathbb{Z}_q^d$, consider a vector $\vec{\mathbf{f}} \in \mathcal{R}_q^m$ and a vector $\hat{\mathbf{f}} \in \mathcal{R}_q^m$ such that $\mathsf{NTT}(\hat{\mathbf{f}}) = \mathsf{Coef}(\vec{\mathbf{f}})$. Let $f_1, \ldots, f_d$ be the multilinear extensions of the columns of $\mathsf{Coef}(\vec{\mathbf{f}}) \in \mathbb{Z}_q^{m \times d}$. The lemma asserts that a multilinear evaluation of $\mathsf{mle}\left[\hat{\mathbf{f}}\right]$ can be mapped to the evaluations of $f_1, \ldots, f_d$ (over $\mathbb{Z}_q$) through the NTT isomorphism. This lemma helps maintain the invariant of the evaluation statement $\hat{\mathbf{v}} = \mathsf{mle}\left[\hat{\mathbf{f}}\right](\vec{\mathbf{r}})$ after folding the witness $\vec{\mathbf{f}}$, even if the evaluation point $\vec{\mathbf{r}} \in \mathcal{R}_q^{\log m}$ changes after the folding.

Lemma A.1. *Let $\mathcal{R}_q \cong \mathbb{F}_{q^\tau}^{d/\tau}$ for some $\tau \in \mathbb{N}$ where $\tau \mid d$. Let $m \in \mathbb{N}$ be a power of two. For any $\vec{\mathbf{f}} \in \mathcal{R}_q^m$, let $\hat{\mathbf{f}} := (\hat{\mathbf{f}}_1, \ldots, \hat{\mathbf{f}}_\tau) \in \mathcal{R}_q^{m \times \tau}$ denote the vector such that $\mathsf{NTT}(\hat{\mathbf{f}}) = \mathsf{Coef}(\vec{\mathbf{f}}) \in \mathbb{Z}_q^{m \times d}$. Let $\vec{\mathbf{r}} \in \mathcal{R}_q^{\log m}$ and denote $(\vec{\mathbf{r}}_1^*, \ldots, \vec{\mathbf{r}}_{d/\tau}^*) \in (\mathbb{F}_{q^\tau}^{\log m})^{d/\tau}$ the columns of $\mathsf{NTT}(\vec{\mathbf{r}})$. For every $i \in [\tau]$, we have that $\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}) \in \mathcal{R}_q$ is mapped to*

$$\left(\mathsf{mle}\left[\mathsf{Coef}_{1+(i-1)\cdot d/\tau}(\vec{\mathbf{f}})\right](\vec{\mathbf{r}}_1^*), \ldots, \mathsf{mle}\left[\mathsf{Coef}_{i\cdot d/\tau}(\vec{\mathbf{f}})\right](\vec{\mathbf{r}}_{d/\tau}^*)\right) \in \mathbb{F}_{q^\tau}^{d/\tau}$$

by the NTT isomorphism. Recall that $\mathsf{mle}[\cdot]$ denotes multilinear extensions (Definition 2.4) and $\mathsf{Coef}_j(\vec{\mathbf{f}}) \in \mathbb{Z}_q^m$ is the jth $(1 \le j \le d)$ column of $\mathsf{Coef}(\vec{\mathbf{f}})$.

Proof. By definition of $\hat{\mathbf{f}}$, we have that

$$\mathsf{NTT}(\hat{\mathbf{f}}) = \left[\mathsf{NTT}(\hat{\mathbf{f}}_1), \ldots, \mathsf{NTT}(\hat{\mathbf{f}}_\tau)\right] = \left[\mathsf{Coef}_1(\vec{\mathbf{f}}), \ldots, \mathsf{Coef}_d(\vec{\mathbf{f}})\right]. \tag{42}$$

Also observe that

$$\left(\mathsf{mle}\left[\hat{\mathbf{f}}_1\right](\vec{\mathbf{r}}),\ldots,\mathsf{mle}\left[\hat{\mathbf{f}}_\tau\right](\vec{\mathbf{r}})\right)=\left(\left\langle\hat{\mathbf{f}}_j,\bigotimes_{i=1}^{\log m}(\vec{\mathbf{r}}_i,1-\vec{\mathbf{r}}_i)\right\rangle\right)_{j=1}^{\tau} \tag{43}$$

where $\bigotimes$ denotes tensor product over $\mathcal{R}_q$. By the Chinese Remainder Theorem, the NTT map is an isomorphism, and the lemma follows from the properties of an isomorphism. □

B Deferred Proofs

B.1 Proof of Lemma 4.1

Proof. *Public reducibility:* Given input instance $\mathbb{x} = (\mathsf{cm}, \mathbb{x}_{\mathsf{ccs}})$ and transcript that includes $(\vec{\mathbf{r}}_o, \hat{\mathbf{v}}, [\mathbf{u}_j]_{j=1}^t)$, the algorithm f outputs $\mathbb{x}_o = (\vec{\mathbf{r}}_o, \hat{\mathbf{v}}, \mathsf{cm}, [\mathbf{u}_j]_{j=1}^t, \mathbb{x}_{\mathsf{ccs}}, 1)$ if the verifier check passes; and $\perp$ otherwise.

Completeness: Let $(\mathbb{x}; \mathbb{w}) := \left((\mathsf{cm}, \mathbb{x}_{\mathsf{ccs}}); (\vec{\mathbf{f}}, \mathbb{w}_{\mathsf{ccs}})\right) \leftarrow \mathcal{A}(\mathsf{pp})$ denotes adversary $\mathcal{A}$'s output for $\mathcal{R}_1 := \mathcal{R}^B_{\mathsf{cmccs}}$, where $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$ is the public parameter. WLOG, we assume that $(\mathsf{pp}, \mathbb{x}; \mathbb{w}) \in \mathcal{R}^B_{\mathsf{cmccs}}$ (where $\mathcal{R}^B_{\mathsf{cmccs}}$ is specified in Definition 4.2). The protocol $\langle \mathsf{P}(\mathsf{pp}, \mathbb{x}, \mathbb{w}), \mathsf{V}(\mathsf{pp}, \mathbb{x})\rangle$ proceeds as follows:

1. After running the sum-check and receiving the challenge vector $\vec{\mathbf{r}}_o \in \mathcal{C}^{\log m}$, P sends V the value $\hat{\mathbf{v}} := \mathsf{mle}\left[\hat{\mathbf{f}}\right](\vec{\mathbf{r}}_o)$. Moreover, let $\mathbf{z}_{\mathsf{ccs}} := (\mathbb{x}_{\mathsf{ccs}}, 1, \mathbb{w}_{\mathsf{ccs}})$, for every $j \in [t]$, P sends $\mathbf{u}_j := \sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log n_c}} \mathsf{mle}\left[M_j\right](\vec{\mathbf{r}}_o, \vec{\mathbf{b}}) \cdot \mathsf{mle}\left[\mathbf{z}_{\mathsf{ccs}}\right](\vec{\mathbf{b}})$.
2. V outputs $\perp$ and halts if the check at Step 4 fails.
3. P outputs $\mathbb{w}_o := \mathbb{w} = (\vec{\mathbf{f}}, \mathbb{w}_{\mathsf{ccs}})$. V outputs $\mathbb{x}_o := (\vec{\mathbf{r}}_o, \hat{\mathbf{v}}, \mathsf{cm}, [\mathbf{u}_j]_{j=1}^t, \mathbb{x}_{\mathsf{ccs}}, 1)$.

