Compressed Sigma Protocols: New Model and Aggregation Techniques

Yuxi Xue[1], Tianyu Zheng[1] ⋆, Shang Gao[1], Bin Xiao[1], and Man Ho Au[1]

The Hong Kong Polytechnic University, Hong Kong
{yuxi-ivy.xue, tian-yu.zheng}@connect.polyu.hk
{shang-jason.gao, b.xiao,mhaau}@polyu.edu.hk

Abstract. Sigma protocols (Σ-protocols) provide a foundational paradigm for constructing secure algorithms in privacy-preserving applications. To enhance efficiency, several extended models [BG18], [BBB+18], [AC20] incorporating various optimization techniques have been proposed as "replacements" for the original Σ-protocol. However, these models often lack the expressiveness needed to handle complex relations and hinder designers from applying appropriate instantiation and optimization strategies. In this paper, we introduce a novel compressed Σ-protocol model that effectively addresses these limitations by providing concrete constructions for relations involving non-linear constraints. Our approach is sufficiently expressive to encompass a wide range of relations. Central to our model is the definition of *doubly folded commitments*, which, along with a proposed Argument of Knowledge, generalizes the compression and amortization processes found in previous models. Despite the ability to express more relations, this innovation also provides a foundation to discuss a general aggregation technique, optimizing the proof size of instantiated schemes. To demonstrate the above statements, we provide a brief review of several existing protocols that can be instantiated using our model to demonstrate the versatility of our construction. We also present use cases where our generalized model enhances applications traditionally considered "less compact", such as binary proofs [BCC+15] and k-out-of-n proofs [ACF21]. In conclusion, our new model offers a more efficient and expressive alternative to the current use of Σ-protocols, paving the way for broader applicability and optimization in cryptographic applications.

Keywords: Σ-Protocols · Zero-Knowledge Proofs · Bulletproofs

1 Introduction

Zero-knowledge proofs are fundamental cryptographic primitives that find applications in various fields, including privacy preservation, secret sharing, secure multi-party computation, and other fields [18, 22]. Among these, Sigma protocols (Σ-protocols) [12] are particularly notable for their simplicity and versatility.

⋆ Tianyu has the same contribution to the work as with Yuxi, who should be regarded as co-first authors.

They offer a straightforward, plug-and-play reference model that allows users to design secure zero-knowledge proof schemes, especially for statements defined over algebraic structures such as prime-order and RSA-type groups [24].

However, directly applying Σ-protocols to real-world scenarios often results in inefficient protocols. While it may not be difficult to design a Σ-protocol for some specific applications, their real-world performance can fall short of expectations, particularly when handling complex relations. A major cause is that the traditional Σ-protocol model only allows for independent instantiation for each witness to be proved if no extension on the security assumptions is given. For instance, when proving knowledge of multiple secrets, directly instantiating the Σ-protocol with Pedersen commitments for each secret generates a proof size that scales linearly on the number of secrets. Although several optimization techniques exist, such as batch verification and proof compressions, properly deploying them requires a sufficient familiarity with Σ-protocols and raises additional effort for practitioners. Concretely, when considering real-world implementations, designers of Σ-protocols may encounter challenges in formalizing multiple concrete witness representations, allocating commitment keys for each witness, and checking constraints that involve multiple openings. Improper handling of these tasks will lead to significant performance issues, including:

- **Redundant proof:** The prover may include unnecessary elements that could be aggregated, incurring extra communication costs.
- **Lack of batch verification:** The verifier may check multiple constraints individually, rather than efficiently verify them in a batch.

To address these limitations, researchers have proposed new models capturing more optimizations and taking them as replacements for the original Σ protocol, like [4,5,8,9]. A seminal work among them is Bulletproofs [10], which introduces an inner-product-argument-based compression model with logarithmic proof size. However, Bulletproofs require the constraints to be first reformulated into inner-product forms, necessitating the redesign of existing schemes and limiting their applicability. Building upon Bulletproofs [10], several extended models have been developed. Attema et al. [2, 3] propose a compressed Σ-protocol that achieves logarithmic proof size. Bünz et al. [11] generalize the inner-product argument in Bulletproofs for group elements as well as a model. While these approaches propose effective proof optimizations, they still lack the expressiveness required to handle complex relations and omit certain optimization strategies, such as the aggregation technique discussed in this paper. As a result, designers still need to navigate numerous low-level details without the benefit of standardized methodologies, hindering the construction of efficient Σ protocols.

1.1 Our contributions

In this work, we propose a new model as a "drop-in replacement" for Σ-protocol in a wider range of applications. This new model offers better expressiveness than other models and supports the deployment of optimizations previously not covered. Concretely, we develop our new model in three steps as below.

(1) Generalizing the folding operations. To provide a basic protocol for further discussions, we first review and explore the similarity between compression and amortization processes in current compressed Σ-protocol theory [2]. This analysis leads us to introduce the doubly folded commitment in the form of $\mathsf{Com}(g(X), m(X))$, which generalizes the message and the commitment key parts into polynomial forms to allow potential random linear combinations used in the two processes, i.e., evaluating $g(X), m(X)$ on a random point $x \in \mathbb{F}$. Furthermore, we propose an Argument of Knowledge (AoK) protocol that converts multiple commitments to our new doubly folded commitments. This protocol captures the folding operations in the compression and amortization processes in previous models [2, 10].

(2) Extending for non-linear constraints. Building upon the above protocol, we add extra non-linear constraints on witnesses formulated in polynomial terms. Specifically, we consider a constraint function that maps multiple polynomials to a single polynomial, which enables us to extend the doubly folded commitment with more complex inputs as $\mathsf{Com}(g(X); f(\boldsymbol{m}(X)))$. By integrating this with the AoK protocol from step 1, we obtain a new protocol that reduces multiple commitments to a vector of doubly folded commitments $\boldsymbol{m}(X)$ under $g(X)$, while showing that the commitment vector satisfies a specified constraint in $f(\cdot)$.

(3) Adopting aggregation techniques. The protocol derived in steps 1 & 2 provides a sufficient foundation for formalizing the aggregation techniques that were difficult to capture by previous models. In this part, we focus on all commitments generated during the AoK protocol (auxiliary commitments in Section 3.3). We demonstrate that auxiliary commitments of the same monomial but having different commitment keys can be aggregated. The security of this aggregation is formally proven by reducing it to the binding property of the underlying commitment scheme. This technique aggregates multiple commitments into a more compact form, reducing proof sizes and verification costs.

The above 3 steps yield a new model for the compressed Σ-protocol with desirable features. To illustrate the expressiveness, we show how to capture existing amortization and compression processes with our model in Section 4.1, the comparison is given in Table 1. To demonstrate the practicality, we apply it to optimize the performance of several commonly used proof schemes in Section 4.2, including binary proofs and their application to ring signatures [16] and k-out-of-n proofs [3]. According to the results presented Section 4, our model improves the concrete performance of different applications. For example, our model can aggregate 2 group elements in the binary proofs, reducing the ring signature size in [7] by 4 group elements. As for the k-out-of-n proofs given in [3], our model can halve the proof size from $4\log(n)$ to $2\log(n)$ without the requirement of pairing-friendly curves.

In conclusion, our model is more expressive than the original Σ-protocol model and other "replacement" schemes. It is more friendly for designers to implement and optimize their proof schemes. Moreover, our model finds a wider range of applications in the real world, especially for constructing lightweight

zero-knowledge proofs in private blockchain systems and authentication protocols [1, 10, 25].

Table 1: Existing "replacement" Σ-protocols with logarithmic size and comparison of their relations and applications. The PCS and SNARK stand for Polynomial Commitment Schemes and Succinct Non-interactive Argument of Knowledge respectively.

