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

Dan Boneh[*] and Binyi Chen[*]

[*]Stanford University

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. 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. Our evaluation of the final proof system suggests that it is as performant as Hypernova, while providing post-quantum security.

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 many others. 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 2.4 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 for a suitable choice of parameters.

A natural starting point is to try a 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 2.2):

- 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 $\vec{\mathbf{f}}' := \sum_{i=1}^{k} \rho_i \vec{\mathbf{f}}_i$ 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.

Instead, our solution 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 $\mathcal{R}^B_{\mathsf{cm}}$ 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 illustrates all the tools needed to build a folding-based IVC and PCD from the MSIS assumption. However, a relation such as $\mathcal{R}^B_{\mathsf{cm}}$ that proves knowledge of a committed value is not enough to implement an IVC or PCD. One would also need to incorporate into $\mathcal{R}^B_{\mathsf{cm}}$ a computation checking relation, such as verifying a witness for an R1CS relation. We do so in Section 4 by extending $\mathcal{R}^B_{\mathsf{cm}}$ 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 $\mathcal{R}^B_{\mathsf{cm}}$ 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. We note that Protostar [BC23] does not use such a linearization step, and instead handles cross terms by collapsing them using a random linear combination. Their approach, however, does not readily apply in our settings because it requires committing to a random high norm Vandermonde vector of weights, which we cannot do efficiently using Ajtai commitments.

Evaluation. In Section 5 we provide a concrete evaluation of the resulting system. We show that for a CCS relation that uses high degree gates, LatticeFold performs better than Hypernova. For relations that only use degree-2 gates the two systems are comparable. In addition, LatticeFold is post-quantum secure.

One reason LatticeFold performs so 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 $\mathcal{R}_q$. The same does not hold 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 greatly increases the complexity of folding. Furthermore, LatticeFold uses a small 64-bit field, whereas Hypernova uses a 256-bit field, due to the use of Pedersen commitments.

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.

Several post-quantum SNARKs were constructed from hash-based Merkle commitments. Some examples include Stark [BBHR18], Ligero [AHIV17], Aurora [Ben+19], and Brakedown [Gol+23]. 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+18; BLNS20; Alb+22; BCS23]. However, these systems are not competitive with the hash based systems listed above. Other lattice-based proof systems, such as [ENS20; LNP22], perform well for small statements, but their proof size is linear in the size of the witness.

More recently, 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.

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**. For vectors $\vec{u}, \vec{v}$ with same lengths, $\langle \vec{u}, \vec{v} \rangle$ denotes the inner product between $\vec{u}$ and $\vec{v}$. 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 $\bar{\mathcal{R}}^{\leq d}[X_1, \ldots, X_\mu]$ denotes 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 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 let 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 $(X^{d/t}-\zeta_j)$'s are *irreducible*. 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,

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

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

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

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 $\mathsf{vec}(\mathbf{a}) := (a_1, \ldots, a_d)^\top \in \mathbb{Z}_q^d$ to denote the coefficient vector of $\mathbf{a}$ and denote $\mathsf{vec}_i(\mathbf{a}) := a_i$ 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 $\mathsf{vec}(\vec{\mathbf{a}})$ to denote the matrix $(\mathsf{vec}(\mathbf{a}_1), \ldots, \mathsf{vec}(\mathbf{a}_m))^\top \in \mathbb{Z}_q^{m\times d}$ and denote $\mathsf{fvec}(\vec{\mathbf{a}}) \in \mathbb{Z}_q^{dm}$ as the vector which concatenates $\mathsf{vec}(\vec{\mathbf{a}})$'s row vectors. For every $i \in [d]$, we define $\mathsf{vec}_i(\vec{\mathbf{a}}) := (\mathsf{vec}_i(\mathbf{a}_1), \ldots, \mathsf{vec}_i(\mathbf{a}_m))^\top \in \mathbb{Z}_q^m$ as the i-th column of $\mathsf{vec}(\vec{\mathbf{a}})$. Define $\mathsf{Rot}(\mathbf{a}) := (\mathsf{vec}(\mathbf{a}), \mathsf{vec}(X \cdot \mathbf{a}), \ldots, \mathsf{vec}(X^{d-1} \cdot \mathbf{a})) \in \mathbb{Z}_q^{d\times d}$. We observe that

$$\mathsf{vec}(\mathbf{a} \cdot \mathbf{b}) = \mathsf{Rot}(\mathbf{a})\mathsf{vec}(\mathbf{b}) \tag{2}$$

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

$$\mathsf{Rot}(\mathbf{A}) := \begin{bmatrix} \mathsf{Rot}(\mathbf{A}_{1,1}) & \ldots & \mathsf{Rot}(\mathbf{A}_{1,m}) \\ \vdots & \ddots & \vdots \\ \mathsf{Rot}(\mathbf{A}_{\kappa,1}) & \ldots & \mathsf{Rot}(\mathbf{A}_{\kappa,m}) \end{bmatrix} \tag{3}$$

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

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 vector $\vec{B} := (B_1, \ldots, B_d) \in \mathbb{F}_{q^\tau}^d$, we define function $\mathsf{RotSum} : \mathcal{R}_q \times \mathbb{F}_{q^\tau}^d \to \mathbb{F}_{q^\tau}^d$ as*

$$\mathsf{RotSum}(\mathbf{a}, \vec{B}) := \sum_{i=1}^{d} B_i \cdot \mathsf{vec}(X^{i-1}\mathbf{a}) \,, \tag{4}$$

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

1. *For any $a \in \mathbb{Z}_q$ and any $\mathbf{b} \in \mathcal{R}_q$, we have that $\mathsf{vec}(a \cdot \mathbf{b}) = a \cdot \mathsf{vec}(\mathbf{b})$.*
2. *For any $\mathbf{a}, \mathbf{b} \in \mathcal{R}_q$ (where $\mathsf{vec}(\mathbf{b}) \in \mathbb{Z}_q^d \subseteq \mathbb{F}_{q^\tau}^d$), we have that*
$$\mathsf{RotSum}(\mathbf{a}, \mathsf{vec}(\mathbf{b})) = \mathsf{vec}(\mathbf{a} \cdot \mathbf{b}) \,.$$
3. *For any $\mathbf{a}, \mathbf{b}_1, \ldots, \mathbf{b}_\tau \in \mathcal{R}_q$, define $\vec{B} := \sum_{i=1}^{\tau} \mathsf{vec}(\mathbf{b}_i) \cdot X^{i-1} \in \mathbb{F}_{q^\tau}^d$, we have that*
$$\mathsf{RotSum}(\mathbf{a}, \vec{B}) = \sum_{i=1}^{\tau} \mathsf{vec}(\mathbf{a} \cdot \mathbf{b}_i) \cdot X^{i-1} \in \mathbb{F}_{q^\tau}^d \,.$$

Proof. The 1st claim is clear as $\mathcal{R}_q$ is a $\mathbb{Z}_q$-module. For the 2nd claim, observe that $\mathsf{RotSum}(\mathbf{a}, \mathsf{vec}(\mathbf{b})) = \mathsf{Rot}(\mathbf{a})\mathsf{vec}(\mathbf{b})$ by definition of $\mathsf{Rot}(\mathbf{a})$, thus the claim holds by Eqn. 2.

For the last claim, recall that $\mathsf{vec}(X^{i-1}\mathbf{a}) \in \mathbb{Z}_q^d$ is the i-th column $[\mathsf{Rot}(\mathbf{a})]_i$ of $\mathsf{Rot}(\mathbf{a})$ for all $i \in [d]$. Denote $\vec{B}_i := \sum_{j=1}^{\tau} b_{i,j} X^{j-1} \in \mathbb{F}_{q^\tau}$ as the i-th ($1 \le i \le d$) element of $\vec{B}_i$. We have that

$$\begin{aligned}\mathsf{RotSum}(\mathbf{a}, \vec{B}) &:= \sum_{i=1}^{d} \left(\sum_{j=1}^{\tau} b_{i,j} X^{j-1} \right) \cdot [\mathsf{Rot}(\mathbf{a})]_i = \sum_{j=1}^{\tau} \left(\sum_{i=1}^{d} b_{i,j} \cdot [\mathsf{Rot}(\mathbf{a})]_i \right) \cdot X^{j-1} \\ &= \sum_{j=1}^{\tau} \mathsf{RotSum}(\mathbf{a}, \mathsf{vec}(\mathbf{b}_i)) \cdot X^{j-1} = \sum_{j=1}^{\tau} \mathsf{vec}(\mathbf{a} \cdot \mathbf{b}_i) \cdot X^{i-1} \,.\end{aligned}$$

The 1st equality is by definition of RotSum; the 2nd equality is by rearranging the terms; the 3rd equality is by definition of RotSum; the last equality is by the 2nd claim in the lemma. $\square$

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 $\|\vec{\mathbf{f}}\|_2 \le \sqrt{dk}\|\vec{\mathbf{f}}\|_\infty$ 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 a bit and denote $\|\vec{\mathbf{f}}\|_\infty$ as the norm after lifting $\vec{\mathbf{f}}$ to $\mathcal{R}^k$. The lifting works by mapping the $\mathbb{Z}_q$-coefficients of $\vec{\mathbf{f}}$ to the interval $\lceil -q/2, q/2 \rfloor \in \mathbb{Z}$.*

2.1 Module SIS

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

Definition 2.1 (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 problem with norm bound $B_{\mathsf{SIS}} \approx \min(q, 2^{2\sqrt{\log(1.0045) d\kappa \log q}})$ is expected to have 128-bits of security [MR09; Esg+19; APS15]. For ease of exposition, 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's clear that $\mathsf{MSIS}^{\infty;q}_{\kappa,m,B}$ is at least as hard as $\mathsf{MSIS}^q_{\kappa,m,\sqrt{dm}B}$.

Definition 2.2 (Module SIS with ℓ_∞-norms [ACK21])**.** *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 $\mathsf{MSIS}^{\infty;q}_{\kappa,m,B}$ 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 $\mathcal{R}_q$.*

2.2 The Ajtai Compact Commitment

A commitment scheme allows one to compute a string cm from a message m so that it can later open cm to m. Specifically, 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 m and randomness r, and outputs a commitment cm. We say that CM is **compact** if the commitment cm is shorter than the committed message m. We say CM is **binding** if it is hard to find a commitment cm and two different openings (m_1, r_1), (m_2, r_2) such that $\mathsf{cm} = \mathsf{Commit}(\mathsf{pp}, m_1, r_1) = \mathsf{Commit}(\mathsf{pp}, m_2, r_2)$. We say that CM is **hiding** if cm does not reveal information about m.

We review the Ajtai commitment scheme [Ajt96] 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 [Ajt96])**.** *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 if the MSIS problem $\mathsf{MSIS}^{\infty,q}_{\kappa,m,2B}$ is hard. Suppose not, i.e., 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 $\|\vec{\mathbf{x}}_1 - \vec{\mathbf{x}}_2\|_\infty < 2B$.

2.3 Sum-Checks and Multilinear Extensions over Rings

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

Definition 2.3. *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 $\mathcal{C}_{\mathsf{small}} \subseteq \mathcal{R}_q$ whose elements have small norm (after lifting to $\mathcal{R}$). The **expansion factor** of $\mathcal{C}_{\mathsf{small}} \subseteq \mathcal{R}_q$ is

$$\|\mathcal{C}_{\mathsf{small}}\|_{\mathsf{op}} := \sup_{\rho\in\mathcal{C}_{\mathsf{small}},\mathbf{v}\in\mathcal{R}} \frac{\|\rho\mathbf{v}\|_\infty}{\|\mathbf{v}\|_\infty}\,. \tag{5}$$

Here, the multiplication $\rho \times \mathbf{v}$ is performed over the ring $\mathcal{R}$ where we lift $\rho \in \mathcal{R}_q$ to $\mathcal{R}$ (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* $\mathbf{u}, \mathbf{v} \in \mathcal{R}$*, we have that*

$$\frac{\|\mathbf{u}\mathbf{v}\|_\infty}{\|\mathbf{v}\|_\infty} \le d \cdot \|\mathbf{u}\|_\infty\,.$$

The next lemma shows that an element in $\mathcal{R}_q$ is invertible if its norm (after lifting to $\mathcal{R}$) is small. This implies that we can find large strong sampling sets *in* $\mathcal{R}_q$ with small norm elements. Because given two distinct small elements (with norm less than $q/4$), their difference still has a small norm (as there is no modulus overflow) and thus is invertible.

Lemma 2.3 (Corollary 1.2 of [LS18])**.** *Let* $d \ge t > 1$ *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* $\mathcal{R}$*.*

Generalized Schwartz-Zippel Lemma. We recall a generalization of the Schwartz-Zippel lemma to the 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 following protocol for proving that $s = \sum_{\vec{\mathbf{b}} \in \{0,1\}^\mu} f(\vec{\mathbf{b}})$ 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 $h_i(0) + h_i(1) \overset{?}{=} h_{i-1}(\mathbf{r}_{i-1})$ 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 $h_\mu(\mathbf{r}_i) \overset{?}{=} f(\mathbf{r}_1, \ldots, \mathbf{r}_\mu)$.*

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

Multilinear extensions over rings. We define the multilinear extensions over rings and show its uniqueness.

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 $eq(\vec{\mathbf{b}}, \vec{\mathbf{X}}) := \prod_{i=1}^{\mu} \left[(1 - \vec{\mathbf{b}}_i)(1 - \vec{\mathbf{X}}_i) + \vec{\mathbf{b}}_i \vec{\mathbf{X}}_i\right]$.

Similar to the field setting, the multilinear extensions over rings have two nice properties. First, the multilinear extension of a Boolean function f is *unique*. Second, we can represent a multilinear polynomial g uniquely in its interpolated form, that is, it can be represented as an interpolation of g's evaluations on the Boolean hypercube.

Lemma 2.6 (Uniqueness of MLEs over Rings [CCKP19])**.** *The multilinear extension* $\mathsf{mle}[f]$ *of* $f : \{0,1\}^\mu \to \bar{\mathcal{R}}$ *is the unique multilinear polynomial that agrees with* f *on set* $\{0,1\}^\mu$.

Proof. See Appendix B2 of [CCKP19]. Note that the proof only exploits the properties of rings (i.e., commutativity and distributivity of addition and multiplication, and the existence of identity 1) and thus easily extends to the ring setting. $\square$

From Lemma 2.6, we extend Lemma 1 of Hypernova [KS23b] to the ring setting, i.e., we can represent a multilinear polynomial in its multilinear extension form.

Corollary 2.1. *For a multilinear polynomial* $f \in \bar{\mathcal{R}}^{\leq 1}[X_1, \ldots, X_\mu]$ *over a ring* $\bar{\mathcal{R}}$, *we have* $f(\vec{\mathbf{X}}) = \sum_{\vec{\mathbf{b}} \in \{0,1\}^\mu} f(\vec{\mathbf{b}}) \cdot eq(\vec{\mathbf{b}}, \vec{\mathbf{X}})$.

Proof. Denote $g(\vec{\mathbf{X}}) := \sum_{\vec{\mathbf{b}} \in \{0,1\}^\mu} f(\vec{\mathbf{b}}) eq(\vec{\mathbf{b}}, \vec{\mathbf{X}})$. Note that f and g have the same evaluations on set $\{0,1\}^\mu$. By Lemma 2.6, f and g must be the same polynomial. $\square$

2.4 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 $(\mathbb{x}_1, \mathbb{w}_1) \in \mathcal{R}_1$. Let us recall the formal 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* $\mathbb{x}_2 := \perp$; *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 knowing the public parameter $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$, the protocol execution $(\mathbb{x}_2, \mathbb{w}_2) \leftarrow \langle \mathsf{P}(\mathsf{pp}, \mathbb{x}_1, \mathbb{w}_1), \mathsf{V}(\mathsf{pp}, \mathbb{x}_1)\rangle$ satisfies that $(\mathsf{pp}, \mathbb{x}_2, \mathbb{w}_2)$ is in $\mathcal{R}_2$ if $(\mathsf{pp}, \mathbb{x}_1, \mathbb{w}_1)$ is in $\mathcal{R}_1$.*

Knowledge soundness. *For every expected polynomial-time adversaries $\mathcal{A}$ and P^* there is an expected polynomial-time extractor Ext such that given $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$ and $(\mathbb{x}_1, \mathsf{st}) \leftarrow \mathcal{A}(\mathsf{pp})$, it holds that*

$$\Pr\big[(\mathsf{pp}, \mathbb{x}_1, \mathsf{Ext}(\mathsf{pp}, \mathbb{x}_1, \mathsf{st})) \in \mathcal{R}_1\big] \approx \Pr\big[(\mathsf{pp}, \langle \mathsf{P}^*(\mathsf{pp}, \mathbb{x}_1, \mathsf{st}), \mathsf{V}(\mathsf{pp}, \mathbb{x}_1)\rangle) \in \mathcal{R}_2\big] .$$

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*

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

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.

Theorem 2.2 (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 2.3 (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 property of Defn. 2.5 should hold for expected polynomial-time adversaries/extractors. This is for the convenience of proving the composition theorems. Looking ahead, this implies that the MSIS hardness assumption in Defn. 2.2 should also hold for expected poly-time adversaries. This is without loss of generality because the MSIS assumption is falsifiable and [LPS24] (Appendix A) shows that if a falsifiable assumption holds for strict PPT adversaries (namely 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 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 relation $\mathcal{R}_{\mathsf{acc}}$.*

3 A Folding Scheme for Ajtai Commitment Openings

In this section, we construct a folding scheme for the Ajtai commitment opening relation. In Sect. 3.1, we define an algebraic relation $\mathcal{R}^B_{\mathsf{cm}}$ over $\mathcal{R}_q$ that captures the commitment opening relation, and augment it to $\mathcal{R}^B_{\mathsf{eval}}$ with a multilinear evaluation statement. In Sect. 3.2, we construct a reduction of knowledge from relation $\mathcal{R}^B_{\mathsf{eval}} \times \mathcal{R}^B_{\mathsf{cm}}$ to $\mathcal{R}^B_{\mathsf{eval}}$, which leads to a folding scheme for the Ajtai commitment opening relation. In Sect. 3.3, we describe an optimization that enables us to choose small prime modulus q for improved efficiency.

