Proof-Carrying Data from Multi-folding Schemes

Zibo Zhou[1], Zongyang Zhang[1], and Jin Dong[2]

[1] School of Cyber Science and Technology, Beihang University, Beijing, China
{zbzhou,zongyangzhang}@buaa.edu.cn
[2] Beijing Academy of Blockchain and Edge Computing, BABEC
dongjin@baec.org.cn

Abstract. Proof-carrying data (PCD) is a powerful cryptographic primitive that allows mutually distrustful parties to perform distributed computation defined on directed acyclic graphs in an efficiently verifiable manner. Important efficiency parameters include prover's cost at each step and the recursion overhead that measures the additional cost apart from proving the computation.

In this paper, we construct a PCD scheme having the smallest prover's cost and recursion overhead in the literature. Specifically, the prover's cost at each step is dominated by only one $O(|C|)$-sized multi-scalar multiplication (MSM), and the recursion overhead is dominated by only one $2r$-sized MSM, where $|C|$ is the computation size and r is the number of incoming edges at certain step. In contrast, the state-of-the-art PCD scheme requires $4r + 12$ $O(|C|)$-sized MSMs w.r.t. the prover's cost and six $2r$-sized MSMs, one $6r$-sized MSM w.r.t. the recursion overhead. In addition, our PCD scheme supports more expressive constraint system for computations—*customizable constraint system (CCS)* that supports high-degree constraints efficiently, in contrast with *rank-1 constraint system (R1CS)* that supports only quadratic constraints used in existing PCD schemes.

Underlying our PCD scheme is a multi-folding scheme that reduces the task of checking multiple instances into the task of checking one. We generalize existing construction to support arbitrary number of instances.

Keywords: proof-carrying data · folding schemes · recursive zero-knowledge proofs.

1 Introduction

Proof-carrying data (PCD), introduced by Chiesa and Tromer [13], is a powerful cryptographic primitive that enables mutually distrustful parties to perform distributed computations defined on directed acyclic graphs, while every intermediate state of the computation can be verified efficiently. It generalizes incrementally verifiable computation (IVC) [32] which enables a possibly infinite computation defined on path graphs such that the correctness can be verified efficiently at any point. PCD has found numerous applications to enforcing language semantics [15], verifiable MapReduce computations [14], verifiable photo editing [27], verifiable registries [31], and blockchains [4,11,20].

There has been tremendous interest and progress in designing efficient PCD schemes. A classic method for constructing PCD is via *recursive composition* of succinct non-interactive argument of knowledge (SNARK) [2,1,12]. Informally, at each step i, the prover uses a SNARK to prove that the i-th step of the computation is executed correctly and the SNARK verifier expressed as an arithmetic circuit has accepted the SNARK proof from step $i - 1$. Important efficiency parameters include the prover's cost at each step and the *recursion overhead*, i.e., the verifier's operations expressed as an arithmetic circuit that the prover must prove at each step besides proving the computation. This method yields a secure PCD construction but the concrete efficiency is limited by the use of cycles of expensive pairing-friendly elliptic curves for pairing-based SNARKs [1] or heavy use of cryptographic hash functions for hash-based SNARKs [12].

Bünz et al. [8], denoted as BCMS20, introduced an alternative method for constructing PCD by formalizing a novel notion—atomic accumulation scheme from Halo [5]. Instead of expressing the whole SNARK verifier as a circuit, BCMS20 only requires expressing the verifier of the atomic accumulation scheme as a circuit, whose size is much smaller. However, the prover at each step has to perform expensive fast Fourier transforms (FFTs) and the recursion overhead is still dominated by group operations that scale logarithmically with the size of the computation. Later, Bünz et al. [7], denoted as BCLMS21, extended the notion of atomic accumulation scheme to split accumulation scheme. By designing a non-interactive argument of knowledge with a split accumulation scheme for *rank-1 constraint system (R1CS)* [19], they constructed a PCD scheme whose recursion overhead is dominated by constant number of group operations. But the prover at each step still has to generate a proof, resulting in slightly high prover's cost and recursion overhead. Kothapalli and Setty [23] avoided generating a proof by introducing and constructing a multi-folding scheme for *customizable constraint system (CCS)* [29]. Nevertheless, their multi-folding scheme is tailored to fold two instances and thus results in an IVC scheme that is only a special case of PCD.

1.1 Our Contributions

In this paper, we generalize the multi-folding scheme of Kothapalli and Setty [23] and then construct a PCD scheme having the smallest prover's cost and recursion overhead in the literature. Table 1 shows the comparison between the state-of-the-art PCD scheme and ours. We elaborate our contributions below.

A Multi-folding Scheme for Arbitrary Number of Instances. We generalize the multi-folding scheme of Kothapalli and Setty [23] to support folding arbitrary number of instances. Our scheme could reduce the task of checking multiple instances into the task of checking a single instance. By using the Fiat-Shamir transformation [18] we could make it non-interactive, which is the basis of our PCD scheme.

An Efficient PCD Scheme Supporting *Customizable Constraint System.* We construct an efficient PCD scheme that supports a more expressive constraint system—*customizable constraint system (CCS)*. The recursion overhead of our PCD scheme is dominated by only one multi-scalar multiplication (MSM)[3] of size $2r$, and the prover's cost at each step is dominated by only one MSM of size $O(n)$, where r is the number of incoming edges of a node at certain step and n is the number of variables in the constraint system. The proof size is logarithmic and the verifier's cost is dominated by only one MSM of size n. Therefore, our PCD scheme not only has smaller prover's cost and recursion overhead, but also reduces the proof size exponentially and reduces the verifier's cost significantly. Besides, the CCS our PCD scheme supports can generalize not only R1CS without overhead but also Plonkish which supports constraints with degree larger than two and thereby represents certain program executions more succinctly. In particular, when $t = 3, q = 2, d = 2$, the data showed in Table 1 measures the efficiency of our PCD scheme using R1CS.

However, achieving zero-knowledge in our PCD scheme requires the multi-folding scheme to be zero-knowledge and an additional invocation of a general zero-knowledge succinct non-interactive argument of knowledge (zkSNARK), which we leave in future work.

[3] For field elements $a_1, \ldots . a_r$ and elliptic curve group elements $G_1, \ldots, G_r$, the multi-scalar multiplication denotes the operation $a_1G_1 + \cdots + a_rG_r$. The group scalar multiplication denotes the operation a_1G_1.

1.2 Related Work

IVC/PCD from SNARKs. Chiesa and Tromer [13] introduced the notion of proof-carrying data (PCD) which generalizes incrementally verifiable computation (IVC) [32]. Bitansky et al. [2] proved that any SNARK for machine computation can be efficiently transformed into a corresponding PCD scheme for constant-depth graphs via recursive composition. But it was not realized in practice due to enormous computational cost. Ben-Sasson et al. [1] achieved the first implementation of PCD using pairing-based SNARKs instantiated via pairing-friendly cycles of elliptic curves. But these curves must be constructed over large (780-bit) fields due to their low embedding degrees, resulting in poor concrete efficiency. Chiesa et al. [12] obtained the first efficient realization of post-quantum transparent PCD, but the number of constraints needed to express the verifier's computation in a circuit is dominated by numerous hash function invocations.

Table 1. Comparison of proof-carrying data schemes

	BCLMS21 [7]	This work
Constraint system	R1CS	CCS
Recursion overhead	6 $(2r)$-MSM 1 $(6r)$-MSM $5r$ $\mathbb{G}$ $r+3$ RO	$O\big(d\log m + r\cdot(\log m + t + dq)\big)$ $\mathbb{F}$ 1 $(2r)$-MSM $r+1$ H $2\log_2 m + 2$ RO
Prover's cost (each step)	$O(r\cdot(N+m+n))$ $\mathbb{F}$ $4r+12$ (m)-MSM 6 $(2r)$-MSM 1 $(6r)$-MSM $5r$ $\mathbb{G}$ $r+4$ RO	$O\big(r\cdot(N+tm+n+qmd\log^2 d)\big)$ $\mathbb{F}$ 1 $(2r)$-MSM 1 $O(n)$-MSM $2\log_2 m + 2$ RO
Proof size	$O(m+n)$ $\mathbb{F}$ 15 $\mathbb{G}$	$O(d\log m + t + \log n)$ $\mathbb{F}$ $O(\log n)$ $\mathbb{G}$
Verifier's cost	$O(N)$ $\mathbb{F}$ 10 $O(m)$-MSM 1 RO	$O(d\log m + t + dq + \log n)$ $\mathbb{F}$ 1 $O(n)$-MSM $O(\log m + \log n)$ RO
Zero-knowledge	yes	no

Notations: R1CS denotes rank-1 constraint system with parameters (m, n, N) where $N = \Omega(\max(m, n))$. CCS denotes customizable constraint system with parameters (m, n, N, t, q, d). r denotes the number of incoming edges of a node at certain step. $\mathbb{G}$ denotes group scalar multiplications when measuring overhead/cost or group elements when measuring proof size. Similarly, $\mathbb{F}$ denotes field operations or field elements. H denotes the invocation of hash functions. RO denotes the invocation of random oracles. 1 $(2r)$-MSM denotes one multi-scalar multiplication of size $2r$.

IVC/PCD from accumulation/folding schemes. Bünz et al. [8] developed and formalized a novel approach from Bowe et al. [5] to construct PCD, i.e., PCD from atomic accumulation schemes. Following this line of works, Boneh et al. [3] formalized a method to construct PCD from additive polynomial commitment schemes with a (public/private) aggregation scheme. Meanwhile, Bünz et al. [7] improved [8] by introducing the notion of split accumulation schemes to construct PCD.

Later, many researchers focus on IVC instead of PCD for its conceptual simplicity and practical applications in verifiable delay functions and zkEVM. Kothapalli et al. [24] introduced the notion of folding schemes and constructed an IVC scheme—Nova whose recursion overhead is dominated by only two group scalar multiplications. Kothapalli and Setty [21] adapted Nova to SuperNova that realizes non-uniform IVC with the same recursion overhead. Nova and SuperNova both express each step of an incremental computation with R1CS, which is restricted to checking quadratic constraints in a specific form. In practice, practitioners often use custom constraint systems such as Plonkish that allows gates to compute custom high-degree polynomials. To overcome this limitation, Mohnblatt [26] introduced Sangria which adapted the folding scheme of Nova to handle Plonkish, but the number of group operations grows linearly in the gate degree. Recently, Kothapalli and Setty [23] introduced HyperNova, an IVC scheme for CCS that simultaneously generalizes Plonkish, R1CS, and algebraic intermediate representation (AIR) without overheads. Bünz and Chen [10] introduced ProtoStar, a non-uniform IVC scheme for Plonkish/CCS. Eagen and Gabizon [17] continued the work of ProtoStar and constructed an efficient folding scheme—ProtoGalaxy for multiple instances. Kothapalli and Setty [22] introduced CycleFold—a conceptually simple approach to instantiate folding-scheme-based IVCs over a cycle of elliptic curves.

2 Preliminaries

For $t \in \mathbb{N}$, let $[t]$ denote the set $\{1, 2, \ldots, t\}$. We use $\mathbb{F}$ to denote a finite field, $\mathbb{F}^t$ to denote a vector space of dimension t over $\mathbb{F}$, and $\mathbb{F}[\ell]$ to denote the family of ℓ-variate multilinear polynomials over $\mathbb{F}$. For $x \in \mathbb{F}^t$, let x_i denote the i-th element. We use $y \xleftarrow{\$} S$ to denote the assignment of a uniformly random element in set S to y. We use $y \coloneqq c$ to denote the assignment of the constant value c to y. When A is an algorithm, we use $y \leftarrow A(x)$ to denote the assignment of the output of A on input x to y. We use λ to denote the security parameter and will drop it from the notation when it is implicit. We use $\mathsf{negl}(\lambda)$ to denote a negligible function in λ. Let PPT denote probabilistic polynomial time. A *multiset* is an extension of the concept of a set where every element has a positive multiplicity. For a tuple containing a semicolon, those variables listed before the semicolon are public (known to both the prover and the verifier), and those listed after it are secret (known only to the prover).

Lemma 1 (Multilinear Extension [16]). *For every function $f : \{0,1\}^\ell \to \mathbb{F}$, there is a unique ℓ-variate multilinear polynomial $\widetilde{f} : \mathbb{F}^\ell \to \mathbb{F}$ such that $\widetilde{f}(x) = f(x)$ for all $x \in \{0,1\}^\ell$. We call $\widetilde{f}$ the multilinear extension (MLE) of function f, and $\widetilde{f}$ can be expressed as*

$$\widetilde{f}(X) = \sum_{x \in \{0,1\}^\ell} f(x) \cdot \widetilde{eq}(X, x),$$

where $\widetilde{eq}(X, x) = \prod_{i=1}^{\ell} \big(x_i X_i + (1 - x_i)(1 - X_i)\big)$ and itself is the MLE of function

$$eq(X, x) : \{0,1\}^\ell \to \mathbb{F}, \text{ and } eq(X, x) = \begin{cases} 1 & \text{if } X = x \\ 0 & \text{otherwise} \end{cases}.$$

Lemma 2 (Lagrange Interpolation of Multilinear Polynomials [30, Lemma 3.6]). *For $\ell \in \mathbb{N}$, any ℓ-variate multilinear polynomial P could be expressed as $P(X) = \sum_{x \in \{0,1\}^\ell} \widetilde{eq}(X, x) \cdot P(x)$.*

Lemma 3 (Schwartz-Zippel Lemma [28]). *Let $f : \mathbb{F}^\ell \to \mathbb{F}$ be a non-zero ℓ-variate polynomial of total degree d. Let S be any finite subset of $\mathbb{F}$. Then for $x \xleftarrow{\$} S^\ell$, $\Pr[f(x) = 0] \leq \frac{d}{|S|}$.*

2.1 The Sum-Check Protocol

We adapt the description from Kothapalli and Setty [23]. The sum-check protocol [25] is an interactive protocol allowing a prover $\mathcal{P}$ to convince a verier $\mathcal{V}$ of the validity of the statement

$$T = \sum_{x_1 \in \{0,1\}} \cdots \sum_{x_\ell \in \{0,1\}} g(x_1, \ldots, x_\ell),$$

where $g : \mathbb{F}^\ell \to \mathbb{F}$ is an ℓ-variate polynomial over some finite field $\mathbb{F}$, and the degree of each variable is at most d. While $\mathcal{V}$ could directly compute T using $O(2^\ell)$ evaluation of g, the sum-check protocol reduces $\mathcal{V}$'s work to be polynomial in ℓ. In the protocol, $\mathcal{V}$ takes as input randomness $r \in \mathbb{F}^\ell$ and interacts with $\mathcal{P}$ over a sequence of ℓ rounds. At the end of this interaction, $\mathcal{V}$ outputs a claim about the evaluation $g(r)$. We denote the sum-check protocol as $c \leftarrow \langle \mathcal{P}, \mathcal{V}(r) \rangle (g, \ell, d, T)$, and it satisfies the following properties.

- **Completeness.** If $T = \sum_{x \in \{0,1\}^\ell} g(x)$, then for an honest $\mathcal{P}$ and for all $r \in \mathbb{F}^\ell$, $\Pr_r \big[c \leftarrow \langle \mathcal{P}, \mathcal{V}(r) \rangle (g, \ell, d, T) \wedge g(r) = c \big] = 1$.
- **Soundness.** If $T \neq \sum_{x \in \{0,1\}^\ell} g(x)$, then for any adversary $\mathcal{P}^*$ and for all $r \in \mathbb{F}^\ell$, $\Pr_r \big[c \leftarrow \langle \mathcal{P}^*, \mathcal{V}(r) \rangle (g, \ell, d, T) \wedge g(r) = c \big] \leq \ell \cdot d / \mathbb{F}$.
- **Succinctness.** The communication cost is $O(\ell \cdot d)$ elements in $\mathbb{F}$.