First, by definitions of $[\mathbf{u}_j]_{j=1}^t$, we have that V passes the check at Step 4 and accepts.

Next we argue that the protocol's output $(\mathsf{pp}, \mathbb{x}_o; \mathbb{w}_o)$ is in the relation $\mathcal{R}_2 := \mathcal{R}^B_{\mathsf{evalccs}}$ (Eq. (35)) and completeness holds. It suffices to check that $\mathbf{z}_{\mathsf{ccs}} = \mathbf{G}\vec{\mathbf{f}}$ and certain statements are in $\mathcal{R}^B_{\mathsf{eval}}$ and $\mathcal{R}_{\mathsf{lccs}}$ respectively. We argue them one by one:

1. $\mathbf{z}_{\mathsf{ccs}} = \mathbf{G}\vec{\mathbf{f}}$ because $(\mathsf{pp}, \mathbb{x}; \mathbb{w}) \in \mathcal{R}^B_{\mathsf{cmccs}}$ by assumption.
2. $(\mathsf{pp}, (\vec{\mathbf{r}}_o, \hat{\mathbf{v}}, \mathsf{cm}); \vec{\mathbf{f}})$ is in $\mathcal{R}^B_{\mathsf{eval}}$ because (i) $\hat{\mathbf{v}} = \mathsf{mle}\left[\hat{\mathbf{f}}\right](\vec{\mathbf{r}}_o)$ and (ii) $(\mathsf{pp}, \mathsf{cm}, \vec{\mathbf{f}})$ is in $\mathcal{R}^B_{\mathsf{cm}}$ by the assumption that $(\mathsf{pp}, \mathbb{x}; \mathbb{w})$ is in $\mathcal{R}_1 := \mathcal{R}^B_{\mathsf{cmccs}}$.
3. $(\mathsf{pp}_{\mathsf{ccs}}, (\vec{\mathbf{r}}_o, [\mathbf{u}_j]_{j=1}^t, \mathbb{x}_{\mathsf{ccs}}, \mathbf{h}); \mathbb{w}_{\mathsf{ccs}})$ is in $\mathcal{R}_{\mathsf{lccs}}$ by the assignments of $[\mathbf{u}_j]_{j=1}^t$.

Knowledge soundness: Let $(\mathbb{x} := (\mathsf{cm}, \mathbb{x}_{\mathsf{ccs}}), \mathsf{state}) \leftarrow \mathcal{A}(\mathsf{pp})$ denote adversary $\mathcal{A}$'s chosen input for $\mathcal{R}_1 := \mathcal{R}^B_{\mathsf{cmccs}}$, where $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$ is the public parameter. The extractor Ext proceeds as follows:

1. Simulate the protocol $\langle \mathsf{P}^*(\mathsf{pp}, \mathbb{x}, \mathsf{state}), \mathsf{V}(\mathsf{pp}, \mathbb{x})\rangle$ where P^* is the malicious prover.
2. Abort and output $\perp$ if V rejects.
3. Otherwise, let $(\mathbb{x}_o, \mathbb{w}_o)$ be the protocol output where $\mathbb{x}_o := (\vec{\mathbf{r}}_o, \hat{\mathbf{v}}, \mathsf{cm}, [\mathbf{u}_j]_{j=1}^t, \mathbb{x}_{\mathsf{ccs}}, 1)$ and $\mathbb{w}_o := (\vec{\mathbf{f}}, \mathbb{w}_{\mathsf{ccs}})$. Abort and output $\perp$ if $(\mathsf{pp}, \mathbb{x}_o, \mathbb{w}_o) \notin \mathcal{R}^B_{\mathsf{evalccs}}$ (Eq. (35)).
4. Otherwise, if $(\mathsf{pp}, \mathbb{x}_o, \mathbb{w}_o) \in \mathcal{R}^B_{\mathsf{evalccs}}$, the extractor outputs $\mathbb{w} := (\vec{\mathbf{f}}, \mathbb{w}_{\mathsf{ccs}})$.

Let E_{bad} denote the bad event that (i) V accepts and the protocol's output $(\mathbb{x}_o, \mathbb{w}_o)$ satisfies that $(\mathsf{pp}, \mathbb{x}_o, \mathbb{w}_o) \in \mathcal{R}_2 := \mathcal{R}^B_{\mathsf{evalccs}}$ but (ii) $(\mathsf{pp}, \mathbb{x}; \mathbb{w})$ is *not* in $\mathcal{R}_1 := \mathcal{R}^B_{\mathsf{cmccs}}$ (Eq. (33)). To prove knowledge soundness, it suffices to argue that $\Pr[E_{\mathsf{bad}}] \leq \frac{(2b+1)\log m}{|\mathcal{C}|}$.

Note that if $(\mathsf{pp}, \mathbb{x}_o, \mathbb{w}_o)$ is in $\mathcal{R}_2 := \mathcal{R}^B_{\mathsf{evalccs}}$ (Eq. (35)), the following hold: (i) $(\mathbf{z}_{\mathsf{ccs}} = \mathbf{G}\vec{\mathbf{f}})$, (ii) $(\mathsf{pp}, \mathsf{cm}; \vec{\mathbf{f}})$ is in $\mathcal{R}^B_{\mathsf{cm}}$, and (iii) $(\mathsf{pp}_{\mathsf{ccs}}, (\vec{\mathbf{r}}_o, [\mathbf{u}]_{i=1}^t, \mathbb{x}_{\mathsf{ccs}}, 1); \mathbb{w}_{\mathsf{ccs}})$ is in $\mathcal{R}_{\mathsf{lccs}}$ (Eq. (34)). From (i) and (ii), the sumcheck polynomial g is fixed before sampling $\vec{\mathbf{r}}_o$, as the witness $\mathbf{z}_{\mathsf{ccs}}$ is bound to cm. Additionally, from (iii) and because the verifier check at Step 4 passes, the sum-check random evaluation $g(\vec{\mathbf{r}}_o) \stackrel{?}{=} s$ passes. Therefore, if $\sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log m}} g(\vec{\mathbf{b}}) \neq 0$, by sumcheck soundness, the probability that $(\mathsf{pp}, \mathbb{x}_o, \mathbb{w}_o) \in \mathcal{R}^B_{\mathsf{evalccs}}$ is at most $\frac{2b \log m}{|\mathcal{C}|}$ (over the sum-check challenges).