Designing a folding scheme for the relation $\mathcal{R}^B_{\mathsf{cm}}$ is the core challenge in building a IVC/PCD scheme from Ajtai commitments. However, on its own, this is not sufficient for an IVC/PCD. The relation would need 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 $\mathcal{R}^B_{\mathsf{cmccs}}$ (Eqn. 25) that is sufficient to use in IVC/PCD where each step of the computation is expressed as a CCS relation.

3.1 The Relation for Commitment Openings

In this section, we explain how to represent the Ajtai commitment opening relation in a way that enables efficient folding. The core idea (inspired and adapted from [BLS19]) is to interpret the norm bound constraint as a product relation over rings. Let $\mathcal{R}_q$ denote the ring $\mathcal{R}_q := \mathcal{R}/q\mathcal{R}$ where $\mathcal{R} := \mathbb{Z}[X]/(X^d+1)$ and q is a prime. Let $\mathsf{pp} := (\kappa, m, B, \mathbf{A})$ be the public parameters where $\mathbf{A} \in \mathcal{R}_q^{\kappa \times m}$ is the sampled MSIS matrix and the norm bound $B < q/2$. We define the relation $\mathcal{R}^B_{\mathsf{MSIS}^\infty}$ for Ajtai commitment openings (Sect. 2.2) as

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

Here cm is an Ajtai commitment, and $\vec{\mathbf{x}}$ is a vector of m elements over $\mathcal{R}$. Since $B < q/2$, we can uniquely represent $\vec{\mathbf{x}}$ as an element 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 can rewrite $\mathcal{R}^B_{\mathsf{MSIS}^\infty}$ as

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

We observe that $\mathcal{R}^B_{\mathsf{MSIS}^\infty}$ is equivalent to the following relation $\mathcal{R}^B_{\mathsf{SIS}^\infty}$ over $\mathbb{Z}_q$: Let $\bar{\mathbf{A}} := \mathsf{Rot}(\mathbf{A})$ denote the rotation matrix of $\mathbf{A}$ (defined in Eqn. 3) such that the coefficient embedding of $\mathsf{cm} = \mathbf{A}\vec{\mathbf{x}}$ is exactly $\bar{\mathsf{cm}} = \bar{\mathbf{A}}\vec{x}$ (where $\vec{x} = \mathsf{fvec}(\vec{\mathbf{x}}) \in \mathbb{Z}_q^{md}$ is concatenation of the coefficient embeddings of $\vec{\mathbf{x}} \in \mathcal{R}_q^m$). The relation $\mathcal{R}^B_{\mathsf{SIS}^\infty}$ is

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

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

$$\mathcal{R}^B_{\mathsf{SISProd}} := \left\{ (\mathsf{pp}, \bar{\mathsf{cm}} \in \mathbb{Z}_q^{\kappa d}; \vec{x} \in \mathbb{Z}_q^{md}) \ : \ \begin{array}{c} (\bar{\mathsf{cm}} = \bar{\mathbf{A}}\vec{x}) \wedge \\ \left(\vec{x} \circ \left[\bigcirc_{i=1}^{B-1} (\vec{x} - \vec{i}) \circ (\vec{x} + \vec{i})\right] = \vec{0}\right) \end{array} \right\} \tag{6}$$

where $\vec{i} \in \mathbb{Z}_q^{md}$ is i multiplied by the identity vector $I_{md} := 1^{md}$.

The final step involves replacing the Hadamard product relation over $\mathbb{Z}_q$ to a Hadamard product relation over $\mathcal{R}_q$. For brevity, we assume that $q \equiv 1 \bmod 2d$ (Eqn. 1) and thus $\mathcal{R}_q \cong \mathbb{Z}_q^d$. (In Section 3.3 we will explain how to generalize to other primes.) Therefore, there are two ways to understand $\vec{x} \in \mathbb{Z}_q^{md}$: first, it can be understood as the coefficient embeddings of some $\vec{\mathbf{f}} \in \mathcal{R}_q^m$; alternatively, it can be understood as the NTT representations of some $\hat{\mathbf{f}} \in \mathcal{R}_q^m$, that is, $\mathsf{NTT}(\hat{\mathbf{f}}) = \mathsf{vec}(\vec{\mathbf{f}})$. Moreover, the *Hadamard product* between the *NTT forms* of two ring elements is equivalent to the *multiplication* of the two ring elements. E.g., we have that $\vec{x} \circ \vec{x} = \mathsf{NTT}(\hat{\mathbf{f}}) \circ \mathsf{NTT}(\hat{\mathbf{f}}) \cong \hat{\mathbf{f}} \circ \hat{\mathbf{f}}$ where $\vec{x} \circ \vec{x}$ is a Hadamard product over $\mathbb{Z}_q$ while $\hat{\mathbf{f}} \circ \hat{\mathbf{f}}$ is over $\mathcal{R}_q$. Thus, we can transform $\mathcal{R}^B_{\mathsf{SISProd}}$ to the following relation $\mathcal{R}^B_{\mathsf{cm}}$ over $\mathcal{R}_q$:

$$\mathcal{R}^B_{\mathsf{cm}} := \left\{ (\mathsf{pp}, \mathsf{cm} \in \mathcal{R}_q^{\kappa}; \vec{\mathbf{f}} \in \mathcal{R}_q^{m}) \ : \ \begin{array}{c} (\mathsf{cm} = \mathbf{A}\vec{\mathbf{f}}) \wedge \\ \left(\hat{\mathbf{f}} \circ \left[\bigcirc_{i=1}^{B-1} (\hat{\mathbf{f}} - \hat{i}) \circ (\hat{\mathbf{f}} + \hat{i})\right] = \hat{0}\right) \end{array} \right\}. \tag{7}$$

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

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, this is because in our folding scheme, the verifier will run a sum-check to reduce the norm bound constraint in $\mathcal{R}^B_{\mathsf{cm}}$ to an evaluation statement, hence we need to include such 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

$$\mathcal{R}^B_{\mathsf{eval}} := \left\{ (\mathsf{pp}; (\vec{\mathbf{r}}, \mathbf{v}, \mathsf{cm}) \in \mathcal{R}_q^{\log m} \times \mathcal{R}_q \times \mathcal{R}_q^{\kappa}; \vec{\mathbf{f}} \in \mathcal{R}_q^{m}) \ : \ \begin{array}{c} (\mathsf{pp}, \mathsf{cm}; \vec{\mathbf{f}}) \in \mathcal{R}^B_{\mathsf{cm}} \\ \wedge \mathsf{mle}\left[\hat{\mathbf{f}}\right](\vec{\mathbf{r}}) = \mathbf{v} \end{array} \right\}, \tag{8}$$

here $\mathsf{mle}\left[\hat{\mathbf{f}}\right] \in \mathcal{R}_q^{\leq 1}[X_1, \ldots, X_{\log m}]$ is the multilinear extension (Defn. 2.4) of $\hat{\mathbf{f}}$, where $\mathsf{NTT}(\hat{\mathbf{f}})$ agrees with the coefficient embedding matrix $\mathsf{vec}(\vec{\mathbf{f}}) \in \mathbb{Z}_q^{m \times d}$.

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 (Defn. 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, which consists of three steps below.

Step 1: Expansion. First, we reduce the relation $\mathcal{R}^B_{\mathsf{eval}} \times \mathcal{R}^B_{\mathsf{cm}}$ to the relation $\mathcal{R}^B_{\mathsf{eval}} \times \mathcal{R}^B_{\mathsf{eval}}$ via a reduction of knolwedge Π_{cm} (Fig. 1) from $\mathcal{R}^B_{\mathsf{cm}}$ to $\mathcal{R}^B_{\mathsf{eval}}$.

Step 2: Decomposition. Next, using a decomposition protocol Π_{dec} (Fig. 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 $k \in \mathbb{N}$ where $b^k = B$. Note that b, k are parameters that can be chosen dynamically.

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} (Fig. 3).

By the composition theorems for reductions of knowledge (Theorem 2.2, Theorem 2.3), 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 formally state the result in Theorem 3.1.

Before describing the three protocols, we define the common setup.

Setup and notation. Let $\mathcal{R}_q$ be 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 $\mathsf{vec}(\vec{\mathbf{f}})$ where $\mathsf{vec}(\vec{\mathbf{f}}) \in \mathbb{Z}_q^{m \times d}$ is the coefficient embedding matrix of $\vec{\mathbf{f}}$ (defined near Eqn. 2).

We first show a useful lemma. Informally, in the special case where $\mathcal{R}_q \cong \mathbb{Z}_q^d$, it states that a multilinear evaluation of a polynomial $\hat{\mathbf{f}}$ over $\mathcal{R}_q$ (where $\mathsf{NTT}(\hat{\mathbf{f}})$ corresponds to the coefficient embedding of a vector $\vec{\mathbf{f}} \in \mathcal{R}_q^m$), is isomorphic to the multilinear evaluations of the d polynomials $f_1, \ldots, f_d$ (over $\mathbb{Z}_q$) at a single point (where the coefficients of $f_1, \ldots, f_d$ are the columns of the coefficient embedding matrix $\mathsf{vec}(\vec{\mathbf{f}})$). Looking ahead, this lemma is helpful to ensure that $\mathbf{v} = \mathsf{mle}\left[\hat{\mathbf{f}}\right](\vec{\mathbf{r}})$ continues to be consistent with the witness vector $\vec{\mathbf{f}}$ after folding.

Lemma 3.1. *Let $m \in \mathbb{N}$ be a power-of-two and $\mathcal{R}_q \cong \mathbb{F}_{q^\tau}^{d/\tau}$ for some $\tau \in \mathbb{N}$ where $\tau \mid d$. For any $\vec{\mathbf{f}} \in \mathcal{R}_q^m$ and any vector $\vec{\mathbf{r}} \in \mathcal{R}_q^{\log m}$ such that $\mathsf{NTT}(\vec{\mathbf{r}}) = \underbrace{(\vec{\mathbf{r}}^*, \ldots, \vec{\mathbf{r}}^*)}_{d/\tau}$ (where $\vec{\mathbf{r}}^* \in \mathbb{F}_{q^\tau}^{\log m}$), 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 $\mathsf{NTT}(\hat{\mathbf{f}}) = \mathsf{vec}(\vec{\mathbf{f}})$. We have that $\mathsf{mle}\left[\hat{\mathbf{f}}\right](\vec{\mathbf{r}}) \cong \mathsf{mle}\left[\mathsf{vec}(\vec{\mathbf{f}})\right](\vec{\mathbf{r}}^*)$ where*

$$\mathsf{mle}\left[\hat{\mathbf{f}}\right](\vec{\mathbf{r}}) := \left(\mathsf{mle}\left[\hat{\mathbf{f}}_1\right](\vec{\mathbf{r}}), \ldots, \mathsf{mle}\left[\hat{\mathbf{f}}_\tau\right](\vec{\mathbf{r}})\right) \in \mathcal{R}_q^\tau,$$
$$\mathsf{mle}\left[\mathsf{vec}(\vec{\mathbf{f}})\right](\vec{\mathbf{r}}^*) := \left(\mathsf{mle}\left[\mathsf{vec}_1(\vec{\mathbf{f}})\right](\vec{\mathbf{r}}^*), \ldots, \mathsf{mle}\left[\mathsf{vec}_d(\vec{\mathbf{f}})\right](\vec{\mathbf{r}}^*)\right) \in \mathbb{F}_{q^\tau}^d.$$

Recall that $\mathsf{mle}[\cdot]$ denotes multilinear extensions (Defn. 2.4) and $\mathsf{vec}_i(\vec{\mathbf{f}}) \in \mathbb{Z}_q^m$ is the i-th $(1 \le i \le d)$ column of the coefficient embedding matrix $\mathsf{vec}(\vec{\mathbf{f}})$.

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

$$\mathsf{NTT}(\hat{\mathbf{f}}) = (\mathsf{NTT}(\hat{\mathbf{f}}_1), \ldots, \mathsf{NTT}(\hat{\mathbf{f}}_\tau)) = (\mathsf{vec}_1(\vec{\mathbf{f}}), \ldots, \mathsf{vec}_d(\vec{\mathbf{f}})). \tag{9}$$

Also observe that

$$\left(\mathsf{mle}\left[\hat{\mathbf{f}}_1\right](\vec{\mathbf{r}}), \ldots, \mathsf{mle}\left[\hat{\mathbf{f}}_\tau\right](\vec{\mathbf{r}})\right) = \left(\left\langle \hat{\mathbf{f}}_j, \bigotimes_{i=1}^{\log m} (\vec{\mathbf{r}}_i, 1 - \vec{\mathbf{r}}_i) \right\rangle\right)_{j=1}^{\tau} \tag{10}$$

where $\bigotimes$ denotes tensor product over $\mathcal{R}_q$. Thus the lemma holds by the Chinese Remainder Theorem. □

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 Sect. 3.3, we generalize to an arbitrary prime modulus.

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

By the parallel composition theorem (Theorem 2.3), in order to reduce from $\mathcal{R}_{\mathsf{eval}}^B \times \mathcal{R}_{\mathsf{cm}}^B$ to $\mathcal{R}_{\mathsf{eval}}^B \times \mathcal{R}_{\mathsf{eval}}^B$, it suffices to construct a protocol that reduces $\mathcal{R}_{\mathsf{cm}}^B$ (Eqn. 7) to $\mathcal{R}_{\mathsf{eval}}^B$ (Eqn. 8). We describe the protocol Π_{cm} in Fig. 1.

Lemma 3.2. *Π_{cm} is a reduction of knowledge from $\mathcal{R}_{\mathsf{cm}}^B$ to $\mathcal{R}_{\mathsf{eval}}^B$ for any $B \in \mathbb{N}$.*

Proof. *Public reducibility:* Given instance $\mathbb{x} = \mathsf{cm}$ and transcript $\mathbf{v}$, one can output $\mathbb{x}_o = ((0^{\log m}, \mathbf{v}, \mathsf{cm}))$.

Completeness: Given $(\mathsf{pp}, \mathsf{cm}; \vec{\mathbf{f}}) \in \mathcal{R}_{\mathsf{cm}}^B$, the honest prover can compute and send $\mathbf{v} := \mathsf{mle}\left[\hat{\mathbf{f}}\right](0^{\log m})$ such that $((0^{\log m}, \mathbf{v}, \mathsf{cm}); \vec{\mathbf{f}}) \in \mathcal{R}_{\mathsf{eval}}^B$. The honest verifier will output instance $(0^{\log m}, \mathbf{v}, \mathsf{cm})$ and the honest prover will output $\vec{\mathbf{f}}$.

Knowledge soundness: By definition of $\mathcal{R}_{\mathsf{eval}}^B$ (Eqn. 8), given any $((\vec{\mathbf{r}}, \mathbf{v}, \mathsf{cm}); \vec{\mathbf{f}}) \in \mathcal{R}_{\mathsf{eval}}^B$, we can extract witness $(\mathsf{cm}; \vec{\mathbf{f}})$ that is in the relation $\mathcal{R}_{\mathsf{cm}}^B$. □

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

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

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

Intuitively, the decomposition step split the two witness vectors of norms less than B into $2k$ witness vectors with a much smaller norm b, so that later (in the folding step) they can be folded back to a vector with norm less than B.

By Theorem 2.3, 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 essentially $\Pi_{\mathsf{dec}} := \Pi^*_{\mathsf{dec}} \times \Pi^*_{\mathsf{dec}}$ 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 $(\mathcal{R}^b_{\mathsf{hom}})^k$. Here $\mathcal{R}^B_{\mathsf{hom}}$ is a generalization of $\mathcal{R}^B_{\mathsf{eval}}$ (Eqn. 8) defined as

$$\mathcal{R}^B_{\mathsf{hom}} := \left\{ (\mathcal{L}; (\vec{\mathbf{r}} \in \mathcal{R}_q^{\log m}, \mathbf{v} \in \mathcal{R}_q, y \in \mathcal{Y}); \vec{\mathbf{f}} \in \mathcal{R}_q^m) \ : \ \begin{array}{c} y = \mathcal{L}(\vec{\mathbf{f}}) \wedge \|\vec{\mathbf{f}}\|_\infty < B \\ \wedge \mathsf{mle}\left[\hat{\mathbf{f}}\right](\vec{\mathbf{r}}) = \mathbf{v} \end{array} \right\}, \tag{11}$$

where $\mathcal{L}$ is any $\mathcal{R}_q$-module homomorphism from $\mathcal{R}_q^m$ to an $\mathcal{R}_q$-module $\mathcal{Y}$. This $\mathcal{L}$ is treated as a public parameter. Clearly, $\mathcal{R}^B_{\mathsf{eval}}$ (Eqn. 8) 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 decomposition of $\vec{\mathbf{f}}$ 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

$$\vec{\mathbf{f}} = \vec{\mathbf{F}} \cdot \left[1, b, b^2, \ldots, b^{k-1}\right]^\top = \sum_{i=0}^{k-1} b^i \cdot \vec{\mathbf{f}}_i \,. \tag{12}$$

For example, for $k = 2$, $b = \lceil\sqrt{B}\,\rceil$, and $m = 1$, 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 $|c_i| < b$ and $|d_i| < b$, and $f = f_0 + b f_1$.