2.2 Polynomial Commitment Schemes

Definition 1 (Polynomial Commitment Schemes for Multilinear Polynomials [9]). *A polynomial commitment scheme for multilinear polynomials is a tuple $\mathsf{PCS} = (\mathsf{Setup}, \mathsf{Com}, \mathsf{Open}, \mathsf{Eval})$ where*

- $\mathsf{pp}_{\mathsf{pcs}} \leftarrow \mathsf{Setup}(1^\lambda, \ell)$ *takes the security parameter λ with the unary form and the number of variables ℓ in a multilinear polynomial as input, and outputs some public parameter $\mathsf{pp}_{\mathsf{pcs}}$;*
- $C \leftarrow \mathsf{Com}(\mathsf{pp}_{\mathsf{pcs}}, g)$ *takes $\mathsf{pp}_{\mathsf{pcs}}$ and an ℓ-variate multilinear polynomial over a finite field $g \in \mathbb{F}[\ell]$ as input, and outputs a commitment C;*
- $1/0 \leftarrow \mathsf{Open}(\mathsf{pp}_{\mathsf{pcs}}, C, g)$ *takes $\mathsf{pp}_{\mathsf{pcs}}, C, g$ as input, and outputs 1/0 to denote that C is indeed a commitment to g or not.*
- $1/0 \leftarrow \mathsf{Eval}(\mathsf{pp}_{\mathsf{pcs}}, C, \ell, r, v; g)$ *is an interactive protocol between a prover $\mathcal{P}$ and a verifier $\mathcal{V}$. Both of them know $\mathsf{pp}_{\mathsf{pcs}}, C, \ell, r \in \mathbb{F}^\ell, v \in \mathbb{F}$. $\mathcal{P}$ additionally knows g and attempts to convince $\mathcal{V}$ that $C = \mathsf{Com}(\mathsf{pp}_{\mathsf{pcs}}, g)$ and $g(r) = v$. The outputs 1/0 denote that $\mathcal{V}$ accepts or not.*

A polynomial commitment scheme PCS should satisfy completeness, binding and knowledge soundness defined below.

- **Completeness.** PCS *has completeness if for any ℓ-variate multilinear polynomial $g \in \mathbb{F}[\ell]$,*

$$\Pr \left[\begin{array}{l} \mathsf{Eval}(\mathsf{pp}_{\mathsf{pcs}}, C, \ell, r, v; g) = 1 \\ \wedge\, g(r) = v \end{array} \middle| \begin{array}{r} \mathsf{pp}_{\mathsf{pcs}} \leftarrow \mathsf{Setup}(1^\lambda, \ell); \\ C \leftarrow \mathsf{Com}(\mathsf{pp}_{\mathsf{pcs}}, g) \end{array} \right] \geq 1 - \mathsf{negl}(\lambda).$$

- **Binding.** PCS *has binding if for any PPT adversary* $\mathcal{A}$*, size parameter* $\ell \geq 1$*,*

$$\Pr\left[\begin{matrix} b_0 = b_1 \neq 0 \\ \wedge\, g_0 \neq g_1 \end{matrix} \middle| \begin{matrix} \mathsf{pp}_{\mathsf{pcs}} \leftarrow \mathsf{Setup}(1^\lambda, \ell); (C, g_0, g_1) \leftarrow \mathcal{A}(\mathsf{pp}_{\mathsf{pcs}}); \\ b_0 \leftarrow \mathsf{Open}(\mathsf{pp}_{\mathsf{pcs}}, C, g_0); b_1 \leftarrow \mathsf{Open}(\mathsf{pp}_{\mathsf{pcs}}, C, g_1) \end{matrix}\right] \leq \mathsf{negl}(\lambda).$$

- **Knowledge soundness.** PCS *has knowledge soundness if given* $\mathsf{pp}_{\mathsf{pcs}} \leftarrow \mathsf{Setup}(1^\lambda, \ell)$*,* Eval *is a succinct argument of knowledge for NP relation*

$$\mathcal{R}_{\mathsf{Eval}}(\mathsf{pp}_{\mathsf{pcs}}) = \left\{(C, r, v; g) : g \in \mathbb{F}[\ell] \wedge g(r) = v \wedge \mathsf{Open}(\mathsf{pp}_{\mathsf{pcs}}, C, g) = 1\right\}.$$

PCS *is additively homomorphic if for all* ℓ *and public parameters* $\mathsf{pp}_{\mathsf{pcs}}$*, and for any* ℓ*-variate multilinear polynomial* $g_1, g_2 \in \mathbb{F}[\ell]$*,* $\mathsf{Com}(\mathsf{pp}_{\mathsf{pcs}}, g_1) + \mathsf{Com}(\mathsf{pp}_{\mathsf{pcs}}, g_2) = \mathsf{Com}(\mathsf{pp}_{\mathsf{pcs}}, g_1 + g_2)$*.*

A polynomial commitment to an ℓ*-variate multilinear polynomial can be viewed as a commitment to a* 2^ℓ*-sized vector. In this paper, we instantiate the polynomial commitment scheme for multilinear polynomials with Bulletproofs [6], whose proof size is dominated by* $O(\ell)$ *group elements and verifier's cost is dominated by one MSM of size* $O(2^\ell)$ *in the* Eval *protocol.*

2.3 Multi-folding Schemes

Definition 2 (Multi-folding Schemes [23]). *Consider relations* $\mathcal{R}_1$ *and* $\mathcal{R}_2$ *over public parameters, structure, instance, and witness tuples, a predicate* compat *that structures in* $\mathcal{R}_1$ *and* $\mathcal{R}_2$ *must satisfy, and size parameters* $\mu, \nu \in \mathbb{N}$*. A multi-folding scheme for* $(\mathcal{R}_1, \mathcal{R}_2, \mathsf{compat}, \mu, \nu)$ *is a tuple of algorithms* $\mathrm{MFS} = (\mathcal{G}, \mathcal{K}, \mathcal{P}, \mathcal{V})$ *where*

- $\mathsf{fpp} \leftarrow \mathcal{G}\left(1^\lambda\right)$ *is a PPT generator algorithm. It takes the security parameter* λ *with the unary form as input, and outputs some public parameter* fpp*;*
- $(\mathsf{fpk}, \mathsf{fvk}) \leftarrow \mathcal{K}(\mathsf{fpp}, (\mathsf{s}_1, \mathsf{s}_2))$ *is a deterministic encoder algorithm. It takes* fpp *and structure* $\mathsf{s}_1, \mathsf{s}_2$ *among the instances to be folded as input, and outputs a prover key* fpk *and a verifier key* fvk*;*
- $(\mathsf{u}, \mathsf{w}) \leftarrow \left\langle \mathcal{P}(\mathsf{fpk}, \overrightarrow{\mathsf{w}_1}, \overrightarrow{\mathsf{w}_2}), \mathcal{V}(\mathsf{fvk}) \right\rangle \left(\overrightarrow{\mathsf{u}_1}, \overrightarrow{\mathsf{u}_2}\right)$ *denotes the interaction between a PPT prover* $\mathcal{P}$ *and a PPT verifier* $\mathcal{V}$*.* $\mathcal{P}$ *inputs* fpk*, a vector of instances* $\overrightarrow{\mathsf{u}_1}$ *in* $\mathcal{R}_1$ *of size* μ *with structure* s_1*, a vector of instances* $\overrightarrow{\mathsf{u}_2}$ *in* $\mathcal{R}_2$ *of size* ν *with structure* s_2*, and corresponding witnesses* $\overrightarrow{\mathsf{w}_1}, \overrightarrow{\mathsf{w}_2}$*.* $\mathcal{V}$ *inputs* $\mathsf{fvk}, \overrightarrow{\mathsf{u}_1}, \overrightarrow{\mathsf{u}_2}$*. At the end of the interaction,* $\mathcal{P}$ *outputs a folded instance-witness pair* (u, w) *in* $\mathcal{R}_1$ *with structure* s_1*, and* $\mathcal{V}$ *outputs a folded instance* u *in* $\mathcal{R}_1$ *with structure* s_1*.*

Let $\mathcal{R}^{(n)}$ *be the relation so that* $\left(\mathsf{fpp}, \mathsf{s}, \overrightarrow{\mathsf{u}}, \overrightarrow{\mathsf{w}}\right) \in \mathcal{R}^{(n)}$ *if and only if* $\left(\mathsf{fpp}, \mathsf{s}, \overrightarrow{\mathsf{u}}_i, \overrightarrow{\mathsf{w}}_i\right) \in \mathcal{R}$ *for all* $i \in [n]$*. A multi-folding scheme* MFS *should sastisfy perfect completeness and knowledge soundness defined below.*

- **Perfect Completeness.** MFS *has perfect completeness if for all PPT adversaries* $\mathcal{A}$*,*

$$\Pr\left[(\mathsf{fpp}, \mathsf{s}_1, \mathsf{u}, \mathsf{w}) \in \mathcal{R}_1 \middle| \begin{matrix} \mathsf{fpp} \leftarrow \mathcal{G}\left(1^\lambda\right); \\ \left((\mathsf{s}_1, \mathsf{s}_2), (\overrightarrow{\mathsf{u}_1}, \overrightarrow{\mathsf{w}_1}), (\overrightarrow{\mathsf{u}_2}, \overrightarrow{\mathsf{w}_2})\right) \leftarrow \mathcal{A}(\mathsf{fpp}); \\ \mathsf{compat}(\mathsf{s}_1, \mathsf{s}_2) = \mathsf{true}; \\ \left(\mathsf{fpp}, \mathsf{s}_1, \overrightarrow{\mathsf{u}_1}, \overrightarrow{\mathsf{w}_1}\right) \in \mathcal{R}_1^{(\mu)}; \left(\mathsf{fpp}, \mathsf{s}_2, \overrightarrow{\mathsf{u}_2}, \overrightarrow{\mathsf{w}_2}\right) \in \mathcal{R}_2^{(\nu)}; \\ (\mathsf{fpk}, \mathsf{fvk}) \leftarrow \mathcal{K}(\mathsf{fpp}, (\mathsf{s}_1, \mathsf{s}_2)); \\ (\mathsf{u}, \mathsf{w}) \leftarrow \left\langle \mathcal{P}(\mathsf{fpk}, \overrightarrow{\mathsf{w}_1}, \overrightarrow{\mathsf{w}_2}), \mathcal{V}(\mathsf{fvk}) \right\rangle \left(\overrightarrow{\mathsf{u}_1}, \overrightarrow{\mathsf{u}_2}\right) \end{matrix}\right] = 1.$$

- **Knowledge Soundness.** MFS *has knowledge soundness if for any expected polynomial time adversary $\mathcal{A}$ and $\mathcal{P}^*$, there exists an expected polynomial time extractor $\mathcal{E}$ such that over all randomness ρ,*

$$\Pr\left[\begin{array}{l|r} \left(\mathsf{fpp}, \mathsf{s}_1, \overrightarrow{\mathsf{u}_1}, \overrightarrow{\mathsf{w}_1}\right) \in \mathcal{R}_1^{(\mu)} & \mathsf{fpp} \leftarrow \mathcal{G}\left(1^\lambda\right); \\ \wedge \left(\mathsf{fpp}, \mathsf{s}_2, \overrightarrow{\mathsf{u}_2}, \overrightarrow{\mathsf{w}_2}\right) \in \mathcal{R}_2^{(\nu)} & \left((\mathsf{s}_1, \mathsf{s}_2), (\overrightarrow{\mathsf{u}_1}, \overrightarrow{\mathsf{u}_2}), \mathsf{st}\right) \leftarrow \mathcal{A}(\mathsf{fpp}, \rho); \\ & \mathsf{compat}(\mathsf{s}_1, \mathsf{s}_2) = \mathsf{true}; \\ & (\overrightarrow{\mathsf{w}_1}, \overrightarrow{\mathsf{w}_2}) \leftarrow \mathcal{E}(\mathsf{fpp}, \rho) \end{array}\right] \geq$$

$$\Pr\left[\begin{array}{l|r} (\mathsf{fpp}, \mathsf{s}_1, \mathsf{u}, \mathsf{w}) \in \mathcal{R}_1 & \mathsf{fpp} \leftarrow \mathcal{G}\left(1^\lambda\right); \\ & \left((\mathsf{s}_1, \mathsf{s}_2), (\overrightarrow{\mathsf{u}_1}, \overrightarrow{\mathsf{u}_2}), \mathsf{st}\right) \leftarrow \mathcal{A}(\mathsf{fpp}, \rho); \\ & \mathsf{compat}(\mathsf{s}_1, \mathsf{s}_2) = \mathsf{true}; \\ & (\mathsf{fpk}, \mathsf{fvk}) \leftarrow \mathcal{K}(\mathsf{fpp}, (\mathsf{s}_1, \mathsf{s}_2)); \\ & (\mathsf{u}, \mathsf{w}) \leftarrow \left\langle \mathcal{P}^*(\mathsf{fpk}, \mathsf{st}), \mathcal{V}(\mathsf{fvk}) \right\rangle (\overrightarrow{\mathsf{u}_1}, \overrightarrow{\mathsf{u}_2}) \end{array}\right] - \mathsf{negl}(\lambda).$$

- **Efficiency.** *We consider* MFS *non-trivial if the communication costs and $\mathcal{V}$'s computation are lower in the case where $\mathcal{V}$ participates in the multi-folding scheme and then checks a witness sent by $\mathcal{P}$ for the folded instance than the case where $\mathcal{V}$ checks witnesses sent by $\mathcal{P}$ for each of the original instances.*

Construction 1 (Non-interactive Multi-folding Schemes [23]). *A public-coin multi-folding scheme where all the messages sent from $\mathcal{V}$ to $\mathcal{P}$ are chosen uniformly at random could be transformed into a non-interactive multi-folding scheme* $\text{NIMFS} = (\mathcal{G}', \mathcal{K}', \mathcal{P}', \mathcal{V}')$ *using the Fiat-Shamir transformation [18] where*

- $\mathsf{fpp}' \leftarrow \mathcal{G}'\left(1^\lambda\right)$ *takes* 1^λ *as input, and outputs some public parameter* fpp'*;*
- $(\mathsf{fpk}', \mathsf{fvk}') \leftarrow \mathcal{K}'(\mathsf{fpp}', (\mathsf{s}_1, \mathsf{s}_2)))$ *takes* fpp' *and structures* $\mathsf{s}_1, \mathsf{s}_2$ *as input, and outputs a prover key* fpk' *and a verifier key* fvk'*;*
- $(\mathsf{u}, \mathsf{w}, \pi) \leftarrow \mathcal{P}'\big(\mathsf{fpk}', (\overrightarrow{\mathsf{u}_1}, \overrightarrow{\mathsf{w}_1}), (\overrightarrow{\mathsf{u}_2}, \overrightarrow{\mathsf{w}_2})\big)$ *takes* $\mathsf{fpk}', (\overrightarrow{\mathsf{u}_1}, \overrightarrow{\mathsf{w}_1}), (\overrightarrow{\mathsf{u}_2}, \overrightarrow{\mathsf{w}_2})$ *as input, and outputs a folded instance-witness pair* (u, w) *and a proof* π*;*
- $\mathsf{u} \leftarrow \mathcal{V}'\big(\mathsf{fvk}', \overrightarrow{\mathsf{u}_1}, \overrightarrow{\mathsf{u}_2}, \pi\big)$ *takes* $\mathsf{fvk}', \overrightarrow{\mathsf{u}_1}, \overrightarrow{\mathsf{u}_2}, \pi$ *as input, and outputs a folded instance* u*.*

2.4 (Linearized) Committed Customizable Constraint System

Setty et al. [29] introduced customizable constraint systems (CCS), a new constraint system that generalizes arithmetic circuits. Later, Kothapalli and Setty [23] described two variants of CCS—committed customizable constraint systems (CCCS) and linearized committed customizable constraint systems (LCCCS)—that are amenable to construct their multi-folding schemes. In particular, they let $\mathcal{R}_1$ be the LCCCS relation $\mathcal{R}_{\text{LCCCS}}$ and $\mathcal{R}_2$ be the CCCS relation $\mathcal{R}_{\text{CCCS}}$, a NP-complete relation. We follow their work and adapt their definition of CCCS and LCCCS.