Next we argue that if $(\mathsf{pp}_{\mathsf{ccs}}, \mathbb{x}_{\mathsf{ccs}}; \mathbb{w}_{\mathsf{ccs}}) \notin \mathcal{R}_{\mathsf{ccs}}$, then $\sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log m}} g(\vec{\mathbf{b}}) \neq 0$ with high probability. Define multilinear polynomial $p(\vec{\mathbf{x}})$ as

$$p(\vec{\mathbf{x}}) := \sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log m}} eq(\vec{\mathbf{x}}, \vec{\mathbf{b}}) \cdot \left(\sum_{i=1}^{n_s} c_i \cdot \prod_{j \in S_i} \left(\sum_{\vec{\mathbf{y}} \in \{0,1\}^{\log n_c}} \mathsf{mle}\left[M_j\right](\vec{\mathbf{b}}, \vec{\mathbf{y}}) \cdot \mathsf{mle}\left[\mathbf{z}_{\mathsf{ccs}}\right](\vec{\mathbf{y}}) \right) \right) .$$

By definition of g, it's clear that $\sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log m}} g(\vec{\mathbf{b}}) = 0$ if and only if $p(\vec{\beta}) = 0$. Moreover, $(\mathsf{pp}_{\mathsf{ccs}}, \mathbb{x}_{\mathsf{ccs}}; \mathbb{w}_{\mathsf{ccs}}) \notin \mathcal{R}_{\mathsf{ccs}}$ (Definition 4.1) implies that $p(\vec{\mathbf{x}}) \neq 0$ for some $\vec{\mathbf{x}} \in \{0,1\}^{\log m}$ and $p(\vec{\mathbf{x}})$ is not a zero polynomial. Since $\vec{\beta} \xleftarrow{\$} \mathcal{C}^{\log m}$ is uniformly chosen from the sampling set $\mathcal{C}$ (Defn 2.1), by the Generalized Schwartz-Zippel Lemma (Lemma 2.4), the probability that $p(\vec{\mathbf{x}}) \neq 0$ while $p(\vec{\beta}) = 0$ is at most $\log m / |\mathcal{C}|$.

In summary, the probability that E_{bad} occurs is at most $\frac{\log m}{|\mathcal{C}|} + \frac{2b \log m}{|\mathcal{C}|}$, which finishes the proof. □

B.2 Proof of Lemma 4.2

Proof. *Public reducibility:* Given instance $\mathbb{x} = (\vec{\mathbf{r}}, \hat{\mathbf{v}}, y, \mathbf{u}, \mathbb{x}_w)$ and transcript $[\hat{\mathbf{v}}_i, y_i, \mathbf{u}_i, \mathbb{x}_{w,i}]_{i=0}^{k-1}$, one outputs $[\mathbb{x}_i := (\vec{\mathbf{r}}, \hat{\mathbf{v}}_i, y_i, \mathbf{u}_i, \mathbb{x}_{w,i})]_{i=0}^{k-1}$ if the verifier checks pass; otherwise it outputs $\perp$.

Completeness: Let $(\mathbb{x}, \mathbb{w}) := \left((\vec{\mathbf{r}}, \hat{\mathbf{v}}, y, \mathbf{u}, \mathbb{x}_w), (\vec{\mathbf{f}}, \vec{\mathbf{w}})\right) \leftarrow \mathcal{A}(\mathsf{pp})$ denote adversary $\mathcal{A}$'s chosen instance-witness pair for $\mathcal{R}_1 := \mathcal{R}^B_{\mathsf{ccshom}}$ (Eq. (36)), where $\mathsf{pp} := (\mathcal{L}, \mathcal{L}_w, \overline{M}) \leftarrow \mathsf{Setup}(1^\lambda)$

is the public parameter. WLOG we assume that $(\mathsf{pp}, \mathbb{x}; \mathbb{w})$ is in $\mathcal{R}^B_{\mathsf{ccshom}}$. The protocol $\langle \mathsf{P}(\mathsf{pp}, \mathbb{x}, \mathbb{w}), \mathsf{V}(\mathsf{pp}, \mathbb{x}) \rangle$ proceeds as follows:

1. P computes $\vec{\mathbf{F}} := (\vec{\mathbf{f}}_0, \ldots, \vec{\mathbf{f}}_{k-1}) := \mathsf{split}_{b,k}(\vec{\mathbf{f}})$ (Eq. (11)) and sends V values

$$y_i := \mathcal{L}(\vec{\mathbf{f}}_i)\,, \qquad \hat{\mathbf{v}}_i := \mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}})\,,$$
$$\mathbf{u}_i := \left\langle \overline{M} \times \mathsf{tensor}(\vec{\mathbf{r}}), \mathcal{L}_w(\vec{\mathbf{f}}_i) \right\rangle\,, \qquad (\mathbb{x}_{w,i} || \vec{\mathbf{w}}_i) := \mathcal{L}_w(\vec{\mathbf{f}}_i)\,.$$

   for every $i \in [0, k)$.

2. V outputs $\perp$ and halts if the check $\sum_{i=0}^{k-1} b^i \cdot [y_i, \hat{\mathbf{v}}_i, \mathbf{u}_i, \mathbb{x}_{w,i}] \stackrel{?}{=} [y, \hat{\mathbf{v}}, \mathbf{u}, \mathbb{x}_w]$ fails.

3. P outputs $[\vec{\mathbf{f}}_i, \vec{\mathbf{w}}_i]_{i=0}^{k-1}$. V outputs $[\mathbb{x}_i := (\vec{\mathbf{r}}, \hat{\mathbf{v}}_i, y_i, \mathbf{u}_i, \mathbb{x}_{w,i})]_{i=0}^{k-1}$.

We first show that if V accepts, then the protocol's output satisfies that for all $i \in [0, k)$,

$$(\mathsf{pp}, (\vec{\mathbf{r}}, \hat{\mathbf{v}}_i, y_i, \mathbf{u}_i, \mathbb{x}_{w,i}); (\vec{\mathbf{f}}_i, \vec{\mathbf{w}}_i)) \in \mathcal{R}^b_{\mathsf{ccshom}}\,.$$

This follows given how $[y_i, \hat{\mathbf{v}}_i, \mathbf{u}_i, \mathbb{x}_{w,i}]_{i=0}^{k-1}$ are computed, and because $\|\vec{\mathbf{f}}_i\|_\infty < b$ for all $i \in [0, k)$ by the property of the algorithm $\mathsf{split}_{b,k}(\vec{\mathbf{f}})$.