With this notation in place, we describe the protocol Π^*_{dec} in Fig. 2. Before proving that Π^*_{dec} is a reduction of knowledge, we state a useful lemma. Informally, it states that

Input: $\mathbb{x} := (\vec{\mathbf{r}}, \mathbf{v}, y)$ and $\mathbb{w} := \vec{\mathbf{f}}$.
Output: $[\mathbb{x}_i = (\vec{\mathbf{r}}, \mathbf{v}_i, y_i), \mathbb{w}_i = \vec{\mathbf{f}}_i]_{i=0}^{k-1}$.
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 $[y_i, \mathbf{v}_i]_{i=0}^{k-1}$ where
$$y_i := \mathcal{L}(\vec{\mathbf{f}}_i)\,, \qquad \mathbf{v}_i := \mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}})$$
for every $i \in [0, k)$.
2. V checks that $\sum_{i=0}^{k-1} b^i \cdot y_i \overset{?}{=} y$, and $\sum_{i=0}^{k-1} b^i \cdot \mathbf{v}_i \overset{?}{=} \mathbf{v}$.
3. V outputs $[\mathbb{x}_i = (\vec{\mathbf{r}}, \mathbf{v}_i, y_i)]_{i=0}^{k-1}$. 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 $(\mathcal{R}^b_{\mathsf{hom}})^k$.

the folded instance will be "consistent" with the folded witness if the input instances and witnesses are "consistent" before folding.

Lemma 3.3. *Given any $\ell \in \mathbb{N}$ and 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 any $\mathcal{R}_q$-module homomorphism. Given any $[\rho_i]_{i=1}^{\ell} \in \mathcal{R}_q^{\ell}$ and any $[\mathbf{v}_i, y_i; \vec{\mathbf{f}}_i]_{i=1}^{\ell}$ such that $y_i = \mathcal{L}(\vec{\mathbf{f}}_i)$ and $\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}) = \mathbf{v}_i$ for all $i \in [\ell]$. Set $\mathbf{v}_o, y_o, \vec{\mathbf{f}}_o$ such that*
$$\mathsf{NTT}(\mathbf{v}_o) = \sum_{i=1}^{\ell} \mathsf{RotSum}(\rho_i, \mathsf{NTT}(\mathbf{v}_i))\,, \qquad y_o := \sum_{i=1}^{\ell} \rho_i \cdot y_i\,, \qquad \vec{\mathbf{f}}_o := \sum_{i=1}^{\ell} \rho_i \cdot \vec{\mathbf{f}}_i\,,$$
where RotSum is defined in Lemma 2.1. Then we have that $y_o = \mathcal{L}(\vec{\mathbf{f}}_o)$ and $\mathsf{mle}\left[\hat{\mathbf{f}}_o\right](\vec{\mathbf{r}}) = \mathbf{v}_o$.

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 1st equality follows by definition of $\vec{\mathbf{f}}_o$; the 2nd equality holds by the homomorphic property of $\mathcal{L}$; the 3rd equality holds as $y_i = \mathcal{L}(\vec{\mathbf{f}}_i)$ for all $i \in [0, k)$ by the premise of the lemma; the last equality holds by definition of y_o.

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 $\mathsf{NTT}(\mathbf{v}_o) =$

$\mathsf{vec}(\bar{\mathbf{v}}_o)$ and $\mathsf{NTT}(\mathbf{v}_i) = \mathsf{vec}(\bar{\mathbf{v}}_i)$ for all $i \in [\ell]$. We argue that

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

The 1st equality follows by definition of $\mathsf{vec}(\bar{\mathbf{v}}_o)$; the 2nd equality holds given how $\mathbf{v}_o$ is defined in the lemma; the 3rd equality is by definition of $\mathsf{vec}(\bar{\mathbf{v}}_i)$; the 4th equality holds by the premise that $\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}) = \mathbf{v}_i$ and by Lemma 3.1, which states that $\mathsf{vec}(\bar{\mathbf{v}}_i) = \mathsf{NTT}(\mathbf{v}_i) = \mathsf{mle}\left[\mathsf{vec}(\vec{\mathbf{f}}_i)\right][\vec{\mathbf{r}}] = \left\langle \mathsf{vec}(\vec{\mathbf{f}}_i), \mathsf{tensor}(\vec{\mathbf{r}}) \right\rangle$; the 5th equality holds by rearranging the terms and by the property of inner products; the 6th equality is by the 2nd claim in Lemma 2.1 (i.e. $\mathsf{RotSum}(\mathbf{a}, \mathsf{vec}(\mathbf{b})) = \mathsf{vec}(\mathbf{a} \cdot \mathbf{b})$ for any $\mathbf{a}, \mathbf{b} \in \mathcal{R}_q$); the 7th equality is by additivity of inner products and coefficient embedding; the last equality is by definition of $\vec{\mathbf{f}}_o$. Therefore, by Lemma 3.1, we have that

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

□

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

Lemma 3.4. *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 3.5 and Lemma 3.6 below.

Lemma 3.5. *Π^*_{dec} satisfies public reducibility and completeness.*

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

Completeness: Let $(\mathbb{x} = (\vec{\mathbf{r}}, \mathbf{v}, y); \mathbb{w} := \vec{\mathbf{f}}) \leftarrow \mathcal{A}(\mathsf{pp})$ denote adversary $\mathcal{A}$'s chosen input for $\mathcal{R}_1 := \mathcal{R}^B_{\mathsf{hom}}$ where $\mathsf{pp} := \mathcal{L} \leftarrow \mathsf{Setup}(1^\lambda)$ 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 $\vec{\mathbf{F}} := (\vec{\mathbf{f}}_0, \dots, \vec{\mathbf{f}}_{k-1}) \leftarrow \mathsf{split}_{b,k}(\vec{\mathbf{f}})$, and sends $y_i = \mathcal{L}(\vec{\mathbf{f}}_i)$, $\mathbf{v}_i = \mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}})$ for every $i \in [0, k)$.

2. V checks that $y \overset{?}{=} \sum_{i=0}^{k-1} b^i \cdot y_i$ and $\mathbf{v} \overset{?}{=} \sum_{i=0}^{k-1} b^i \cdot \mathbf{v}_i$. It outputs $\perp$ and halts if the check fails.

3. P outputs $[\vec{\mathbf{f}}_i]_{i=0}^{k-1}$. V accepts and outputs $[\vec{\mathbf{r}}, \mathbf{v}_i, y_i]_{i=0}^{k-1}$.

We first shows that V accepts. 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 holds 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 $\vec{\mathbf{f}} = \sum_{i=0}^{k-1} b^i \cdot \vec{\mathbf{f}}_i$ (Eqn 12) and $\mathcal{L}(\vec{\mathbf{f}}) = y$ by the assumption that $(\mathcal{L}, \mathbb{x}; \mathbb{w}) \in \mathcal{R}^B_{\mathsf{hom}}$.

Similarly, we have that

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

The 1st equality is by definition of $\mathbf{v}_i$; the 2nd equality holds because the map $g_{\vec{\mathbf{r}}}(f) := f(\vec{\mathbf{r}})$ is linearly homomorphic (over $\mathbb{Z}_q$) and because $\hat{\mathbf{f}} = \sum_{i=0}^{k-1} b^i \cdot \hat{\mathbf{f}}_i$ (which is implied by the fact that $\vec{\mathbf{f}} = \sum_{i=0}^{k-1} b^i \cdot \vec{\mathbf{f}}_i$ and by the 1st claim in Lemma 2.1). The last equality holds because $(\mathcal{L}, \mathbb{x}; \mathbb{w}) \in \mathcal{R}^B_{\mathsf{hom}}$ by assumption. Thus V will accept and output the desired instances.

Next, we show that $(\mathcal{L}, [(\vec{\mathbf{r}}, \mathbf{v}_i, y_i); \vec{\mathbf{f}}_i]_{i=0}^{k-1})$ is in $\mathcal{R}_2 := (\mathcal{R}^b_{\mathsf{hom}})^k$. By definition of $\mathsf{split}_{b,k}(\vec{\mathbf{f}})$ (Eqn. 12), we have that $\|\vec{\mathbf{f}}_i\|_\infty < b$ for all $i \in [0, k)$. Also recall that $y_i = \mathcal{L}(\vec{\mathbf{f}}_i)$, $\mathbf{v}_i = \mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}})$ for every $i \in [0, k)$. Thus $(\mathcal{L}, (\vec{\mathbf{r}}, \mathbf{v}_i, y_i); \vec{\mathbf{f}}_i) \in \mathcal{R}^b_{\mathsf{hom}}$ for all $i \in [0, k)$ and completeness holds. □

Lemma 3.6. Π^*_{dec} *satisfies knowledge soundness.*

Proof. Let $(\mathbb{x} := (\vec{\mathbf{r}}, \mathbf{v}, y); \mathsf{state}) \leftarrow \mathcal{A}(\mathsf{pp})$ denote adversary $\mathcal{A}$'s chosen input instance for $\mathcal{R}_1 := \mathcal{R}^B_{\mathsf{hom}}$, where $\mathsf{pp} := \mathcal{L} \leftarrow \mathsf{Setup}(1^\lambda)$ 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 $(\mathbb{x}_o, \mathbb{w}_o) := [(\vec{\mathbf{r}}, \mathbf{v}_i, y_i); \vec{\mathbf{f}}_i]_{i=0}^{k-1}$ 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

$$\mathbb{w} := \vec{\mathbf{f}} := \sum_{i=0}^{k-1} b^i \cdot \vec{\mathbf{f}}_i\,. \tag{13}$$

To prove knowledge soundness, it suffices to 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 $(\vec{\mathbf{r}}, \mathbf{v}, y; \vec{\mathbf{f}}) \in \mathcal{R}_1 := \mathcal{R}^B_{\mathsf{hom}}$.

Since V accepts, we have that $y = \sum_{i=0}^{2k-1} b^i \cdot y_i$ and $\mathbf{v} = \sum_{i=0}^{2k-1} b^i \cdot \mathbf{v}_i$. Recall that $y_i = \mathcal{L}(\vec{\mathbf{f}}_i)$ and $\mathbf{v}_i = \mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}})$ for all $i \in [0,k)$ by assumption, thus by Lemma 3.3, we have that $y = \mathcal{L}(\vec{\mathbf{f}})$ and $\mathbf{v} = \mathsf{mle}\left[\hat{\mathbf{f}}\right](\vec{\mathbf{r}})$. Moreover, note that $\|\vec{\mathbf{f}}_i\|_\infty < b$ for all $i \in [0,k)$ because $(\vec{\mathbf{r}}, \mathbf{v}_i, y_i; \vec{\mathbf{f}}_i)$ 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, $y = \mathcal{L}(\vec{\mathbf{f}})$, $\mathbf{v} = \mathsf{mle}\left[\hat{\mathbf{f}}\right](\vec{\mathbf{r}})$ and $\|\vec{\mathbf{f}}\|_\infty < B$ and thus $(\vec{\mathbf{r}}, \mathbf{v}, y; \vec{\mathbf{f}}) \in \mathcal{R}^B_{\mathsf{hom}}$ as desired. □

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

Finally, we describe the core protocol Π_{fold} that folds $2k$ instance-witness pairs of $\mathcal{R}^b_{\mathsf{eval}}$ into a single instance-witness pair in $\mathcal{R}^B_{\mathsf{eval}}$. Intuitively, it 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 from a strong sampling set $\mathcal{C}_{\mathsf{small}}$ (over $\mathcal{R}_q$).

More generally, the protocol is a reduction of knowledge from $(\mathcal{R}^b_{\mathsf{hom}})^{2k}$ to $\mathcal{R}^B_{\mathsf{hom}}$ with a further restriction that the public parameter pp, which is a *sampled* homomorphism $\mathcal{L}$, is **binding** on the domain $\{\vec{\mathbf{f}} : \|\vec{\mathbf{f}}\|_\infty < B\}$. To distinguish the difference, in the following context, we use $\mathcal{R}^b_{\mathsf{bind}}$ to denote this condition. Note that $\mathcal{R}^B_{\mathsf{eval}}$ (Eqn. 8) is a special case of $\mathcal{R}^B_{\mathsf{bind}}$ where the sampled homomorphism $\mathcal{L}$ is defined as $\mathcal{L}(\vec{\mathbf{f}}) := \mathbf{A}\vec{\mathbf{f}}$, and the binding property follows from the hardness of $\mathsf{MSIS}^{\infty,q}_{\kappa,m,2B}$.

Importantly, 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 random combination of $4k$ separate sum-checks for polynomials $[g_{1,i}, g_{2,i}]_{i=1}^{2k}$, respectively. Here the sum-check for $g_{1,i}$ (defined in Eqn. 15) is used to verify that the evaluation statement $\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_i) = \mathbf{v}_i$ holds true. The sum-check for $g_{2,i}$ (defined in Eqn. 16) is equivalent to check that the polynomial

$$\prod_{j=-(b-1)}^{b-1} \left(\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{x}}) - j\right)$$

evaluates to zero on every point $\vec{\mathbf{x}}$ in the Boolean hypercube, which is the same as the Hadamard product check (or equivalently, the norm bound check) $\hat{\mathbf{f}}_i \circ \left[\bigcirc_{j=1}^{b-1}(\hat{\mathbf{f}}_i - \hat{j}) \circ (\hat{\mathbf{f}}_i + \hat{j})\right] = \hat{0}$ defined in relation $\mathcal{R}^b_{\mathsf{cm}}$ (Eqn. 7).

Parameters: $c \in \mathbb{N}$, $\mathcal{C} := \mathbb{Z}_q \subseteq \mathcal{R}_q$, and a strong sampling set $\mathcal{C}_{\mathsf{small}} \subseteq \mathcal{R}_q$ with expansion factor $\|\mathcal{C}_{\mathsf{small}}\|_{\mathsf{op}} \leq c$ (Defn. 5).
Input: $[\mathbb{x}_i := (\vec{\mathbf{r}}_i, \mathbf{v}_i, y_i)]_{i=1}^{2k}$ and $[\mathbb{w}_i := \vec{\mathbf{f}}_i]_{i=1}^{2k}$.
Output: $\mathbb{x}_o := (\vec{\mathbf{r}}_o, \mathbf{v}_o, y_o)$, $\mathbb{w}_o := \vec{\mathbf{f}}_o$.
The protocol $\langle \mathsf{P}(\mathsf{pp}, \mathbb{x}; \mathbb{w}), \mathsf{V}(\mathsf{pp}, \mathbb{x})\rangle$**:**

1. $\mathsf{V} \to \mathsf{P}$: V sends P challenges $[\alpha_i]_{i=1}^{2k} \xleftarrow{\$} \mathcal{C}^{2k}$, $[\mu_i]_{i=1}^{2k-1} \xleftarrow{\$} \mathcal{C}^{2k-1}$ 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

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

Set $\mu_{2k} := 1$, here the polynomial $g(\vec{\mathbf{x}}) \in \mathcal{R}_q^{\leq 2b}[X_1, \ldots, X_{\log m}]$ is defined as

$$g(\vec{\mathbf{x}}) := \sum_{i=1}^{2k} \left[\alpha_i g_{1,i}(\vec{\mathbf{x}}) + \mu_i g_{2,i}(\vec{\mathbf{x}})\right] \,, \tag{14}$$

$$\forall i \in [2k] : \; g_{1,i}(\vec{\mathbf{x}}) := eq(\vec{\mathbf{r}}_i, \vec{\mathbf{x}}) \cdot \mathsf{mle}\left[\hat{\vec{\mathbf{f}}}_i\right](\vec{\mathbf{x}}) \,, \tag{15}$$

$$\forall i \in [2k] : \; g_{2,i}(\vec{\mathbf{x}}) := eq(\vec{\beta}, \vec{\mathbf{x}}) \cdot \prod_{j=-(b-1)}^{b-1} \left(\mathsf{mle}\left[\hat{\vec{\mathbf{f}}}_i\right](\vec{\mathbf{x}}) - j\right) \,. \tag{16}$$

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 vector sampled by V.
3. $\mathsf{P} \to \mathsf{V}$: P sends V values $\left[\theta_i := \mathsf{mle}\left[\hat{\vec{\mathbf{f}}}_i\right](\vec{\mathbf{r}}_o)\right]_{i=1}^{2k}$.
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 $[\rho_i]_{i=2}^{2k} \xleftarrow{\$} \mathcal{C}_{\mathsf{small}}^{2k-1}$. Set $\rho_1 := 1$.
6. V outputs $\mathbb{x}_o := (\vec{\mathbf{r}}_o, \mathbf{v}_o, y_o)$ where

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

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

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

We describe the protocol Π_{fold} in Fig. 3. The following lemma shows that the protocol Π_{fold} in Fig. 3 is a reduction of knowledge from $(\mathcal{R}^b_{\mathsf{bind}})^{2k}$ to $\mathcal{R}^B_{\mathsf{bind}}$.

Lemma 3.7. *Given a ring $\mathcal{R}_q \cong \mathbb{Z}_q^d$ and 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 binding on the domain $\{\vec{\mathbf{f}} : \|\vec{\mathbf{f}}\|_\infty < B\}$. Let $c \in \mathbb{N}$ and let $\mathcal{C}$, $\mathcal{C}_{\mathsf{small}}$ be strong sampling sets (over $\mathcal{R}_q$) where $1/|\mathcal{C}|, 1/|\mathcal{C}_{\mathsf{small}}| = \mathsf{negl}(\lambda)$ and the expansion factor $\|\mathcal{C}_{\mathsf{small}}\|_{\mathsf{op}} \le c$ (Defn. 5). For any b, k such that $2kc(b-1) < B$, the protocol Π_{fold} is a reduction of knowledge from $(\mathcal{R}^b_{\mathsf{bind}})^{2k}$ to $\mathcal{R}^B_{\mathsf{bind}}$.*

The proof follows from Lemma 3.8 and Lemma 3.9 below.

Lemma 3.8. *Π_{fold} satisfies public reducibility and completeness.*

Proof. *Public reducibility:* Given input instances $[\vec{\mathbf{r}}_i, \mathbf{v}_i, y_i]_{i=1}^{2k}$ and the transcript that includes challenges $\vec{\mathbf{r}}_o$, evaluations $[\theta_i]_{i=1}^{2k}$ and folding challenges $[\rho_i]_{i=2}^{2k}$. Set $\rho_1 := 1$. One can output $\mathbb{x}_o := (\vec{\mathbf{r}}_o, \mathbf{v}_o, y_o := \sum_{i=1}^{2k} \rho_i \cdot y_i)$ where $\mathsf{NTT}(\mathbf{v}_o) = \sum_{i=1}^{2k} \mathsf{RotSum}(\rho_i, \mathsf{NTT}(\theta_i))$ if the verification passes and $\perp$ otherwise.