For a matrix $M \in \mathbb{F}^{m \times n}$ where $m, n \in \mathbb{N}$ and let $s = \log_2 m, s' = \log_2 n$, interpret it as a function $\{0,1\}^s \times \{0,1\}^{s'} \to \mathbb{F}$. Then we could define its unique multilinear extension $\widetilde{M}$ as a multilinear polynomial in $s+s'$ variables such that for all $x \in \{0,1\}^s, y \in \{0,1\}^{s'}$, $\widetilde{M}(x,y) = M(x,y)$. Similarly, for a vector $w \in \mathbb{F}^m$, interpret it as a function $\{0,1\}^s \to \mathbb{F}$. Then we let $\widetilde{w}$ denote its unique multilinear extension.

Definition 3 (Committed Customizable Constraint System [23]). *A CCCS relation* $\mathcal{R}_{\text{CCCS}}$ *consists of tuples containing public parameters, structure, instance and witness where*

– *public parameters consist of size bounds* $m, n, N, \ell, t, q, d \in \mathbb{N}$ *and* $\mathsf{pp}_{\mathsf{pcs}}$*, where assume that* $n = 2 \cdot (\ell + 1)$ *for simplicity,* $\mathsf{pp}_{\mathsf{pcs}} \leftarrow \mathsf{Setup}(1^\lambda, \log_2 n - 1)$*;*
– *the structure consists of:*
  - *a sequence of sparse multilinear polynomials in* $s + s'$ *variables* $\widetilde{M_1}, \ldots, \widetilde{M_t}$ *such that they evaluate to a non-zero value in at most* $N = \Omega(\max(m, n))$ *locations over the Boolean hypercube* $\{0,1\}^s \times \{0,1\}^{s'}$*, where assume that* $s = \log_2 m, s' = \log_2 n$*, and for* $i \in [t]$*,* $\widetilde{M_i}$ *is the unique multilinear extension of matrice* $M_i \in \mathbb{F}^{m \times n}$*;*
  - *a sequence of* q *multisets* $(S_1, \ldots, S_q)$*, where an element in each multiset is from the set* $[t]$ *and the cardinality of each multiset is at most* d*;*
  - *a sequence of* q *constants* $(c_1, \ldots, c_q)$*, where each constant is from* $\mathbb{F}$*;*
– *the instance is* (C, x)*, where* C *is a commitment to a multilinear polynomial in* $s' - 1$ *variables and* $\mathsf{x} \in \mathbb{F}^\ell$*;*
– *the witness is a multilinear polynomial* $\widetilde{w}$ *in* $s' - 1$ *variables, where* $\widetilde{w}$ *is the unique multilinear extension of vector* $w \in \mathbb{F}^{\ell+1}$*.*

Given public parameters, a $\mathcal{R}_{\text{CCCS}}$ *structure-instance tuple is satisfied by a* $\mathcal{R}_{\text{CCCS}}$ *witness if* $C = \mathsf{Com}\big(\mathsf{pp}_{\mathsf{pcs}}, \widetilde{w}\big)$ *and for all* $x \in \{0,1\}^s$*,*

$$\sum_{i=1}^{q} c_i \cdot \Big(\prod_{j \in S_i} \Big(\sum_{y \in \{0,1\}^{s'}} \widetilde{M_j}(x, y) \cdot \widetilde{z}(y) \Big) \Big) = 0,$$

where $\widetilde{z}$ *is the unique multilinear extension of vector* $(1, \mathsf{x}, w) \in \mathbb{F}^n$*, i.e., an* s'*-variate multilinear polynomial such that* $\widetilde{z}(y) = \widetilde{(1, \mathsf{x}, w)}(y)$ *for all* $y \in \{0,1\}^{s'}$*.*

Definition 4 (Linearized Committed Customizable Constraint System [23]). *A LCCCS relation* $\mathcal{R}_{\text{LCCCS}}$ *consists of tuples containing public parameters, structure, instance and witness where the public parameters and structure are the same as those in a CCCS relation. The instance and witness are as follows:*

– *the instance is* $(C, u, \mathsf{x}, r, v_1, \ldots, v_t)$*, where* $u \in \mathbb{F}, \mathsf{x} \in \mathbb{F}^\ell, r \in \mathbb{F}^s, v_i \in \mathbb{F}$ *for all* $i \in [t]$*, and* C *is a commitment to a multilinear polynomial in* $s' - 1$ *variables;*
– *the witness is a multilinear polynomial* $\widetilde{w}$ *in* $s' - 1$ *variables, where* $\widetilde{w}$ *is the unique multilinear extension of vector* $w \in \mathbb{F}^{\ell+1}$*.*

Given public parameters, a $\mathcal{R}_{\text{LCCCS}}$ *structure-instance tuple is satisfied by a* $\mathcal{R}_{\text{LCCCS}}$ *witness if* $C = \mathsf{Com}\big(\mathsf{pp}_{\mathsf{pcs}}, \widetilde{w}\big)$ *and for all* $i \in [t]$*,*

$$v_i = \sum_{y \in \{0,1\}^{s'}} \widetilde{M_i}(r, y) \cdot \widetilde{z}(y),$$

where $\widetilde{z}$ *is is the unique multilinear extension of vector* $(u, \mathsf{x}, w) \in \mathbb{F}^n$*, i.e.,an* s'*-variate multilinear polynomial such that* $\widetilde{z}(y) = \widetilde{(u, \mathsf{x}, w)}(y)$ *for all* $y \in \{0,1\}^{s'}$*.*

2.5 Proof-Carrying Data

We refer to the definition of Bünz et al [7]. Define a transcript T as a directed acyclic graph where each vertex $v \in V(\mathsf{T})$ is labeled by local data $z_{\mathsf{loc}}^{(v)}$ and each edge $e \in E(\mathsf{T})$ is labeled by a message

$z^{(e)} \neq \perp$. The output $\mathsf{o}(\mathsf{T})$ of a transcript T is $z^{(e)}$ where $e = (v, v')$ is the lexicographically-first edge such that v' is a sink. For a class of compliance predicates F, define that a vertex $v \in V(\mathsf{T})$ is φ-compliant for $\varphi \in \mathsf{F}$ if for all outgoing edges $e = (v, v') \in E(\mathsf{T})$:

- (base case) if v has no incoming edges, $\varphi\big(z^{(e)}, z^{(v)}_{\mathsf{loc}}, \perp, \ldots, \perp\big)$ accepts;
- (recursive case) if v has incoming edges $e_1, \ldots, e_r$, $\varphi\big(z^{(e)}, z^{(v)}_{\mathsf{loc}}, z^{(e_1)}, \ldots, z^{(e_r)}\big)$ accepts.

We say that T is φ-compliant if all of its vertices are φ-compliant.

Definition 5 (Proof-Carrying Data [7]). *A proof-carrying data scheme for a class of compliance predicates* F *is a tuple of algorithms* $\mathsf{PCD} = (\mathsf{G}, \mathsf{K}, \mathsf{P}, \mathsf{V})$ *where*

- $\mathsf{pp} \leftarrow \mathsf{G}\left(1^\lambda\right)$ *takes the security parameter* λ *with the unary form as input, and outputs some public parameter* pp*;*
- $(\mathsf{pk}, \mathsf{vk}) \leftarrow \mathsf{K}(\mathsf{pp}, \varphi)$ *takes* pp *and a compliance predicate* $\varphi \in \mathsf{F}$ *as input, and outputs a prover key* pk *and a verifier key* vk*;*
- $\Pi \leftarrow \mathsf{P}\big(\mathsf{pk}, z, z_{\mathsf{loc}}, \{z_i, \Pi_i\}_{i=1}^r\big)$ *takes* pk*, message* z *of the outgoing edge, local data* z_{loc}*, messages* $\{z_i\}_{i=1}^r$ *of incoming edges and their corresponding proofs* $\{\Pi_i\}_{i=1}^r$ *as input, and outputs a new proof* Π *to attest the correctness of* z*;*
- $0/1 \leftarrow \mathsf{V}\big(\mathsf{vk}, z, \Pi\big)$ *takes* vk, z, Π *as input, and outputs 0/1 to reject or accept.*

A proof-carrying data scheme PCD *should sastify perfect completeness and knowledge soundness defined below. Note that we do not require it to satisfy zero-knowledge here as our* PCD *scheme needs an auxiliary zkSNARK to achieve zero-knowledge which is not the main focus of this paper.*

- **Perfect Completeness.** PCD *has perfect completeness if for every adversary* $\mathcal{A}$*,*

$$\Pr\left[\mathsf{V}\big(\mathsf{vk}, z, \Pi\big) = 1 \;\middle|\; \begin{array}{r} \mathsf{pp} \leftarrow \mathsf{G}\left(1^\lambda\right); \\ \big(\varphi, z, z_{\mathsf{loc}}, \{z_i, \Pi_i\}_{i=1}^r\big) \leftarrow \mathcal{A}(\mathsf{pp}); \\ (\mathsf{pk}, \mathsf{vk}) \leftarrow \mathsf{K}(\mathsf{pp}, \varphi); \\ \varphi \in \mathsf{F}; \varphi(z, z_{\mathsf{loc}}, z_1, \ldots, z_r) = 1; \\ \forall i \in [r], z_i = \perp \text{ or } \mathsf{V}\big(\mathsf{vk}, z_i, \Pi_i\big) = 1; \\ \Pi \leftarrow \mathsf{P}\big(\mathsf{pk}, z, z_{\mathsf{loc}}, \{z_i, \Pi_i\}_{i=1}^r\big) \end{array}\right] = 1.$$

- **Knowledge Soundness.** PCD *has knowledge soundness (w.r.t. an auxiliary input distribution* $\mathcal{D}$*) if for every expected polynomial time adversary* P^**, there exists an expected polynomial time extractor* $\mathcal{E}_{\mathsf{P}^*}$ *such that for every set* Z*,*

$$\Pr\left[\begin{array}{c} \varphi \in \mathsf{F} \\ \wedge\ (\mathsf{pp}, \mathsf{ai}, \varphi, \mathsf{o}(\mathsf{T}), \mathsf{ao}) \in Z \\ \wedge\ \mathsf{T} \text{ is } \varphi\text{-compliant} \end{array} \;\middle|\; \begin{array}{r} \mathsf{pp} \leftarrow \mathsf{G}\left(1^\lambda\right); \\ \mathsf{ai} \leftarrow \mathcal{D}(\mathsf{pp}); \\ (\varphi, \mathsf{T}, \mathsf{ao}) \leftarrow \mathcal{E}_{\mathsf{P}^*}(\mathsf{pp}, \mathsf{ai}) \end{array}\right] \geq$$
$$\Pr\left[\begin{array}{c} \varphi \in \mathsf{F} \\ \wedge\ (\mathsf{pp}, \mathsf{ai}, \varphi, \mathsf{o}, \mathsf{ao}) \in Z \\ \wedge\ \mathsf{V}\big(\mathsf{vk}, \mathsf{o}, \Pi\big) = 1 \end{array} \;\middle|\; \begin{array}{r} \mathsf{pp} \leftarrow \mathsf{G}\left(1^\lambda\right); \\ \mathsf{ai} \leftarrow \mathcal{D}(\mathsf{pp}); \\ (\varphi, \mathsf{o}, \Pi, \mathsf{ao}) \leftarrow \mathsf{P}^*(\mathsf{pp}, \mathsf{ai}); \\ (\mathsf{pk}, \mathsf{vk}) \leftarrow \mathsf{K}(\mathsf{pp}, \varphi) \end{array}\right] - \mathsf{negl}(\lambda).$$

3 A Multi-folding Scheme for Arbitrary Number of Instances

Recall that a multi-folding scheme for $(\mathcal{R}_1, \mathcal{R}_2, \mathsf{compat}, \mu, \nu)$ allow a prover and a verifier to reduce the task of checking μ instances in $\mathcal{R}_1$ with structure s_1 and ν instances in $\mathcal{R}_2$ with structure s_2 into the task of checking a single instance in $\mathcal{R}_1$ with structure s_1. In our case, we let $\mathcal{R}_1 = \mathcal{R}_{\text{LCCCS}}$, $\mathcal{R}_2 = \mathcal{R}_{\text{CCCS}}$, and $\mathsf{compat}(\mathsf{s}_1, \mathsf{s}_2)$ require $\mathsf{s}_1 = \mathsf{s}_2$. Kothapalli and Setty [23] constructed a multi-folding scheme for $(\mathcal{R}_{\text{LCCCS}}, \mathcal{R}_{\text{CCCS}}, \mathsf{compat}, 1, 1)$. We continue their work and construct a multi-folding scheme for arbitrary values of μ, ν, which is the basis of our proof-carrying data scheme.

3.1 Overview

Suppose that there are μ $\mathcal{R}_{\text{LCCCS}}$ instance-witness pairs $\{\phi_k\}_{k\in[\mu]}$, ν $\mathcal{R}_{\text{CCCS}}$ instance-witness pairs $\{\psi_{k'}\}_{k'\in[\nu]}$, where $\phi_k \coloneqq (C_1, u, \mathsf{x}_1, r_x, v_1, \ldots, v_t; \widetilde{w_1})$, $\psi_{k'} \coloneqq (C_2, \mathsf{x}_2; \widetilde{w_2})$. Let $\phi_k.\widetilde{z_1} \coloneqq (\phi_k.u, \widetilde{\phi_k.\mathsf{x}_1}, \phi_k.w_1)$, $\psi_{k'}.\widetilde{z_2} \coloneqq (1, \widetilde{\psi_{k'}.\mathsf{x}_2}, \psi_{k'}.w_2)$. The prover $\mathcal{P}$ and the verifier $\mathcal{V}$ both input μ $\mathcal{R}_{\text{LCCCS}}$ instances $\{\phi_k.(C_1, u, \mathsf{x}_1, r_x, v_1, \ldots, v_t)\}_{k\in[\mu]}$ and ν $\mathcal{R}_{\text{CCCS}}$ instances $\{\psi_{k'}.(C_2, \mathsf{x}_2)\}_{k'\in[\nu]}$. $\mathcal{P}$ additionally inputs the corresponding witnesses $\{\phi_k.\widetilde{w_1}\}_{k\in[\mu]}, \{\psi_{k'}.\widetilde{w_2}\}_{k'\in[\nu]}$. To obtain the folded instance and witness, we rely on the random linear combination technique.

For $k \in [\mu]$, according to $\mathcal{R}_{\text{LCCCS}}$, for $i \in [t]$, we have

$$\begin{aligned}
\phi_k.v_i &= \sum_{y\in\{0,1\}^{s'}} \widetilde{M_i}(\phi_k.r_x, y) \cdot \phi_k.\widetilde{z_1}(y) \\
&= \sum_{x\in\{0,1\}^{s}} \widetilde{eq}(\phi_k.r_x, x) \cdot \Big(\sum_{y\in\{0,1\}^{s'}} \widetilde{M_i}(x, y) \cdot \phi_k.\widetilde{z_1}(y) \Big) \qquad \text{By Lemma 2}
\end{aligned}$$

We first perform a random linear combination of these values $\{\phi_k.v_i\}_{k\in[\mu],i\in[t]}$, i.e., for $\gamma \xleftarrow{\$} \mathbb{F}$,

$$\sum_{k=1}^{\mu}\sum_{i=1}^{t} \gamma^{(k-1)t+i} \phi_k.v_i = \sum_{k=1}^{\mu}\sum_{i=1}^{t} \gamma^{(k-1)t+i} \cdot \sum_{x\in\{0,1\}^{s}} L_{k,i}(x), \tag{1}$$

where $L_{k,i}(X) \coloneqq \widetilde{eq}(\phi_k.r_x, X) \cdot \Big(\sum_{y\in\{0,1\}^{s'}} \widetilde{M_i}(X, y) \cdot \phi_k.\widetilde{z_1}(y)\Big)$.