"Replacement" models	**Expressiveness**	**Applications**
Bulletproofs [10]	Medium (IPA)	Range Proofs
Compressed Σ-protocols [2]	Low (linear)	Range Proofs
Compressed Σ-protocols [3]	High (homomorphisms)	Partial knowledge proofs
Inner Pairing Products [11]	High (GIPA)	PCS/SNARK aggregation
This work	Highest (all of above)	All of above

1.2 Related Work

There has been a series of research dedicated to building an efficient zero-knowledge proof system from Σ-protocol. Groth [15] proposes a public-coin zero-knowledge proof with linear computation complexity for both the prover and verifier. In his work, Groth designs specific relations for committed matrices to achieve sub-linear communication complexity. Bootle et al. [8] present an efficient zero-knowledge proof for arithmetic circuit satisfiability with linear computation cost and logarithmic communication complexity. Their approach allows the protocol to be recursively executed, resulting in proofs of size $O(\log N)$. Bulletproofs [10] further improves the performance of the inner-product argument (IPA) in [8] by reducing the number of commitments. Bulletproofs reformulate the relations for range proofs and arithmetic circuit satisfiability into inner-product forms, thereby achieving logarithmic-sized proofs. When dealing with multiple proofs of the same relation, Bulletproofs also supports the aggregation technique to reduce the proof size. To apply the above optimization of proof size to more general cases, Attema and Cramer [2] provide a compression mechanism for Σ-protocols of general linear relations. They reduce the communication complexity from linear to logarithmic by reconciling Bulletproofs [10] with Σ-protocol. Although they construct relatively compact zero-knowledge proof protocols, their mechanism can only be applied to linear relations. In 2021, Attema et al. [3] extend their protocol for proving homomorphic relations, and introduce its applications in partial knowledge proofs. To provide a more efficient verification for pairing-based applications, Bünz et al. [11] propose a generalized inner product argument (GIPA) that leads to a logarithmic time verifier. They formalize a doubly homomorphic commitment property to unify the generalization of inner product arguments. Another research area related to our work is reducing the proof size with some aggregation techniques (or batch verification) [6, 19]. Bayer and Groth [4] introduce a batching technique for verifying multiple polynomials simultaneously, reducing communication costs compared to the parallel repetition of individual polynomial evaluations. Inspired by Bayer's work, Bootle and Groth [9] propose a batch argument for polynomial relations. Their approach embeds multiple statements into a single polynomial using Lagrange

interpolation, allowing proof and verification of many instances of the same relation within a single argument. Tomescu et al. [21] formalize an aggregatable subvector commitment scheme that supports both aggregating and updating proofs. Their scheme aggregates multiple proofs into a single subvector proof with constant-sized polynomial commitments. Gorbunov et al. [14] construct an aggregatable proof scheme to commit to a vector with small commitments and proofs, allowing the aggregation of multiple proofs across different commitments into one short proof to reduce storage requirements.

2 Preliminary

2.1 Notations

Let $\lambda \in \mathbb{N}$ be the security parameter. Based on λ, generate a cyclic group of prime order q as $\mathbb{G}$ and a ring of integers modulo q as $\mathbb{Z}_q$. We use $y \leftarrow\!\!\$\ S$ to denote the process of uniformly sampling y from a finite set S. Typically, the lowercase letters, e.g., a denote the field elements in $\mathbb{Z}_q$ and the uppercase letters, e.g., A denote the group elements in $\mathbb{G}$. For simplicity, we take all $\mathbb{G}$'s as *additive* groups, which means that addition and scalar multiplication are used for computation. To represent a set, we use $\{a_i\}_{i=1}^n$ as a short-hand for $\{a_1, ..., a_n\}$.

Vector Notation We denote vectors in $\mathbb{Z}_q$ as $\boldsymbol{a} = (a_1, ..., a_n)$ and vectors in $\mathbb{G}$ as $\boldsymbol{A} = (A_1, ..., A_n)$. Sometimes the notations of vectors and sets are used interchangeably like $\boldsymbol{a} = \{a_i\}_{i=1}^n$ for simplicity. $|\boldsymbol{a}|$ denotes the length of vector $\boldsymbol{a}$. For any two vectors $\boldsymbol{a}$ and $\boldsymbol{b}$, their inner product and Hadamard product are denoted as $\langle \boldsymbol{a}, \boldsymbol{b} \rangle$ and $\boldsymbol{a} \circ \boldsymbol{b}$ respectively. The subvector $\boldsymbol{a}[i{:}j]$ contains all the i-th to the j-th elements in $\boldsymbol{a}$.

Polynomial Notation We use $\mathbb{F}^{(<d)}[X]$ for the set of polynomials in $\mathbb{F}[X]$ of degree at most d and the same applies to $\mathbb{G}$. We will take the superscript for degree $(< d)$ implicitly if the context is clear. To denote a polynomial vector, we write $\boldsymbol{m}(X) = \{m_i(X)\}_{i=1}^u$, where for $i \in \{1, \ldots, u\}, m_i(X) \in \mathbb{F}^{(<d)}[X]$.

2.2 Commitment Schemes

A commitment scheme is a pair of polynomial time algorithms $(\mathsf{Gen}, \mathsf{Com})$ [16].

- $\mathsf{ck} \leftarrow \mathsf{Gen}(1^\lambda)$: on input security parameter λ, outputs a commitment key ck, where ck specifies a message space $\mathcal{M}_c$, a key space $\mathcal{K}_c$, a commitment space $\mathcal{C}_c$, a randomness space $\mathcal{R}_c$ and an opener space $\mathcal{O}_c$.
- $c_m \leftarrow \mathsf{Com}(\mathsf{ck}; m)$: on input commitment key $\mathsf{ck} \in \mathcal{K}_c$, a message $m \in \mathcal{M}_c$, a randomness $r \in \mathcal{R}_c$, outputs a commitment c_m. Parameter r can be omitted when the sampling is implicit.

Typically, a commitment scheme should satisfy binding and hiding properties defined in Appendix A.1.

2.3 Doubly Homomorphic Property.

Based on the definitions in Section 2.2, we define another important property that needs to be satisfied by the commitment schemes in this paper.

Definition 1. (Doubly Homomorphic Commitment Scheme [11]) *A commitment scheme* $(\mathsf{KeyGen}, \mathsf{Com}, \mathsf{Open})$ *is* doubly homomorphic *if the abelian groups constructed by the images of* Com *satisfies the following two properties for all* $\mathsf{ck}, \mathsf{ck}' \in \mathcal{K}$ *and* $m, m' \in \mathcal{M}$*:*

$$(1)\ \mathsf{Com}(\mathsf{ck}; m) + \mathsf{Com}(\mathsf{ck}; m') = \mathsf{Com}(\mathsf{ck}; m + m'),$$
$$(2)\ \mathsf{Com}(\mathsf{ck}; m) + \mathsf{Com}(\mathsf{ck}'; m) = \mathsf{Com}(\mathsf{ck} + \mathsf{ck}'; m).$$

Note that the message space and key space are *both* additively homomorphic. To distinguish, we denote this type of commitment scheme as Com. This paper mainly focuses on two concrete schemes. The first is the plain commitment scheme based on the Discrete Logarithm (DL) assumption, which computes $\mathsf{Com}(\mathsf{ck}; m) = m \cdot G$ to achieve binding, where G is a generator of a cyclic group from $\mathcal{K}$. The second is the Pedersen commitment scheme [20], which computes $\mathsf{Com}(\mathsf{ck}; m) = m \cdot G + r \cdot H$ with additional random $r \in \mathcal{R}$ and generator H to achieve hiding property. According to the previous work [11], we claim that both of them satisfy the doubly homomorphic property in Definition 1. In addition to the above two schemes, our techniques can include other existing solutions, as long as they satisfy the doubly homomorphic properties. We derive a straightforward corollary frequently used in our following content from the above property.

Lemma 1. *For any message* $m \in \mathcal{M}$ *committed under the key* $\mathsf{ck} \in \mathcal{K}$ *and a scalar* $x \in \mathbb{Z}_q$*, the doubly homomorphic commitment satisfies [11]*

$$(1)\ \mathsf{Com}(x \cdot \mathsf{ck}; m) = x \cdot \mathsf{Com}(\mathsf{ck}; m), \quad (2)\ \mathsf{Com}(\mathsf{ck}; x \cdot m) = x \cdot \mathsf{Com}(\mathsf{ck}; m).$$

Besides, we mention that if the message is a vector such as $\boldsymbol{m} \in \mathbb{F}_q^n$ (assume message $\boldsymbol{m}$ contains one randomiser), the commitment $\mathsf{Com}(\mathsf{ck}; \boldsymbol{m})$ is also expected to satisfy the above properties. For example, a Pedersen vector commitment computes $\mathsf{Com}(\mathsf{ck}; \boldsymbol{m}) = \sum_{i=1}^{n} m_i \cdot G_i$ is computationally binding, perfectly hiding, and doubly homomorphic according to [10]. We also want to clarify that we *only* borrow the definition of doubly homomorphic commitments in [11]. Thus, the security of our commitment scheme only relies on the concrete chosen implementation. Therefore, we do not need to particularly define the q-ASDBP and q-SDH assumptions used in their paper.