Completeness: Let $(\mathbb{x}, \mathbb{w}) := [\mathbb{x}_i = (\vec{\mathbf{r}}_i, \mathbf{v}_i, y_i), \mathbb{w}_i = \vec{\mathbf{f}}_i]_{i=1}^{2k} \leftarrow \mathcal{A}(\mathsf{pp})$ denote adversary $\mathcal{A}$'s chosen input for $\mathcal{R}_1 := (\mathcal{R}^b_{\mathsf{bind}})^{2k}$, where $\mathsf{pp} := \mathcal{L} \leftarrow \mathsf{Setup}(1^\lambda)$ is the public parameter. WLOG we assume that $(\mathsf{pp}, \mathbb{x}_i, \mathbb{w}_i) \in \mathcal{R}^b_{\mathsf{bind}}$ 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 $[\rho_i]_{i=2}^{2k}$ be verifier's last folding challenges and set $\rho_1 := 1$. P outputs $\mathbb{w}_o := \vec{\mathbf{f}}_o := \sum_{i=1}^{2k} \rho_i \cdot \vec{\mathbf{f}}_i$ and V outputs $\mathbb{x}_o := (\vec{\mathbf{r}}_o, \mathbf{v}_o, y_o)$ where $\vec{\mathbf{r}}_o$ is V's sum-check challenges and $(\mathbf{v}_o, y_o)$ are defined such that
$$\mathsf{NTT}(\mathbf{v}_o) = \sum_{i=1}^{2k} \mathsf{RotSum}(\rho_i, \mathsf{NTT}(\theta_i)), \qquad y_o := \sum_{i=1}^{2k} \rho_i \cdot y_i .$$

We first show that V accepts, i.e., the check at Step 4 passes. This follows by definition of polynomial g (Eqn. 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 $(\mathsf{pp}, \mathbb{x}_o, \mathbb{w}_o) \in \mathcal{R}_2 := \mathcal{R}^B_{\mathsf{bind}}$ (Eqn. 11). First, because $(\mathsf{pp}, \mathbb{x}_i, \mathbb{w}_i) \in \mathcal{R}^b_{\mathsf{bind}}$ for all $i \in [2k]$, by Lemma 3.3, it holds that $\mathcal{L}(\vec{\mathbf{f}}_o) = y_o$ and $\mathsf{mle}\left[\hat{\mathbf{f}}_o\right](\vec{\mathbf{r}}_o) = \mathbf{v}_o$. Moreover,
$$\|\vec{\mathbf{f}}_o\|_\infty = \|\sum_{i=1}^{2k} \rho_i \cdot \vec{\mathbf{f}}_i\|_\infty \le \sum_{i=1}^{2k} \|\rho_i \cdot \vec{\mathbf{f}}_i\|_\infty \le \sum_{i=1}^{2k} c \cdot \|\vec{\mathbf{f}}_i\|_\infty \le \sum_{i=1}^{2k} c \cdot (b-1) < B .$$

The first equality is by definition of $\vec{\mathbf{f}}_o$. The next inequality holds because $\|a+b\|_\infty \leq \|a\|_\infty + \|b\|_\infty$ for any $a, b \in \mathcal{R}_q^m$ where $\|a\|_\infty + \|b\|_\infty < B < q/2$.[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 $(\mathsf{pp}, \mathbb{x}_i; \vec{\mathbf{f}}_i) \in \mathcal{R}^b_{\mathsf{bind}}$; the last inequality holds as $2kc(b-1) < B < q/2$ by the premise of the lemma. Thus $(\mathsf{pp}, \mathbb{x}_o, \mathbb{w}_o) \in \mathcal{R}^B_{\mathsf{bind}}$ (Eqn. 11) as desired. □

Lemma 3.9. Π_{fold} *satisfies knowledge soundness.*

Proof. Set $k^* := 2k$. For brevity, we assume that $k = 1$ and $k^* = 2$. The proof can extend to any constant $k > 1$. Let $\mathsf{pp} := \mathcal{L} \leftarrow \mathsf{Setup}(1^\lambda)$. Given adversary $\mathcal{A}$ that provides input $[\mathbb{x}_i := (\vec{\mathbf{r}}_i, \mathbf{v}_i, y_i)]_{i=1}^{k^*}$ and malicious prover P^* that outputs $(\mathbb{x}_o := (\vec{\mathbf{r}}_o, \mathbf{v}_o, y_o), \mathbb{w}_o := \vec{\mathbf{f}}_o)$ such that $(\mathsf{pp}, \mathbb{x}_o; \mathbb{w}_o)$ is in $\mathcal{R}^B_{\mathsf{bind}}$ (defined in Sect. 3.2.3) with non-negligible probability ϵ, we describe an expected poly-time extractor Ext that outputs $[\mathbb{w}_i := \vec{\mathbf{f}}_i]_{i=1}^{k^*}$ such that $(\mathsf{pp}, [\mathbb{x}_i; \mathbb{w}_i]_{i=1}^{k^*}) \in (\mathcal{R}^b_{\mathsf{bind}})^{k^*}$ with probability $\epsilon - \mathsf{negl}(\lambda)$.

Without loss of generality, we assume that P^* is deterministic. We start by describing an extractor Ext and analyze its runtime. Next, we argue that Ext produces $k^* = 2$ different accepting transcripts with probability $\epsilon - \mathsf{negl}(\lambda)$. Finally, we show that with probability $\epsilon - \mathsf{negl}(\lambda)$, Ext outputs valid witnesses for the k^* input instances.

Extractor. The extractor algorithm $\mathsf{Ext}(\mathsf{pp}, r)$ (with randomness r):

1. Run adversary $\mathcal{A}$ to obtain input instances $[[\mathbb{x}_i]_{i=1}^2, \mathsf{state}] \leftarrow \mathcal{A}(\mathsf{pp}, r)$.

2. Simulate the execution $(\mathbb{x}_o^{(1)}, \mathbb{w}_o^{(1)}) \leftarrow \langle \mathsf{P}^*, \mathsf{V} \rangle \left[\mathsf{pp}, [\mathbb{x}_i]_{i=1}^2; \mathsf{st}\right]$ once with final verifier challenge $\rho^{(1)} \xleftarrow{\$} \mathcal{C}_{\mathsf{small}}$. Abort if $(\mathsf{pp}, \mathbb{x}_o^{(1)}; \mathbb{w}_o^{(1)}) \notin \mathcal{R}^B_{\mathsf{bind}}$.

3. Rewind the execution $\langle \mathsf{P}^*, \mathsf{V} \rangle$ to the stage where the last verifier challenge is not sent yet. Sample a different last verifier challenge $\rho^{(2)} \xleftarrow{\$} \mathcal{C}_{\mathsf{small}} \setminus \{\rho^{(1)}\}$ repetitively (while preserving the prior randomness) until P^*'s output $(\mathbb{x}_o^{(2)}, \mathbb{w}_o^{(2)})$ satisfies $(\mathsf{pp}, \mathbb{x}_o^{(2)}, \mathbb{w}_o^{(2)}) \in \mathcal{R}^B_{\mathsf{bind}}$ or if the number of calls to $\langle \mathsf{P}^*, \mathsf{V} \rangle$ exceeds $|\mathcal{C}_{\mathsf{small}}|$.

4. Abort if the number of calls to $\langle \mathsf{P}^*, \mathsf{V} \rangle$ exceeds $|\mathcal{C}_{\mathsf{small}}|$. Otherwise parse $\mathbb{w}_o^{(1)} = \vec{\mathbf{f}}_o^{(1)}$, $\mathbb{w}_o^{(2)} = \vec{\mathbf{f}}_o^{(2)}$. Solve the system of linear equations to get $\vec{\mathbf{f}}_1$, $\vec{\mathbf{f}}_2$ such that

$$\vec{\mathbf{f}}_1 + \rho^{(1)} \cdot \vec{\mathbf{f}}_2 = \vec{\mathbf{f}}_o^{(1)}\,, \qquad \vec{\mathbf{f}}_1 + \rho^{(2)} \cdot \vec{\mathbf{f}}_2 = \vec{\mathbf{f}}_o^{(2)}\,. \tag{17}$$

   Note that we can always do the interpolation because $\rho^{(1)} - \rho^{(2)}$ is invertible given that $\mathcal{C}_{\mathsf{small}}$ is a strong sampling set (Defn. 2.3).

5. Output $[\mathbb{w}_i := \vec{\mathbf{f}}_i]_{i=1}^2$.

[1]The norm $\|a\|_\infty$ for an element a in $\mathcal{R}_q$ is defined in Remark 2.1.

Running time. We argue that the extractor Ext runs in expected polynomial time. For each choice of verifier randomness rand that *excludes the last verifier challenge* ρ, we use n_{rand} to denote the number of choices of ρ such that P^* can succeed. After 1 call to $\langle \mathsf{P}^*, \mathsf{V}\rangle$ at step 2, the extractor can proceed to step 3 with probability $n_{\mathsf{rand}}/|\mathcal{C}_{\mathsf{small}}|$ and the expected number of calls to $\langle \mathsf{P}^*, \mathsf{V}\rangle$ at step 3 is $\min(\frac{|\mathcal{C}_{\mathsf{small}}|}{n_{\mathsf{rand}}-1}, |\mathcal{C}_{\mathsf{small}}|)$. Therefore, the expected number of calls to $\langle \mathsf{P}^*, \mathsf{V}\rangle$ is

$$
\begin{aligned}
& 1 + \sum_{\mathsf{rand}^*} \Pr[\mathsf{rand} = \mathsf{rand}^*] \cdot \frac{n_{\mathsf{rand}^*}}{|\mathcal{C}_{\mathsf{small}}|} \cdot \min\left(\frac{|\mathcal{C}_{\mathsf{small}}|}{n_{\mathsf{rand}^*}-1}, |\mathcal{C}_{\mathsf{small}}|\right) \\
\leq\, & 1 + \sum_{\mathsf{rand}^*} \Pr[\mathsf{rand} = \mathsf{rand}^*] \cdot \min\left(\frac{n_{\mathsf{rand}^*}}{n_{\mathsf{rand}^*}-1}, n_{\mathsf{rand}^*}\right) \\
\leq\, & 1 + \sum_{\mathsf{rand}^*} \Pr[\mathsf{rand} = \mathsf{rand}^*] \cdot 2 = 3\,.
\end{aligned}
$$

The last inequality holds because $\min\left(\frac{n_{\mathsf{rand}^*}}{n_{\mathsf{rand}^*}-1}, n_{\mathsf{rand}^*}\right) \leq 2$.

The aborting probability. Next, we argue that with probability $\epsilon - \mathsf{negl}(\lambda)$, the extractor Ext won't abort. Note that Ext aborts only if one of the following events happens:

- The output of the first call to $\langle \mathsf{P}^*, \mathsf{V}\rangle$ (in Step 2) is not in the relation $\mathcal{R}^{B}_{\mathsf{bind}}$. This happens with probability at most $1-\epsilon$ by the assumption that P^* succeeds with probability ϵ.
- The number of calls to $\langle \mathsf{P}^*, \mathsf{V}\rangle$ in Step 3 exceeds $|\mathcal{C}_{\mathsf{small}}|$. As shown previously, the expected number of calls to $\langle \mathsf{P}^*, \mathsf{V}\rangle$ is at most 3, by Markov inequality, the probability that Ext calls $\langle \mathsf{P}^*, \mathsf{V}\rangle$ for more than $|\mathcal{C}_{\mathsf{small}}|$ times is at most $3/|\mathcal{C}_{\mathsf{small}}| = \mathsf{negl}(\lambda)$.

By union bound, the probability that Ext aborts is at most $1-\epsilon+\mathsf{negl}(\lambda)$, thus Ext produces an output with probability at least $\epsilon - \mathsf{negl}(\lambda)$.

The probability of successful extraction. Next, if Ext does not abort, it must output some $[\vec{\mathbf{f}}_i]_{i=1}^2$ such that the system of equations in Eqn. 17 holds w.r.t. $[\vec{\mathbf{f}}_o^{(i)}]_{i=1}^2$. Moreover, for every $i \in [2]$, define $\mathbf{v}_o^{(i)}$ and $y_o^{(i)}$ such that

$$
\mathsf{NTT}(\mathbf{v}_o^{(i)}) = \mathsf{NTT}(\theta_1) + \mathsf{RotSum}(\rho^{(i)}, \mathsf{NTT}(\theta_2))\,, \qquad y_o^{(i)} := y_1 + \rho^{(i)} \cdot y_2 \tag{18}
$$

(where θ_1,θ_2 are sent by P^* at Step 3 in Fig. 3), we have $(\mathsf{pp}, (\vec{\mathbf{r}}_o, \mathbf{v}_o^{(i)}, y_o^{(i)}), \vec{\mathbf{f}}_o^{(i)}) \in \mathcal{R}^{B}_{\mathsf{bind}}$ by the terminating condition at Step 3 in the extractor.

Next, we show that Ext outputs correct witness conditioned on it does not abort. We start with a lemma showing that the extracted witness $[\vec{\mathbf{f}}_i]_{i=1}^2$ are pre-images of the binding commitments $[y_i]_{i=1}^2$ and $\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_o) = \theta_i$ for all $i \in [2]$ where $[\theta_i]_{i=1}^2$ are the evaluations sent by P^*. Looking ahead, this implies that the sum-check claim holds w.r.t. the polynomial g (Eqn 14).

Claim 1. *If* Ext *does not abort (and thus* $(\mathsf{pp}, (\vec{\mathbf{r}}_o, \mathbf{v}_o^{(i)}, y_o^{(i)}), \vec{\mathbf{f}}_o^{(i)}) \in \mathcal{R}_{\mathsf{bind}}^B$ *for all* $i \in [2]$*), then the extracted witness* $\vec{\mathbf{f}}_1, \vec{\mathbf{f}}_2$ *satisfies that for every* $i \in [2]$*,* $\mathcal{L}(\vec{\mathbf{f}}_i) = y_i$ *and* $\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_o) = \theta_i$*.*

Proof. Recall that $\rho^{(1)} - \rho^{(2)}$ is invertible by definition of strong sampling sets. Therefore, the equations on the right hand side of Eqn. 18 have unique solutions $[y_i]_{i=1}^2$. On the other hand, $(\mathsf{pp}, (\vec{\mathbf{r}}_o, \mathbf{v}_o^{(i)}, y_o^{(i)}), \vec{\mathbf{f}}_o^{(i)}) \in \mathcal{R}_{\mathsf{bind}}^B$ implies that $\mathcal{L}(\vec{\mathbf{f}}_o^{(i)}) = y_o^{(i)}$. Moreover, $\mathcal{L}$ is an $\mathcal{R}_q$-module homomorphism, by definition of $\vec{\mathbf{f}}_1$, $\vec{\mathbf{f}}_2$, we have that

$$y_o^{(i)} = \mathcal{L}(\vec{\mathbf{f}}_o^{(i)}) = \mathcal{L}(\vec{\mathbf{f}}_1 + \rho^{(i)} \cdot \vec{\mathbf{f}}_2) = \mathcal{L}(\vec{\mathbf{f}}_1) + \rho^{(i)}\mathcal{L}(\vec{\mathbf{f}}_2)$$

for all $i \in [2]$. Thus $[\mathcal{L}(\vec{\mathbf{f}}_i)]_{i=1}^2$ is also a solution to the equations in Eqn. 18 and therefore $\mathcal{L}(\vec{\mathbf{f}}_i) = y_i$ for all $i \in [2]$.

Next we prove that $\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_o) = \theta_i$ for all $i \in [2]$. For ease of exposition, we define $[\bar{\theta}_i, \bar{\mathbf{v}}_o^{(i)}]_{i=1}^2$ as the $\mathcal{R}_q$-elements such that $\mathsf{NTT}(\theta_i) = \mathsf{vec}(\bar{\theta}_i)$ and $\mathsf{NTT}(\mathbf{v}_o^{(i)}) = \mathsf{vec}(\bar{\mathbf{v}}_o^{(i)})$ for all $i \in [2]$. Thus the left hand side of Eqn. 18 can be rewritten as that for all $i \in [2]$:

$$\mathsf{vec}(\bar{\mathbf{v}}_o^{(i)}) = \mathsf{vec}(\bar{\theta}_1) + \mathsf{RotSum}(\rho^{(i)}, \mathsf{vec}(\bar{\theta}_2)) = \mathsf{vec}(\bar{\theta}_1) + \mathsf{vec}(\rho^{(i)} \cdot \bar{\theta}_2) = \mathsf{vec}(\bar{\theta}_1 + \rho^{(i)} \cdot \bar{\theta}_2) \quad (19)$$

where the 2nd equality is by the 2nd claim in Lemma 2.1 and the 3rd equality is by additivity of coefficient embedding.

Recall that $(\mathsf{pp}, (\vec{\mathbf{r}}_o, \mathbf{v}_o^{(i)}, y_o^{(i)}), \vec{\mathbf{f}}_o^{(i)}) \in \mathcal{R}_{\mathsf{bind}}^B$ for all $i \in [2]$ by the premise of the lemma, hence $\mathsf{mle}\left[\hat{\mathbf{f}}_o^{(i)}\right](\vec{\mathbf{r}}_o) = \mathbf{v}_o^{(i)}$ for all $i \in [2]$. Since $\vec{\mathbf{f}}_1 + \rho^{(i)} \cdot \vec{\mathbf{f}}_2 = \vec{\mathbf{f}}_o^{(i)}$ for all $i \in [2]$ (Eqn. 17), by Lemma 3.1, it holds that $[\mathsf{mle}\left[\mathsf{vec}(\vec{\mathbf{f}}_i)\right](\vec{\mathbf{r}}_o) \in \mathbb{Z}_q^d]_{i=1}^2$ is a solution for $[\bar{\theta}_i]_{i=1}^2$ in Eqn. 19. Applying Lemma 3.1 again, we have that $\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_o) = \theta_i$ for all $i \in [2]$. □

Next, we show that the extracted witness $[\vec{\mathbf{f}}_i]_{i=1}^2$ are correct with high probability.