It remains to argue that V will accept, that is, the check $\sum_{i=0}^{k-1} b^i \cdot [y_i, \hat{\mathbf{v}}_i, \mathbf{u}_i, \mathbb{x}_{w,i}] \stackrel{?}{=} [y, \hat{\mathbf{v}}, \mathbf{u}, \mathbb{x}_w]$ passes. By Lemma 3.4, we have that

$$\sum_{i=0}^{k-1} b^i \cdot y_i = y\,, \qquad \sum_{i=0}^{k-1} b^i \cdot \hat{\mathbf{v}}_i = \hat{\mathbf{v}}\,.$$

Since $\mathcal{L}_w$ is an $\mathcal{R}_q$-module homomorphism, by the same argument for proving $\sum_{i=0}^{k-1} b^i \cdot y_i = y$, it also holds that $\sum_{i=0}^{k-1} b^i \cdot \mathbb{x}_{w,i} = \mathbb{x}_w$. Finally, we have that

$$\sum_{i=0}^{k-1} b^i \cdot \mathbf{u}_i = \sum_{i=0}^{k-1} b^i \cdot \left\langle \overline{M} \times \mathsf{tensor}(\vec{\mathbf{r}}), \mathcal{L}_w(\vec{\mathbf{f}}_i) \right\rangle = \left\langle \overline{M} \times \mathsf{tensor}(\vec{\mathbf{r}}), \sum_{i=0}^{k-1} b^i \cdot \mathcal{L}_w(\vec{\mathbf{f}}_i) \right\rangle$$
$$= \left\langle \overline{M} \times \mathsf{tensor}(\vec{\mathbf{r}}), \mathcal{L}_w\left(\sum_{i=0}^{k-1} b^i \cdot \vec{\mathbf{f}}_i\right) \right\rangle = \left\langle \overline{M} \times \mathsf{tensor}(\vec{\mathbf{r}}), \mathcal{L}_w(\vec{\mathbf{f}}) \right\rangle = \mathbf{u}\,.$$

The 1st equality is by definition of $[\mathbf{u}_i]_{i=0}^{k-1}$; the 2nd equality is by the property of inner products; the 3rd equality holds because $\mathcal{L}_w$ is an $\mathcal{R}_q$-module homomorphism; the 4th equality is by definition of decomposition (Eq. (11)); the last equality holds because $(\mathsf{pp}, \mathbb{x}; \mathbb{w})$ is in $\mathcal{R}_1 := \mathcal{R}^B_{\mathsf{ccshom}}$ (Eq. (36)) by assumption, whereas $\mathcal{R}^B_{\mathsf{ccshom}}$ checks that $\left\langle \overline{M} \times \mathsf{tensor}(\vec{\mathbf{r}}), \mathcal{L}_w(\vec{\mathbf{f}}) \right\rangle = \mathbf{u}$. Therefore, V will accept and thus completeness holds.

Knowledge soundness: Let $(\mathbb{x} := (\hat{\mathbf{v}}, y, \mathbf{u}, \mathbb{x}_w), \mathsf{state}) \leftarrow \mathcal{A}(\mathsf{pp})$ denote adversary $\mathcal{A}$'s chosen input instance for $\mathcal{R}_1 := \mathcal{R}^B_{\mathsf{ccshom}}$, where $\mathsf{pp} := (\mathcal{L}, \mathcal{L}_w, \overline{M}) \leftarrow \mathsf{Setup}(1^\lambda)$ is the public parameter. The extractor Ext proceeds as follows:

1. Simulate the protocol $\langle \mathsf{P}^*(\mathsf{pp}, \mathbb{x}, \mathsf{state}), \mathsf{V}(\mathsf{pp}, \mathbb{x})\rangle$ where P^* is the malicious prover.

2. Output $\perp$ if V rejects. Otherwise let $(\mathbb{x}_o, \mathbb{w}_o) := [(\vec{\mathbf{r}}, \hat{\mathbf{v}}_i, y_i, \mathbf{u}_i, \mathbb{x}_{w,i}); (\vec{\mathbf{f}}_i, \vec{\mathbf{w}}_i)]_{i=0}^{k-1}$ be the protocol output (note that $\vec{\mathbf{r}}$ is the same as that in the input instance $\mathbb{x}$ to pass the verification check). The extractor outputs witness $\mathbb{w} := (\vec{\mathbf{f}}, \vec{\mathbf{w}})$ where

$$\vec{\mathbf{f}} := \sum_{i=0}^{k-1} b^i \cdot \vec{\mathbf{f}}_i\,, \qquad \vec{\mathbf{w}} := \sum_{i=0}^{k-1} b^i \cdot \mathcal{L}_w(\vec{\mathbf{f}}_i)[n_{\mathsf{in}}+1, n_{\mathsf{in}}+n]\,. \tag{44}$$

To prove knowledge soundness, it suffices to show that if V accepts and the output $(\mathbb{x}_o, \mathbb{w}_o)$ satisfies that $(\mathsf{pp}, \mathbb{x}_o, \mathbb{w}_o) \in \mathcal{R}_2 := (\mathcal{R}^b_{\mathsf{ccshom}})^k$, then the extracted witness $\mathbb{w}$ satisfies that $(\mathsf{pp}, \mathbb{x}, \mathbb{w}) \in \mathcal{R}_1 := \mathcal{R}^B_{\mathsf{ccshom}}$.