For $k' \in [\nu]$, according to $\mathcal{R}_{\text{CCCS}}$, we have for all $x \in \{0,1\}^s$,

$$\sum_{i=1}^{q} c_i \cdot \prod_{j\in S_i} \Big(\sum_{y\in\{0,1\}^{s'}} \widetilde{M_j}(x, y) \cdot \psi_{k'}.\widetilde{z_2}(y) \Big) = 0.$$

Denoting the left-hand side of the above equation as a polynomial $q_{k'}(X)$, we have for $x \in \{0,1\}^s$, $q_{k'}(x) = 0$. Then the polynomial

$$G_{k'}(X) \coloneqq \sum_{x\in\{0,1\}^{s}} \widetilde{eq}(X, x) \cdot q_{k'}(x)$$

is multilinear and vanishes on all $x \in \{0,1\}^s$. Hence, we have $G_{k'}(X)$ is a zero polynomial. For $\beta \xleftarrow{\$} \mathbb{F}^s$, $G_{k'}(\beta) = 0$. Let $Q_{k'}(X) \coloneqq \widetilde{eq}(\beta, X) \cdot q_{k'}(X)$. We have for $k' \in [\nu]$, $\sum_{x\in\{0,1\}^s} Q_{k'}(x) =$

$G_{k'}(\beta) = 0$. Then based on Equation (1), we further perform a random linear combination that

$$\sum_{k=1}^{\mu}\sum_{i=1}^{t}\gamma^{(k-1)t+i}\phi_k.v_i = \sum_{k=1}^{\mu}\sum_{i=1}^{t}\gamma^{(k-1)t+i}\cdot\sum_{x\in\{0,1\}^s}L_{k,i}(x) + \sum_{k'=1}^{\nu}\gamma^{\mu t+k'}\cdot\sum_{x\in\{0,1\}^s}Q_{k'}(x).$$

Let

$$g(X) \coloneqq \sum_{k=1}^{\mu}\sum_{i=1}^{t}\gamma^{(k-1)t+i}\cdot L_{k,i}(X) + \sum_{k'=1}^{\nu}\gamma^{\mu t+k'}\cdot Q_{k'}(X).$$

We have

$$\sum_{x\in\{0,1\}^s} g(x) = \sum_{k=1}^{\mu}\sum_{i=1}^{t}\gamma^{(k-1)t+i}\phi_k.v_i,$$

which is exactly a statement that a sum-check protocol could prove. Therefore, the task of checking μ $\mathcal{R}_{\text{LCCCS}}$ instances and ν $\mathcal{R}_{\text{CCCS}}$ instances is reduced into the task of performing a sum-check protocol. With $r'_x \xleftarrow{\$} \mathbb{F}^s$, $\mathcal{P}$ and $\mathcal{V}$ run $c \leftarrow \langle\mathcal{P}, \mathcal{V}(r'_x)\rangle\left(g, s, d+1, \sum_{k=1}^{\mu}\sum_{i=1}^{t}\gamma^{(k-1)t+i}\phi_k.v_i\right)$. Now, $\mathcal{V}$ has to check the equation $g(r'_x) = c$. We let $\mathcal{P}$ first compute some intermediate values and then $\mathcal{V}$ compute $g(r'_x)$ using these values. Specifically, $\mathcal{P}$ computes and sends $\{\sigma_{k,i}, \theta_{k',i}\}_{k\in[\mu],k'\in[\nu],i\in[t]}$ to $\mathcal{V}$, where

$$\sigma_{k,i} \coloneqq \sum_{y\in\{0,1\}^{s'}}\widetilde{M}_i(r'_x, y)\cdot\phi_k.\widetilde{z}_1(y), \qquad \theta_{k',i} \coloneqq \sum_{y\in\{0,1\}^{s'}}\widetilde{M}_i(r'_x, y)\cdot\psi_{k'}.\widetilde{z}_2(y). \tag{2}$$

Then $\mathcal{V}$ computes

$$g(r'_x) \coloneqq \sum_{k=1}^{\mu}\sum_{i=1}^{t}\gamma^{(k-1)t+i}\cdot\widetilde{eq}(\phi_k.r_x, r'_x)\cdot\sigma_{k,i} + \sum_{k'=1}^{\nu}\gamma^{\mu t+k'}\cdot\widetilde{eq}(\beta, r'_x)\cdot\left(\sum_{i=1}^{q}c_i\cdot\prod_{j\in S_i}\theta_{k',j}\right)$$

and compares it with c. Nevertheless, $\mathcal{V}$ still has to check the correctness of $\{\sigma_{k,i}, \theta_{k',i}\}_{k\in[\mu],k'\in[\nu],i\in[t]}$. Observe that equations in (2) are exactly LCCCS relations. Thus, we could first perform a random linear combination of $\{\sigma_{k,i}, \theta_{k',i}\}_{k\in[\mu],k'\in[\nu]}$ for $i \in [t]$ and then prove the correctness of the folded $\mathcal{R}_{\text{LCCCS}}$ instance. Now, the task of checking μ $\mathcal{R}_{\text{LCCCS}}$ instances and ν $\mathcal{R}_{\text{CCCS}}$ instances is reduced into the task of checking a single $\mathcal{R}_{\text{LCCCS}}$ instance.

3.2 Formal Description

Construction 2 (A Multi-folding Scheme for Arbitrary Number of Instances). We formally present our multi-folding scheme as follows.

$\underline{\mathsf{fpp} \leftarrow \mathcal{G}\left(1^\lambda\right)}$:

1. Sample size bounds $m, n, N, \ell, t, q, d \in \mathbb{N}$ with $n = 2\cdot(\ell+1)$.
2. Compute $\mathsf{pp}_{\mathsf{pcs}} \leftarrow \mathsf{Setup}(1^\lambda, \log_2 n - 1)$, and output $\mathsf{fpp} \coloneqq (m, n, N, \ell, t, q, d, \mathsf{pp}_{\mathsf{pcs}})$.

$\underline{(\mathsf{fpk}, \mathsf{fvk}) \leftarrow \mathcal{K}(\mathsf{fpp}, (\mathsf{s}_1, \mathsf{s}_2))}$:

1. Parse $\mathsf{s}_1 = \mathsf{s}_2$ as $\left((\widetilde{M}_1, \ldots, \widetilde{M}_t), (S_1, \ldots, S_q), (c_1, \ldots, c_q)\right)$.

2. Output $\mathsf{fpk} \coloneqq (\mathsf{fpp}, \mathsf{s}_1)$, $\mathsf{fvk} \coloneqq (\mathsf{fpp}, \mathsf{s}_1)$.

$\underline{(\mathsf{u}, \mathsf{w}) \leftarrow \langle \mathcal{P}(\mathsf{fpk}, \overrightarrow{\mathsf{w}_1}, \overrightarrow{\mathsf{w}_2}), \mathcal{V}(\mathsf{fvk}) \rangle (\overrightarrow{\mathsf{u}_1}, \overrightarrow{\mathsf{u}_2})}$

1. Parse $(\overrightarrow{\mathsf{u}_1}, \overrightarrow{\mathsf{w}_1})$ as $\{\phi_k\}_{k\in[\mu]}$, $(\overrightarrow{\mathsf{u}_2}, \overrightarrow{\mathsf{w}_2})$ as $\{\psi_{k'}\}_{k'\in[\nu]}$.
2. $\mathcal{V}$ samples $\gamma \xleftarrow{\$} \mathbb{F}, \beta \xleftarrow{\$} \mathbb{F}^s$ and sends them to $\mathcal{P}$. $\mathcal{V}$ then samples $r'_x \xleftarrow{\$} \mathbb{F}^s$.
3. $\mathcal{P}$ and $\mathcal{V}$ run the sum-check protocol

$$c \leftarrow \langle \mathcal{P}, \mathcal{V}(r'_x) \rangle \left(g, s, d+1, \sum_{k=1}^{\mu}\sum_{i=1}^{t} \gamma^{(k-1)t+i}\phi_k.v_i\right).$$

   $\mathcal{V}$ aborts once he outputs "reject" in the sum-check protocol.
4. $\mathcal{P}$ computes and sends $\{\sigma_{k,i}, \theta_{k',i}\}_{k\in[\mu],k'\in[\nu],i\in[t]}$ to $\mathcal{V}$.
5. $\mathcal{V}$ computes $\{e_{k,1} \coloneqq \widetilde{eq}(\phi_k.r_x, r'_x)\}_{k\in[\mu]}, e_2 \coloneqq \widetilde{eq}(\beta, r'_x)$, and aborts if

$$c \neq \sum_{k=1}^{\mu}\sum_{i=1}^{t} \gamma^{(k-1)t+i} \cdot e_{k,1} \cdot \sigma_{k,i} + \sum_{k'=1}^{\nu} \gamma^{\mu t+k'} \cdot e_2 \cdot \left(\sum_{i=1}^{q} c_i \cdot \prod_{j\in S_i} \theta_{k',j}\right).$$

6. $\mathcal{V}$ samples $\rho \xleftarrow{\$} \mathbb{F}$ and sends it to $\mathcal{P}$.
7. $\mathcal{P}$ and $\mathcal{V}$ output the folded $\mathcal{R}_{\text{LCCCS}}$ instance $\mathsf{u} \coloneqq (C', u', \mathsf{x}', r'_x, v'_1, \ldots, v'_t)$, where for all $i \in [t]$,

$$\begin{aligned}
C' &\coloneqq \sum_{k=1}^{\mu} \rho^{k-1} \cdot \phi_k.C_1 + \sum_{k'=1}^{\nu} \rho^{\mu-1+k'} \cdot \psi_{k'}.C_2,\\
u' &\coloneqq \sum_{k=1}^{\mu} \rho^{k-1} \cdot \phi_k.u + \sum_{k'=1}^{\nu} \rho^{\mu-1+k'} \cdot 1,\\
\mathsf{x}' &\coloneqq \sum_{k=1}^{\mu} \rho^{k-1} \cdot \phi_k.\mathsf{x}_1 + \sum_{k'=1}^{\nu} \rho^{\mu-1+k'} \cdot \psi_{k'}.\mathsf{x}_2,\\
v'_i &\coloneqq \sum_{k=1}^{\mu} \rho^{k-1} \cdot \sigma_{k,i} + \sum_{k'=1}^{\nu} \rho^{\mu-1+k'} \cdot \theta_{k',i}.
\end{aligned}$$

8. $\mathcal{P}$ outputs the folded $\mathcal{R}_{\text{LCCCS}}$ witness $\mathsf{w} \coloneqq \widetilde{w'}$, where

$$\widetilde{w'} \coloneqq \sum_{k=1}^{\mu} \rho^{k-1} \cdot \phi_k.\widetilde{w_1} + \sum_{k'=1}^{\nu} \rho^{\mu-1+k'} \cdot \psi_{k'}.\widetilde{w_2}.$$

Theorem 1. *Construction 2 is a public-coin multi-folding scheme for* $(\mathcal{R}_{\text{LCCCS}}, \mathcal{R}_{\text{CCCS}}, \mathsf{compat}, \mu, \nu)$ *with perfect completeness and knowledge soundness.*

Proof. We now describe the proof of Theorem 1. We mainly refer to the proof of the multi-folding scheme of Kothapalli and Setty [23].

Perfect Completeness. For public parameters $\mathsf{fpp} = (m, n, N, \ell, t, q, d, \mathsf{pp}_{\mathsf{pcs}})$ and $s = \log_2 m$, $s' = \log_2 n$, consider arbitrary adversarially chosen structure $\mathsf{s}_1 = \mathsf{s}_2 = \big((\widetilde{M}_1, \ldots, \widetilde{M}_t), (S_1, \ldots, S_q), (c_1, \ldots, c_q)\big)$, μ satisfied $\mathcal{R}_{\mathrm{LCCCS}}$ instance-witness pairs $\{\phi_k\}_{k\in[\mu]}$ and ν satisfied $\mathcal{R}_{\mathrm{CCCS}}$ instance-witness pairs $\{\psi_{k'}\}_{k'\in[\nu]}$. We show that $\mathcal{V}$ will not abort and the folded $\mathcal{R}_{\mathrm{LCCCS}}$ instance $(C', u', \mathsf{x}', r'_x, v'_1, \ldots, v'_t)$ is satisfied by the folded $\mathcal{R}_{\mathrm{LCCCS}}$ witness $\widetilde{w}'$.

Firstly, since $\{\phi_k\}_{k\in[\mu]}$ are satisfied $\mathcal{R}_{\mathrm{LCCCS}}$ instance-witness pairs, we have for $k \in [\mu], i \in [t]$ and $\phi_k.\widetilde{z_1} = (\phi_k.u, \widetilde{\phi_k.\mathsf{x}_1}, \phi_k.w_1)$,

$$\begin{aligned}
\phi_k.v_i &= \sum_{y\in\{0,1\}^{s'}} \widetilde{M}_i(\phi_k.r_x, y)\cdot \phi_k.\widetilde{z_1}(y) && \text{By precondition}\\
&= \sum_{x\in\{0,1\}^{s}} \widetilde{eq}(\phi_k.r_x, x)\cdot \Big(\sum_{y\in\{0,1\}^{s'}} \widetilde{M}_i(x,y)\cdot \phi_k.\widetilde{z_1}(y)\Big) && \text{By Lemma 2}\\
&= \sum_{x\in\{0,1\}^{s}} L_{k,i}(x) && \text{By construction.}
\end{aligned}$$

Furthermore, since $\{\psi_{k'}\}_{k'\in[\nu]}$ are satisfied $\mathcal{R}_{\mathrm{CCCS}}$ instance-witness pairs, we have for $k' \in [\nu]$, $\psi_{k'}.\widetilde{z_2} = (1, \widetilde{\psi_{k'}.\mathsf{x}_2}, \psi_{k'}.w_2)$ and $x \in \{0,1\}^s$,

$$\sum_{i=1}^{q} c_i \cdot \prod_{j\in S_i}\Big(\sum_{y\in\{0,1\}^{s'}} \widetilde{M}_j(x,y)\cdot \psi_{k'}.\widetilde{z_2}(y)\Big) = 0.$$

Denoting the left-hand side of the above equation as a polynomial $q_{k'}(X)$, we have for $x \in \{0,1\}^s$, $q_{k'}(x) = 0$. Then the polynomial $G_{k'}(X) \coloneqq \sum_{x\in\{0,1\}^s} \widetilde{eq}(X, x)\cdot q_{k'}(x)$ is multilinear and vanishes on all $x \in \{0,1\}^s$. Hence, we have $G_{k'}(X)$ is a zero polynomial. For $\beta \xleftarrow{\$} \mathbb{F}^s$, $G_{k'}(\beta) = 0$. By construction, we have for $k' \in [\nu]$, $\sum_{x\in\{0,1\}^s} Q_{k'}(x) = G_{k'}(\beta) = 0$.

Therefore, for $\gamma \xleftarrow{\$} \mathbb{F}$, we have

$$\begin{aligned}
\sum_{k=1}^{\mu}\sum_{i=1}^{t} \gamma^{(k-1)t+i}\phi_k.v_i &= \sum_{k=1}^{\mu}\sum_{i=1}^{t} \gamma^{(k-1)t+i}\Big(\sum_{x\in\{0,1\}^s} L_{k,i}(x)\Big) + \sum_{k'=1}^{\nu}\gamma^{\mu t+k'}\sum_{x\in\{0,1\}^s} Q_{k'}(x)\\
&= \sum_{x\in\{0,1\}^s}\Big(\sum_{k=1}^{\mu}\sum_{i=1}^{t}\gamma^{(k-1)t+i}L_{k,i}(x) + \sum_{k'=1}^{\nu}\gamma^{\mu t+k'}Q_{k'}(x)\Big)\\
&= \sum_{x\in\{0,1\}^s} g(x).
\end{aligned}$$

Thus, by the completeness of the sum-check protocol, $\mathcal{V}$ will not output "reject" inside it. Moreover, we have

$$\begin{aligned}
c = g(r'_x) &= \sum_{k=1}^{\mu}\sum_{i=1}^{t}\gamma^{(k-1)t+i}L_{k,i}(r'_x) + \sum_{k'=1}^{\nu}\gamma^{\mu t+k'}Q_{k'}(r'_x)\\
&= \sum_{k=1}^{\mu}\sum_{i=1}^{t}\gamma^{(k-1)t+i}\cdot e_{k,1}\cdot\sigma_{k,i} + \sum_{k'=1}^{\nu}\gamma^{\mu t+k'}\cdot e_2\cdot\Big(\sum_{i=1}^{q} c_i\cdot\prod_{j\in S_i}\theta_{k',j}\Big).
\end{aligned}$$

We finally have that $\mathcal{V}$ will not abort.