Claim 2. *Conditioned on* Ext *does not abort, with overwhelming probability (over* Ext*'s randomness), the extracted witness* $\vec{\mathbf{f}}_1, \vec{\mathbf{f}}_2$ *satisfies that* $\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_i) = \mathbf{v}_i$ *and* $\|\hat{\mathbf{f}}_i\|_\infty < b$ *for every* $i \in [2]$ *(where* $\hat{\mathbf{f}}_i$ *is defined in Eqn. 9 with* $\mathcal{R}_q \cong \mathbb{Z}_q^d$*).*

Proof. By Claim 1, we have that $\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_o) = \theta_i$ for all $i \in [2]$. Since the verifier check at Step 4 of Fig. 3 passes, we have that the sum-check random evaluation claim holds, that is, $g(\vec{\mathbf{r}}_o) = s$ where g is defined according to Eqn 14 in which $\mathsf{mle}\left[\hat{\mathbf{f}}_1\right]$ and $\mathsf{mle}\left[\hat{\mathbf{f}}_2\right]$ are obtained from Ext's output $\vec{\mathbf{f}}_1$, $\vec{\mathbf{f}}_2$.

Next, we argue that the sum-check polynomial g defined above is independent of the verifier challenges $\vec{\mathbf{r}}_o$ with overwhelming probability. (This is essential for us to use sum-check soundness later.) Recall that $(\mathsf{pp}, (\vec{\mathbf{r}}_o, \mathbf{v}_o^{(i)}, y_o^{(i)}), \vec{\mathbf{f}}_o^{(i)}) \in \mathcal{R}^B_{\mathsf{bind}}$ for all $i \in [2]$ if Ext does not abort. By the binding property of $\mathcal{L}$, we know that with overwhelming probability, $[\vec{\mathbf{f}}_o^{(i)}]_{i=1}^2$ are bound with $[y_o^{(i)} = y_1 + \rho^{(i)} \cdot y_2]_{i=1}^2$ respectively, which implies that the extracted witness $\vec{\mathbf{f}}_1$, $\vec{\mathbf{f}}_2$ are also bound with y_1, y_2 respectively (as the interpolation is unique). Since y_1,y_2 are fixed before sampling $\vec{\mathbf{r}}_o$, the vectors $\vec{\mathbf{f}}_1$, $\vec{\mathbf{f}}_2$ as well as the sumcheck polynomial g defined above are independent of the verifier challenges $\vec{\mathbf{r}}_o$ with overwhelming probability.

Since $g(\vec{\mathbf{r}}_o) = s$ and g is independent of $\vec{\mathbf{r}}_o$ with overwhelming probability, by sum-check soundness, with overwhelming probability (over the random choice of sum-check challenges), the sumcheck claim holds, that is

$$\sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log m}} g(\vec{\mathbf{b}}) = \sum_{i=1}^{2} \alpha_i \mathbf{v}_i \,. \tag{20}$$

On the other hand, define $p_i(\vec{\mathbf{x}}) := \prod_{j=1-b}^{b-1} \left(\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{x}}) - j\right)$. By the uniqueness of MLE (Corollary 2.1), we can rewrite the evaluation $p_i(\vec{\beta})$ as an interpolation of p_i's evaluations on the Boolean hypercube, that is,

$$p_i(\vec{\beta}) = \sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log m}} eq(\vec{\beta}, \vec{\mathbf{b}}) \cdot p_i(\vec{\mathbf{b}}) = \sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log m}} g_{2,i}(\vec{\mathbf{b}}) \,, \tag{21}$$

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

$$\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_i) = \sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log m}} eq(\vec{\mathbf{r}}_i, \vec{\mathbf{b}}) \cdot \mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{b}}) = \sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log m}} g_{1,i}(\vec{\mathbf{b}}) \,, \tag{22}$$

where $g_{1,i}$ is defined in Eqn. 15. Recall that g in Eqn. 14 is defined as

$$g(\vec{\mathbf{b}}) := \sum_{i=1}^{2} \left[\alpha_i g_{1,i}(\vec{\mathbf{b}}) + \mu_i g_{2,i}(\vec{\mathbf{b}})\right] .$$

By plugging-in the LHS of Eqn. 21 and Eqn. 22, we can rewrite Eqn. 20 as

$$\sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log m}} g(\vec{\mathbf{b}}) = \sum_{i=1}^{2} \alpha_i \cdot \mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_i) + \sum_{i=1}^{2} \mu_i \cdot p_i(\vec{\beta}) = \sum_{i=1}^{2} \alpha_i \mathbf{v}_i \tag{23}$$

where μ_1 is the challenge sent by V at Step 1 of Fig. 3 and $\mu_2 := 1$. Re-arranging the terms, we have that

$$h(\alpha_1, \alpha_2, \mu_1) := \sum_{i=1}^{2} \left(\mathbf{v}_i - \mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_i)\right) \cdot \alpha_i + p_1(\vec{\beta}) \cdot \mu_1 + p_2(\vec{\beta}) = 0\,.$$

Since $[\alpha_i]_{i=1}^2$ and μ_1 are uniformly chosen from the sampling set $\mathcal{C}$, by the Generalized Schwartz Lemma (Lemma 2.4), with overwhelming probability (over $[\alpha_i]_{i=1}^2$, μ_1), we have that for every $i \in [2]$, (i) $\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_i) = \mathbf{v}_i$, and (ii) the polynomial $p_i(\vec{\mathbf{x}})$ is identically zero, that is,

$$p_i(\vec{\mathbf{x}}) := \prod_{j=1-b}^{b-1} \left(\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{x}}) - j\right) = 0\,.$$

Recall from Sect. 3.1 that (ii) implies $\|\vec{\mathbf{f}}_i\|_\infty < b$, which finishes the proof. □

By Claim 1 and Claim 2, we obtain that Ext outputs valid witnesses with overwhelming probability conditioned on it does not abort. Since Ext doesn't abort with probability $\epsilon - \mathsf{negl}(\lambda)$, it holds that Ext outputs valid witnesses with probability at least $\epsilon - \mathsf{negl}(\lambda)$. This completes the proof of Lemma 3.9. □

3.2.4 Putting it all together

Finally, by composing the protocols Π_{cm}, Π_{dec} and Π_{fold}, we obtain a reduction of knowledge from $\mathcal{R}^B_{\mathsf{eval}} \times \mathcal{R}^B_{\mathsf{cm}}$ to $\mathcal{R}^B_{\mathsf{eval}}$ as desired.

Theorem 3.1. *Given any ring $\mathcal{R}_q \cong \mathbb{Z}_q^d$, let $\mathsf{pp} := (\kappa, m, \mathbf{A}, B < q/2)$ be the public parameters such that $\mathsf{MSIS}^{\infty,q}_{\kappa,m,2B}$ is hard. Let $c \in \mathbb{N}$ and let $\mathcal{C}$, $\mathcal{C}_{\mathsf{small}}$ be strong sampling sets (over $\mathcal{R}_q$) where $1/|\mathcal{C}|, 1/|\mathcal{C}_{\mathsf{small}}| = \mathsf{negl}(\lambda)$ and the expansion factor $\|\mathcal{C}_{\mathsf{small}}\|_{\mathsf{op}} \le c$ (Defn. 5). Set b, k such that $2kc(b-1) < B$ and $b^k = B$. The composed protocol $\Pi_{\mathsf{mfold}} := \Pi_{\mathsf{fold}} \circ \Pi_{\mathsf{dec}} \circ \Pi_{\mathsf{cm}}$ is a public-coin reduction of knolwedge from relation $\mathcal{R}^B_{\mathsf{eval}} \times \mathcal{R}^B_{\mathsf{cm}}$ to relation $\mathcal{R}^B_{\mathsf{eval}}$.*

Proof. The protocol is public-coin as Π_{cm} and Π_{dec} are non-interactive and Π_{fold} is public-coin. The Theorem follows from Lemma 3.2, Lemma 3.4, Lemma 3.7 and the knowledge composition theorems (Theorem 2.2 and Theorem 2.3). □

3.3 Supporting Small Prime Modulus

In the protocol Π_{fold} (Fig. 3), 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 coordinate of their NTT representation. Hence the 1st coordinate 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.

Thus to achieve 128-bit security, we need to use an 128-bit prime modulus in Π_{fold}. In practice, however, it would be significantly more efficient to use a smaller modulus that facilitates fast computations on CPU/hardwares. E.g., a 32-bit prime is a perfect fit for GPUs that operate on 32-bit data types and for CPUs that use 32/64-bit integer types. We describe an optimization that extends Π_{fold} to support 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 still cannot be too small in order to 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 Sect. 2 that we have $\mathcal{R}_q \cong \mathbb{F}_{q^\tau}^t$ via the NTT transform. Thus we can rewrite the commitment opening relation $\mathcal{R}_{\mathsf{cm}}^{B}$ (Eqn. 7) 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, $\mathsf{NTT}(\hat{\mathbf{f}}) = \mathsf{vec}(\vec{\mathbf{f}})$. Given $\mathcal{R}_{\mathsf{cm}}^{\tau,B}$, we can similarly generalize the expanded commitment opening relation $\mathcal{R}_{\mathsf{eval}}^{B}$ (Eqn. 8) to $\mathcal{R}_{\mathsf{eval}}^{\tau,B}$ defined as

$$\mathcal{R}_{\mathsf{eval}}^{\tau,B} := \left\{ (\mathsf{pp}; (\vec{\mathbf{r}}, [\mathbf{v}_j]_{j=1}^{\tau}, \mathsf{cm}) \in \mathcal{R}_q^{\log m}\times\mathcal{R}_q^{\tau}\times\mathcal{R}_q^{\kappa}; \vec{\mathbf{f}}\in\mathcal{R}_q^m) \;:\; \begin{array}{c} (\mathsf{pp},\mathsf{cm};\vec{\mathbf{f}})\in\mathcal{R}_{\mathsf{cm}}^{B}\wedge \\ \left(\forall j\in[\tau]: \mathsf{mle}\left[\hat{\mathbf{f}}_j\right](\vec{\mathbf{r}}) = \mathbf{v}_j\right) \end{array} \right\}, \tag{24}$$

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} (Fig. 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 $I_t := 1^t$ (where i is enumerated over $\mathbb{F}_{q^\tau}$), that is,
$$\mathcal{C} := \{\mathbf{a}_i\in\mathcal{R}_q : \mathsf{NTT}(\mathbf{a}_i) = i\cdot I_t\}_{i\in\mathbb{F}_{q^\tau}}.$$
This ensures that $\mathcal{C}$ is a strong sampling set with size $q^\tau\approx 2^{128}$ (as the difference of any two distinct elements in $\mathcal{C}$ is isomorphic to $a\cdot I_t$ for some a in $\mathbb{F}_{q^\tau}^\times$, 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).

- Let $[\rho_i]_{i=1}^{2k}$ be the last folding challenges (in Fig. 3). For every $i \in [2k]$, we use $V_i \in \mathcal{R}_q^\tau$ to denote $V_i := [\theta_{i,j}]_{j=1}^{\tau}$ (where $\theta_{i,j} \in \mathcal{R}_q$ are the MLE evaluations to be folded) and denote $\mathsf{NTT}(V_i) := (\mathsf{NTT}(\theta_{i,1}), \ldots, \mathsf{NTT}(\theta_{i,\tau})) \in \mathbb{F}_{q^\tau}^d$. The folding verifier computes $V_o := [\mathbf{v}_{o,j}]_{j=1}^{\tau} \in \mathcal{R}_q^\tau$ such that $\mathsf{NTT}(V_o) \in \mathbb{F}_{q^\tau}^d$ satisfies that

$$\mathsf{NTT}(V_o) = \sum_{i=1}^{2k} \mathsf{RotSum}(\rho_i, \mathsf{NTT}(V_i)),$$

where RotSum is defined in Lemma 2.1. By the 3rd claim in Lemma 2.1, we can extend Eqn. 19 to the more general setting where $\mathcal{R}_q \cong \mathbb{F}_{q^\tau}^{d/\tau}$,[2] and all proofs in Sect. 3.2 will still go through.

4 A Lattice-based Folding Scheme for CCS

In this section, we construct a folding scheme for the customizable constraint systems (CCS) [STW23a], which is a generalization of the Rank-1 Constraint Systems (R1CS) for supporting high-degree custom gates. As mentioned in the head of Sect. 3, this enables us to build IVC/PCD from Ajtai commitments. Our construction is highly modular and generic. We first adapt 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[3] and let $\bar{\mathcal{R}}$ be an arbitrary ring. Let $\mathbb{i}$ be an index that consists of 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 $c_1, \ldots, c_{n_s} \in \bar{\mathcal{R}}$.*

Denote $\mathsf{pp}_{\mathsf{ccs}} := (\mathsf{pp}, \mathbb{i})$. A tuple $(\mathsf{pp}_{\mathsf{ccs}}, \mathbb{x} \in \bar{\mathcal{R}}^{\ell_{\mathsf{in}}}; \mathbb{w} \in \bar{\mathcal{R}}^{n_c - \ell_{\mathsf{in}} - 1})$ is in the relation $\mathcal{R}_{\mathsf{ccs}}$ (over $\bar{\mathcal{R}}$) if and only if

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

where $\vec{\mathbf{z}} := (\mathbb{x}, 1, \mathbb{w}) \in \bar{\mathcal{R}}^{n_c}$ and $\bigcirc$ 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). *Suppose the ring $\bar{\mathcal{R}} \cong \mathbb{F}^k$ for a field $\mathbb{F}$. We can pack k tuples in the CCS relation over $\mathbb{F}$ into a single tuple in the CCS relation over $\bar{\mathcal{R}}$. More precisely, given k tuples $((\mathsf{pp}, \mathbb{i}_i), \mathbb{x}_i, \mathbb{w}_i)_{i=1}^{k}$, the k tuples 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 so that $\mathsf{NTT}(\mathbf{e}) = (e_1, \ldots, e_k)$ where $e_i \in \mathbb{F}$ is the corresponding entry in $(\mathbb{i}_i, \mathbb{x}_i, \mathbb{w}_i)$ $(1 \leq i \leq k)$.*

[2]The single linear equation (over $\mathcal{R}_q$) in Eqn. 19 will be extended to τ linear equations.

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

4.1 The Relation for 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 Eqn. 7 to the CCS setting. As mentioned in the header of Section 3, a folding scheme for $\mathcal{R}^B_{\mathsf{cmccs}}$ would allow us to build an IVC/PCD scheme from Ajtai commitments.

Definition 4.2 (Lattice-based committed CCS relation)**.** *Let $\mathcal{R}_q := \mathbb{Z}_q[X]/(X^d + 1)$. 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 sampled parameter for $\mathcal{R}^B_{\mathsf{cm}}$ (Eqn. 7) and $\mathsf{pp}_{\mathsf{ccs}} = (\mathsf{pp}, \mathbb{i})$ for $\mathcal{R}_{\mathsf{ccs}}$ (over $\mathcal{R}_q$) is defined in Defn. 4.1. Set $\ell := m/n_c \in \mathbb{N}$. It further satisfies 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*

$$\mathcal{R}^B_{\mathsf{cmccs}} := \left\{ \begin{array}{c} \left(\mathsf{pp}, \mathbb{x} := (\mathsf{cm} \in \mathcal{R}_q^\kappa, \mathbb{x}_{\mathsf{ccs}} \in \mathcal{R}_q^{\ell_{\mathsf{in}}}); \mathbb{w} := (\vec{\mathbf{f}} \in \mathcal{R}_q^m, \mathbb{w}_{\mathsf{ccs}} \in \mathcal{R}_q^{n-\ell_{\mathsf{in}}-1})\right) \ s.t. \\ (\mathsf{pp}_{\mathsf{cm}}, \mathsf{cm}; \vec{\mathbf{f}}) \in \mathcal{R}^B_{\mathsf{cm}} \wedge (\mathsf{pp}_{\mathsf{ccs}}, \mathbb{x}_{\mathsf{ccs}}; \mathbb{w}_{\mathsf{ccs}}) \in \mathcal{R}_{\mathsf{ccs}} \wedge (\mathbf{z}_{\mathsf{ccs}} = \mathbf{G}\vec{\mathbf{f}}) \end{array} \right\}, \tag{25}$$

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 $\mathbf{z}_{\mathsf{ccs}} = \mathbf{G}\vec{\mathbf{f}}$ is used to capture the relation that $\vec{\mathbf{f}}$ is the "base-B" representation of the original witness $\mathbf{z}_{\mathsf{ccs}}$ in CCS. Crucially, if $\vec{\mathbf{f}}$ is a witness in $\mathcal{R}^B_{\mathsf{cm}}$, then $(\mathbb{x}_{\mathsf{ccs}}, 1, \mathbb{w}_{\mathsf{ccs}}) = \mathbf{G}\vec{\mathbf{f}}$ implies that $\mathbb{w}_{\mathsf{ccs}}$ is binding with cm, because $\vec{\mathbf{f}}$ is a valid opening for the Ajtai commitment cm.*

Remark 4.3. *We set $(\mathbb{x}_{\mathsf{ccs}}, 1, \mathbb{w}_{\mathsf{ccs}}) = \mathbf{G}\vec{\mathbf{f}}$ only for ease of exposition. In practice, however, it is sufficient decompose only $\mathbb{w}_{\mathsf{ccs}}$ to $\vec{\mathbf{f}}$ and constrain $\mathbb{w}_{\mathsf{ccs}} = \mathbf{G}\vec{\mathbf{f}}$, because the integrity of $\mathbb{x}_{\mathsf{ccs}}$ is already guaranteed by the verifier checks.*

The expanded relation. Similar to the treatment in Sect. 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 and replace the high-degree custom gate relation in $\mathcal{R}^B_{\mathsf{cmccs}}$ with a linearized relation $\mathcal{R}_{\mathsf{lccs}}$. Looking ahead, this is because in our folding scheme, the verifier will run sum-checks to reduce the norm constraints and the high-degree custom gate relation in $\mathcal{R}^B_{\mathsf{cmccs}}$ into some linearized relations. Hence, we need to adjust the accumulated relation accordingly. We note that $\mathcal{R}^B_{\mathsf{evalccs}}$ is an extension to $\mathcal{R}^B_{\mathsf{eval}}$ in Eqn. 8.