By Lemma 3.5, we have that $(\vec{\mathbf{r}}, \hat{\mathbf{v}}, y; \vec{\mathbf{f}}) \in \mathcal{R}^B_{\mathsf{hom}}$ if $[(\vec{\mathbf{r}}, \hat{\mathbf{v}}_i, y_i); \vec{\mathbf{f}}_i]_{i=0}^{k-1}$ is in $(\mathcal{R}^b_{\mathsf{hom}})^k$. By definition of $\mathcal{R}^B_{\mathsf{ccshom}}$ (Eq. (36)), it remains to argue that

$$\mathbf{z}_i := (\mathbb{x}_{w,i} || \vec{\mathbf{w}}_i) = \mathcal{L}_w(\vec{\mathbf{f}}_i)\ \forall i \in [0,k) \implies \mathbf{z} := (\mathbb{x}_w || \vec{\mathbf{w}}) = \mathcal{L}_w(\vec{\mathbf{f}})\,; \tag{45}$$

$$\mathbf{u}_i = \left\langle \overline{M} \times \mathsf{tensor}(\vec{\mathbf{r}}), \mathbf{z}_i \right\rangle\ \forall i \in [0,k) \implies \mathbf{u} = \left\langle \overline{M} \times \mathsf{tensor}(\vec{\mathbf{r}}), \mathbf{z} \right\rangle\,. \tag{46}$$

We first prove Eq. (45). By the verifier check $\sum_{i=0}^{k-1} b^i \cdot \mathbb{x}_{w,i} = \mathbb{x}_w$ and by Eq. (44), we have that $\mathbf{z} = \sum_{i=0}^{k-1} b^i \cdot \mathbf{z}_i$. Since $\mathcal{L}_w$ is an $\mathcal{R}_q$-module homomorphism, Eq. (45) holds by the same argument as in Lemma 3.2 for proving $y = \mathcal{L}(\vec{\mathbf{f}})$ (where we replace $y, \mathcal{L}$ with $\mathbf{z}, \mathcal{L}_w$ respectively).

Similarly, Eq. (46) holds because

$$\mathbf{u} = \sum_{i=0}^{k-1} b^i \cdot \mathbf{u}_i = \sum_{i=0}^{k-1} b^i \cdot \left\langle \overline{M} \times \mathsf{tensor}(\vec{\mathbf{r}}), \mathbf{z}_i \right\rangle = \left\langle \overline{M} \times \mathsf{tensor}(\vec{\mathbf{r}}), \sum_{i=0}^{k-1} b^i \cdot \mathbf{z}_i \right\rangle = \left\langle \overline{M} \times \mathsf{tensor}(\vec{\mathbf{r}}), \mathbf{z} \right\rangle$$

where the 1st equality is by the verifier check; the 2nd equality follows from the premise in Eq. (46); the 3rd equality follows from the linearly homomorphic property of inner products; the last equality holds as we've proved previously that $\mathbf{z} = \sum_{i=0}^{k-1} b^i \cdot \mathbf{z}_i$.

In summary, $(\vec{\mathbf{r}}, \hat{\mathbf{v}}, y; \vec{\mathbf{f}})$ is in $\mathcal{R}^B_{\mathsf{hom}}$ and Eq. (45), Eq. (46) hold true. Therefore, conditioned on that $(\mathsf{pp}, [(\vec{\mathbf{r}}, \hat{\mathbf{v}}_i, y_i, \mathbf{u}_i, \mathbb{x}_{w,i}); (\vec{\mathbf{f}}_i, \vec{\mathbf{w}}_i)]_{i=0}^{k})$ is in $\mathcal{R}_2 := (\mathcal{R}^b_{\mathsf{ccshom}})^k$, the extracted witness $(\vec{\mathbf{f}}, \vec{\mathbf{w}})$ will satisfy that $(\mathsf{pp}, (\vec{\mathbf{r}}, \hat{\mathbf{v}}, y, \mathbf{u}, \mathbb{x}_w); (\vec{\mathbf{f}}, \vec{\mathbf{w}}))$ is in $\mathcal{R}_1 := \mathcal{R}^B_{\mathsf{ccshom}}$, which completes the proof. □

B.3 Proof of Theorem 4.2

Proof. *Public reducibility:* Given input instances $[\vec{\mathbf{r}}_i, \hat{\mathbf{v}}_i, y_i, \mathbf{u}_i, \mathbb{x}_{w,i}]_{i=1}^{2k}$ and the transcript that includes challenges $\vec{\mathbf{r}}_o$, evaluations $[\theta_i]_{i=1}^{2k}$, values $[\eta_i]_{i=1}^{2k}$, and folding challenges $[\rho_i]_{i=1}^{2k}$.

The algorithm f outputs $\mathbb{x}_o := (\vec{\mathbf{r}}_o, \hat{\mathbf{v}}_o, y_o, \mathbf{u}_o, \mathbb{x}_{w,o})$ such that

$$\mathsf{NTT}(\hat{\mathbf{v}}_o) = \sum_{i=1}^{2k} \mathsf{RotSum}(\rho_i, \mathsf{NTT}(\theta_i)) \,, \quad [y_o, \mathbf{u}_o, \mathbb{x}_{w,o}] := \sum_{i=1}^{2k} \rho_i \cdot [y_i, \eta_i, \mathbb{x}_{w,i}]$$

if the verifier checks pass; and output $\perp$ otherwise. (RotSum defined in Lemma 2.1.)

Completeness: Let

$$(\mathbb{x}, \mathbb{w}) := [\mathbb{x}_i = (\vec{\mathbf{r}}_i, \hat{\mathbf{v}}_i, y_i, \mathbf{u}_i, \mathbb{x}_{w,i}), \mathbb{w}_i = (\vec{\mathbf{f}}_i, \vec{\mathbf{w}}_i)]_{i=1}^{2k} \leftarrow \mathcal{A}(\mathsf{pp})$$

denote adversary $\mathcal{A}$'s chosen instance-witness pair for $\mathcal{R}_1 := (\mathcal{R}^b_{\mathsf{hom}})^{2k}$, where $\mathsf{pp} := (\mathcal{L}, \mathcal{L}_w, \overline{M} = M_1) \leftarrow \mathsf{Setup}(1^\lambda)$ is the public parameter. WLOG we assume that $(\mathsf{pp}, \mathbb{x}, \mathbb{w})$ is in $\mathcal{R}_1 := (\mathcal{R}^b_{\mathsf{ccshom}})^{2k}$. The protocol $\langle \mathsf{P}(\mathsf{pp}, \mathbb{x}, \mathbb{w}), \mathsf{V}(\mathsf{pp}, \mathbb{x}) \rangle$ proceeds as follows:

1. P and V honestly run the sum-check and P sends values $[\theta_i, \eta_i]_{i=1}^{2k}$ honestly such that for every $i \in [2k]$,

$$\eta_i := \left\langle \overline{M} \times \mathsf{tensor}(\vec{\mathbf{r}}_o), (\mathbb{x}_{w,i} || \vec{\mathbf{w}}_i) \right\rangle \,, \qquad \theta_i := \mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_o) \,. \tag{47}$$

   Here $\mathsf{tensor}(\cdot)$ is defined in Eq. (37) and $\vec{\mathbf{r}}_o$ is the sum-check challenge vector.