Secondly, by construction, we have that for $k \in [\mu], k' \in [\nu]$,

$$\big(\phi_k.C_1, \phi_k.u, \phi_k.\mathsf{x}_1, r'_x, \sigma_{k,1}, \ldots, \sigma_{k,t}; \phi_k.\widetilde{w_1}\big),$$
$$\big(\psi_{k'}.C_2, 1, \psi_{k'}.\mathsf{x}_2, r'_x, \theta_{k',1}, \ldots, \theta_{k',t}; \psi_{k'}.\widetilde{w_2}\big)$$

are all satisfied $\mathcal{R}_{\text{LCCCS}}$ instance-witness pairs. Therefore, for $\widetilde{z'} = \widetilde{(u', \mathsf{x}', w')}$ and $i \in [t]$, we have that

$$\begin{aligned}
v'_i &= \sum_{k=1}^{\mu} \rho^{k-1} \cdot \sigma_{k,i} + \sum_{k'=1}^{\nu} \rho^{\mu-1+k'} \cdot \theta_{k',i} && \text{By construction} \\
&= \sum_{k=1}^{\mu} \rho^{k-1} \cdot \Big(\sum_{y \in \{0,1\}^{s'}} \widetilde{M}_i(r'_x, y) \cdot \phi_k.\widetilde{z}_1(y) \Big) + \sum_{k'=1}^{\nu} \rho^{\mu-1+k'} \cdot \Big(\sum_{y \in \{0,1\}^{s'}} \widetilde{M}_i(r'_x, y) \cdot \psi_{k'}.\widetilde{z}_2(y) \Big) \\
&= \sum_{y \in \{0,1\}^{s'}} \widetilde{M}_i(r'_x, y) \cdot \Big(\sum_{k=1}^{\mu} \rho^{k-1} \cdot \phi_k.\widetilde{z}_1(y) + \sum_{k'=1}^{\nu} \rho^{\mu-1+k'} \cdot \psi_{k'}.\widetilde{z}_2(y) \Big) \\
&= \sum_{y \in \{0,1\}^{s'}} \widetilde{M}_i(r'_x, y) \cdot \widetilde{z'}(y).
\end{aligned}$$

By the additive homomorphism property of the polynomial commitment scheme, we have that $C' = \mathsf{Com}\big(\mathsf{pp}_{\mathsf{pcs}}, \widetilde{w'}\big)$. Therefore, $(C', u', \mathsf{x}', r'_x, v'_1, \ldots, v'_t)$ is a satisfied $\mathcal{R}_{\text{LCCCS}}$ instance and $\widetilde{w'}$ is the corresponding witness.

We conclude that Construction 2 has perfect completeness.

Knowledge Soundness. Consider a malicious prover $\mathcal{P}^*$ that succeeds with probability ϵ. For public parameters $\mathsf{fpp} = (m, n, N, \ell, t, q, d, \mathsf{pp}_{\mathsf{pcs}})$ and $s = \log_2 m, s' = \log_2 n$. Consider an adversary $\mathcal{A}$ that adaptively picks structures $\mathsf{s}_1 = \mathsf{s}_2 = \big((\widetilde{M_1}, \ldots, \widetilde{M_t}), (S_1, \ldots, S_q), (c_1, \ldots, c_q)\big)$ that satisfy compat, μ $\mathcal{R}_{\text{LCCCS}}$ instances $\{\phi_k.\mathsf{u}_1 \coloneqq \phi_k.(C_1, u, \mathsf{x}_1, r_x, v_1, \ldots, v_t)\}_{k \in [\mu]}$, ν $\mathcal{R}_{\text{CCCS}}$ instances $\{\psi_{k'}.\mathsf{u}_2 \coloneqq \psi_{k'}.(C_2, \mathsf{x}_2)\}_{k' \in [\nu]}$ and some auxiliary state st. We construct an expected polynomial time extractor $\mathcal{E}$ that succeeds with probability $\epsilon - \mathsf{negl}(\lambda)$ in obtaining satisfied witnesses for the original instances as follows.

$\big(\{\phi_k.\widetilde{w_1}\}_{k \in [\mu]}, \{\psi_{k'}.\widetilde{w_2}\}_{k' \in [\nu]}\big) \leftarrow \mathcal{E}\,(\mathsf{fpp}, \rho)$:

1. Invoke $\mathcal{A}$ to obtain the output tuple: $\big((\mathsf{s}_1, \mathsf{s}_2), \{\phi_k.\mathsf{u}_1\}_{k \in [\mu]}, \{\psi_{k'}.\mathsf{u}_2\}_{k' \in [\nu]}, \mathsf{st}\big) \leftarrow \mathcal{A}\,(\mathsf{fpp}, \rho)\,.$
2. Compute $(\mathsf{fpk}, \mathsf{fvk}) \leftarrow \mathcal{K}(\mathsf{fpp}, (\mathsf{s}_1, \mathsf{s}_2))$.
3. Run the interaction $\big(\mathsf{u}^{(1)}, \widetilde{w}^{(1)}\big) \leftarrow \big\langle \mathcal{P}^*(\mathsf{fpk}, \mathsf{st}), \mathcal{V}(\mathsf{fvk}) \big\rangle \big(\{\phi_k.\mathsf{u}_1\}_{k \in [\mu]}, \{\psi_{k'}.\mathsf{u}_2\}_{k' \in [\nu]}\big)$ *once* with the final verifier challenge $\rho^{(1)} \xleftarrow{\$} \mathbb{F}$.
4. Abort if $\big(\mathsf{fpp}, \mathsf{s}_1, \mathsf{u}^{(1)}, \widetilde{w}^{(1)}\big) \notin \mathcal{R}_{\text{LCCCS}}$.
5. Rerun the interaction $\big(\mathsf{u}^{(1)}, \widetilde{w}^{(1)}\big) \leftarrow \big\langle \mathcal{P}^*(\mathsf{fpk}, \mathsf{st}), \mathcal{V}(\mathsf{fvk}) \big\rangle \big(\{\phi_k.\mathsf{u}_1\}_{k \in [\mu]}, \{\psi_{k'}.\mathsf{u}_2\}_{k' \in [\nu]}\big)$ with different verifier's final challenges while maintaining the same prior randomness. Keep doing so until it obtains $\mu + \nu - 1$ folded $\mathcal{R}_{\text{LCCCS}}$ instance-witness pairs $\big\{\big(\mathsf{u}^{(k'')}, \widetilde{w}^{(k'')}\big)\big\}_{k'' \in [2, \ldots, \mu+\nu]}$ such that $\big(\mathsf{fpp}, \mathsf{s}_1, \mathsf{u}^{(k'')}, \widetilde{w}^{(k'')}\big) \in \mathcal{R}_{\text{LCCCS}}$ for all $k'' \in [2, \ldots, \mu + \nu]$. Let $\big\{\rho^{(k'')}\big\}$ denote the corresponding verifier's final challenges.

6. Abort if there exists a collision in the verifier's final challenges.
7. Interpolate points $\left\{\left(\rho^{(k'')}, \widetilde{w}^{(k'')}\right)\right\}_{k''\in[\mu+\nu]}$ to retrieve witnesses $\left(\{\phi_k.\widetilde{w_1}\}_{k\in[\mu]}, \{\psi_{k'}.\widetilde{w_2}\}_{k'\in[\nu]}\right)$ such that for $k'' \in [\mu+\nu]$,

$$\sum_{k=1}^{\mu} \rho^{(k'')k-1} \cdot \phi_k.\widetilde{w_1} + \sum_{k'=1}^{\nu} \rho^{(k'')\mu-1+k'} \cdot \psi_{k'}.\widetilde{w_2} = \widetilde{w}^{(k'')}. \tag{3}$$

8. Output $\left(\{\phi_k.\widetilde{w_1}\}_{k\in[\mu]}, \{\psi_{k'}.\widetilde{w_2}\}_{k'\in[\nu]}\right)$.

We now demonstrate that the extractor $\mathcal{E}$ runs in expected polynomial time and succeeds with probability $\epsilon - \mathsf{negl}(\lambda)$.

Firstly, the extractor $\mathcal{E}$ runs the interaction $\langle \mathcal{P}^*, \mathcal{V}\rangle$ once, and if it does not abort, keeps rerunning the interaction until it obtains $\mu+\nu-1$ satisfied folded $\mathcal{R}_{\text{LCCCS}}$ instance-witness pairs. Thus, the expected number of times $\mathcal{E}$ runs the interaction is

$$\begin{aligned}
&\Pr[\text{First call to } \langle \mathcal{P}^*, \mathcal{V}\rangle \text{ fails}] \cdot 1 + \\
&\Pr[\text{First call to } \langle \mathcal{P}^*, \mathcal{V}\rangle \text{ succeeds}] \cdot \left(1 + \frac{\mu+\nu-1}{\Pr[\langle \mathcal{P}^*, \mathcal{V}\rangle \text{ succeeds}]}\right) \\
=&(1-\epsilon)\cdot 1 + \epsilon \cdot \left(1 + \frac{\mu+\nu-1}{\epsilon}\right) = \mu+\nu.
\end{aligned}$$

Assuming that μ, ν are polynomial in the security parameter, we have that $\mathcal{E}$ runs in expected polynomial time.

Secondly, let E_1 denote the event that $\mathcal{E}$ successfully produces some outputs in less than T times of running the interaction $\langle \mathcal{P}^*, \mathcal{V}\rangle$. Given E_1, let E_2 denote the event that the outputs of $\mathcal{E}$ are satisfied witnesses. We have that

$$\Pr[\mathcal{E} \text{ succeeds}] = \Pr[E_1] \cdot \Pr[E_2].$$

We now analyze $\Pr[E_1]$ and $\Pr[E_2]$. By the success probability of $\mathcal{P}^*$, we have that $\mathcal{E}$ does not abort in step (4) with probability ϵ. Given that $\mathcal{E}$ does not abort in step (4), by Markov's inequality, we have that $\mathcal{E}$ runs the interaction $\langle \mathcal{P}^*, \mathcal{V}\rangle$ more than T times with probability $\frac{\mu+\nu}{T}$. Given that $\mathcal{E}$ runs the interaction less than T times, which has probability $1 - \frac{\mu+\nu}{T}$, we have $\mathcal{E}$ tests at most T values for ρ. Thus, the probability that $\mathcal{E}$ does not abort in step (6) is $1 - \frac{T^2}{|\mathbb{F}|}$. Thus, we have

$$\Pr[E_1] = \left(1 - \frac{\mu+\nu}{T}\right) \cdot \epsilon \cdot \left(1 - \frac{T^2}{|\mathbb{F}|}\right).$$

Setting $T = \sqrt[3]{|\mathbb{F}|}$ and assuming $T \geq \mu+\nu$, we have that $\Pr[E_1] = \epsilon - \mathsf{negl}(\lambda)$.

To analyze $\Pr[E_2]$, we first show that the retrieved witnesses are valid openings to the corresponding polynomial commitments in the instance, and then show that they satisfy the remaining algebraic relations with some probability. For $k'' \in [\mu+\nu]$, let $\mathsf{u}^{(k'')} = \left(C^{(k'')}, u^{(k'')}, \mathsf{x}^{(k'')}, r'_x, v_1^{(k'')}, \ldots, v_t^{(k'')}\right)$.

Since $\widetilde{w}^{(k'')}$ is a satisfied $\mathcal{R}_{\text{LCCCS}}$ witness, we have that

$$\begin{aligned}
&\sum_{k=1}^{\mu} \rho^{(k'')k-1} \cdot \mathsf{Com}\big(\mathsf{pp}_{\mathsf{pcs}}, \phi_k.\widetilde{w_1}\big) + \sum_{k'=1}^{\nu} \rho^{(k'')\mu-1+k'} \cdot \mathsf{Com}\big(\mathsf{pp}_{\mathsf{pcs}}, \psi_{k'}.\widetilde{w_2}\big) \\
=&\mathsf{Com}\Big(\mathsf{pp}_{\mathsf{pcs}}, \sum_{k=1}^{\mu} \rho^{(k'')k-1} \cdot \phi_k.\widetilde{w_1} + \sum_{k'=1}^{\nu} \rho^{(k'')\mu-1+k'} \cdot \psi_{k'}.\widetilde{w_2}\Big) \\
& \qquad\qquad \text{By additive homomorphism} \\
=&\mathsf{Com}\big(\mathsf{pp}_{\mathsf{pcs}}, \widetilde{w}^{(k'')}\big) \qquad\qquad \text{By Equation (3)} \\
=&C^{(k'')} \qquad\qquad \text{Witness } \widetilde{w}^{(k'')} \text{ is a satisfied opening} \\
=&\sum_{k=1}^{\mu} \rho^{(k'')k-1} \cdot \phi_k.C_1 + \sum_{k'=1}^{\nu} \rho^{(k'')\mu-1+k'} \cdot \psi_{k'}.C_2 \text{ By the verifier's computation}
\end{aligned}$$

Treat the above equation as a univariate polynomial equation in $\rho^{(k'')}$. Since it holds for all $k'' \in [\mu+\nu]$, we must have that for $k \in [\mu], k' \in [\nu]$,

$$\phi_k.C_1 = \mathsf{Com}\big(\mathsf{pp}_{\mathsf{pcs}}, \phi_k.\widetilde{w_1}\big), \qquad \psi_{k'}.C_2 = \mathsf{Com}\big(\mathsf{pp}_{\mathsf{pcs}}, \psi_{k'}.\widetilde{w_2}\big),$$

which means that $\big(\{\phi_k.\widetilde{w_1}\}_{k\in[\mu]}, \{\psi_{k'}.\widetilde{w_2}\}_{k'\in[\nu]}\big)$ are valid openings.