2.4 Sigma Protocol

Let $\mathcal{R} = \{(\mathsf{ck}, x, w)\}$ be an NP relation where ck is the common reference string, x is the statement and w is the witness. Sometimes we omit the ck for simplicity when it has already been explicitly given. A Σ-protocol Π_Σ is a type of interactive three-move protocol between a prover P and a verifier V consisting of a tuple of PPT algorithms $\Pi_\Sigma = (\mathsf{Setup}, \mathsf{P}, \mathsf{V})$ [16]:

- $\mathsf{ck} \leftarrow \mathsf{Setup}(1^\lambda)$: on input the security parameter λ, outputs the common reference string ck.
- $t \leftarrow \mathsf{P}(\mathsf{ck}, x, w)$: on input common reference string ck, statement x, witness w, outputs an initial message t.
- $c \leftarrow \mathcal{C}_s$: sample a public coin challenge c uniformly at random from the challenge space $\mathcal{C}_s$.
- $z \leftarrow \mathsf{P}(c)$: on input the challenge c, responds with z.
- $b \leftarrow \mathsf{V}(\mathsf{ck}, x, t, c, z) = 1$: on input common reference string ck, statement x, prover's message t, z, challenge c, the verifier outputs a bit to determine whether the proof should be accepted or rejected as 0 or 1.

A Σ protocol should provide completeness, special-soundness and honest verifier zero-knowledge defined in Appendix A.2. For convenience, we assume the Sigma protocol is instantiated based on Pedersen commitments and prove theorems for security under the DL assumption.

3 Compact Sigma Protocol

In this section, we first review the existing amortized Σ-protocol and discuss its relationship with the compression process. Based on the observation, we propose a new protocol to capture the same folding operation in amortizations and compressions. This protocol reduces multiple commitments into a doubly folded commitment, which generalizes the commitment key part as well. Furthermore, we extend the constraints of the new protocol to the forms of mappings, making it more expressive instead of merely arguing the knowledge of the witness. Finally, we introduce an efficient aggregation technique to reduce the proof size.

3.1 Review of Compressed Σ-Protocol

Before introducing our new protocol, we briefly review the original compressed Σ-protocol in [2]. For simplicity, we focus on R_{DL} defined as follows, which implies an opening of the Pedersen vector commitment under the DL assumption:

$$R_{\mathrm{Ped}} = \left\{ A \in \mathbb{G}, \mathbf{ck} \in \mathbb{G}^n, \boldsymbol{m} \in \mathbb{Z}_q^n : A = \langle \boldsymbol{m}, \mathbf{ck} \rangle \right\},$$

where $\mathbf{ck} = \{\mathsf{ck}_i\}_{i=1}^n$ and $\boldsymbol{m} = \{m_i\}_{i=1}^n$. We will discuss proving additional non-linear constraints for the witness in Section 3.3. Now assume a prover wants to prove the knowledge of k instances $\{A_i, \boldsymbol{m}_i\}_{i=1}^n$ under the same commitment key $\mathbf{ck}$. Attema et al. [2] modeled them in a framework with two major phases:

- **Amortization.** Upon receiving a challenge x from the verifier, the prover computes the linear combination (amortization) of k instances into one $\{A, \boldsymbol{m}\} \in R_{\mathrm{Ped}}$ by setting $A := \sum_{i=1}^k x^i \cdot A_i$ and $\boldsymbol{m} := \sum_{i=1}^k x^i \cdot \boldsymbol{m}_i$.
- **Compression.** To show that the amortized instance $\{A, \boldsymbol{m}\}$ satisfies R_{Ped}, the prover and verifier engage in a Bulletproofs compression with $\log n$ iterations to reduce $\boldsymbol{m}$ to a scalar m^*, such that $A^* = m^* \cdot \mathsf{ck}^*$, where ck^* and A^* are publicly derived from $\mathbf{ck}$ and $\boldsymbol{A}$ respectively during the process.

Note that for Σ-protocols, the masking procedure can be viewed as randomly sampling a R_{Ped} instance $\{A_0, \boldsymbol{m}_0\}$ and amortizing it with existing ones, as depicted in Figure 1 (in a compressed Σ-protocol, the prover and verifier engage in the compression procedure in checking $A \stackrel{?}{=} \langle \boldsymbol{m}, \mathsf{ck} \rangle$, rather than sending and verifying $\boldsymbol{m}$ directly). Accordingly, $\{A, \boldsymbol{m}\}$ becomes a masked instance, and the compression procedure does not need to be zero-knowledge. We describe one iteration of compression in Figure 2. According to the security proofs given in [2],

Amortized Σ-protocol Π_{Amor} for relation $\{R_{\text{Ped}}\}_{i=1}^{k}$

Prover$(\mathsf{ck}, \{A_i, \boldsymbol{m}_i\}_{i=1}^{k})$		**Verifier**$(\mathsf{ck}, \{A_i\}_{i=1}^{k})$
$\boldsymbol{m}_0 \leftarrow\!\$\, \mathbb{Z}_q^n$		
$A_0 := \langle \boldsymbol{m}_0, \mathsf{ck} \rangle$	$\xrightarrow{A_0}$	
	$\xleftarrow{x}$	$x \leftarrow\!\$\, \mathbb{Z}_q$
$\boldsymbol{m} := \boldsymbol{m}_0 + \sum_{i=1}^{n} x^i \cdot \boldsymbol{m}_i$	$\xrightarrow{\boldsymbol{m}}$	$A := \sum_{i=0}^{n} x^i \cdot A_i$
		Check: $A \stackrel{?}{=} \langle \boldsymbol{m}, \mathsf{ck} \rangle$

Fig. 1: Procedure of amortized Σ-protocol Π_{Amor}.

both Π_{Amor} and Π_{Comp} provide security guarantees described in the following theorem.

Theorem 1. *Π_{Amor} is a 3-move protocol for k-many R_{Ped} relations. It has perfect completeness, $k+1$-special soundness, and special honest-verifier zero-knowledge. Π_{Comp} is a 3-move protocol for a R_{Ped} relation. It has perfect completeness and 3-special soundness.*

Argument of Knowledge Π_{Comp} for relation R_{Ped}

Prover$(\mathsf{ck}, A, \boldsymbol{m})$		**Verifier**(ck, A)
split $\mathsf{ck}, \boldsymbol{m}$ and compute E_1, E_2		
$\mathsf{ck}_L = \{\mathsf{ck}_i\}_{i=1}^{n/2}, \mathsf{ck}_R = \{\mathsf{ck}_{\frac{n}{2}+i}\}_{i=1}^{n/2}$		
$\boldsymbol{m}_L = \{m_i\}_{i=1}^{n/2}, \boldsymbol{m}_R = \{m_{\frac{n}{2}+i}\}_{i=1}^{n/2}$		
$E_1 := \langle \boldsymbol{m}_L, \mathsf{ck}_R \rangle, E_2 := \langle \boldsymbol{m}_R, \mathsf{ck}_L \rangle$	$\xrightarrow{E_1, E_2}$	
	$\xleftarrow{y}$	$y \leftarrow\!\$\, \mathbb{Z}_q$
$\boldsymbol{m}^* := \boldsymbol{m}_L + y \cdot \boldsymbol{m}_R$		$\mathsf{ck}^* := y \cdot \mathsf{ck}_L + \mathsf{ck}_R$
	$\xrightarrow{\boldsymbol{m}^*}$	$A^* := E_1 + y \cdot A + y^2 \cdot E_2$
		Check: $A^* \stackrel{?}{=} \langle \boldsymbol{m}^*, \mathsf{ck}^* \rangle$

Fig. 2: One iteration of compression Π_{Comp}.