2. V outputs $\perp$ and halts if the check at Step 4 fails.

3. Otherwise, let $[\rho_i]_{i=1}^{2k}$ be verifier's last folding challenges. Set $(\hat{\mathbf{v}}_o, y_o, \mathbf{u}_o, \mathbb{x}_{w,o}, \vec{\mathbf{f}}_o, \vec{\mathbf{w}}_o)$ such that

$$\mathsf{NTT}(\hat{\mathbf{v}}_o) = \sum_{i=1}^{2k} \mathsf{RotSum}(\rho_i, \mathsf{NTT}(\theta_i))$$

   and

$$[y_o, \mathbf{u}_o, \vec{\mathbf{f}}_o, (\mathbb{x}_{w,o} || \vec{\mathbf{w}}_o)] := \sum_{i=1}^{2k} \rho_i \cdot [y_i, \eta_i, \vec{\mathbf{f}}_i, (\mathbb{x}_{w,i} || \vec{\mathbf{w}}_i)] \,. \tag{48}$$

4. P outputs $\mathbb{w}_o := (\vec{\mathbf{f}}_o, \vec{\mathbf{w}}_o)$. V outputs $\mathbb{x}_o := (\vec{\mathbf{r}}_o, \hat{\mathbf{v}}_o, y_o, \mathbf{u}_o, \mathbb{x}_{w,o})$.

First, we show that V accepts, meaning the verifier check at Step 4 will pass. This follows from the definition of polynomial g (Eq. (38)) and the definitions of $(\eta_i, \theta_i)_{i=1}^{2k}$.

It remains to argue that the protocol output $(\mathbb{x}_o, \mathbb{w}_o)$ satisfies that $(\mathsf{pp} := (\mathcal{L}, \mathcal{L}_w, \overline{M}), \mathbb{x}_o; \mathbb{w}_o) \in \mathcal{R}_2 := \mathcal{R}^B_{\mathsf{ccshom}}$ (Eq. (36)). First, by Lemma 3.6, we have that $(\mathcal{L}, (\vec{\mathbf{r}}_o, \hat{\mathbf{v}}_o, y_o); \vec{\mathbf{f}}_o) \in \mathcal{R}^B_{\mathsf{hom}}$. Let $\mathbf{z}_o := (\mathbb{x}_{w,o} || \vec{\mathbf{w}}_o)$. It remains to prove that (i) $\mathbf{z}_o = \mathcal{L}_w(\vec{\mathbf{f}}_o)$ and (ii) $\mathbf{u}_o = \left\langle \overline{M} \times \mathsf{tensor}(\vec{\mathbf{r}}_o), \mathbf{z}_o \right\rangle$. Note that (i) holds true because

$$\mathbf{z}_o := (\mathbb{x}_{w,o} || \vec{\mathbf{w}}_o) = \sum_{i=1}^{2k} \rho_i \cdot (\mathbb{x}_{w,i} || \vec{\mathbf{w}}_i) = \sum_{i=1}^{2k} \rho_i \cdot \mathcal{L}_w(\vec{\mathbf{f}}_i) = \mathcal{L}_w(\vec{\mathbf{f}}_o) \,.$$

The 1st equality follows from Eq. (48); the 2nd equality holds because $\mathcal{L}_w(\vec{\mathbf{f}}_i) = (\mathbb{x}_{w,i}||\vec{\mathbf{w}}_i)$ for all $i \in [2k]$, given the premise that $(\mathsf{pp}, \mathbb{x}, \mathbb{w}) \in (\mathcal{R}^b_{\mathsf{ccshom}})^{2k}$; the last equality holds due to the fact that $\vec{\mathbf{f}}_o = \sum_{i=1}^{2k} \rho_i \cdot \vec{\mathbf{f}}_i$ and by the homomorphic property of $\mathcal{L}_w$.

Similarly, (ii) holds true because

$$\mathbf{u}_o = \sum_{i=1}^{2k} \rho_i \cdot \eta_i = \sum_{i=1}^{2k} \rho_i \cdot \left\langle \overline{M} \times \mathsf{tensor}(\vec{\mathbf{r}}_o), (\mathbb{x}_{w,i}||\vec{\mathbf{w}}_i) \right\rangle = \left\langle \overline{M} \times \mathsf{tensor}(\vec{\mathbf{r}}_o), \mathbf{z}_o \right\rangle .$$

The 1st equality is by the definition of $\mathbf{u}_o$; the 2nd equality follows from the definition of η_i in Eq. (47); the last equality holds given the assignment of $\mathbf{z}_o := (\mathbb{x}_{w,o}||\vec{\mathbf{w}}_o)$ in Eq. (48) and the homomorphic property of inner products.

In sum, $(\mathsf{pp}, \mathbb{x}_o, \mathbb{w}_o)$ is in $\mathcal{R}_2 := \mathcal{R}^B_{\mathsf{ccshom}}$ (Eq. (36)) and the completeness holds.

Knowledge soundness: The extractor $\mathsf{Ext}^{\mathcal{A},\mathsf{P}^*}$ and the running time analysis are almost identical to the proof of Theorem 3.3. The only difference is that the extractor $\mathsf{Ext}^{\mathcal{A},\mathsf{P}^*}$, besides outputting $[\vec{\mathbf{f}}_i]_{i=1}^{2k}$, also outputs $[\vec{\mathbf{w}}_i := \mathcal{L}_w(\vec{\mathbf{f}}_i)[n_{\mathsf{in}}+1, n_{\mathsf{in}}+n]]_{i=1}^{2k}$.

Let E_{ext} denote the event defined in the proof of Theorem 3.3, indicating that the extractor recovers two *identical* witnesses $\mathsf{out}_1, \mathsf{out}_2 \neq \perp$ using two good sets of randomness ψ and ψ'. To argue the success probability of extraction, we replace the events E_{hom} (Eq. (21)) and E_{eval} (Eq. (22)) with E^*_{hom} and E^*_{eval} defined as follows.

$$E^*_{\mathsf{hom}} := E_{\mathsf{ext}} \wedge \left(\forall i \in [2k] \ : \ (\mathcal{L}(\vec{\mathbf{f}}_i) = y_i) \wedge \left((\mathbb{x}_{w,i}||\vec{\mathbf{w}}_i) = \mathcal{L}_w(\vec{\mathbf{f}}_i) \right) \right) , \tag{49}$$

$$E^*_{\mathsf{eval}} := E_{\mathsf{ext}} \wedge \left(\forall i \in [2k] \ : \ \left(\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}'_o) = \theta'_i \right) \wedge \left(\eta'_i = \left\langle \overline{M} \cdot \mathsf{tensor}(\vec{\mathbf{r}}'_o), (\mathbb{x}_{w,i}||\vec{\mathbf{w}}_i) \right\rangle \right) \right) . \tag{50}$$

Moreover, we redefine the event E_{bad} (Eq. (23)) as E^*_{bad} such that E^*_{bad} holds if and only $E_{\mathsf{ext}} = 1$ and there exists $i \in [2k]$ such that

$$\mathbf{u}_i \neq \left\langle \overline{M} \cdot \mathsf{tensor}(\vec{\mathbf{r}}_i), (\mathbb{x}_{w,i}||\vec{\mathbf{w}}_i) \right\rangle \quad \text{or} \quad \mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_i) \neq \hat{\mathbf{v}}_i \quad \text{or} \quad p_i(\vec{\mathbf{x}}) \neq 0 . \tag{51}$$

To reuse the proof of Theorem 3.3, it suffices to prove the following claims.