Next, by the extractor's construction we have that $\{\sigma_{k,i}, \theta_{k',i}\}_{k\in[\mu],k'\in[\nu],i\in[t]}$ sent by the prover are identical across all executions of the interaction $\langle \mathcal{P}^*, \mathcal{V}\rangle$. By the verifier's computation, we have that for $k'' \in [\mu+\nu], i \in [t]$,

$$v_i^{(k'')} = \sum_{k=1}^{\mu} \rho^{(k'')k-1} \cdot \sigma_{k,i} + \sum_{k'=1}^{\nu} \rho^{(k'')\mu-1+k'} \cdot \theta_{k',i}. \tag{4}$$

Since $\big\{\widetilde{w}^{(k'')}\big\}_{k''\in[\mu+\nu]}$ are satisfied $\mathcal{R}_{\text{LCCCS}}$ witnesses, we have that for $k'' \in [\mu+\nu], i \in [t]$,

$$v_i^{(k'')} = \sum_{y\in\{0,1\}^{s'}} \widetilde{M}_i(r'_x, y) \cdot \widetilde{z}^{(k'')}(y), \tag{5}$$

where $\widetilde{z}^{(k'')} = \big(u^{(k'')}, \widetilde{\mathsf{x}^{(k'')}}, \widetilde{w}^{(k'')}\big)$. By Equations (3), (4), (5) and the verifier's computation, we have that for $k'' \in [\mu+\nu], i \in [t]$,

$$\begin{aligned}
&\sum_{k=1}^{\mu} \rho^{(k'')k-1} \cdot \sigma_{k,i} + \sum_{k'=1}^{\nu} \rho^{(k'')\mu-1+k'} \cdot \theta_{k',i} \\
=& \sum_{y\in\{0,1\}^{s'}} \widetilde{M}_i(r'_x, y) \cdot \widetilde{z}^{(k'')}(y) \\
=& \sum_{y\in\{0,1\}^{s'}} \widetilde{M}_i(r'_x, y) \cdot \Big(\sum_{k=1}^{\mu} \rho^{(k'')k-1} \cdot \phi_k.\widetilde{z_1}(y) + \sum_{k'=1}^{\nu} \rho^{(k'')\mu-1+k'} \cdot \psi_{k'}.\widetilde{z_2}(y)\Big) \\
=& \sum_{k=1}^{\mu} \rho^{(k'')k-1} \cdot \sum_{y\in\{0,1\}^{s'}} \widetilde{M}_i(r'_x, y) \cdot \phi_k.\widetilde{z_1}(y) + \sum_{k'=1}^{\nu} \rho^{(k'')\mu-1+k'} \cdot \sum_{y\in\{0,1\}^{s'}} \widetilde{M}_i(r'_x, y) \cdot \psi_{k'}.\widetilde{z_2}(y),
\end{aligned}$$

where $\phi_k.\widetilde{z_1} = (\phi_k.u, \widetilde{\phi_k.\mathsf{x}_1}, \phi_k.w_1)$ and $\psi_{k'}.\widetilde{z_2} = (1, \widetilde{\psi_{k'}.\mathsf{x}_2}, \psi_{k'}.w_2)$. Treat the above equation as a univariate polynomial equation in $\rho^{(k'')}$. Since it holds for all $k'' \in [\mu+\nu]$, we must have that for $k \in [\mu], k' \in [\nu], i \in [t]$,

$$\sigma_{k,i} = \sum_{y \in \{0,1\}^{s'}} \widetilde{M_i}(r'_x, y) \cdot \phi_k.\widetilde{z_1}(y), \qquad \theta_{k',i} = \sum_{y \in \{0,1\}^{s'}} \widetilde{M_i}(r'_x, y) \cdot \psi_{k'}.\widetilde{z_2}(y).$$

Then since the verifier does not abort, we have that

$$\begin{aligned}
c &= \sum_{k=1}^{\mu}\sum_{i=1}^{t} \gamma^{(k-1)t+i} \cdot e_{k,1} \cdot \sigma_{k,i} + \sum_{k'=1}^{\nu} \gamma^{\mu t+k'} \cdot e_2 \cdot \Big(\sum_{i=1}^{q} c_i \cdot \prod_{j \in S_i} \theta_{k',j}\Big) \\
&= \sum_{k=1}^{\mu}\sum_{i=1}^{t} \gamma^{(k-1)t+i} \cdot \widetilde{eq}(\phi_k.r_x, r'_x) \cdot \sigma_{k,i} + \sum_{k'=1}^{\nu} \gamma^{\mu t+k'} \cdot \widetilde{eq}(\beta, r'_x) \cdot \Big(\sum_{i=1}^{q} c_i \cdot \prod_{j \in S_i} \theta_{k',j}\Big) \\
&= \sum_{k=1}^{\mu}\sum_{i=1}^{t} \gamma^{(k-1)t+i} \cdot L_{k,i}(r'_x) + \sum_{k'=1}^{\nu} \gamma^{\mu t+k'} \cdot Q_{k'}(r'_x) \\
&= g(r'_x),
\end{aligned}$$

which by the soundness of the sum-check protocol, implies that with probability $1 - O(s \cdot d)/|\mathbb{F}| = 1 - \mathsf{negl}(\lambda)$ over the choice of r'_x,

$$\begin{aligned}
\sum_{k=1}^{\mu}\sum_{i=1}^{t} \gamma^{(k-1)t+i} \phi_k.v_i &= \sum_{x \in \{0,1\}^s} g(x) \\
&= \sum_{x \in \{0,1\}^s} \Big(\sum_{k=1}^{\mu}\sum_{i=1}^{t} \gamma^{(k-1)t+i} \cdot L_{k,i}(x) + \sum_{k'=1}^{\nu} \gamma^{\mu t+k'} \cdot Q_{k'}(x)\Big) \\
&= \sum_{k=1}^{\mu}\sum_{i=1}^{t} \gamma^{(k-1)t+i} \cdot \sum_{x \in \{0,1\}^s} L_{k,i}(x) + \sum_{k'=1}^{\nu} \gamma^{\mu t+k'} \cdot \sum_{x \in \{0,1\}^s} Q_{k'}(x).
\end{aligned}$$

By the Schwartz-Zippel lemma [28], this implies that with probability $1 - O(\mu \cdot t + \nu)/|\mathbb{F}| = 1 - \mathsf{negl}(\lambda)$ over the choice of γ, we have for $k \in [\mu], i \in [t]$,

$$\phi_k.v_i = \sum_{x \in \{0,1\}^s} L_{k,i}(x), \tag{6}$$

and for $k' \in [\nu]$,

$$0 = \sum_{x \in \{0,1\}^s} Q_{k'}(x). \tag{7}$$

Expanding Equation (6), we have that

$$\begin{aligned}
\phi_k.v_i &= \sum_{x \in \{0,1\}^s} L_{k,i}(x) \\
&= \sum_{x \in \{0,1\}^s} \widetilde{eq}(\phi_k.r_x, x) \cdot \Big(\sum_{y \in \{0,1\}^{s'}} \widetilde{M_i}(x, y) \cdot \phi_k.\widetilde{z_1}(y)\Big) \\
&= \sum_{y \in \{0,1\}^{s'}} \widetilde{M_i}(\phi_k.r_x, y) \cdot \phi_k.\widetilde{z_1}(y). && \text{By Lemma 2}
\end{aligned}$$

Since we have argued that $\{\phi_k.\widetilde{w_1}\}_{k\in[\mu]}$ are valid openings, we have that they are satisfied $\mathcal{R}_{\text{LCCCS}}$ witnesses with probability $1-\mathsf{negl}(\lambda)$.

Expanding Equation (7), we have that

$$\begin{aligned}
0 &= \sum_{x\in\{0,1\}^s} Q_{k'}(x) \\
&= \sum_{x\in\{0,1\}^s} \widetilde{eq}(\beta,x)\cdot\Big(\sum_{i=1}^{q} c_i\cdot\prod_{j\in S_i}\Big(\sum_{y\in\{0,1\}^{s'}}\widetilde{M_j}(x,y)\cdot\psi_{k'}.\widetilde{z_2}(y)\Big)\Big) \\
&= \sum_{i=1}^{q} c_i\cdot\prod_{j\in S_i}\Big(\sum_{y\in\{0,1\}^{s'}}\widetilde{M_j}(\beta,y)\cdot\psi_{k'}.\widetilde{z_2}(y)\Big).
\end{aligned}$$

By the Schwartz-Zippel lemma, this implies that with probability $1-O(s\cdot d)/|\mathbb{F}| = 1-\mathsf{negl}(\lambda)$ over the choice of β, we have that for all $x\in\{0,1\}^s$,

$$\sum_{i=1}^{q} c_i\cdot\prod_{j\in S_i}\Big(\sum_{y\in\{0,1\}^{s'}}\widetilde{M_j}(x,y)\cdot\psi_{k'}.\widetilde{z_2}(y)\Big) = 0.$$

Since we have argued that $\{\psi_{k'}.\widetilde{w_2}\}_{k'\in[\nu]}$ are valid openings, we have that they are satisfied $\mathcal{R}_{\text{CCCS}}$ witnesses with probability $1-\mathsf{negl}(\lambda)$. Hence, we have that $\Pr[E_2] = 1-\mathsf{negl}(\lambda)$.

Therefore, we have that

$$\Pr[\mathcal{E}\text{ succeeds}] = \Pr[E_1]\cdot\Pr[E_2] = \big(\epsilon-\mathsf{negl}(\lambda)\big)\cdot\big(1-\mathsf{negl}(\lambda)\big) = \epsilon-\mathsf{negl}(\lambda).$$

We conclude that Construction 2 has knowledge soundness. □

A Non-interactive Multi-folding Scheme. Since our multi-folding scheme is public-coin, we could transform it into a non-interactive multi-folding scheme $\text{NIMFS} = (\mathcal{G}', \mathcal{K}', \mathcal{P}', \mathcal{V}')$ for the tuple $(\mathcal{R}_{\text{LCCCS}}, \mathcal{R}_{\text{CCCS}}, \mathsf{compat}, \mu, \nu)$ using the Fiat-Shamir transformation, according to the Lemma 3 of Kothapalli and Setty [23].

Efficiency. Fot the prover's cost, the sum-check protocol requires the prover to generate some proof, which could be completed with $O\big(\mu(N+tm)+\nu(N+tm+qmd\log^2 d)\big)$ field operations according to [29]. For $\{\sigma_{k,i},\theta_{k',i}\}_{k\in[\mu],k'\in[\nu],i\in[t]}$, we have that

$$\begin{aligned}
\sigma_{k,i} &\coloneqq \sum_{y\in\{0,1\}^{s'}}\widetilde{M_i}(r'_x,y)\cdot\phi_k.\widetilde{z_1}(y) \\
&= \sum_{y\in\{0,1\}^{s'}}\Big(\sum_{a\in\{0,1\}^s}\sum_{b\in\{0,1\}^{s'}} M_i(a,b)\cdot\widetilde{eq}\big((r'_x,y),(a,b)\big)\Big)\cdot\phi_k.\widetilde{z_1}(y) \quad \text{By multilinear extension} \\
&= \sum_{y\in\{0,1\}^{s'}}\Big(\sum_{a\in\{0,1\}^s}\widetilde{eq}(r'_x,a)\sum_{b\in\{0,1\}^{s'}} M_i(a,b)\cdot\widetilde{eq}(y,b)\Big)\cdot\phi_k.\widetilde{z_1}(y) \\
&= \sum_{a\in\{0,1\}^s}\widetilde{eq}(r'_x,a)\cdot\sum_{b\in\{0,1\}^{s'}} M_i(a,b)\cdot\phi_k.\widetilde{z_1}(b) \quad \text{By the definition of } \widetilde{eq}
\end{aligned}$$

According to [33], $\{\widetilde{eq}(r'_x, a)\}_{a\in\{0,1\}^s}$ could be computed with $O(m)$ field operations. Based on these values, the computation of $\{\sigma_{k,i}, \theta_{k',i}\}_{k\in[\mu],k'\in[\nu],i\in[t]}$ could be accomplished with $O\big(m+(\mu+\nu)\cdot N\big)$ field operations. When instantiating the polynomial commitment scheme with Bulletproofs [6], the computation of C' and $\widetilde{w}'$ requires one MSM of size $\mu+\nu$ and $O\big((\mu+\nu)\cdot n\big)$ field operations, respectively. Thus, the total cost is dominated by $O\big(\mu(N+tm+n)+\nu(N+tm+n+qmd\log^2 d)\big)$ field operations and one MSM of size $\mu+\nu$.

For the verifier's cost, verification in the sum-check protocol requires $O(d\log m)$ field operations [30]. The computation of $\{e_{k,1} \coloneqq \widetilde{eq}(\phi_k.r_x, r'_x)\}_{k\in[\mu]}, e_2 \coloneqq \widetilde{eq}(\beta, r'_x)$ and checking c require $O(\mu\log m+\mu t+\nu dq)$ field operations. The computation of C' requires one MSM of size $\mu+\nu$. Thus, the total cost is dominated by $O(d\log m+\mu\log m+\mu t+\nu dq)$ field operations and one MSM of size $\mu+\nu$.

The communication cost consists of elements sent from $\mathcal{P}$ to $\mathcal{V}$ in the sum-check protocol and $\{\sigma_{k,i}, \theta_{k',i}\}_{k\in[\mu],k'\in[\nu],i\in[t]}$, which sums up to $O(d\log m)+t\cdot(\mu+\nu)$ field elements.

4 PCD from Non-interactive Multi-folding Schemes

Recall that PCD enables a set of parties to carry out an indefinitely long distributed computation where every step along the way is accompanied by a proof of correctness. We rely on our non-interactive multi-folding scheme to construct PCD scheme where the computation and the circuit expressing verifier at each step are together expressed as CCCS, a NP-complete constraint system.

4.1 Overview

At each step, the prover P receives r previous outputs $\{z_i\}_{i\in[r]}$ each with a proof Π_i that consists of a satisfied $\mathcal{R}_{\text{LCCCS}}$ instance-witness pair $(\mathsf{U}_i, \mathsf{W}_i)$ and a satisfied $\mathcal{R}_{\text{CCCS}}$ instance-witness pair $(\mathsf{u}_i, \mathsf{w}_i)$ in our case. In addition, P has some local input z_{loc}. With $z_{\mathsf{loc}}, z_1, \ldots, z_r$, P computes z such that $\varphi(z, z_{\mathsf{loc}}, z_1, \ldots, z_r)$ accepts. Then he should provide a proof to the next party that attests not only to the correctness of his local computation, but also to the correctness of all his inputs $\{z_i\}_{i\in[r]}$. To this end, P first invokes the non-interactive multi-folding scheme to fold $\{(\mathsf{U}_i, \mathsf{W}_i), (\mathsf{u}_i, \mathsf{w}_i)\}_{i\in[r]}$ into a single $\mathcal{R}_{\text{LCCCS}}$ instance-witness pair (U, W). Define a circuit R_φ that represents the compliance predicate φ as well as the verifier of the non-interactive multi-folding scheme, which could be expressed as a $\mathcal{R}_{\text{CCCS}}$ structure. Then P generates a satisfied $\mathcal{R}_{\text{CCCS}}$ instance-witness pair (u, w) by computing R_φ on input appropriate values from his prior computations.

Let the new PCD proof Π consist of $(\mathsf{U}, \mathsf{W}), (\mathsf{u}, \mathsf{w})$. Now, by checking (u, w), we have that $\varphi(z, z_{\mathsf{loc}}, z_1, \ldots, z_r)$ accepts and $\{\mathsf{U}_i, \mathsf{u}_i\}_{i\in[r]}$ is corrected folded into U. Further by checking (U, W) and the knowledge soundness of the multi-folding scheme, we have that there exists satisfied witnesses $\{\mathsf{W}_i, \mathsf{w}_i\}_{i\in[r]}$ for instances $\{\mathsf{U}_i, \mathsf{u}_i\}_{i\in[r]}$, which attests to the correctness of $\{z_i\}_{i\in[r]}$. Therefore, we maintain the invariant that if (U, W) is a satisfied $\mathcal{R}_{\text{LCCCS}}$ instance-witness pair and (u, w) is a satisfied $\mathcal{R}_{\text{CCCS}}$ instance-witness pair, then the distributed computation is correct up to this step.

However, the above description elides some subtlety. Since the proof Π passed to the next party includes U, the public IO of the circuit R_φ, i.e., $\mathsf{u.x}$ should include U. The next party will fold u, U into a new $\mathcal{R}_{\text{LCCCS}}$ instance as described above. But now U is part of u, they can not be folded. To address this issue, inspired by Nova [24], we modify R_φ to additionally hash z, U and let the output be the circuit's public IO, which ensures that $\mathsf{u.x}$ does not contain U but still binds with it. To achieve recursion, we further modify R_φ to include the verifier's work of checking that the inputs satisfy the hash relation, i.e., $\mathsf{u}_i.\mathsf{x}$ is the hash of z_i, U_i for $i \in [r]$.

Note that since the PCD proof includes the entire witnesses W, w, its size is linear in the size of the circuit. However, as in HyperNova [23], we could use a general SNARK such as SuperSpartan [29] to prove the knowledge of a valid PCD proof, which could reduce the proof size exponentially.

4.2 Formal Description

Construction 3 (A PCD Scheme from Non-interactive Multi-folding Schemes). We formally present our PCD scheme as follows. Let $(\mathsf{u}_\perp, \mathsf{w}_\perp)$ be a default trivially satisfied $\mathcal{R}_{\text{LCCCS}}$ instance-witness pair. Given a compliance predicate φ, the circuit R_φ that realizes the recursion is as follows.