Definition 4.3 (Lattice-based linearized CCS relation)**.** *Let $\mathsf{pp} := (\mathsf{pp}_{\mathsf{cm}}, \mathsf{pp}_{\mathsf{ccs}})$ be the public parameters in Defn. 4.2 where $\ell := m/n_c \in \mathbb{N}$ and $B^\ell \geq q/2$. Without loss of generality, we assume that the number of rows n_r in CCS matrices equals to the committed* *witness length m.*[4] $\mathcal{R}^B_{\mathsf{evalccs}}$ *is defined as*

$$\mathcal{R}^B_{\mathsf{evalccs}} := \left\{ \begin{array}{c} \Big(\mathsf{pp}, \mathbb{x} := (\vec{\mathbf{r}} \in \mathcal{R}_q^{\log m}, \mathsf{cm}, \mathbf{v}, [\mathbf{u}_i]_{i=1}^t, \mathbb{x}_{\mathsf{ccs}}, \mathbf{h}); \\ \mathbb{w} := (\vec{\mathbf{f}} \in \mathcal{R}_q^m, \mathbb{w}_{\mathsf{ccs}} \in \mathcal{R}_q^{n_c - \ell_{\mathsf{in}} - 1})\Big) \;\; s.t. \\ (\mathbf{z}_{\mathsf{ccs}} = \mathbf{G}\vec{\mathbf{f}}) \wedge (\mathsf{pp}_{\mathsf{cm}}, (\mathsf{cm}, \vec{\mathbf{r}}, \mathbf{v}); \vec{\mathbf{f}}) \in \mathcal{R}^B_{\mathsf{eval}} \\ \wedge (\mathsf{pp}_{\mathsf{ccs}}, (\vec{\mathbf{r}}_{\mathsf{ccs}}, [\mathbf{u}_i]_{i=1}^t, \mathbb{x}_{\mathsf{ccs}}, \mathbf{h}); \mathbb{w}_{\mathsf{ccs}}) \in \mathcal{R}_{\mathsf{lccs}} \end{array} \right\}, \tag{26}$$

where $\mathbf{z}_{\mathsf{ccs}} := (\mathbb{x}_{\mathsf{ccs}}, \mathbf{h}, \mathbb{w}_{\mathsf{ccs}}) \in \mathcal{R}_q^{n_c}$*;* $\mathbf{G} := \mathbf{I}_{n_c} \otimes [1, B, \ldots, B^{\ell-1}] \in \mathbb{Z}_q^{n_c \times m}$ *is the* ***gadget matrix****;* $\mathcal{R}^B_{\mathsf{eval}}$ *is defined in Eqn. 8; and* $\mathcal{R}_{\mathsf{lccs}}$ *consists of the tuples* $(\mathsf{pp}_{\mathsf{ccs}}, (\vec{\mathbf{r}}, [\mathbf{u}_i]_{i=1}^t, \mathbb{x}_{\mathsf{ccs}}, \mathbf{h}); \mathbb{w}_{\mathsf{ccs}})$ *where for all* $i \in [t]$*, it holds that*

$$\mathbf{u}_i = \sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log n_c}} \mathsf{mle}\,[M_i]\,(\vec{\mathbf{r}}, \vec{\mathbf{b}}) \cdot \mathsf{mle}\,[\mathbf{z}_{\mathsf{ccs}}]\,(\vec{\mathbf{b}})\,. \tag{27}$$

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

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, it is a reduction of knowledge (Defn. 2.5) from $\mathcal{R}^B_{\mathsf{evalccs}} \times \mathcal{R}^B_{\mathsf{cmccs}}$ to $\mathcal{R}^B_{\mathsf{evalccs}}$. Similar to the strategy in Sect. 3.2, the construction consists of three steps.

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} (Fig. 4) that reduces $\mathcal{R}^B_{\mathsf{cmccs}}$ to $\mathcal{R}^B_{\mathsf{evalccs}}$. Intuitively, it runs a sum-check protocol to reduce the high-degree custom gates check to a degree-1 check (e.g., a multilinear evaluation check).

Step 2: Decomposition. Next, using a protocol Π_{ccsdec} (Fig. 5), 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} (Fig. 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} (Fig. 6). Note that Π_{ccsfold} is an adaptation to Π_{fold} (Fig. 3) with similar analysis.

By the composition theorems for reductions of knowledge (Theorem 2.2, Theorem 2.3), 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.

[4]We can always pad dummy constraints (or dummy committed witness) so that the number of constraints n_r is equal to m.

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

By Theorem 2.3, 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 $\mathcal{R}^B_{\mathsf{cmccs}}$ (Eqn. 25) to $\mathcal{R}^B_{\mathsf{evalccs}}$ (Eqn. 26). We describe the protocol Π_{ccs} in Fig. 4. Intuitively, it runs a sum-check to reduce the high-degree CCS relation to a multilinear evaluation relation that has degree 1.

Parameters: A strong sampling set $\mathcal{C} := \mathbb{Z}_q \subseteq \mathcal{R}_q$ (Defn. 2.3).
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: $\mathbb{x}_o := (\vec{\mathbf{r}}_o \in \mathcal{R}_q^{\log m}, \mathbf{v} \in \mathcal{R}_q, \mathsf{cm}, [\mathbf{u}_i \in \mathcal{R}_q]_{i=1}^t, \mathbb{x}_{\mathsf{ccs}}, 1)$ 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. $\mathsf{P} \leftrightarrow \mathsf{V}$: 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 $\mathbf{z}_{\mathsf{ccs}} := (\mathbb{x}_{\mathsf{ccs}}, 1, \mathbb{w}_{\mathsf{ccs}})$. $g \in \mathcal{R}_q^{\leq \mathsf{deg}+1}[X_1, \ldots, X_{\log m}]$ is defined as[a]
$$g(\vec{\mathbf{x}}) := eq(\vec{\beta}, \vec{\mathbf{x}}) \cdot \left(\sum_{i=1}^{n_s} c_i \cdot \left[\prod_{j \in S_i} \left(\sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log n_c}} \mathsf{mle}\left[M_j\right](\vec{\mathbf{x}}, \vec{\mathbf{b}}) \cdot \mathsf{mle}\left[\mathbf{z}_{\mathsf{ccs}}\right](\vec{\mathbf{b}}) \right) \right] \right) .$$
The protocol reduces to a random evaluation check $g(\vec{\mathbf{r}}_o) \overset{?}{=} s$ for some $s \in \mathcal{R}_q$, and $\vec{\mathbf{r}}_o \xleftarrow{\$} \mathcal{C}^{\log m}$ is the sum-check challenge vector sampled by V.
3. $\mathsf{P} \to \mathsf{V}$: P sends V the values $(\mathbf{v}, [\mathbf{u}_i]_{i=1}^t)$ where $\mathbf{v} := \mathsf{mle}\left[\hat{\mathbf{f}}\right](\vec{\mathbf{r}}_o)$ and for every $i \in [t]$, $\mathbf{u}_i$ is computed as
$$\mathbf{u}_i := \sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log n_c}} \mathsf{mle}\left[M_i\right](\vec{\mathbf{r}}_o, \vec{\mathbf{b}}) \cdot \mathsf{mle}\left[\mathbf{z}_{\mathsf{ccs}}\right](\vec{\mathbf{b}}) .$$
4. V computes $\mathbf{e} := eq(\vec{\beta}, \vec{\mathbf{r}}_o)$ and checks that
$$\mathbf{e} \cdot \left(\sum_{i=1}^{n_s} c_i \cdot \prod_{j \in S_i} \mathbf{u}_j \right) \overset{?}{=} s .$$
5. V outputs $\mathbb{x}_o := (\vec{\mathbf{r}}_o, \mathbf{v}, \mathsf{cm}, [\mathbf{u}_i]_{i=1}^t, \mathbb{x}_{\mathsf{ccs}}, 1)$. P outputs $\mathbb{w}_o := (\vec{\mathbf{f}}, \mathbb{w}_{\mathsf{ccs}})$.

[a] deg is the CCS gate degree.

Figure 4: The protocol Π_{ccs} that reduces $\mathcal{R}^B_{\mathsf{cmccs}}$ to $\mathcal{R}^B_{\mathsf{evalccs}}$.

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. Public reducibility: Given input instance $\mathbb{x} = (\mathsf{cm}, \mathbb{x}_{\mathsf{ccs}})$ and transcript that includes $(\vec{\mathbf{r}}_o, \mathbf{v}, [\mathbf{u}_i]_{i=1}^t)$, one can output $\mathbb{x}_o = (\vec{\mathbf{r}}_o, \mathbf{v}, \mathsf{cm}, [\mathbf{u}_i]_{i=1}^t, \mathbb{x}_{\mathsf{ccs}}, 1)$ if the verifier check passes; and $\perp$ otherwise.

Completeness: Let $(\mathbb{x}; \mathbb{w}) := \Big((\mathsf{cm}, \mathbb{x}_{\mathsf{ccs}}); (\vec{\mathbf{f}}, \mathbb{w}_{\mathsf{ccs}})\Big) \leftarrow \mathcal{A}(\mathsf{pp})$ 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 $(\mathsf{pp}, \mathbb{x}; \mathbb{w}) \in \mathcal{R}^B_{\mathsf{cmccs}}$. 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 values $\mathbf{v} := \mathsf{mle}\left[\hat{\mathbf{f}}\right](\vec{\mathbf{r}}_o)$ and $\mathbf{u}_i := \sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log n_c}} \mathsf{mle}[M_i](\vec{\mathbf{r}}_o, \vec{\mathbf{b}}) \cdot \mathsf{mle}[\mathbf{z}_{\mathsf{ccs}}](\vec{\mathbf{b}})$ for every $i \in [t]$ (where $\mathbf{z}_{\mathsf{ccs}} := (\mathbb{x}_{\mathsf{ccs}}, 1, \mathbb{w}_{\mathsf{ccs}})$).

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 $\mathbb{x}_o := (\vec{\mathbf{r}}_o, \mathbf{v}, \mathsf{cm}, [\mathbf{u}]_{i=1}^t, \mathbb{x}_{\mathsf{ccs}}, 1)$.

We first show that V passes the check at Step 4 and accepts, this follows by definitions of $[\mathbf{u}_i]_{i=1}^t$.

Next we show that the protocol output $(\mathbb{x}_o; \mathbb{w}_o)$ satisfies that $(\mathsf{pp}, \mathbb{x}_o; \mathbb{w}_o)$ is in $\mathcal{R}_2 := \mathcal{R}^B_{\mathsf{evalccs}}$ (Eqn. 26): First, $(\mathsf{pp}, (\vec{\mathbf{r}}_o, \mathbf{v}, \mathsf{cm}); \vec{\mathbf{f}})$ is in $\mathcal{R}^B_{\mathsf{eval}}$ because $\mathbf{v} = \mathsf{mle}\left[\hat{\mathbf{f}}\right](\vec{\mathbf{r}}_o)$ and $(\mathsf{pp}, \mathsf{cm}, \vec{\mathbf{f}}) \in \mathcal{R}^B_{\mathsf{cm}}$ (as $(\mathsf{pp}, \mathbb{x}; \mathbb{w}) \in \mathcal{R}_1 := \mathcal{R}^B_{\mathsf{cmccs}}$). Second, $\mathbf{z}_{\mathsf{ccs}} = \mathbf{G}\vec{\mathbf{f}}$ because $(\mathsf{pp}, \mathbb{x}; \mathbb{w}) \in \mathcal{R}^B_{\mathsf{cmccs}}$. Finally, $(\mathsf{pp}_{\mathsf{ccs}}, (\vec{\mathbf{r}}_o, [\mathbf{u}_i]_{i=1}^t, \mathbb{x}_{\mathsf{ccs}}, \mathbf{h}); \mathbb{w}_{\mathsf{ccs}})$ is in $\mathcal{R}_{\mathsf{lccs}}$ by definitions of $[\mathbf{u}_i]_{i=1}^t$. Thus, $(\mathsf{pp}, \mathbb{x}_o, \mathbb{w}_o)$ is in $\mathcal{R}^B_{\mathsf{evalccs}}$ and completeness holds.

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. Output $\perp$ if V rejects. Otherwise let $(\mathbb{x}_o, \mathbb{w}_o)$ be the protocol output where $\mathbb{x}_o := (\vec{\mathbf{r}}_o, \mathbf{v}, \mathsf{cm}, [\mathbf{u}_i]_{i=1}^t, \mathbb{x}_{\mathsf{ccs}}, 1)$ and $\mathbb{w}_o := (\vec{\mathbf{f}}, \mathbb{w}_{\mathsf{ccs}})$.

3. If $(\mathsf{pp}, \mathbb{x}_o, \mathbb{w}_o)$ is in $\mathcal{R}^B_{\mathsf{evalccs}}$ (Eqn. 26), the extractor outputs witness $\mathbb{w} := (\vec{\mathbf{f}}, \mathbb{w}_{\mathsf{ccs}})$. Otherwise outputs $\perp$.

To prove knowledge soundness, it suffices to show that if V accepts and the protocol's output $(\mathbb{x}_o, \mathbb{w}_o)$ satisfies that $(\mathsf{pp}, \mathbb{x}_o, \mathbb{w}_o) \in \mathcal{R}_2 := \mathcal{R}^B_{\mathsf{evalccs}}$, then $(\mathsf{pp}, \mathbb{x}; \mathbb{w})$ is also in $\mathcal{R}_1 := \mathcal{R}^B_{\mathsf{cmccs}}$ (Eqn. 25) with overwhelming probability (over the random choices of sum-check challenges and the sampled pp).

Note that because $(\mathsf{pp}, \mathbb{x}_o, \mathbb{w}_o)$ is in $\mathcal{R}_2 := \mathcal{R}^B_{\mathsf{evalccs}}$ (Eqn. 26) by assumption, we have that (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 $\mathcal{R}_{\mathsf{lccs}}$ (Eqn. 27). By definition of $\mathcal{R}^B_{\mathsf{cmccs}}$ (Eqn. 25), it remains to argue that $(\mathsf{pp}_{\mathsf{ccs}}, \mathbb{x}_{\mathsf{ccs}}; \mathbb{w}_{\mathsf{ccs}})$ is in $\mathcal{R}_{\mathsf{ccs}}$ (Defn. 4.1) (with overwhelming probability).

First, by (i), (ii) above, the sumcheck polynomial g is fixed before sampling $\vec{\mathbf{r}}_o$, as the witness $\mathbf{z}_{\mathsf{ccs}}$ is binding with cm. Second, by (iii), and because the verifier check at Step 4 passes, the sum-check random evaluation check $g(\vec{\mathbf{r}}_o) \overset{?}{=} s$ passes. Therefore, by sumcheck soundness, with overwhelming probability (over the sum-check challenges), the sumcheck claim $\sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log m}} g(\vec{\mathbf{b}}) = 0$ holds. By the uniqueness of MLE (Corollary 2.1), we have that $p(\vec{\beta}) = 0$ for the polynomial

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

Since $\vec{\beta} \xleftarrow{\$} \mathcal{C}^{\log m}$ is uniformly chosen from the sampling set $\mathcal{C}$ (Defn 2.3), by the Generalized Schwartz-Zippel Lemma (Lemma 2.4), the polynomial $p(\vec{\mathbf{x}})$ is identically zero. Thus $p(\vec{\mathbf{x}}) = 0$ for all $\vec{\mathbf{x}} \in \{0,1\}^{\log m}$, which implies that $(\mathsf{pp}_{\mathsf{ccs}}, \mathbb{x}_{\mathsf{ccs}}; \mathbb{w}_{\mathsf{ccs}}) \in \mathcal{R}_{\mathsf{ccs}}$ (Defn. 4.1).