Although the compression process follows a typical "split-and-fold" paradigm, i.e., splitting a vector $\boldsymbol{a}$ into two parts and folding them together, we can still spot some similar patterns compared to the amortization process. Specifically, both compression and amortization processes conduct folding operations by computing a random linear combination of several vectors in terms of a challenge x, which reduces multiple commitments into a more compact commitment (the folding operation in Π_{Comp} does allow folding multiple vectors in [8]). Besides, since the prover in Π_{Comp} folds $\boldsymbol{ck}$ as well, it is natural to wonder if we can also apply the same generalization on Π_{Amor}, which bring us some desirable features:

- Existing amortization process only considers linear constraints, i.e., $A = \mathsf{Com}(\mathsf{ck}, \boldsymbol{m}) \wedge f(\boldsymbol{m}) = y$, where $f(\cdot)$ is a linear function. By generalizing the folding operation for $\boldsymbol{ck}$, it is feasible to enhance its ability to handle non-linear constraints.
- The aggregation techniques given in Bulletproofs' compression can be extended to a more general case, allowing us to optimize the performance for both amortization and compression processes.

So far, our target is clear: we aim to present a new protocol for capturing similar folding operations in both processes in the first step. Next, we accommodate the new protocol to non-linear constraints not discussed in previous work. Finally, we discuss a generalized aggregation technique covering previous works and apply it to optimize the proof size of our new protocol.

3.2 The Basic Protocol

In this part, we aim to propose a general protocol that captures the folding operations in amortization and compression processes for R_{Ped}. To achieve this, we first introduce a generalized commitment on both the commitment key part and the message part. Without loss of generality, we start from a basic relation for doubly homomorphic commitments defined below.

Definition 2 (Relation for doubly homomorphic commitments [11]).

$$R_0 = \left\{ A \in \mathbb{G}, \mathsf{ck} \in \mathbb{G}; m \in \mathbb{Z}_q : \mathsf{Com}(ck; m) = A. \right\}.$$

Given n commitments $\{A_i\}_{i=1}^n$ each satisfying R_0, we consider modeling the folding operation for "squeezing" them into one commitment. To achieve this, we first introduce a generalized notion as a doubly folded commitment.

Doubly Folded Commitment As discussed in Section 1, existing Σ-protocols are limited in handling complicated relations. A primary reason for this limitation is that most of them only consider homomorphism among the same commitment key ck. However, homomorphism among multiple commitment keys is necessary for capturing complicated relations like k-out-of-n proofs in [7] or R1CS

in [8]. To address this limitation, we extend the commitment defined in Definition 1 on the commitment key and message part simultaneously. Concretely, $\mathsf{Com}(ck; m)$ is rewritten as $\mathsf{Com}(g(X); m(X))$ with polynomials

$$g(X) = \sum_{j=1}^{n} X^j \cdot ck_j \in \mathbb{G}^{(<n)}[X], \quad m(X) = \sum_{j=1}^{n} X^j \cdot m_j \in \mathbb{Z}^{(<n)}[X].$$

Here we borrow the "folding" concept to denote $\mathsf{Com}(g(X); m(X))$ as *doubly folded commitment*. Without loss of generality, we claim that the new commitments scheme satisfies the following lemma:

Lemma 2. *If the commitment scheme* $\mathsf{Com}(ck; m)$ *with* $\mathsf{ck} \in \mathbb{G}$, $m \in \mathbb{Z}_q$ *is doubly homomorphic, the doubly folded commitment scheme* $\mathsf{Com}(g(X); m(X))$ *with* $g(X) \in \mathbb{G}^{(<n)}[X]$, $m(X) \in \mathbb{Z}_q^{(<n)}[X]$ *also satisfies the doubly homomorphic properties for all* $g_1(X), g_2(X), m_1(X), m_2(X)$*:*

1. $\mathsf{Com}(g_1(X); m_1(X)) + \mathsf{Com}(g_1(X); m_2(X)) = \mathsf{Com}(g_1(X); m_1(X) + m_2(X))$,
2. $\mathsf{Com}(g_1(X); m_1(X)) + \mathsf{Com}(g_2(X); m_1(X)) = \mathsf{Com}(g_1(X) + g_2(X); m_1(X))$.

We further derive a corollary from the above lemma as:

Corollary 1. *Under any message* $m \in \mathbb{Z}_q$ *committed under key* $\mathsf{ck} \in \mathbb{G}$*, the doubly folded commitment satisfies for all* $X \in \mathbb{Z}_q$*:*

1. $\mathsf{Com}(\sum_{j=1}^{n} X^j \cdot \mathsf{ck}_j; m) = \sum_{j=1}^{n} X^j \cdot \mathsf{Com}(\mathsf{ck}_j; m)$,
2. $\mathsf{Com}(\mathsf{ck}; \sum_{j=1}^{n} X^j \cdot m_j) = \sum_{j=1}^{n} X^j \cdot \mathsf{Com}(\mathsf{ck}; m_j)$.

Next, we illustrate how to compute a doubly folded commitment from the plain doubly homomorphic commitments. This part will provide insight into building a general folding protocol capturing amortization and compression. Concretely, we write the doubly folded commitment $\mathsf{Com}(g(X); m(X))$ with polynomials in coefficient form and further combine the terms with the same degree into one commitment by utilizing the corollary 1 above.

$$\begin{aligned}
\mathsf{Com}(g(X); m(X)) &= \mathsf{Com}\left(\textstyle\sum_{j=1}^{n} X^j \cdot \mathsf{ck}_j; \sum_{i=1}^{n} X^i \cdot m_i\right) \\
&= \textstyle\sum_{j=1}^{n} X^j \cdot \sum_{i=1}^{n} X^i \cdot \mathsf{Com}(\mathsf{ck}_j; m_i) \ // \text{ by corollary 1} \\
&= \textstyle\sum_{i=1}^{n} X^{2i} \cdot \mathsf{Com}(\mathsf{ck}_i; m_i) + \sum_{i=3}^{2n-1} X^i \cdot \mathsf{Com}(\mathsf{ck}'_i; m'_i) \\
&= \textstyle\sum_{i=1}^{n} X^{2i} \cdot A_i + \sum_{i=3}^{2n-1} X^i \cdot E_i. \qquad (1)
\end{aligned}$$

where $E_i = \sum_{j+k=i, j\neq k} \mathsf{Com}(\mathsf{ck}_j; m_k), i = 3, ..., 2n-1$ represents the sum of commitments for all cross terms with the same power X^i. We illustrate the detailed computation of these group elements with a matrix in Figure 3. Concretely, m_i, ck_i are the coefficients of X^i term in $g(X), m(X)$ respectively. $A_i = \mathsf{Com}(\mathsf{ck}_i; m_i)$ is the commitment along the main diagonal. E_i is the sum of commitments for all cross terms with the same X^i. For example, E_{n+1} is the sum of all commitments in $(n+1)$-degree terms $\{\mathsf{Com}(\mathsf{ck}_i; m_{n+1-i})\}_{i=1}^{n}$ in the secondary diagonal. Note that group elements $\{A_i\}_{i=1}^{n}, \{E_i\}_{i=3}^{2n-1}$ are all independent of X. Therefore, the prover can send these group elements to prove the validity of a doubly folded commitment.

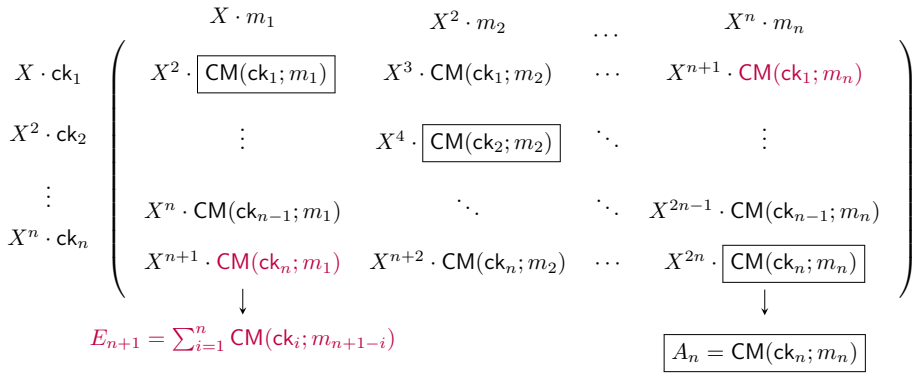

Fig. 3: Transforming a doubly folded commitment into plain commitments.