$\underline{0/1 \leftarrow R_\varphi\big(h; (z, z_{\mathsf{loc}}, \{z_i, \mathsf{U}_i, \mathsf{u}_i\}_{i=1}^r, \mathsf{fvk}', \mathsf{U}, \pi)\big)}$:

1. Check that the compliance predicate $\varphi(z, z_{\mathsf{loc}}, z_1, \ldots, z_r)$ accepts.
2. If $z_i = \perp$ for all $i \in [r]$, then check that $h = \mathsf{hash}(\mathsf{fvk}', z, \mathsf{u}_\perp)$.

   Else, check that
   (a) for $i \in [r]$, $\mathsf{u}_i.\mathsf{x} = \mathsf{hash}(\mathsf{fvk}', z_i, \mathsf{U}_i)$, where $\mathsf{u}_i.\mathsf{x}$ is the public IO of u_i.
   (b) $\mathsf{U} = \text{NIMFS}.\mathcal{V}'\big(\mathsf{fvk}', \{\mathsf{U}_i\}_{i\in[r]}, \{\mathsf{u}_i\}_{i\in[r]}, \pi\big)$.
   (c) $h = \mathsf{hash}(\mathsf{fvk}', z, \mathsf{U})$.
3. If the above checks hold, output 1; otherwise, output 0.

Since R_φ can be computed in polynomial time, it can be represented as a $\mathcal{R}_{\text{CCCS}}$ structure. Let

$$(\mathsf{u}, \mathsf{w}) \leftarrow \mathsf{trace}\Big(R_\varphi, \big(h, (z, z_{\mathsf{loc}}, \{z_i, \mathsf{U}_i, \mathsf{u}_i\}_{i=1}^r, \mathsf{fvk}', \mathsf{U}, \pi)\big)\Big)$$

denote the satisfied $\mathcal{R}_{\text{CCCS}}$ instance-witness pair for the execution of R_φ on input $\big(h, (z, z_{\mathsf{loc}}, \{z_i, \mathsf{U}_i, \mathsf{u}_i\}_{i=1}^r, \mathsf{fvk}', \mathsf{U}, \pi)\big)$.

We define the PCD scheme $(\mathsf{G}, \mathsf{K}, \mathsf{P}, \mathsf{V})$ as follows.

$\underline{\mathsf{pp} \leftarrow \mathsf{G}\left(1^\lambda\right)}$:

1. Compute $\mathsf{fpp}' \leftarrow \text{NIMFS}.\mathcal{G}'\left(1^\lambda\right)$, and output $\mathsf{pp} \coloneqq \mathsf{fpp}'$.

$\underline{(\mathsf{pk}, \mathsf{vk}) \leftarrow \mathsf{K}(\mathsf{pp}, \varphi)}$:

1. Compute $(\mathsf{fpk}', \mathsf{fvk}') \leftarrow \text{NIMFS}.\mathcal{K}'(\mathsf{fpp}', R_\varphi)$, and output $(\mathsf{pk}, \mathsf{vk}) \coloneqq \big((\mathsf{fpk}', \mathsf{fvk}'), \mathsf{fvk}'\big)$.

$\underline{\mathit{\Pi} \leftarrow \mathsf{P}\big(\mathsf{pk}, z, z_{\mathsf{loc}}, \{z_i, \mathit{\Pi}_i\}_{i=1}^r\big)}$:

1. For $i \in [r]$, parse $\mathit{\Pi}_i$ as $\big((\mathsf{U}_i, \mathsf{W}_i), (\mathsf{u}_i, \mathsf{w}_i)\big)$, where $(\mathsf{U}_i, \mathsf{W}_i)$ is a satisfied $\mathcal{R}_{\text{LCCCS}}$ instance-witness pair and $(\mathsf{u}_i, \mathsf{w}_i)$ is a satisfied $\mathcal{R}_{\text{CCCS}}$ instance-witness pair.
2. If $z_i = \perp$ for all $i \in [r]$, then set $(\mathsf{U}, \mathsf{W}, \pi) \coloneqq (\mathsf{u}_\perp, \mathsf{w}_\perp, \perp)$.

   Else, compute $(\mathsf{U}, \mathsf{W}, \pi) \leftarrow \text{NIMFS}.\mathcal{P}'\big(\mathsf{fpk}', \{(\mathsf{U}_i, \mathsf{W}_i)\}_{i\in[r]}, \{(\mathsf{u}_i, \mathsf{w}_i)\}_{i\in[r]}\big)$
3. Compute $h \leftarrow \mathsf{hash}(\mathsf{fvk}', z, \mathsf{U})$.

4. $(\mathsf{u}, \mathsf{w}) \leftarrow \mathsf{trace}\Big(R_\varphi, \big(h, (z, z_{\mathsf{loc}}, \{z_i, \mathsf{U}_i, \mathsf{u}_i\}_{i=1}^r, \mathsf{fvk}', \mathsf{U}, \pi)\big)\Big)$.
5. Output $\mathit{\Pi} \coloneqq \big((\mathsf{U}, \mathsf{W}), (\mathsf{u}, \mathsf{w})\big)$.

$0/1 \leftarrow \mathsf{V}\big(\mathsf{vk}, z, \mathit{\Pi}\big)$:

1. Parse $\mathit{\Pi}$ as $\big((\mathsf{U}, \mathsf{W}), (\mathsf{u}, \mathsf{w})\big)$.
2. Check that $\mathsf{u.x} = \mathsf{hash}(\mathsf{fvk}', z, \mathsf{U})$.
3. Check that W is a satisfied $\mathcal{R}_{\mathrm{LCCCS}}$ witness to U and w is a satisfied $\mathcal{R}_{\mathrm{CCCS}}$ witness to u.
4. If the above checks hold, output 1; otherwise, output 0.

Theorem 2. *Construction 3 is a PCD scheme with perfect completeness and knowledge soundness.*

Proof. We now describe the proof of Theorem 2.

Perfect Completeness. For public parameter pp, consider arbitrary adversarially chosen $\big(\varphi, z, z_{\mathsf{loc}}, \{z_i, \mathit{\Pi}_i\}_{i=1}^r\big)$ such that the perfect completeness precondition is satisfied. We show that given $\mathit{\Pi} \leftarrow \mathsf{P}\big(\mathsf{pk}, z, z_{\mathsf{loc}}, \{z_i, \mathit{\Pi}_i\}_{i=1}^r\big)$, $\mathsf{V}\big(\mathsf{vk}, z, \mathit{\Pi}\big) = 1$ with probability 1.

If $z_i = \perp$ for all $i \in [r]$, by the construction of P, we have that $(\mathsf{U}, \mathsf{W}) = (\mathsf{u}_\perp, \mathsf{w}_\perp)$ is a trivially satisfied $\mathcal{R}_{\mathrm{LCCCS}}$ instance-witness pair, and $h = \mathsf{hash}(\mathsf{fvk}', z, \mathsf{U})$. By the perfect completeness precondition, we have that $\varphi(z, z_{\mathsf{loc}}, z_1, \ldots, z_r) = 1$. Therefore, P could construct a satisfied $\mathcal{R}_{\mathrm{CCCS}}$ instance-witness pair (u, w) that represents the correct computation of R_φ. Moreover, by the construction of R_φ, we have $\mathsf{u.x} = \mathsf{hash}(\mathsf{fvk}', z, \mathsf{U})$. Therefore, $\mathsf{V}\big(\mathsf{vk}, z, \mathit{\Pi}\big) = 1$ with probability 1.

If $\exists i \in [r]$ such that $z_i \neq \perp$, by the perfect completeness precondition, $\{(\mathsf{U}_i, \mathsf{W}_i)\}_{i\in[r]}$ are satisfied $\mathcal{R}_{\mathrm{LCCCS}}$ instance-witness pairs, $\{(\mathsf{u}_i, \mathsf{w}_i)\}_{i\in[r]}$ are satisfied $\mathcal{R}_{\mathrm{CCCS}}$ instance-witness pairs, and $\mathsf{u}_i\mathsf{.x} = \mathsf{hash}(\mathsf{fvk}', z_i, \mathsf{U}_i)$. Then by the perfect completeness of the multi-folding scheme, we have that (U, W) is a satisfied $\mathcal{R}_{\mathrm{LCCCS}}$ intance-witness pair. Therefore, P could construct a satisfied $\mathcal{R}_{\mathrm{CCCS}}$ instance-witness pair (u, w) that represents the correct computation of R_φ. Additionally, by the construction of R_φ, we have that $\mathsf{u.x} = \mathsf{hash}(\mathsf{fvk}', z, \mathsf{U})$. Therefore, $\mathsf{V}\big(\mathsf{vk}, z, \mathit{\Pi}\big) = 1$ with probability 1.

In conclusion, we conclude that Construction 3 has perfect completeness.

Knowledge Soundness. We mainly refer to the proof of the PCD scheme of Bünz et al [7]. Fix a set Z, and for $\mathsf{pp} \leftarrow \mathsf{G}\left(1^\lambda\right), \mathsf{ai} \leftarrow \mathcal{D}(\mathsf{pp})$, consider an expected polynomial time adversary P^* that succeeds with probability ϵ. We construct an expected polynomial time extractor $\mathcal{E}_{\mathsf{P}^*}$ that with input $(\mathsf{pp}, \mathsf{ai})$, outputs $(\varphi, \mathsf{T}, \mathsf{ao})$ such that $\varphi \in \mathsf{F}, (\mathsf{pp}, \mathsf{ai}, \varphi, \mathsf{o}(\mathsf{T}), \mathsf{ao}) \in Z$ and T is φ-compliant with probability $\epsilon - \mathrm{negl}(\lambda)$.

Referring to existing works [8,7], we assume that every node has a unique outgoing edge. Thus, the extracted transcript T will be a tree. Let every node v be labeled with a local data $z_{\mathsf{loc}}^{(v)}$, the label $z^{(v)}$ of its unique outgoing edge and a proof $\mathit{\Pi}^{(v)}$ that proves the correctness of $z^{(v)}$. We first present the construction of $\mathcal{E}_{\mathsf{P}^*}$ that extracts the labels of all the nodes in T, and then explain its correctness.

We construct $\mathcal{E}_{\mathsf{P}^*}$ via an iterative process that constructs a sequence of extractors $\mathcal{E}_0, \mathcal{E}_1, \ldots, \mathcal{E}_d$, where d is the depth of T and for $i \in \{0, 1, \ldots, d\}$, $\mathcal{E}_i$ outputs a tree T_i of depth $i+1$. In particular, the nodes of T_d at depth $d+1$ are all empty nodes. We argue later that the extractor $\mathcal{E}_d$ is then equal to $\mathcal{E}_{\mathsf{P}^*}$.

In the base case, we define $\mathcal{E}_0(\mathsf{pp}, \mathsf{ai})$ to compute $(\varphi, \mathsf{o}, \mathit{\Pi}, \mathsf{ao}) \leftarrow \mathsf{P}^*(\mathsf{pp}, \mathsf{ai})$ and output $(\varphi, \mathsf{T}_0, \mathsf{ao})$, where T_0 is a single node labeled with $(\mathsf{o}, \mathit{\Pi})$.

Next, we construct the extractor $\mathcal{E}_i$ inductively for each $i \in [d]$, given that we have already constructed $\mathcal{E}_{i-1}$. We denote $S_{\mathsf{T}}(i)$ as the set of nodes of T at depth i. We first construct an adversary $\mathcal{P}^*_{i-1}$ for the non-interactive multi-folding scheme using $\mathcal{E}_{i-1}$, which implies an extractor $\mathcal{E}_{\mathcal{P}^*_{i-1}}$ corresponding to $\mathcal{P}^*_{i-1}$ by the knowledge soundness of the multi-folding scheme, and then construct $\mathcal{E}_i$ using $\mathcal{P}^*_{i-1}, \mathcal{E}_{\mathcal{P}^*_{i-1}}$.

$\underline{\mathcal{P}^*_{i-1}(\mathsf{pp}, \mathsf{ai}, \rho)}$:

1. Compute $(\varphi, \mathsf{T}_{i-1}, \mathsf{ao}) \leftarrow \mathcal{E}_{i-1}(\mathsf{pp}, \mathsf{ai})$. If T_{i-1} is not a tree of depth i, abort.
2. For each node $v \in S_{\mathsf{T}_{i-1}}(i)$, denote its label as $(z^{(v)}, \mathit{\Pi}^{(v)})$.
3. Parse $\mathit{\Pi}^{(v)}$ as $\big((\mathsf{U}^{(v)}, \mathsf{W}^{(v)}), (\mathsf{u}^{(v)}, \mathsf{w}^{(v)})\big)$.
4. Obtain $\big\{\mathsf{U}_j^{(v)}, \mathsf{u}_j^{(v)}, z_j^{(v)}\big\}_{j\in[r]}, \pi^{(v)}$ from $\mathsf{w}^{(v)}$.
5. Let $S_{i-1} \coloneqq \big\{v \in S_{\mathsf{T}_{i-1}}(i) \mid \exists j \in [r], z_j^{(v)} \neq \perp\big\}$.
6. Output $\Big(\Big\{\big\{\mathsf{U}_j^{(v)}, \mathsf{u}_j^{(v)}\big\}_{j\in[r]}, \mathsf{U}^{(v)}, \mathsf{W}^{(v)}, \pi^{(v)}\Big\}_{v\in S_{i-1}}, (\varphi, \mathsf{T}_{i-1}, \mathsf{ao})\Big)$.

By the knowledge soundness of the multi-folding scheme, there exists an extractor $\mathcal{E}_{\mathcal{P}^*_{i-1}}$ that for $v \in S_{i-1}$, outputs $\big\{\mathsf{W}_j^{(v)}, \mathsf{w}_j^{(v)}\big\}_{j\in[r]}$ such that $\big\{\big(\mathsf{U}_j^{(v)}, \mathsf{W}_j^{(v)}\big)\big\}_{j\in[r]}$ are satisfied $\mathcal{R}_{\mathrm{LCCCS}}$ instance-witness pairs and $\big\{\big(\mathsf{u}_j^{(v)}, \mathsf{w}_j^{(v)}\big)\big\}_{j\in[r]}$ are satisfied $\mathcal{R}_{\mathrm{CCCS}}$ instance-witness pairs.

Given $\mathcal{P}^*_{i-1}, \mathcal{E}_{\mathcal{P}^*_{i-1}}$, we construct $\mathcal{E}_i$ as follows.

$\underline{(\varphi, \mathsf{T}_i, \mathsf{ao}) \leftarrow \mathcal{E}_i(\mathsf{pp}, \mathsf{ai})}$:

1. Compute $\Big(\big\{\big(\mathsf{U}_j^{(v)}, \mathsf{W}_j^{(v)}\big), \big(\mathsf{u}_j^{(v)}, \mathsf{w}_j^{(v)}\big)\big\}_{j\in[r], v\in S_{i-1}}, (\varphi, \mathsf{T}_{i-1}, \mathsf{ao})\Big) \leftarrow \mathcal{E}_{\mathcal{P}^*_{i-1}}(\mathsf{pp}, \mathsf{ai}, \rho)$. If T_{i-1} is not a tree of depth i, abort.
2. Retrieve $\{\mathsf{w}^{(v)}\}_{v\in S_{\mathsf{T}_{i-1}}(i)}$ from the internal state of $\mathcal{P}^*_{i-1}$, and obtain $z_{\mathsf{loc}}^{(v)}, \big\{z_j^{(v)}\big\}_{j\in[r]}$ from $\mathsf{w}^{(v)}$.
3. Append $z_{\mathsf{loc}}^{(v)}$ to the label of $v \in S_{\mathsf{T}_{i-1}}(i)$.
4. For each node $v \in S_{i-1}$, let $S_v \coloneqq \big\{j \in [r] \mid z_j^{(v)} \neq \perp\big\}$. Construct T_i of depth $i+1$ from T_{i-1} by adding, for each node $v \in S_{i-1}$, $\big(z_j^{(v)}, \mathit{\Pi}_j^{(v)}\big)$ to the label of its child $j \in S_v$, where $\mathit{\Pi}_j^{(v)} = \Big(\big(\mathsf{U}_j^{(v)}, \mathsf{W}_j^{(v)}\big), \big(\mathsf{u}_j^{(v)}, \mathsf{w}_j^{(v)}\big)\Big)$.
5. Output $(\varphi, \mathsf{T}_i, \mathsf{ao})$.