In sum, the extractor's output satisfies that $(\mathsf{pp}, \mathbb{x}; \mathbb{w})$ is in $\mathcal{R}_1 := \mathcal{R}^B_{\mathsf{cmccs}}$ with overwhelming probability conditioned on that the simulated execution output of $\langle \mathsf{P}^*, \mathsf{V} \rangle$ is in relation $\mathcal{R}_2 := \mathcal{R}^B_{\mathsf{evalccs}}$. Thus Π_{ccs} is knowledge sound. □

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 2.3, 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 essentially $\Pi_{\mathsf{ccsdec}} := \Pi^*_{\mathsf{ccsdec}} \times \Pi^*_{\mathsf{ccsdec}}$ 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 to both $\mathcal{R}^B_{\mathsf{evalccs}}$ (Defn. 26) and $\mathcal{R}^B_{\mathsf{hom}}$(Eqn. 11), where we generalize Ajtai commitments and gadget matrix multiplications to arbitrary $\mathcal{R}_q$-module homomorphisms:

$$\mathcal{R}^B_{\mathsf{ccshom}} := \left\{ \begin{array}{c} \mathsf{pp} := (\mathcal{L}, \mathcal{L}_w, M), \\ \mathbb{x} := (\vec{\mathbf{r}} \in \mathcal{R}_q^{\log m}, \mathbf{v} \in \mathcal{R}_q, \mathbf{u} \in \mathcal{U}, y \in \mathcal{Y}, \mathbb{x}_w \in \mathcal{R}_q^{n_{\mathsf{in}}}); \\ \mathbb{w} := (\vec{\mathbf{f}} \in \mathcal{R}_q^m, \vec{\mathbf{w}} \in \mathcal{R}_q^n) \text{ s.t.} \\ (\mathcal{L}, (\vec{\mathbf{r}}, \mathbf{v}, y); \vec{\mathbf{f}}) \in \mathcal{R}^B_{\mathsf{hom}} \wedge \\ (\mathbf{z} = \mathcal{L}_w(\vec{\mathbf{f}})) \wedge (\mathbf{u} = \langle M \cdot \mathsf{tensor}(\vec{\mathbf{r}}), \mathbf{z} \rangle) \end{array} \right\}, \tag{28}$$

here $\mathcal{U} := \mathcal{R}_q^t$ for some $t \in \mathbb{N}$, $\mathbf{z} := (\mathbb{x}_w || \vec{\mathbf{w}}) \in \mathcal{R}_q^{n_{\mathsf{in}}+n}$;

$$\mathsf{tensor}(\vec{\mathbf{r}}) := \bigotimes_{i=1}^{\log m} (\vec{\mathbf{r}}_i, 1 - \vec{\mathbf{r}}_i) \in \mathcal{R}_q^m \tag{29}$$

is the tensor product of $\{(\vec{\mathbf{r}}_i, 1 - \vec{\mathbf{r}}_i)\}_{i=1}^{\log m}$; $M \in \mathcal{U}^{(n+n_{\mathsf{in}}) \times m}$ is a matrix over module $\mathcal{U}$. Lastly, $\mathcal{L} : \mathcal{R}_q^m \to \mathcal{Y}$ and $\mathcal{L}_w : \mathcal{R}_q^m \to \mathcal{R}_q^{n+n_{\mathsf{in}}}$ are any $\mathcal{R}_q$-module homomorphisms.

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 $M := (M_1, \ldots, M_t)$, i.e., each entry $(\mathbf{e}_1, \ldots, \mathbf{e}_t) \in \mathcal{U}$ of 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 Fig. 5. The differences from Π^*_{dec} (Fig. 2) are highlighted in red, which are for computing the $\mathbf{u}$-values and the CCS instances.

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

1. P computes $(\vec{\mathbf{f}}_0, \ldots, \vec{\mathbf{f}}_{k-1}) \leftarrow \mathsf{split}_{b,k}(\vec{\mathbf{f}})$ (Eqn. 12).
2. $\mathsf{P} \to \mathsf{V}$: P sends V the values $[y_i, \mathbf{v}_i, \mathbf{u}_i, \mathbb{x}_{w,i}]_{i=0}^{k-1}$:

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

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

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

Figure 5: The protocol Π^*_{ccsdec} that reduces $\mathcal{R}^B_{\mathsf{ccshom}}$ to $(\mathcal{R}^b_{\mathsf{ccshom}})^k$.

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 3.4. We defer the proof to Appendix A.1. □

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 Sect. 3.2.3, we denote that a tuple is in $\mathcal{R}^B_{\mathsf{ccsbind}}$ if and only if the tuple is in $\mathcal{R}^B_{\mathsf{ccshom}}$ (Eqn. 28) and the sampled homomorphism $\mathcal{L}$ in the public parameter is **binding** on the domain $\{\vec{\mathbf{f}} : \|\vec{\mathbf{f}}\|_\infty < B\}$. Note that $\mathcal{R}^B_{\mathsf{evalccs}}$ is a special case of $\mathcal{R}^B_{\mathsf{ccsbind}}$, because $\mathcal{R}^B_{\mathsf{evalccs}}$ is a special case of $\mathcal{R}^B_{\mathsf{ccshom}}$ by Remark 4.4 and the sampled homomorphism $\mathcal{L}(\vec{\mathbf{f}}) := \mathbf{A}\vec{\mathbf{f}}$ is binding given the hardness of $\mathsf{MSIS}^{\infty,q}_{\kappa,m,2B}$.

We describe the protocol Π_{ccsfold} in Fig. 6 that reduces from $(\mathcal{R}^b_{\mathsf{ccsbind}})^{2k}$ to $\mathcal{R}^B_{\mathsf{ccsbind}}$. The idea 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 $\mathcal{U} := \mathcal{R}_q$ and thus $t = 1$. The protocol naturally extends to the case when $\mathcal{U} := \mathcal{R}_q^t$ for some $t > 1$.

Lemma 4.3. *Let $\mathcal{R}_q \cong \mathbb{Z}_q^d$ and public parameter $\mathsf{pp} := (m, n, n_{\mathsf{in}}, B, \mathcal{L}, \mathcal{L}_w)$ (where the sampled $\mathcal{L}$ is binding on the domain $\{\vec{\mathbf{f}} : \|\vec{\mathbf{f}}\|_\infty < B\}$), 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 (Defn. 5). For any b, k such that $2kc(b-1) < B$, Π_{ccsfold} is a reduction of knowledge from $(\mathcal{R}^b_{\mathsf{ccsbind}})^{2k}$ to $\mathcal{R}^B_{\mathsf{ccsbind}}$.*

Proof. The proof is similar to that for Lemma 3.7. We defer the proof to Appendix A.2. □

Putting it all together. Finally, by composing the protocols Π_{ccs}, Π_{ccsdec} and Π_{ccsfold}, we obtain a reduction of knowledge from $\mathcal{R}^B_{\mathsf{evalccs}} \times \mathcal{R}^B_{\mathsf{cmccs}}$ to $\mathcal{R}^B_{\mathsf{evalccs}}$ as desired.

Theorem 4.1. *Let $\mathcal{R}_q \cong \mathbb{Z}_q^d$. Let $\mathsf{pp}_{\mathsf{cm}} := (\kappa, m, \mathbf{A}, B < q/2)$ and $\mathsf{pp}_{\mathsf{ccs}} := (n_r := m, n_c, t, n_s, \mathsf{deg}, \ell_{\mathsf{in}}, [M_i]_{i=1}^t, [S_i, c_i]_{i=1}^{n_s})$ be the public parameters such that $B^{m/n_c} \geq q/2$ and $\mathsf{MSIS}^{\infty,q}_{\kappa,m,2B}$ is hard. Let $c \in \mathbb{N}$ and let $\mathcal{C}$, $\mathcal{C}_{\mathsf{small}}$ be strong sampling sets where $1/|\mathcal{C}|, 1/|\mathcal{C}_{\mathsf{small}}| = \mathsf{negl}(\lambda)$ and the expansion factor $\|\mathcal{C}_{\mathsf{small}}\|_{\mathsf{op}} \leq c$ (Defn. 5). Set b, k such that $2kc(b-1) < B$ and $b^k = B$. The composed protocol $\Pi_{\mathsf{mccsfold}} := \Pi_{\mathsf{ccsfold}} \circ \Pi_{\mathsf{ccsdec}} \circ \Pi_{\mathsf{ccs}}$ is a public-coin reduction of knolwedge 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. The Theorem follows from Lemma 4.1, Lemma 4.2, Lemma 4.3 and the knowledge composition theorems (Theorem 2.2 and Theorem 2.3). □

Remark 4.5 (Supporting small prime modulus)**.** *The same optimization in Sect. 3.3 can be used to extend Theorem 4.1 to support small modulus q. Namely, Theorem 4.1 still holds when $\mathcal{R}_q \cong \mathbb{F}_{q^\tau}^{d/\tau}$ for any $\tau \in \mathbb{N}$ that divides d.*

5 Performance Estimates

The complexity of the folding schemes is described in Table. 1. In the folding scheme for the (committed) CCS relation (Eqn. 25), the instance consists of $\tau + \kappa + t + \ell_{\mathsf{in}} + 1$ elements in $\mathcal{R}_q$ for storing the values $[\mathbf{v}_i]_{i=1}^{\tau}$, $[\mathbf{u}_i]_{i=1}^{t}$, the commitment $\mathsf{cm} \in \mathcal{R}_q^\kappa$ and the public input $\mathbb{x}_w$; additionally it takes $\log m$ field elements to store the challenge vector $\vec{\mathbf{r}}$ in the strong sampling set $\mathcal{C} \cong \mathbb{F}_{q^\tau}$. The prover takes approximately $O(mk(\kappa + t))$ $\mathcal{R}_q$ multiplications to compute the commitments and $\mathbf{u}$-values for the decomposed witness (Fig. 5); and it takes approximately $O(mD \log^2 D)$ $\mathcal{R}_q$ multiplications to run the sum-checks (Fig. 4, Fig. 6), where $D := \max(2b, \mathsf{deg})$.

The verifier takes $4k \cdot (\kappa + \tau + t + \ell_{\mathsf{in}} + 1)$ $\mathcal{R}_q$ multiplications to (i) check the correctness of decomposition in Fig. 5 and (ii) fold the decomposed instances in Fig. 6. It takes

Parameters: $c \in \mathbb{N}, \mathcal{C} := \mathbb{Z}_q \subseteq \mathcal{R}_q$ and a strong sampling set $\mathcal{C}_{\mathsf{small}}$ with expansion factor $\|\mathcal{C}_{\mathsf{small}}\|_{\mathsf{op}} \leq c$.
Input: $\mathbb{x} := [\mathbb{x}_i := (\vec{\mathbf{r}}_i, \mathbf{v}_i, y_i, \mathbf{u}_i, \mathbb{x}_{w,i})]_{i=1}^{2k}$ and $\mathbb{w} := [\mathbb{w}_i := (\vec{\mathbf{f}}_i, \vec{\mathbf{w}}_i)]_{i=1}^{2k}$.
Output: $\mathbb{x}_o := (\vec{\mathbf{r}}_o, \mathbf{v}_o, y_o, \mathbf{u}_o, \mathbb{x}_{w,o})$, $\mathbb{w}_o := (\vec{\mathbf{f}}_o, \vec{\mathbf{w}}_o)$.
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 $[\alpha_i, \zeta_i]_{i=1}^{2k} \xleftarrow{\$} (\mathcal{C} \times \mathcal{C})^{2k}$, $[\mu_i]_{i=1}^{2k-1} \xleftarrow{\$} \mathcal{C}^{2k-1}$, 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

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

Set $\mu_{2k} := 1$, here the polynomial $g(\vec{\mathbf{x}}) \in \mathcal{R}_q^{\leq 2b}[X_1, \ldots, X_{\log m}]$ is defined as

$$g(\vec{\mathbf{x}}) := \sum_{i=1}^{2k} \left[\alpha_i g_{1,i}(\vec{\mathbf{x}}) + \mu_i g_{2,i}(\vec{\mathbf{x}}) + \zeta_i g_{3,i}(\vec{\mathbf{x}})\right] \,, \tag{30}$$

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

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

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

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 $g(\vec{\mathbf{r}}_o) \overset{?}{=} s$ 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 $i \in [2k]$,

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

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

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

5. $\mathsf{V} \to \mathsf{P}$: V sends P random challenges $[\rho_i]_{i=2}^{2k} \xleftarrow{\$} \mathcal{C}_{\mathsf{small}}^{2k-1}$. Set $\rho_1 := 1$.
6. V output $\mathbb{x}_o := (\vec{\mathbf{r}}_o, \mathbf{v}_o, y_o, \mathbf{u}_o, \mathbb{x}_{w,o})$ where $\mathsf{NTT}(\mathbf{v}_o) = \sum_{i=1}^{2k} \mathsf{RotSum}(\rho_i, \mathsf{NTT}(\theta_i))$ 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 6: The protocol Π_{ccsfold} that reduces $(\mathcal{R}^b_{\mathsf{ccsbind}})^{2k}$ to $\mathcal{R}^B_{\mathsf{ccsbind}}$. We set the $\mathcal{R}_q$-module $\mathcal{U}$ to be $\mathcal{U} := \mathcal{R}_q$. The protocol naturally extends to any $\mathcal{U} := \mathcal{R}_q^t$ where $t > 1$.

$n_s \deg + 2k(\tau + t + 2b\tau)$ $\mathcal{R}_q$ multiplications to check the high-degree random evaluation claims and takes $2(2b + \deg)\log m$ $\mathcal{R}_q$ multiplications[5] and $(\deg + 2b)\log m$ hashes to run the sumcheck verifiers in Fig. 4 and Fig. 6. Additionally, it takes $2(2k+2)\log m$ field multiplications to compute the $2k+2$ of eq values. We note that our 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 $\mathbb{F}_{q^\tau}$ and thus does not need to do any NTT inversions.

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

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

Table 1: The complexity of the folding schemes. Let $\mathcal{R}_q \cong \mathbb{F}_{q^\tau}^{d/\tau}$ for some $\tau, d \in \mathbb{N}$ where $\tau \mid d$. The witness in both folding schemes (for $\mathcal{R}_{\mathsf{cm}}$ and $\mathcal{R}_{\mathsf{cmccs}}$) are m elements in $\mathcal{R}_q$ (i.e., the length of the committed witness). (Note that the CCS witness $\mathsf{w}_{\mathsf{ccs}} \in \mathcal{R}_q^{n_c - \ell_{\mathsf{in}} - 1}$ in $\mathcal{R}^B_{\mathsf{cmccs}}$ can be deterministically derived from the Ajtai committed witness.) κ is the number of $\mathcal{R}_q$-elements in the Ajtai commitment; $(t, n_s, \ell_{\mathsf{in}}, \deg)$ are the CCS parameters defined in Defn. 4.1. Let B be the norm bound of the committed witness. b, k are the parameters such that $b^k = B$ and $D := \max(2b, \deg)$ where deg is the CCS gate degree. $\mathcal{R}_q$-mul denotes a multiplication over $\mathcal{R}_q \cong \mathbb{F}_{q^\tau}^{d/\tau}$; $\mathbb{F}$-mul denotes a multiplication over the field $\mathbb{F}_{q^\tau}$; H denotes a hash from $\mathcal{R}_q^2$ to $\mathcal{R}_q$.

Instantiation. Set $d := 64$, $\kappa := 9$ and use a 64-bit prime q such that $\mathcal{R}_q \cong \mathbb{F}_{q^4}^{d/4}$. By the MSIS hardness bound in Sect. 2.1, we can achieve 128-bit security so long as the infinite norm bound B satisfies that $\log B \leq 27 - 0.5 \cdot \log m$.

For simplicity, assume that the CCS parameters $(t, n_s, \ell_{\mathsf{in}})$ satisfies that $\ell_{\mathsf{in}} = 0$, $t = n_s = 1$. Assume that the number of CCS constraints (over $\mathcal{R}_q$) in the IVC/PCD recursive circuit is $m \leq 2^{22}$ and the CCS witness size is $n_c \leq 2^{20}$. Also note that we require that $m \geq \frac{\log(q/2)}{\log B} n_c$. Then we can always set $B := 2^{16}$ so that $\log B \leq 27 - 0.5 \cdot \log m$ (and

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

$\frac{\log(q/2)}{\log B} \leq 4$). 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 $m \leq 2^{22}$ leads to an upper bound 2^{26} on the number of constraints over $\mathbb{F}_{q^4}$, which is usually more than enough.

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. And by Lemma 2.2, $\mathcal{C}_{\mathsf{small}}$ has expansion factor at most $2d = 128$.

Efficiency comparison with Hypernova and Protostar. Let's see the concrete prover cost and verifier circuit size. E.g., we can 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 4.1. The prover time is approximately $180m$ $\mathcal{R}_q$ operations and the number of CCS constraints for simulating the folding verifier is

$$|\mathsf{V}| \approx (1632 + (12 + 2\deg) \cdot \log m) \cdot |\mathcal{R}_q| + (\deg + 4) \log m \cdot |\mathsf{H}|$$

where $|\mathcal{R}_q|$ 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 $|\mathsf{H}| \leq 100$. Let $m := 2^{16}$, this would amount to approximately $8320 + 1632 \cdot \deg$ constraints.

In comparison, Hypernova [KS23b] requires $\approx 1|\mathbb{G}| + \deg \cdot \log m |\mathsf{H}| + (2\deg \cdot \log m + O(1))|\mathbb{F}|$ constraints and Protostar [BC23] requires $\approx 3|\mathbb{G}| + (\deg + O(1)) \cdot (|\mathbb{F}| + |\mathsf{H}|)$ constraints, where $|\mathbb{G}|$ denotes the circuit size for a group scalar multiplication; $|\mathbb{F}|$ denotes the number of constraints for a (non-native) field operation. The caveat of [KS23b; BC23] is that they need to use a cycle-curve pair, hence the recursive verifier circuit needs to be represented in both curves, which involves many non-native field arithmetic simulations. We set $|\mathbb{G}| \approx 1500$ given the newest data from [KS23a], and set $|\mathbb{F}| \approx 64$ according to [KPS18]. Setting $m := 2^{16}$, for Hypernova, it takes $\approx 2500 + 4672 \deg$ constraints; for Protostar, it takes $\approx 6500 + 164 \deg$ constraints. Hence LatticeFold has fewer constraints than Hypernova for all $\deg \geq 2$; and it has comparable number of constraints with Protostar when the gate degree is not large. However, even when the gate degree is large, recall that LatticeFold is post-quantum secure, whereas Protostar is not.

Remark 5.1. *After each folding step, the norm of the folded witness is always less than $2k(b-1)\|\mathcal{C}_{\mathsf{small}}\|_{\mathsf{op}} = B^* < B$ (where $\|\mathcal{C}_{\mathsf{small}}\|_{\mathsf{op}}$ is the expansion factor of $\mathcal{C}_{\mathsf{small}}$). Thus we can use a smaller $k^* < k$ when decomposing the folded witness in relation $\mathcal{R}_{\mathsf{acc}}$, which leads to a minor efficiency improvement. (Here k^* satisfies that $b^{k^*} \geq B^*$.) This also indicates that we can use different decomposition factors k^*, k for the two relations $\mathcal{R}_{\mathsf{acc}}$ and $\mathcal{R}_{\mathsf{comp}}$.*

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 Eqn. 8) 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 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. Third, 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 post-quantum security leads to better performance.