Argument of Knowledge We now introduce a new relation as well as its Argument of Knowledge for capturing the folding operations in both the amortization and compression processes as below:

Definition 3. (Relation for doubly folded commitment)

$$R_1 = \left\{ C \in \mathbb{G}, g(X) \in \mathbb{G}^{(<n)}[X]; m(X) \in \mathbb{Z}_q^{(<n)}[X] : \mathsf{Com}(g(x); m(x)) = C. \right\}.$$

where $C = \sum_{i=1}^{n} x^{2i} \cdot A_i + \sum_{i=3}^{2n-1} X^i \cdot E_i$, *and* $x \in \mathbb{Z}_q$ *is a challenge randomly sampled by the verifier.*

Generally speaking, the relation R_1 plays the same role as the relation R_{Ped} used in the previous compressed Sigma protocol. Differently, R_1 provides us a more general form to simultaneously capture the cases of amortization and compression and, therefore, allows achieving non-linear constraints and aggregations. Based on R_1, we build an Argument of Knowledge for a set of R_0's with respect to $\{A_i\}_{i=1}^n$ and $\{\mathsf{ck}_i\}_{i=1}^n$ that reduces them to R_1 containing only one doubly folded commitment C. We present the step-by-step construction of this protocol in Figure 4 and denote it as Π_1. In Π_1, the prover first computes corresponding group elements $\{E_i\}_{i=3}^{2n-1}$ according to Equation (1) and sends to the verifier. Then the polynomial $m(X)$ is opened on a randomly chosen point x and so does $g(x)$ (computed by the verifier). In the checking equation, the verifier computes the doubly folded commitment on the left-hand side as $\mathsf{Com}(g(x); m(x))$ and checks whether it equals the right-hand side value computed from $\{A_i\}_{i=1}^n$ and $\{E_i\}_{i=3}^{2n-1}$. As a result, we obtain the protocol Π_1 as is a general protocol of folding operation that can capture both Π_{Amor} and Π_{Comp}. Its security is ensured by the following theorem. The formal proof of this theorem is presented in Appendix B.1.

Theorem 2. *Π_1 is a 3-move protocol for relation R_0. It provides perfect completeness and n-special soundness under the binding property of the commitment scheme* Com.

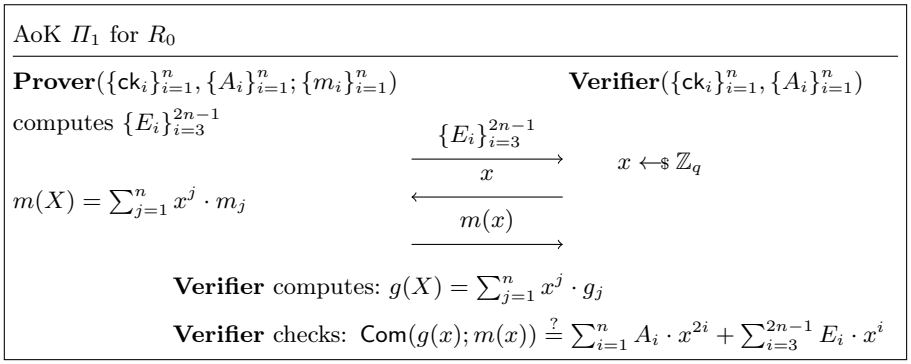

Fig. 4: AoK for reducing R_0's to R_1.

3.3 Adding Non-linear Constraints

This subsection extends protocol Π_1 to support proofs for witnesses with additional constraints. Note that in Π_1, the prover only demonstrates knowledge of a witness $m(X)$, it doesn't allow proving that this witness satisfies further relations For example, we might need to prove that vectors (or equivalently, polynomials in this context) $\boldsymbol{a}$, $\boldsymbol{b}$, and $\boldsymbol{c}$ satisfy the equation $\boldsymbol{a} + \boldsymbol{b} = \boldsymbol{c}$, which corresponds to demonstrating that the polynomials $a(X)$, $b(X)$, and $c(X)$ satisfy $a(X) + b(X) = c(X)$. Thus, we extend the relation R_1 to allow demonstrating that the witness $m(X)$ satisfies additional constraints beyond simply being the committed message, i.e, $\mathsf{Com}(g(x); m(x)) = C, f(m(x)) = y$. Furthermore, to accommodate more general scenarios involving multiple polynomials, we introduce a *mapping* that relates several polynomials to a single one. The mapping is defined as a function $f(\cdot)$:

$$f(\boldsymbol{m}(X)) = m_1(X) \otimes \cdots \otimes m_k(X), \tag{2}$$

where $\boldsymbol{m}(X) = (m_1(X), \cdots, m_k(X))$. The symbol $\otimes$ is either the polynomial addition or multiplication operator in the ring $\mathbb{Z}_q[X]$ (with degree $(< n)$ omitted). For polynomials $m_1(X), m_2(X) \in \mathbb{Z}_q[X]$, their addition satisfies $(m_1(X), m_2(X)) \rightarrow m_3(X)$, where the degree of the output polynomial $m_3(X)$ is at most n. Their multiplication satisfies $(m_1(X), m_2(X)) \rightarrow m_3(X)$, where the degree of the output polynomial $m_3(X)$ is at most $2n$. A simple example is proving the inner-product constraint of two vectors, i.e., $\langle (a_1, a_2), (b_1, b_2) \rangle = u$. The vectors can be first represented in polynomial form as $m_1(X) = a_1 \cdot X + a_2$ and $m_2(X) = b_1 \cdot X + b_2$, then the constraint is denoted as a function $f(\langle m_1(X), m_2(X) \rangle)$ mapping two n-degree polynomials to a $2n$-degree polynomial. We formally define such a relation as below.

Definition 4. (Extended Relation for doubly folded commitments)

$$R_2 = \left\{ \begin{array}{c} C \in \mathbb{G}, g(X) \in \mathbb{G}[X], k \in \mathbb{N}, f : \mathbb{Z}_q^k[X] \mapsto \mathbb{Z}_q^{(<kn)}[X]; \\ \boldsymbol{m}(X) = \{m_j(X) \in \mathbb{Z}_q[X]\}_{j=1}^k : \\ \mathsf{Com}(g(X); m_j(X)) = C_j \ \wedge \ \mathsf{Com}(g(X); f(\boldsymbol{m}(X))) = C \end{array} \right\}.$$

where $C_j = \sum_{i=1}^{n} x^{2i} \cdot A_{i,j} + \sum_{i=3}^{2n-1} X^i \cdot E_{i,j}$, $C = \sum_{l=0}^{kn} X^l \cdot F_l$ *and* $x \in \mathbb{Z}_q$ *is a challenge randomly sampled by the verifier.*

Based on the discussion above, we observe that for any function $f(\cdot)$ with n operations $\otimes$, when taking an k-dimensional vector $\boldsymbol{m}(X)$, its output $f(\boldsymbol{m}(X))$ must be a polynomial with a degree at most kn in $\mathbb{Z}_q^{(<kn)}[X]$. This degree is determined by the maximum degree n of each polynomial in $\boldsymbol{m}(X)$ and the number of polynomials k involved. The commitments $\{F_l\}_{l=0}^{kn}$ capture the coefficients of the polynomial $f(\boldsymbol{m}(X))$. Therefore, for an n-degree polynomial of commitment key $g(X)$, the verifier can check constraint $f(\cdot)$ with the k-dimensional opening vector $\boldsymbol{m}(x)$ and commitments $\{F_l\}_{l=0}^{kn}$ by the equation

$$\mathsf{Com}(g(x); f(\boldsymbol{m}(x))) = \sum_{l=0}^{kn} x^l \cdot F_l. \tag{3}$$

The above equation is computed similarly as Equation (1). We obtain at most kn number of F_l's since $g(x), f(\boldsymbol{m}(x))$ have degrees n, kn respectively. Accordingly, we can extend Π_1 for multiple polynomials $\boldsymbol{m}(x)$ with an additional constraint $f(\cdot)$. The *extended protocol* of Π_1 with additional constraints is presented as Π_2 in Figure 5. Concretely, the prover follows a process similar to Π_1 for each $m_j(X)$ to prove the knowledge of the polynomial vector $\boldsymbol{m}(X)$. The verifier uses commitments $\{A_{i,j}\}_{i,j=1}^{n,k}$ and commitments $\{E_{i,j}\}_{i=3,j=1}^{2n-1,k}$ to check whether equation (1) in Π_2 holds for $j = 1, ..., k$, confirming the prover's knowledge of the witness. With commitments $\{F_l\}_{l=0}^{kn}$, the verifier checks whether equation (2) in Π_2 holds for $\boldsymbol{m}(x)$ and $f(\cdot)$, ensuring that the witness $\boldsymbol{m}(X)$ satisfies the constraint $f(\cdot)$. For the convenience of discussion, we denote the new commitments sent in Π_2, i.e., $E_{i,j}, F_l$ as auxiliary commitments. Finally, the verifier is convinced that the prover does know the witness $\boldsymbol{m}(X)$ satisfying the constraint $f(\cdot)$.