We now show inductively that the extractors are correct. We define the *inductive hypothesis* as that for $i \in \{0, 1, \ldots, d\}$, $\mathcal{E}_i(\mathsf{pp}, \mathsf{ai})$ outputs $(\varphi, \mathsf{T}_i, \mathsf{ao})$ in expected polynomial time such that with probability $\epsilon - \mathrm{negl}(\lambda)$, 1) $\varphi \in \mathsf{F}, (\mathsf{pp}, \mathsf{ai}, \varphi, \mathsf{o}(\mathsf{T}_i), \mathsf{ao}) \in Z$, 2) T_i is φ-compliant up to depth i, and 3) for all $v \in S_{\mathsf{T}_i}(i+1)$, $\mathsf{V}\big(\mathsf{vk}, z^{(v)}, \mathit{\Pi}^{(v)}\big) = 1$.

In the base case, by the premise of P^*, $\mathcal{E}_0$ satisfies the inductive hypothesis.

Next, supposing that $\mathcal{E}_{i-1}$ satisfies the inductive hypothesis, we show that $\mathcal{E}_i$ also satisfies the inductive hypothesis. By the premise of $\mathcal{E}_{i-1}$, we have that with probability $\epsilon - \mathrm{negl}(\lambda)$,

$\varphi \in \mathsf{F}, (\mathsf{pp}, \mathsf{ai}, \varphi, \mathsf{o}(\mathsf{T}_{i-1}), \mathsf{ao}) \in Z$, T_{i-1} is φ-compliant up to depth $i-1$, and for all $v \in S_{\mathsf{T}_{i-1}}(i)$, $\mathsf{V}\big(\mathsf{vk}, z^{(v)}, \mathit{\Pi}^{(v)}\big) = 1$. By the check of V, we have that (1) $\big\{(\mathsf{U}^{(v)}, \mathsf{W}^{(v)}), (\mathsf{u}^{(v)}, \mathsf{w}^{(v)})\big\}_{v \in S_{\mathsf{T}_{i-1}}(i)}$ are satisfied instance-witness pairs. Additionally by the construction of R_φ and the collision-resistant property of the hash function, we have that (2) for $v \in S_{\mathsf{T}_{i-1}}(i)$, $\varphi\big(z^{(v)}, z^{(v)}_{\mathsf{loc}}, z^{(v)}_1, \ldots, z^{(v)}_r\big)$ accepts, (3) for $v \in S_{i-1}$, $\mathsf{U}^{(v)} = \text{NIMFS.}\mathcal{V}'\big(\mathsf{fvk}', \{\mathsf{U}^{(v)}_j\}_{j\in[r]}, \{\mathsf{u}^{(v)}_j\}_{j\in[r]}, \pi^{(v)}\big)$, and (4) for $v \in S_{i-1}, j \in [r]$, $\mathsf{u}^{(v)}_j.\mathsf{x} = \mathsf{hash}\big(\mathsf{fvk}', z^{(v)}_j, \mathsf{U}^{(v)}_j\big)$. Condition (2) implies that T_i is φ-compliant up to depth i, and $\varphi \in \mathsf{F}, (\mathsf{pp}, \mathsf{ai}, \varphi, \mathsf{o}(\mathsf{T}_i), \mathsf{ao}) \in Z$ with probability $\epsilon - \text{negl}(\lambda)$. Conditions (1)(3) imply that $\mathcal{P}^*_{i-1}$ succeeds in producing satisfied folded instance-witness pairs $\big\{\mathsf{U}^{(v)}, \mathsf{W}^{(v)}\big\}_{v\in S_{i-1}}$ for instances $\big\{\mathsf{U}^{(v)}_j, \mathsf{u}^{(v)}_j\big\}_{j\in[r], v\in S_{i-1}}$ with probability $\epsilon - \text{negl}(\lambda)$. Then by the knowledge soundness of the multi-folding scheme, we have that $\mathcal{E}_{\mathcal{P}^*_{i-1}}$ succeeds in outputing satisfied witnesses $\big\{\mathsf{W}^{(v)}_j, \mathsf{w}^{(v)}_j\big\}_{j\in[r], v\in S_{i-1}}$ with probability $\epsilon - \text{negl}(\lambda)$. Additionally by Condition (4), we have that $\mathsf{V}\big(\mathsf{vk}, z^{(v)}, \mathit{\Pi}^{(v)}\big) = 1$ for all $v \in S_{\mathsf{T}_i}(i+1)$ with probability $\epsilon - \text{negl}(\lambda)$. Since $\mathcal{E}_{i-1}$ runs in expected polynomial time, $\mathcal{E}_{\mathcal{P}^*_{i-1}}$ also runs in expected polynomial time, and thereby so does $\mathcal{E}_i$. Therefore, we have that $\mathcal{E}_i$ satisfies the inductive hypothesis.

In conclusion, we conclude that Construction 3 has knowledge soundness. □

Efficiency. The recursion overhead, i.e., verifier's computations expressed as circuits is dominated by the computations in R_φ except for checking the compliance predicate φ, containing $r+1$ calls to hash and one call to NIMFS.$\mathcal{V}'$. The cost of NIMFS.$\mathcal{V}'$ is dominated by $O\big(d\log m + r\cdot(\log m + t + dq)\big)$ field operations, one MSM of size $2r$ and $2\log_2 m + 2$ calls to the random oracle RO to achieve non-interactivity, where RO could be instantiated with an appropriate cryptographic hash function.

The prover's work at each step is dominated by invoking NIMFS.$\mathcal{P}'$, and computing the satisfied $\mathcal{R}_{\text{CCCS}}$ instance-witness pair (u, w) for the execution of R_φ. The cost of NIMFS.$\mathcal{P}'$ is dominated by $O\big(r\cdot(N + tm + n + qmd\log^2 d)\big)$ field operations, one MSM of size $2r$, and $2\log_2 m + 2$ calls to RO to achieve non-interactivity. The cost of computing (u, w) is dominated by computing the commitment C which requires one MSM of size $O(n)$ when instantiating the polynomial commitment scheme with Bulletproofs [6].

The proof $\mathit{\Pi}$ consists of $(\mathsf{U}, \mathsf{W}, \mathsf{u}, \mathsf{w})$ whose size is linear in the size of R_φ. However, as in HyperNova [23], we could use a general SNARK to compress the proof. Specifically, the prover invokes $(\mathsf{U}', \mathsf{W}', \pi') \leftarrow \text{NIMFS.}\mathcal{P}'\big(\mathsf{fpk}', (\mathsf{U}, \mathsf{W}), (\mathsf{u}, \mathsf{w})\big)$, and then uses a general SNARK to generate a proof $\pi_{\mathsf{U}'}$ that proves the knowledge of W'. Now $\mathit{\Pi}$ consists of $(\mathsf{U}, \mathsf{u}, \pi', \pi_{\mathsf{U}'})$. When instantiating the SNARK with SuperSpartan [29] excluding the first sum-check invocation and using the polynomial commitment scheme based on Bulletproofs [6], the proof size is then dominated by $O(d\log m + t + \log n)$ field elements and $O(\log n)$ group elements.

The verifier's work is dominated by checking $(\mathsf{U}, \mathsf{W}), (\mathsf{u}, \mathsf{w})$, whose cost is linear in the size of R_φ. However, by compressing the proof, the work is now dominated by performing $\mathsf{U}' \leftarrow \text{NIMFS.}\mathcal{V}'\big(\mathsf{fvk}', \mathsf{U}, \mathsf{u}, \pi'\big)$ and verifying $\pi_{\mathsf{U}'}$, whose cost is dominated by $O(d\log m + t + dq + \log n)$ field operations, $O(\log m + \log n)$ calls to RO, and one MSM of size $O(n)$.

5 Conclusion

In this paper, we first construct a multi-folding scheme for arbitrary number of instances, which could reduce the task of checking multiple instances into the task of checking one. Based on this

scheme, we construct a PCD scheme having the smallest prover's cost at each step and recursion overhead in the literature. Supposing that there are r incoming edges of a node at certain step and n variables in the constraint system, the prover's cost is dominated by one MSM of size $O(n)$, and the recursion overhead is dominated by one MSM of size $2r$. Besides, our PCD scheme supports more expressive constraint system, i.e., CCCS that allows gates to compute high-degree polynomials.

Recently, Eagen and Gabizon [17] proposed a folding scheme for multiple instances—ProtoGalaxy with novel efficiency characteristics. How to construct PCD schemes based on ProtoGalaxy and analyse the concrete efficiency is an interesting future study.

References

1. Ben-Sasson, E., Chiesa, A., Tromer, E., Virza, M.: Scalable zero knowledge via cycles of elliptic curves. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO. LNCS, vol. 8617, pp. 276–294. Springer (2014)
2. Bitansky, N., Canetti, R., Chiesa, A., Tromer, E.: Recursive composition and bootstrapping for SNARKS and proof-carrying data. In: Boneh, D., Roughgarden, T., Feigenbaum, J. (eds.) Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013. pp. 111–120. ACM (2013)
3. Boneh, D., Drake, J., Fisch, B., Gabizon, A.: Halo infinite: Proof-carrying data from additive polynomial commitments. In: Malkin, T., Peikert, C. (eds.) CRYPTO. LNCS, vol. 12825, pp. 649–680. Springer (2021)
4. Bonneau, J., Meckler, I., Rao, V., Shapiro, E.: Coda: Decentralized cryptocurrency at scale. Cryptology ePrint Archive, Paper 2020/352 (2020), `https://eprint.iacr.org/2020/352`
5. Bowe, S., Grigg, J., Hopwood, D.: Recursive proof composition without a trusted setup. Cryptology ePrint Archive, Paper 2019/1021 (2019), `https://eprint.iacr.org/2019/1021`
6. Bünz, B., Bootle, J., Boneh, D., Poelstra, A., Wuille, P., Maxwell, G.: Bulletproofs: Short proofs for confidential transactions and more. In: S&P. pp. 315–334. IEEE Computer Society (2018)
7. Bünz, B., Chiesa, A., Lin, W., Mishra, P., Spooner, N.: Proof-carrying data without succinct arguments. In: Malkin, T., Peikert, C. (eds.) CRYPTO. LNCS, vol. 12825, pp. 681–710. Springer (2021)
8. Bünz, B., Chiesa, A., Mishra, P., Spooner, N.: Recursive proof composition from accumulation schemes. In: Pass, R., Pietrzak, K. (eds.) TCC. LNCS, vol. 12551, pp. 1–18. Springer (2020)
9. Bünz, B., Fisch, B., Szepieniec, A.: Transparent snarks from DARK compilers. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT. LNCS, vol. 12105, pp. 677–706. Springer (2020)
10. Bünz, B., Chen, B.: Protostar: Generic efficient accumulation/folding for special sound protocols. Cryptology ePrint Archive, Paper 2023/620 (2023), `https://eprint.iacr.org/2023/620`
11. Chen, W., Chiesa, A., Dauterman, E., Ward, N.P.: Reducing participation costs via incremental verification for ledger systems. Cryptology ePrint Archive, Paper 2020/1522 (2020), `https://eprint.iacr.org/2020/1522`
12. Chiesa, A., Ojha, D., Spooner, N.: Fractal: Post-quantum and transparent recursive proofs from holography. In: Canteaut, A., Ishai, Y. (eds.) EUROCRYPT. LNCS, vol. 12105, pp. 769–793. Springer (2020)
13. Chiesa, A., Tromer, E.: Proof-carrying data and hearsay arguments from signature cards. In: Yao, A.C. (ed.) ICS. pp. 310–331. Tsinghua University Press (2010)
14. Chiesa, A., Tromer, E., Virza, M.: Cluster computing in zero knowledge. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT. LNCS, vol. 9057, pp. 371–403. Springer (2015)
15. Chong, S., Tromer, E., Vaughan, J.A.: Enforcing language semantics using proof-carrying data. Cryptology ePrint Archive, Paper 2013/513 (2013), `https://eprint.iacr.org/2013/513`
16. Cormode, G., Mitzenmacher, M., Thaler, J.: Practical verified computation with streaming interactive proofs. In: Goldwasser, S. (ed.) ITCS. pp. 90–112. ACM (2012)
17. Eagen, L., Gabizon, A.: Protogalaxy: Efficient protostar-style folding of multiple instances. Cryptology ePrint Archive, Paper 2023/1106 (2023), `https://eprint.iacr.org/2023/1106`
18. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: Odlyzko, A.M. (ed.) CRYPTO. LNCS, vol. 263, pp. 186–194. Springer (1986)
19. Gennaro, R., Gentry, C., Parno, B., Raykova, M.: Quadratic span programs and succinct nizks without pcps. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT. LNCS, vol. 7881, pp. 626–645. Springer (2013)
20. Kattis, A., Bonneau, J.: Proof of necessary work: Succinct state verification with fairness guarantees. Cryptology ePrint Archive, Paper 2020/190 (2020), `https://eprint.iacr.org/2020/190`
21. Kothapalli, A., Setty, S.: Supernova: Proving universal machine executions without universal circuits. Cryptology ePrint Archive, Paper 2022/1758 (2022), `https://eprint.iacr.org/2022/1758`

22. Kothapalli, A., Setty, S.: Cyclefold: Folding-scheme-based recursive arguments over a cycle of elliptic curves. Cryptology ePrint Archive, Paper 2023/1192 (2023), https://eprint.iacr.org/2023/1192
23. Kothapalli, A., Setty, S.: Hypernova: Recursive arguments for customizable constraint systems. Cryptology ePrint Archive, Paper 2023/573 (2023), https://eprint.iacr.org/2023/573
24. Kothapalli, A., Setty, S.T.V., Tzialla, I.: Nova: Recursive zero-knowledge arguments from folding schemes. In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO. LNCS, vol. 13510, pp. 359–388. Springer (2022)
25. Lund, C., Fortnow, L., Karloff, H.J., Nisan, N.: Algebraic methods for interactive proof systems. In: FOCS. pp. 2–10. IEEE Computer Society (1990)
26. Mohnblatt, N.: Sangria: a folding scheme for plonk (2023), https://github.com/geometryresearch/technical_notes/blob/main/sangria_folding_plonk.pdf
27. Naveh, A., Tromer, E.: Photoproof: Cryptographic image authentication for any set of permissible transformations. In: S&P. pp. 255–271. IEEE Computer Society (2016)
28. Schwartz, J.T.: Fast probabilistic algorithms for verification of polynomial identities. Journal of the ACM **27**(4), 701–717 (1980)
29. Setty, S., Thaler, J., Wahby, R.: Customizable constraint systems for succinct arguments. Cryptology ePrint Archive, Paper 2023/552 (2023), https://eprint.iacr.org/2023/552
30. Thaler, J.: Proofs, arguments, and zero-knowledge. Foundations and Trends in Privacy and Security **4**(2-4), 117–660 (2022)
31. Tyagi, N., Fisch, B., Zitek, A., Bonneau, J., Tessaro, S.: Versa: Verifiable registries with efficient client audits from RSA authenticated dictionaries. In: Yin, H., Stavrou, A., Cremers, C., Shi, E. (eds.) CCS. pp. 2793–2807. ACM (2022)
32. Valiant, P.: Incrementally verifiable computation or proofs of knowledge imply time/space efficiency. In: Canetti, R. (ed.) TCC. LNCS, vol. 4948, pp. 1–18. Springer (2008)
33. Vu, V., Setty, S.T.V., Blumberg, A.J., Walfish, M.: A hybrid architecture for interactive verifiable computation. In: S&P. pp. 223–237. IEEE Computer Society (2013)