Acknowledgments. 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, Heidelberg, Aug. 2021, pp. 549–579. DOI: 10.1007/978-3-030-84245-1_19.

[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, Heidelberg, 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, Heidelberg, Aug. 2022, pp. 102–132. DOI: 10.1007/978-3-031-15979-4_4.

[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+18] 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, Heidelberg, Aug. 2018, pp. 669–699. DOI: 10.1007/978-3-319-96881-0_23.

[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, Heidelberg, Aug. 2018, pp. 757–788. DOI: 10.1007/978-3-319-96884-1_25.

[BBHR18] Eli Ben-Sasson, Iddo Bentov, Yinon Horesh, and Michael Riabzev. *Scalable, transparent, and post-quantum secure computational integrity.* Cryptology ePrint Archive, Report 2018/046. https://eprint.iacr.org/2018/046. 2018.

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

[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, Heidelberg, 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, Heidelberg, 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, 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, Heidelberg, 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, Heidelberg, 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. https://eprint.iacr.org/2019/1021. 2019.

[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, Heidelberg, 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, Heidelberg, Aug. 2019, pp. 176–202. DOI: 10.1007/978-3-030-26948-7_7.

[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, Heidelberg, Aug. 2023, pp. 507–539. DOI: 10.1007/978-3-031-38548-3_17.

[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, Heidelberg, Aug. 2023, pp. 518–548. DOI: 10.1007/978-3-031-38554-4_17.

[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, Heidelberg, 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. https://eprint.iacr.org/2019/762. 2019.

[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, Heidelberg, 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 19*. Ed. by Robert H. Deng, Valérie Gauthier-Umaña, Martín Ochoa, and Moti Yung. Vol. 11464. LNCS. Springer, Heidelberg, June 2019, pp. 67–88. DOI: 10.1007/978-3-030-21568-2_4.

[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, Heidelberg, 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, Heidelberg, 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.

[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, Heidelberg, Aug. 2023, pp. 669–701. DOI: 10.1007/978-3-031-38551-3_21.

[KPS18] Ahmed E. Kosba, Charalampos Papamanthou, and Elaine Shi. “xJsnark: A Framework for Efficient Verifiable Computation”. In: *2018 IEEE Symposium on Security and Privacy*. IEEE Computer Society Press, May 2018, pp. 944–961. DOI: 10.1109/SP.2018.00018.

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

[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, Heidelberg, 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, 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, Heidelberg, 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, Heidelberg, 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.

[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, 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. https://eprint.iacr.org/2022/1576. 2022.

[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, 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.

A Deferred Proofs

A.1 Proof of Lemma 4.2

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

Completeness: Let $(\mathbb{x}, \mathbb{w}) := \Big((\vec{\mathbf{r}}, \mathbf{v}, y, \mathbf{u}, \mathbb{x}_w), (\vec{\mathbf{f}}, \vec{\mathbf{w}})\Big) \leftarrow \mathcal{A}(\mathsf{pp})$ denote adversary $\mathcal{A}$'s chosen instance-witness pair for $\mathcal{R}_1 := \mathcal{R}^B_{\mathsf{ccshom}}$ (Eqn. 28), where $\mathsf{pp} := (\mathcal{L}, \mathcal{L}_w, M) \leftarrow \mathsf{Setup}(1^\lambda)$ 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 $\vec{\mathbf{F}} := (\vec{\mathbf{f}}_0, \ldots, \vec{\mathbf{f}}_{k-1}) := \mathsf{split}_{b,k}(\vec{\mathbf{f}})$ (Eqn. 12) and sends V values

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

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

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

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

We first show that if V accepts, then the protocol's output satisfies that

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

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

It remains to argue that V will accept. By Lemma 3.5, we have that

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

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

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

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 (Eqn. 12); the last equality holds because $(\mathsf{pp}, \mathbb{x}; \mathbb{w})$ is in $\mathcal{R}_1 := \mathcal{R}^B_{\mathsf{ccshom}}$ by assumption. Therefore, V will accept and thus completeness holds.

Knowledge soundness: Let $(\mathbb{x} := (\mathbf{v}, y, \mathbf{u}, \mathbb{x}_w), \mathsf{state}) \leftarrow \mathcal{A}(\mathsf{pp})$ denote adversary $\mathcal{A}$'s chosen input instance for $\mathcal{R}_1 := \mathcal{R}^B_{\mathsf{ccshom}}$, where $\mathsf{pp} := (\mathcal{L}, \mathcal{L}_w, M) \leftarrow \mathsf{Setup}(1^\lambda)$ 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 $(\mathbb{x}_o, \mathbb{w}_o) := [(\vec{\mathbf{r}}, \mathbf{v}_i, y_i, \mathbf{u}_i, \mathbb{x}_{w,i}); (\vec{\mathbf{f}}_i, \vec{\mathbf{w}}_i)]_{i=0}^{k-1}$ 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

$$\vec{\mathbf{f}} := \sum_{i=0}^{k-1} b^i \cdot \vec{\mathbf{f}}_i \,, \qquad \vec{\mathbf{w}} := \sum_{i=0}^{k-1} b^i \cdot \mathcal{L}_w(\vec{\mathbf{f}}_i)[n_{\mathsf{in}} + 1, n_{\mathsf{in}} + n] \,. \tag{31}$$

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

By Lemma 3.6, we have that $(\vec{\mathbf{r}}, \mathbf{v}, y; \vec{\mathbf{f}}) \in \mathcal{R}^B_{\mathsf{hom}}$ if $[(\vec{\mathbf{r}}, \mathbf{v}_i, y_i); \vec{\mathbf{f}}_i]_{i=0}^{k-1}$ is in $(\mathcal{R}^b_{\mathsf{hom}})^k$. By definition of $\mathcal{R}^B_{\mathsf{ccshom}}$ (Eqn. 28), it remains to argue that

$$\mathbf{z}_i := (\mathbb{x}_{w,i} || \vec{\mathbf{w}}_i) = \mathcal{L}_w(\vec{\mathbf{f}}_i)\, \forall i \in [0, k) \implies \mathbf{z} := (\mathbb{x}_w || \vec{\mathbf{w}}) = \mathcal{L}_w(\vec{\mathbf{f}})\,; \tag{32}$$

$$\mathbf{u}_i = \langle M \cdot \mathsf{tensor}(\vec{\mathbf{r}}), \mathbf{z}_i \rangle \; \forall i \in [0, k) \implies \mathbf{u} = \langle M \cdot \mathsf{tensor}(\vec{\mathbf{r}}), \mathbf{z} \rangle \,. \tag{33}$$

We first prove Eqn. 32. By the verifier check $\sum_{i=0}^{k-1} b^i \cdot \mathbb{x}_{w,i} = \mathbb{x}_w$ and by Eqn. 31, 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, Eqn. 32 holds by the same argument in Lemma 3.3 for proving $y = \mathcal{L}(\vec{\mathbf{f}})$ (where we replace $y, \mathcal{L}$ with $\mathbf{z}, \mathcal{L}_w$ respectively).

Similarly, Eqn. 33 holds because

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

where the 1st equality is by the verifier check; the 2nd equality follows by the premise in Eqn. 33; the 3rd equality follows by the linearly homomorphic property of inner products; the last equality holds as we've shown that $\mathbf{z} = \sum_{i=0}^{k-1} b^i \cdot \mathbf{z}_i$.

In sum, $(\vec{\mathbf{r}}, \mathbf{v}, y; \vec{\mathbf{f}})$ is in $\mathcal{R}^B_{\mathsf{hom}}$ and Eqn. 32, Eqn. 33 hold true. Therefore, conditioned on that $(\mathsf{pp}, [(\vec{\mathbf{r}}, \mathbf{v}_i, y_i, \mathbf{u}_i, \mathbb{x}_{w,i}); (\vec{\mathbf{f}}_i, \vec{\mathbf{w}}_i)]_{i=0}^{k})$ is in $\mathcal{R}_2 := (\mathcal{R}^b_{\mathsf{ccshom}})^k$, the extracted witness $(\vec{\mathbf{f}}, \vec{\mathbf{w}})$ will satisfy that $(\mathsf{pp}, (\vec{\mathbf{r}}, \mathbf{v}, y, \mathbf{u}, \mathbb{x}_w); (\vec{\mathbf{f}}, \vec{\mathbf{w}}))$ is in $\mathcal{R}_1 := \mathcal{R}^B_{\mathsf{ccshom}}$, which completes the proof. $\square$

A.2 Proof of Lemma 4.3

Proof. Public reducibility: Given input instances $[\vec{\mathbf{r}}_i, \mathbf{v}_i, y_i, \mathbf{u}_i, \mathbb{x}_{w,i}]_{i=1}^{2k}$ 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 $[\rho_i]_{i=2}^{2k}$. Set $\rho_1 := 1$. One can output $\mathbb{x}_o := (\vec{\mathbf{r}}_o, \mathbf{v}_o, y_o, \mathbf{u}_o, \mathbb{x}_{w,o})$ such that

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

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

Completeness: Let

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

denote adversary $\mathcal{A}$'s chosen instance-witness pair for $\mathcal{R}_1 := (\mathcal{R}^b_{\mathsf{bind}})^{2k}$, where $\mathsf{pp} := (\mathcal{L}, \mathcal{L}_w, M) \leftarrow \mathsf{Setup}(1^\lambda)$ is the public parameter. WLOG we assume that $(\mathsf{pp}, \mathbb{x}, \mathbb{w})$ is in $(\mathcal{R}^b_{\mathsf{bind}})^{2k}$. 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 $i \in [2k]$,

$$\eta_i := \langle M \cdot \mathsf{tensor}(\vec{\mathbf{r}}_o), (\mathbb{x}_{w,i} || \vec{\mathbf{w}}_i) \rangle , \qquad \theta_i := \mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_o) . \tag{34}$$

   Here $\mathsf{tensor}(\cdot)$ is defined in Eqn. 29 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 $[\rho_i]_{i=2}^{2k}$ be verifier's last folding challenges and set $\rho_1 := 1$. Set $(\mathbf{v}_o, y_o, \mathbf{u}_o, \mathbb{x}_{w,o}, \vec{\mathbf{f}}_o, \vec{\mathbf{w}}_o)$ such that

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

   and

$$[y_o, \mathbf{u}_o, \vec{\mathbf{f}}_o, (\mathbb{x}_{w,o} || \vec{\mathbf{w}}_o)] := \sum_{i=1}^{2k} \rho_i \cdot [y_i, \eta_i, \vec{\mathbf{f}}_i, (\mathbb{x}_{w,i} || \vec{\mathbf{w}}_i)] . \tag{35}$$

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

First, we show that V accepts, i.e., the verifier check at Step 4 will pass. This follows by by definition of polynomial g (Eqn. 30) and by 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 $(\mathsf{pp} := (\mathcal{L}, \mathcal{L}_w, M), \mathbb{x}_o; \mathbb{w}_o) \in \mathcal{R}_2 := \mathcal{R}^B_{\mathsf{ccsbind}}$ (Eqn. 28). First, by Lemma 3.8, we have that $(\mathcal{L}, (\vec{\mathbf{r}}_o, \mathbf{v}_o, y_o); \vec{\mathbf{f}}_o) \in \mathcal{R}^B_{\mathsf{bind}}$. Denote $\mathbf{z}_o := (\mathbb{x}_{w,o} || \vec{\mathbf{w}}_o)$. It remains to argue that (i) $\mathbf{z}_o = \mathcal{L}_w(\vec{\mathbf{f}}_o)$ and (ii) $\mathbf{u}_o = \langle M \cdot \mathsf{tensor}(\vec{\mathbf{r}}_o), \mathbf{z}_o \rangle$.

Note that (i) holds true because

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

where the 1st equality is by Eqn. 35; the 2nd equality follows by the premise that $\mathcal{L}_w(\vec{\mathbf{f}}_i) = (\mathbb{x}_{w,i} || \vec{\mathbf{w}}_i)$ for all $i \in [2k]$, the last equality holds by 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

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

where the 1st equality is by definition of $\mathbf{u}_o$; the 2nd equality follows by definition of η_i in Eqn. 34; the last equality follows by the assignment of $\mathbf{z}_o := (\mathbb{x}_{w,o} || \vec{\mathbf{w}}_o)$ in Eqn. 35 and by the homomorphic property of inner products.

In sum, $(\mathsf{pp}, \mathbb{x}_o, \mathbb{w}_o)$ is in $\mathcal{R}_2 := \mathcal{R}^B_{\mathsf{ccsbind}}$ (Eqn. 28) as desired and the completeness holds.

Knowledge soundness: The extractor Ext, the running time analysis, and the aborting probability are almost identical to the proof of Lemma 3.9. The only difference is that the extractor Ext, besides outputting $[\vec{\mathbf{f}}_i]_{i=1}^2$, also outputs $[\vec{\mathbf{w}}_i := \mathcal{L}_w(\vec{\mathbf{f}}_i)[n_{\mathsf{in}}+1, n_{\mathsf{in}}+n]]_{i=1}^2$. To argue the success probability of extraction and finish the proof, it suffices to prove the two claims below that extend Claim 1 and Claim 2 respectively.

Claim 3. *If* Ext *does not abort, then for every* $i \in [2]$*, it holds 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)* $\eta_i = \langle M \cdot \mathsf{tensor}(\vec{\mathbf{r}}_o), (\mathbb{x}_{w,i} || \vec{\mathbf{w}}_i) \rangle$.

Proof. The equations (i),(ii) hold by Claim 1. 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 $\mathbb{x}_{w,i} = \mathcal{L}_w(\vec{\mathbf{f}}_i)[1, n_{\mathsf{in}}]$. Eqn. (iii) then follows by replacing $\mathcal{L}$, y_i with $\mathcal{L}_w(\cdot)[1, n_{\mathsf{in}}]$, $\mathbb{x}_{w,i}$ everywhere respectively and reuse the argument for (i).

To argue (iv), we observe that the map $\phi(\cdot) := \langle M \cdot \mathsf{tensor}(\vec{\mathbf{r}}_o), (\cdot) \rangle$, 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 by (iii), Eqn. (iv) is equivalent to $\eta_i = \Phi(\vec{\mathbf{f}}_i)$, which follows by replacing $\mathcal{L}$, y_i with Φ, η_i everywhere respectively and reuse the argument for (i). $\square$

Claim 4. *Conditioned on* Ext *does not abort, with overwhelming probability, we have that for every* $i \in [2]$*: (i)* $\|\hat{\mathbf{f}}_i\|_\infty < b$*, (ii)* $\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_i) = \mathbf{v}_i$*, and (iii)* $\mathbf{u}_i = \langle M \cdot \mathsf{tensor}(\vec{\mathbf{r}}_i), (\mathbb{x}_{w,i} || \vec{\mathbf{w}}_i) \rangle$.

Proof. The proof is almost identical to that of Claim 2. The only difference is that the sum-check claim in Eqn. 20 becomes

$$\sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log m}} g(\vec{\mathbf{b}}) = \sum_{i=1}^{2} (\alpha_i \mathbf{v}_i + \zeta_i \mathbf{u}_i) \,, \tag{36}$$

where g is defined as

$$g(\vec{\mathbf{b}}) := \sum_{i=1}^{2} \left[\alpha_i g_{1,i}(\vec{\mathbf{b}}) + \mu_i g_{2,i}(\vec{\mathbf{b}}) + \zeta_i g_{3,i}(\vec{\mathbf{b}}) \right]$$

By the uniqueness of MLE (Corollary 2.1), for every $i \in [2]$, denote $\mathbf{z}_i := (\mathbb{x}_{w,i} || \vec{\mathbf{w}}_i)$, we similarly have that

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

Thus similar to Eqn. 23 in Claim 2, by definition of the polynomial g in Eqn. 30, we can rewrite Eqn. 36 as

$$\begin{aligned} \sum_{\vec{\mathbf{b}} \in \{0,1\}^{\log m}} g(\vec{\mathbf{b}}) &= \sum_{i=1}^{2} \left(\alpha_i \cdot \mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_i) + \mu_i \cdot p_i(\vec{\beta}) + \zeta_i \cdot \langle M \cdot \mathsf{tensor}(\vec{\mathbf{r}}_i), \mathbf{z}_i \rangle \right) \\ &= \sum_{i=1}^{2} (\alpha_i \mathbf{v}_i + \zeta_i \mathbf{u}_i) \end{aligned}$$

where $p_i(X)$ and $[\mu_i]_{i=1}^2$ are defined the same way as in Eqn. 23, and $[\alpha_i, \zeta_i]_{i=1}^2$, μ_1 are uniformly sampled from the strong sampling set $\mathcal{C}$. Re-arranging the terms, we have that

$$\begin{aligned} h(\alpha_1, \alpha_2, \mu_1, \zeta_1, \zeta_2) &:= \sum_{i=1}^{2} \left[\alpha_i \cdot \left(\mathbf{v}_i - \mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_i) \right) + \mu_i \cdot p_i(\vec{\beta}) + \zeta_i \cdot (\langle M \cdot \mathsf{tensor}(\vec{\mathbf{r}}_i), \mathbf{z}_i \rangle - \mathbf{u}_i) \right] \\ &= 0 \end{aligned}$$

where $\mu_2 := 1$. By the General Schwartz-Zippel Lemma (Lemma 2.4), the polynomial h is identically zero with overwhelming probability. Thus, for every $i \in [2]$, we have that $\mathsf{mle}\left[\hat{\mathbf{f}}_i\right](\vec{\mathbf{r}}_i) = \mathbf{v}_i$, $p_i(X) = 0$, and $\langle M \cdot \mathsf{tensor}(\vec{\mathbf{r}}_i), \mathbf{z}_i \rangle = \mathbf{u}_i$. With the same argument in the proof of Claim 2, $p_i(X) = 0$ implies that $\|\hat{\mathbf{f}}_i\|_\infty < b$. Thus the claim holds. □

□