Claim 6. *If E_{ext} occurs, then $E^*_{\mathsf{hom}} \wedge E^*_{\mathsf{eval}}$ occurs. That is, for all $i \in [2k]$, we have that (i) $\mathcal{L}(\vec{\mathbf{f}}_i) = y_i$, (ii) $\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}'_o) = \theta'_i$, (iii) $(\mathbb{x}_{w,i}||\vec{\mathbf{w}}_i) = \mathcal{L}_w(\vec{\mathbf{f}}_i)$, and (iv) $\eta'_i = \left\langle \overline{M} \cdot \mathsf{tensor}(\vec{\mathbf{r}}'_o), (\mathbb{x}_{w,i}||\vec{\mathbf{w}}_i) \right\rangle$.*

Proof. The equations (i), (ii) holds given the proof of Claim 2. To prove (iii), since $\vec{\mathbf{w}}_i = \mathcal{L}_w(\vec{\mathbf{f}}_i)[n_{\mathsf{in}}+1, n_{\mathsf{in}}+n]$ by definition, it suffices to show that $\mathbb{x}_{w,i} = \mathcal{L}_w(\vec{\mathbf{f}}_i)[1, n_{\mathsf{in}}]$. Then (iii) follows by replacing $\mathcal{L}$, y_i with $\mathcal{L}_w(\cdot)[1, n_{\mathsf{in}}]$, $\mathbb{x}_{w,i}$ everywhere respectively and reusing the argument for (i).

To argue (iv), we observe that the map $\phi(\cdot) := \left\langle \overline{M} \cdot \mathsf{tensor}(\vec{\mathbf{r}}'_o), (\cdot) \right\rangle$, and the map $\mathcal{L}_w$ are both $\mathcal{R}_q$-module homomorphisms; thus, the composition $\Phi := \phi \circ \mathcal{L}_w$ is also an $\mathcal{R}_q$-module homomorphism. Note that if (iii) holds, then (iv) also holds if $\eta'_i = \Phi(\vec{\mathbf{f}}_i)$, which follows by replacing $\mathcal{L}$, y_i with Φ, η'_i everywhere respectively and reusing the argument for (i). □

Claim 7. $\Pr[E^*_{\mathsf{eval}} \wedge E^*_{\mathsf{bad}}] \leq \frac{(2b+1)\log m + 6k}{|\mathcal{C}|}$.

Proof. The proof is similar to that of Claim 3.7. The difference is that we redefine the polynomial h in Eq. (25) to

$$h([X_i, Y_i, Z_i]_{i=1}^{2k}) := \sum_{i=1}^{2k} \left(\hat{\mathbf{v}}_i - \mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_i)\right) \cdot X_i + \sum_{i=1}^{2k} p_i(\vec{\beta}') \cdot Y_i + \sum_{i=1}^{2k} \left(\left\langle \overline{M} \cdot \mathsf{tensor}(\vec{\mathbf{r}}_i), \mathbf{z}_i \right\rangle - \mathbf{u}_i\right) \cdot Z_i \,.$$

Note that the sumcheck statement holds if and only if $h([\alpha_i, \mu_i, \zeta_i]_{i=1}^{2k}) = 0$. This follows from the fact that for all $i \in [2k]$, let $\mathbf{z}_i := (\mathbb{x}_{w,i} || \vec{\mathbf{w}}_i)$, we have that

$$\begin{aligned}
\sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log m}} g_{3,i}(\vec{\mathbf{b}}) &= \sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log m}} eq(\vec{\mathbf{r}}_i, \vec{\mathbf{b}}) \cdot \left(\sum_{\vec{\mathbf{y}} \in \{0,1\}^{\log(n_{\mathsf{in}}+n)}} \mathsf{mle}\left[\overline{M}\right](\vec{\mathbf{b}}, \vec{\mathbf{y}}) \cdot \mathsf{mle}\left[\mathbf{z}_i\right](\vec{\mathbf{y}}) \,. \right) \\
&= \sum_{\vec{\mathbf{y}} \in \{0,1\}^{\log(n_{\mathsf{in}}+n)}} \mathsf{mle}\left[\overline{M}\right](\vec{\mathbf{r}}_i, \vec{\mathbf{y}}) \cdot \mathsf{mle}\left[\mathbf{z}_i\right](\vec{\mathbf{y}}) \\
&= \left\langle \overline{M} \cdot \mathsf{tensor}(\vec{\mathbf{r}}_i), \mathbf{z}_i \right\rangle \,.
\end{aligned}$$

Therefore, we can redefine the events E_2 and E_3 in the proof of Claim 3.7 as

$$\begin{aligned}
E_2 &:= E_{\mathsf{ext}} \wedge (h([\alpha'_i, \mu'_i, \zeta'_i]_{i=1}^{2k}) = 0) \wedge \\
&\quad \left(\exists i \in [2k] \;:\; \left(\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_i) \neq \hat{\mathbf{v}}_i\right) \vee (p_i(\vec{\beta}') \neq 0) \vee \left(\mathbf{u}_i \neq \left\langle \overline{M} \cdot \mathsf{tensor}(\vec{\mathbf{r}}_i), (\mathbb{x}_{w,i} || \vec{\mathbf{w}}_i) \right\rangle\right)\right) \\
E_3 &:= E^*_{\mathsf{eval}} \wedge (h([\alpha'_i, \mu'_i, \zeta'_i]_{i=1}^{2k}) \neq 0)
\end{aligned}$$

and the same argument used in the proof of Claim 3.7 can be applied here. □

□

B.4 Proof of Lemma 4.3

Proof. *Public reducibility:* Given instances $([\mathbb{x}_i]_{i=1}^k, \mathbb{x}')$, transcripts that includes $[\theta_i, \theta'_i, \rho_i, \rho'_i]_{i=1}^k$, $[\eta^{i,j}, \eta^{i,j}_*]_{i \in [k], j \in [t]}$ and sumcheck challenge $\vec{\mathbf{r}}_o$, if the folding verifier accepts, the algorithm outputs the folded instance according to Step 6 of Π_{batch}, and outputs $\perp$ otherwise.