AoK Π_2 for R_0 with non-linear constraints

Prover$(g(X), \{A_{i,j}\}_{i,j=1}^{n,k}, f(\cdot); \boldsymbol{m}(X))$		**Verifier**$(g(X), \{A_{i,j}\}_{i,j=1}^{n,k}, f(\cdot))$
compute $\{E_{i,j}\}_{i=3,j=1}^{2n-1,k}, \{F_l\}_{l=0}^{kn}$		
	$\xrightarrow{\{E_{i,j}\}_{i=3,j=1}^{2n-1,k}, \{F_l\}_{l=0}^{kn}}$	$x \leftarrow\!\!\$\ \mathbb{Z}_q$
	$\xleftarrow{x}$	
compute $\boldsymbol{m}(x)$	$\xrightarrow{\boldsymbol{m}(x)}$	

Verifier computes: $g(x)$

Verifier checks:

(1) $\mathsf{Com}(g(x); m_j(x)) \stackrel{?}{=} \sum_{i=1}^{n} A_{i,j} \cdot x^{2i} + \sum_{i=3}^{2n-1} E_{i,j} \cdot x^i, \forall\ j = 1, ..., k$

(2) $\mathsf{Com}(g(x); f(\boldsymbol{m}(x))) \stackrel{?}{=} \sum_{l=0}^{kn} F_l \cdot x^l$

Fig. 5: AoK for reducing R_0's to R_2.

In summary, the protocol Π_2 gives us a general model for reducing multiple commitments satisfying certain constraints to a doubly folded commitment scheme (actually Π_2 can prove more than one constraint). We claim that the Theorem 3 below holds for protocol Π_2. The formal proof of this theorem is given in Appendix B.2.

Theorem 3. *Π_2 is a 3-move protocol for relation R_0 with additional constraints. It provides perfect completeness and n-special soundness under the binding property of the commitment scheme* Com.

3.4 Aggregation Technique

Typically, a complex zero-knowledge proof scheme may include multiple inputs with different constraints. For example, a range proof on $v \in [0, 2^{128}-1]$ has two constraints (1) $\boldsymbol{b}$ is a 128-bit binary vector, and (2) $\boldsymbol{b}$ is the binary representation of value v. Consequently, it may not be intuitive for a designer to spot the potential optimization for their proof scheme with such complicated relations. To solve this problem, we present a generic aggregation technique for our protocol Π_2 above, which provides a universal paradigm for possible optimization.

Our technique is mainly based on an important observation: multiple verification equations can be aggregated into one as long as their group elements of the same exponent are committed under distinct and independently chosen keys. We denote equations satisfying the above properties as *mutually independent* equations. Take the plain DL commitment scheme as an example: the security of the setup algorithm requires that two different keys $\mathsf{ck}_1 = G, \mathsf{ck}_2 = H$ are independently chosen generators. The prover can not find a value a for the DL instance $a \cdot G = H$. As a result, the aggregated commitment $m_1 \cdot G + m_2 \cdot H$ ensures that the prover exactly knows m_1 and m_2 with an overwhelming probability.

$$
\begin{array}{cccccccccc}
① & \mathsf{CM}(g(X); m(X)) & = & X^0 \cdot \mathsf{CM}(\mathsf{ck}_0; m_0) & + & X^1 \cdot \mathsf{CM}(\mathsf{ck}_1; m_1) & \cdots & X^n \cdot \mathsf{CM}(\mathsf{ck}_n; m_n) \\
 & & = & X^0 \cdot (E_0) & + & X^1 \cdot (E_1) & \cdots & X^n \cdot (E_n) \\
② & \mathsf{CM}(g'(X); m'(X)) & = & X^0 \cdot \mathsf{CM}(\mathsf{ck}'_0; m'_0) & + & X^1 \cdot \mathsf{CM}(\mathsf{ck}'_1; m'_1) & \cdots & X^n \cdot \mathsf{CM}(\mathsf{ck}'_n; m'_n) \\
 & & = & X^0 \cdot (E'_0) & + & X^1 \cdot (E'_1) & \cdots & X^n \cdot (E'_n) \\
 & & & \downarrow & & \downarrow & & \downarrow \\
③ & = & ① + ② & = & X^0 \cdot (E_0 + E'_0) & + & X^1 \cdot (E_1 + E'_1) & \cdots & X^n \cdot (E_n + E'_n)
\end{array}
$$

Fig. 6: Illustration of the aggregation technique.

As shown in Figure 6, equations ① and ② can be aggregated into ③, where each pair of $(E_i, E'_i), i = 0, ..., n$ are aggregated into $E_i + E'_i$. Since each pair of group elements is generated under different commitment keys $\mathsf{ck}_i, \mathsf{ck}'_i$, it is sufficient to guarantee equations ① and ② hold by ③ under the binding property of commitment scheme. Besides, the two equations are not required to have the same degree of polynomials because we can pad them to the same length with zero terms, e.g., $X^0 \cdot \mathsf{Com}(\mathsf{ck}_0; m_0) + X^1 \cdot \mathsf{Com}(\mathsf{ck}_1; 0)$. Based on

the above observation, we can deduce two lemmas under a more general case below. Note that we use the vector form with index in the remaining part, e.g., $\boldsymbol{E} = (E_0, ..., E_n), \boldsymbol{E}[i] = E_i, \ \forall i = 0, ...n$ for ease of exposition.

Lemma 3. *Given an AoK Π_2 for multiple R_0's with non-linear constraints, an aggregation of auxiliary commitments can be applied if its verification equations are mutually independent. Guaranteed by the binding property of the commitment scheme used, the obtained compact protocol provides perfect completeness and special soundness. If a masking relation is included in the input R_0's, then the protocol also provides SHVZK.*

For example, assume a protocol Π_2 with two constraints $f(\cdot)$, $g(\cdot)$, denote its corresponding auxiliary commitments as $\boldsymbol{F}$, $\boldsymbol{F'}$ respectively. After padding $\boldsymbol{F}$, $\boldsymbol{F'}$ into the same length of $\max(|\boldsymbol{F}|, |\boldsymbol{F'}|)$, we can aggregate them into $\boldsymbol{F''}$ as long as each pair of elements $\boldsymbol{F}[k], \boldsymbol{F'}[k]$ are committed under independent commitment keys (i.e., the equations are mutually independent). Note that the basic verification for each doubly folded commitment $\mathsf{Com}(g(x); m_j(x)), j = 1, ..., k$ can also be aggregated with $f(\cdot)$, $g(\cdot)$ if they are mutually independent (i.e., aggregating $\boldsymbol{E}$ with $\boldsymbol{F}, \boldsymbol{F'}$). We denote the aggregated protocol as a *compact protocol*.

Lemma 4. *Given multiple AoK Π_2's each for multiple R_0's with non-linear constraints, an aggregation of auxiliary commitments can be applied if their verification equations are mutually independent. Guaranteed by the binding property of the commitment scheme used, the obtained compact protocol provides perfect completeness and special soundness. If a masking relation is included in the input R_0's of each Π_2, then the whole protocol also provides SHVZK.*

As an example, assume two compact protocols Π_2, Π_2' with auxiliary commitments as $\boldsymbol{R}, \boldsymbol{E}, \boldsymbol{F}$ and $\boldsymbol{R'}, \boldsymbol{E'}, \boldsymbol{F'}$ respectively. If each pair of $(R[i], R[i]')$, $(E[j], E[j]')$ or $(F[k], F[k]')$ of the same exponent X^i, X^j, X^k are committed under independent commitment keys, we can aggregate them and obtain new auxiliary commitments $\boldsymbol{R''}, \boldsymbol{E''}, \boldsymbol{F''}$. The above two lemmas offer us a useful technique for optimizing the communication cost of instantiated protocols, especially for applications with complex relations. In fact, similar aggregation techniques have already been used in a large number of advanced proof schemes such as one-out-of-many proofs for ring signatures, Bulletproofs, and inner pairing product argument [10, 11, 16], while none of them explicitly summarizes as formal definitions. Our generalized protocol provides a proper platform to discuss and formalize this technique. Lemma 3 and Lemma 4 are trivial to prove under the binding property of the used commitment scheme.

4 Use Cases

In this section, we first demonstrate the versatility of our model by covering processes used in existing models with our new model in Section 4.1. The covered process includes the compression process used in [2, 8, 10, 11], and the amortization process in [2, 3]. Next, we showcase the efficiency of our new model by

optimizing several existing protocols with our aggregation techniques, including the binary proofs and ring signatures in [7] and k-out-of-N proofs [3].

For conciseness, we only show the feasibility of each case without describing the detailed protocol. This is done by taking the Argument of Knowledge protocol given in Section 3 as a black-box algorithm. Concretely, we define an algorithm $\mathsf{AoK}(\boldsymbol{R}, f, \boldsymbol{y}) \to R_{\mathsf{dfc}}$ as an Argument of Knowledge reducing the input of a vector of relations $\boldsymbol{R}$ each with plain commitments, the constraint function f and the image vector $\boldsymbol{y}$, into a single relation R_{dfc} with doubly folded commitments. Note that all relations considered in our model are in the commit-and-prove paradigm, i.e., each relation already contains the commitments computed by the prover ahead of the protocol.

4.1 Existing Cases

This part shows how to capture the processes in the existing generalized Σ-protocol model. Since we only validate for the feasibility, optimizations (applying aggregation techniques) are not mentioned in this subsection.

Covering Amortization In the amortization process given in [2], the prover with inputs of k relations of which the witness satisfies $\mathsf{Com}(\boldsymbol{g}_j, \boldsymbol{m}_j) = C_j$ and $f(\boldsymbol{m}_j) = y_j$ for $j \in [1, k]$, aims to aggregate all the witness into one as $\boldsymbol{m}^* = \sum_{j=1}^{k} \boldsymbol{m}_j \cdot x^j$ with a challenge x. We show how to cover the amortizing operation with our model. Specifically, we define the relation $R_j, j \in [1, k]$ as

$$R_j = \left\{ \boldsymbol{g} \in \mathbb{G}^n; \boldsymbol{m}_j \in \mathbb{F}^n : \langle \boldsymbol{g}_j, \boldsymbol{m}_j \rangle = C_j \wedge f(\boldsymbol{m}_j) = y_j \right\}.$$

Next, the prover executes our protocol with input as relation vector $\boldsymbol{R}$, constraint function and the images each satisfies $f(\boldsymbol{m}_j) = y_j$, the output relation is denoted as $\mathsf{AoK}(\boldsymbol{R}, f, \boldsymbol{y}) \to R_{\mathsf{dfc}}$:

$$R_{\mathsf{dfc}} = \left\{ \begin{matrix} C, u \in \mathbb{G}, \boldsymbol{g}(X) \in \mathbb{G}^{(<k)\times n}[X]; \boldsymbol{m}(X) \in \mathbb{F}^{(<k)\times n}[X] : \\ \langle \boldsymbol{g}(X), \boldsymbol{m}(X) \rangle = C \wedge \mathsf{Com}(u, f(\boldsymbol{m}(X))) = C_y \end{matrix} \right\}.$$

where $\boldsymbol{m}(X) = \sum_{j=1}^{k} X^j \cdot \boldsymbol{m}_j$, $\boldsymbol{g}(X) = \sum_{j=1}^{k} X^j \cdot \boldsymbol{g}_j$, $C = \sum_{j=1}^{k} X^j \cdot C_j$, $C_y = u \cdot \sum_{j=1}^{k} X^j \cdot y_j$. Note that the model above [2] only allows homomorphism constraint $f(\cdot)$. With our new model, we can extend the amortization process to enable non-linear constraints.

Covering Compression (IPA) In the compression process, the prover splits the linear-size opening vector and then folds them into one with only half of the length. We now show how to cover the folding operation within our model. Consider two vectors $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{F}^n$, where we need to prove knowledge of their inner product as $\langle \boldsymbol{a}, \boldsymbol{b} \rangle$. Rather than proving this directly, which would be inefficient for large vectors, we first split each vector into two equal halves. Taking the

vector $\boldsymbol{a}$ as an example, the prover splits it into $\boldsymbol{a}_L, \boldsymbol{a}_R \in \mathbb{F}^{n/2}$ (assuming n can be divided by 2). The relation R_a for splitting vectors $\boldsymbol{a}_L, \boldsymbol{a}_R$ is given below

$$R_a = \left\{ \begin{matrix} A_L, A_R \in \mathbb{G}, \boldsymbol{g}_L, \boldsymbol{g}_R \in \mathbb{G}^{n/2}; \boldsymbol{a}_L, \boldsymbol{a}_R \in \mathbb{F}^{n/2} : \\ \langle \boldsymbol{g}_L, \boldsymbol{a}_L \rangle = A_L \wedge \langle \boldsymbol{g}_R, \boldsymbol{a}_R \rangle = A_R \end{matrix} \right\}.$$

The relation R_b for vectors $\boldsymbol{b}_L, \boldsymbol{b}_R$ are almost the same except that the commitments B_L, B_R are under different keys $\boldsymbol{h}_L, \boldsymbol{h}_R$.

Next, the prover executes our protocol with input as relations R_a, R_b, a constraint function and the inner product value satisfying $f(\boldsymbol{a}, \boldsymbol{b}) = \langle \boldsymbol{a}, \boldsymbol{b} \rangle = v$, the output relation is denoted as $\mathsf{AoK}((R_a, R_b), f, v) \to R_{\mathsf{dfc}}$:

$$R_{\mathsf{dfc}} = \left\{ \begin{matrix} C, u \in \mathbb{G}, \boldsymbol{g}(X), \boldsymbol{h}(X) \in \mathbb{G}^{(<2)\times n/2}[X]; \boldsymbol{a}(X), \boldsymbol{b}(X) \in \mathbb{F}^{(<2)\times n/2}[X] : \\ \mathsf{Com}(\boldsymbol{g}(X); \boldsymbol{a}(X)) = C_a \wedge \mathsf{Com}(\boldsymbol{h}(X); \boldsymbol{b}(X)) = C_b \\ \wedge \mathsf{Com}(u; f(\boldsymbol{a}(X), \boldsymbol{b}(X))) = C \end{matrix} \right\}.$$

where $\boldsymbol{a}(X) = \boldsymbol{a}_L + X \cdot \boldsymbol{a}_R$, $\boldsymbol{b}(X) = X \cdot \boldsymbol{b}_L + \boldsymbol{b}_R$, $\boldsymbol{g}(X) = X \cdot \boldsymbol{g}_L + \boldsymbol{g}_R$, $\boldsymbol{h}(X) = \boldsymbol{h}_L + X \cdot \boldsymbol{h}_R$, $C_a = X^2 \cdot \langle \boldsymbol{g}_L, \boldsymbol{a}_R \rangle + X \cdot A_L \cdot A_R + \langle \boldsymbol{g}_R, \boldsymbol{a}_L \rangle$, $C_b = X^2 \cdot \langle \boldsymbol{h}_R, \boldsymbol{b}_L \rangle + X \cdot B_L \cdot B_R + \langle \boldsymbol{h}_L, \boldsymbol{b}_R \rangle$, $C = u \cdot (X^2 \cdot \langle \boldsymbol{a}_R, \boldsymbol{b}_L \rangle + X \cdot v + \langle \boldsymbol{a}_L, \boldsymbol{b}_R \rangle)$.

The above protocol is executed recursively until the length of the witness is reduced to constants, as in Bulletproofs. Note that this case also indicates that the compression process in [2] and [3] can be covered since they only consider constraints in homomorphism. Moreover, our model can be extended to capture the inner pairing product constraint by following the definitions given in [11].