Completeness: Given adversarily chosen statements that are in the corresponding relations, we need to argue that the verifier will accept in an honest execution and the folded statement is in the output relation. The argument for latter is identical to that in the proof of Theorem 4.2. The argument for the verification check at Step 4 is also similar, but we need to further show that for all $j \in [t]$,

$$\sum_{\ell=1}^{k} b^{\ell-1} \eta^{\ell,j}_* = \sum_{\vec{\mathbf{y}} \in \{0,1\}^{\log(n+n_{\mathsf{in}})}} \mathsf{mle}\left[M_j\right](\vec{\mathbf{r}}_o, \vec{\mathbf{y}}) \cdot \mathsf{mle}\left[\mathbf{z}_{\mathsf{ccs}}\right](\vec{\mathbf{y}}) \,, \tag{52}$$

where $\mathbf{z}_{\mathsf{ccs}} := (\mathbb{x}_{\mathsf{ccs}}, 1, \mathbb{w}_{\mathsf{ccs}})$. This follows from the facts below. First, for all $i \in [k]$, let $\vec{\mathbf{f}}'_i$ denote the i-th low-norm witness vector in $\mathbb{x}'$. For all $j \in [t]$, by definition we have that

$$\eta_*^{i,j} = \sum_{\vec{\mathbf{y}} \in \{0,1\}^{\log(n+n_{\mathsf{in}})}} \mathsf{mle}\left[M_j\right](\vec{\mathbf{r}}_o, \vec{\mathbf{y}}) \cdot \mathsf{mle}\left[\mathcal{L}_w(\vec{\mathbf{f}}'_i)\right](\vec{\mathbf{y}}) .$$

Moreover, for all $\vec{\mathbf{y}} \in \{0,1\}^{\log(n+n_{\mathsf{in}})}$, we have

$$\sum_{\ell=1}^{k} b^{\ell-1} \mathsf{mle}\left[\mathcal{L}_w(\vec{\mathbf{f}}'_i)\right](\vec{\mathbf{y}}) = \mathsf{mle}\left[\sum_{\ell=1}^{k} b^{\ell-1} \mathcal{L}_w(\vec{\mathbf{f}}'_i)\right](\vec{\mathbf{y}}) = \mathsf{mle}\left[\mathcal{L}_w\left(\sum_{\ell=1}^{k} b^{\ell-1} \vec{\mathbf{f}}'_i\right)\right](\vec{\mathbf{y}}) = \mathsf{mle}\left[\mathbf{z}_{\mathsf{ccs}}\right](\vec{\mathbf{y}}) .$$

Therefore, Eq. (52) holds and completeness follows.

Knowledge soundness: The extractor $\mathsf{Ext}^{\mathcal{A},\mathsf{P}^*}$ and the running time analysis are identical to the proof of Theorem 4.2. The main difference lies in the analysis of the success probability. Here we need to further argue that the extracted witness for relation $\mathcal{R}^{b,k}_{\mathsf{splitccs}}$ satisfies the CCS relation (i.e., $(\mathsf{pp}_{\mathsf{ccs}}, \mathbb{x}_{\mathsf{ccs}}; \mathbb{w}_{\mathsf{ccs}}) \in \mathcal{R}_{\mathsf{ccs}}$). To show this, let $z_{\mathsf{ccs}} := (\mathbb{x}_{\mathsf{ccs}}, 1, \mathbb{w}_{\mathsf{ccs}})$, we define the multilinear polynomial $q(\vec{\mathbf{x}})$ the same way as we did in the proof of Lemma 4.1,

$$q(\vec{\mathbf{x}}) := \sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log m}} eq(\vec{\mathbf{x}}, \vec{\mathbf{b}}) \cdot \left(\sum_{i=1}^{n_s} c_i \cdot \prod_{j \in S_i} \left(\sum_{\vec{\mathbf{y}} \in \{0,1\}^{\log n + n_{\mathsf{in}}}} \mathsf{mle}\left[M_j\right](\vec{\mathbf{b}}, \vec{\mathbf{y}}) \cdot \mathsf{mle}\left[\mathbf{z}_{\mathsf{ccs}}\right](\vec{\mathbf{y}}) \right) \right) .$$

Recall that $(\mathsf{pp}_{\mathsf{ccs}}, \mathbb{x}_{\mathsf{ccs}}; \mathbb{w}_{\mathsf{ccs}}) \in \mathcal{R}_{\mathsf{ccs}}$ if and only if the multilinear polynomial $q(\vec{\mathbf{x}})$ is a zero polynomial. Similar to the proof of Theorem 4.2, we need to define the bad event E^*_{bad} and prove that E^*_{bad} happens with small probability. Here the definition of E^*_{bad} is almost identical to that in Eq. (51) except that we add that E^*_{bad} also holds true if $q(\vec{\mathbf{x}}) \neq 0$.

Next, similar to Claim 7, we have to provide an upper-bound on the probability that the evaluation check passes while the bad event happens, that is, $\Pr[E^*_{\mathsf{eval}} \wedge E^*_{\mathsf{bad}}]$. Here, E^*_{eval} is similarly defined as in Eq. (50) except that we add the following predicate: for all $i \in [k]$, let $\vec{\mathbf{f}}'_i$ denote the i-th low-norm extracted witness for $\mathcal{R}^{b,k}_{\mathsf{splitccs}}$, then for all $j \in [t]$, check

$$\eta_*^{i,j} = \sum_{\vec{\mathbf{y}} \in \{0,1\}^{\log(n+n_{\mathsf{in}})}} \mathsf{mle}\left[M_j\right](\vec{\mathbf{r}}_o, \vec{\mathbf{y}}) \cdot \mathsf{mle}\left[\mathcal{L}_w(\vec{\mathbf{f}}'_i)\right](\vec{\mathbf{y}}) .$$

Note that we can obtain an analogous claim as in Claim 6 with exactly the same proof.

Finally, we use the same technique in the proof of Claim 7 to bound the probability $\Pr[E^*_{\mathsf{eval}} \wedge E^*_{\mathsf{bad}}]$. A key observation is that $q(\vec{\beta})$ can be understood as a sumcheck statement, that is,

$$q(\vec{\beta}) = \sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log m}} g'_{\mathsf{ccs}}(\vec{\mathbf{b}})$$

where $g'_{\mathsf{ccs}}(\vec{\mathbf{x}})$ is defined in Eq. (41). Therefore, by analogously defining the polynomial h and adding the term $q(\vec{\beta}) \cdot W$ inside h (where W is a formal variable), we can go through

exactly the same route as in the proof of Claim 7 to argue that $\Pr[E^*_{\mathsf{eval}} \wedge E^*_{\mathsf{bad}}]$ is small. □