4.2 More Compact Proofs

Case 1: Compact Binary Proofs. Bootle *et al.* [7] introduce a ring signature scheme built from binary proofs, i.e., the binary proofs prove that a witness b is either 0 or 1. To prove this relation with Σ-protocol, the prover reduces the relation $b = 0 \vee 1$ to an equation that $b \cdot (1 - b) = 0$, which can be formalized as a function $f : \{0, 1\} \mapsto \mathbb{F}$. Therefore, we can write a relation for a binary b as below

$$R_b = \left\{ B \in \mathbb{G}, g \in \mathbb{G}; b \in \{0, 1\} : \mathsf{Com}(g; b) = B \wedge f(b) = 0 \right\}.$$

Besides, the prover must also include the following relation of the masking value for zero knowledge:

$$R_a = \left\{ A \in \mathbb{G}, g \in \mathbb{G}; a \leftarrow\!\!\$\, \mathbb{F} : \mathsf{Com}(g; a) = A \right\}.$$

In some of the previous work [13,23], extra commitments are sent due to the omission of the aggregation technique. By applying our model, this problem can be hedged. Concretely, to amortize these two relations, the prover can execute our protocol with the input of relations R_b, R_a, a function $f : \{0, 1\} \mapsto \mathbb{F}$, and the image 0. The output relation is $\mathsf{AoK}(R_b, R_a, f, 0) \to R_{\mathsf{dfc}}$:

$$R_{\mathsf{dfc}} = \left\{ \begin{matrix} C \in \mathbb{G}, g \in \mathbb{G}; m(X) \in \mathbb{F}^{(<2)}[X] : \\ \mathsf{Com}(g; m(X)) = C \wedge \mathsf{Com}(g, f(m(X))) = X \cdot E_1 \cdot E_2 \end{matrix} \right\}.$$

where $C = X \cdot B \cdot A$, and $f(m(X)) = (bX + a) \cdot (X - bX + a) = (a - 2ab)X - a^2$ such that $E_1 = \mathsf{Com}(g; a - 2ab), E_2 = \mathsf{Com}(g; -a^2)$. According to our conclusion (Lemma 3) in Section 3.4, it is feasible to aggregate auxiliary commitments as long as they are mutually independent. Concretely, if we replace the commitment key of E_2 with another unknown-order element h, it is feasible to aggregate it with A since they all have the same monomial as X^0. Note that E_1 can not be aggregated to B because B is not an auxiliary commitment. As a result, we can reduce the proof size of the binary proof by one group of elements. Furthermore, the same idea can be applied several times in the ring signature scheme given in [7], reducing the signature size from $\log(N) + 4$ to $\log(N)$, where the N is the ring size.

Case 2: Compact k-out-of-n Partial Knowledge Proofs Attema *et al.* [3] propose zero-knowledge proofs of partial knowledge for k-out-of-n secrets. To achieve a compressed proof, they extend the compressed Σ-protocol theory [2] to enable homomorphism constraints. Upon a general case with the function $f : \mathbb{F}^n \mapsto \mathbb{G}$, Attema *et al.* introduce an optimization technique to reduce the asymptotic proof complexity from $4\log(n)$ to $2\log(n)$ through the pairing based commitments [17]. The downside of this approach is that the size of pairing group $\mathbb{G}_T$ elements is much larger than $\mathbb{G}$ elements (for instance, 12 times larger on bls12-381), and the computation cost of pairing is also substantial. In this work, we introduce an alternative approach to reducing the proof complexity without using pairing.

To begin with, we briefly recap the k-out-of-n relation presented in [3]: the prover wants to show that a witness $\boldsymbol{s}$ to the commitment $P = \langle \boldsymbol{s}, \boldsymbol{g} \rangle$ satisfies that $P_i = s_i \cdot g$ for all $i \in S$, where S is a subset of $[1, n]$ with cardinality k. To prove this relation with the compressed Σ-protocol, the prover can reduce the relation of $\boldsymbol{s}$ to a relation of $\boldsymbol{y}$ satisfying a homomorphism constraint $L : \mathbb{F}^{2n-k} \mapsto \mathbb{G}$:

$$t_i \cdot g \cdot (-\textstyle\sum_j a_j i^j) \cdot P_i = P_i,$$

where $\boldsymbol{y} = (a_1, ..., a_{n-k}, t_1, ..., t_n) \in \mathbb{F}^{2n-k}$ derived from the original $\boldsymbol{s}$. Specifically, $p(X) = 1 + \sum_{j=1}^{n-k} a_j \cdot X^j$ such that $p(0) = 1$ and $p(i) = 0$ for all $i \notin S$ and $t_i = p(i)s_i$. If we temporarily omit the zero-knowledge, i.e., the prover can open the witness, we can observe that the prover is proving the constraint $f : \mathbb{F}^{2n-k} \mapsto \mathbb{F}$ for all $i \in [1, n]$ as $t_i - s_i \cdot \sum_j a_j i^j = s_i$. Therefore, we can write n relations with respect to $\boldsymbol{y}$ as follows

$$R_i = \left\{ \begin{matrix} Y \in \mathbb{G}, \boldsymbol{g} \in \mathbb{G}^{2n-k}; \boldsymbol{y} \in \mathbb{F}^{2n-k}, s_i \in \mathbb{F} : \\ \mathsf{Com}(\boldsymbol{g}; \boldsymbol{y}) = Y \wedge t_i - s_i \cdot \sum_j a_j i^j = s_i, i \in [1, n] \end{matrix} \right\}$$

To amortize these relations, the prover can execute our protocol with the input of all relations $R_i, i = 1, ..., n$, a set of functions $f_i : \mathbb{F}^{2n-k} \mapsto \mathbb{F}, i = 1, ..., n$ and their images $\boldsymbol{x}$. The output relation is $\mathsf{AoK}(\{R_i\}_{i=1}^n, \{f_i\}_{i=1}^n, \boldsymbol{x}) \to R_{\mathsf{dfc}}$:

$$R_{\mathsf{dfc}} = \left\{ \begin{matrix} Y \in \mathbb{G}, \boldsymbol{g} \in \mathbb{G}^{2n-k}; \boldsymbol{y} \in \mathbb{F}^{2n-k} : \\ \mathsf{Com}(\boldsymbol{g}; \boldsymbol{y}) = Y \wedge_{i=1}^n \mathsf{Com}(g; f_i(\boldsymbol{y})) = P_i \end{matrix} \right\}.$$

where $\mathsf{Com}(g; f_i(\boldsymbol{y})) = x_i \cdot g$ indicates that $t_i \cdot g \cdot x_i \cdot (-\sum_j a_j i^j) \cdot g = x_i \cdot g$. Note that $\mathsf{Com}(\boldsymbol{g}; \boldsymbol{y})$ and $\mathsf{Com}(g; f_i(\boldsymbol{y})), i = 1, .., n$ are all auxiliary commitments sent among the protocol for proving the k-out-of-n relation for $\boldsymbol{x}$. According to our conclusions (Lemma 3) in Section 3.4, it is feasible to aggregate all auxiliary commitments up as long as they are mutually independent. However, we observe that their commitment keys do not satisfy the requirements. To amend this, a straightforward approach is to raise the commitment key g to a verifier chosen power r^i for each $\mathsf{Com}(g; f_i(\boldsymbol{y}))$. This aggregation technique yields us an aggregated commitment as $\mathsf{Com}(\boldsymbol{g}; \boldsymbol{y}) \cdot \sum_{i=1}^{n} \mathsf{Com}(r^i \cdot g; f_i(\boldsymbol{y}))$ assuming that $\boldsymbol{g}, \{r^i \cdot g\}_{i=1}^n$ are DL independent. We finally obtain a compact protocol for k-out-of-n relation with proof of $2\log(2N - k + 1) - 1$ group elements, less than the previous $4\log(2N - k + 1) - 5$ group elements in [3].

Acknowledgements. This research has received partial support from HK RGC GRF under Grants PolyU 15205624/QCFD, 15202123/QC1V, 15207522/Q93W, and 15216721/Q86A, and NSFC Youth 62302418/ZGJV.

References

1. Abdalla, M., Benhamouda, F., MacKenzie, P.: Security of the j-pake password-authenticated key exchange protocol. In: 2015 IEEE Symposium on Security and Privacy. pp. 571–587. IEEE (2015)
2. Attema, T., Cramer, R.: Compressed Σ-Protocol Theory and Practical Application to Plug & Play Secure Algorithmics. In: Proc. of the Annual International Cryptology Conference (CRYPTO). pp. 513–543. Springer (2020)
3. Attema, T., Cramer, R., Fehr, S.: Compressing Proofs of k-out-of-n Partial Knowledge. In: Proc. of the Annual International Cryptology Conference (CRYPTO). pp. 65–91. Springer (2021)
4. Bayer, S., Groth, J.: Zero-knowledge argument for polynomial evaluation with application to blacklists. In: Proc. of the Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT). pp. 646–663. Springer (2013)
5. Bayer, S.G.M.: Practical Zero-Knowledge Protocols based on the Discrete Logarithm Assumption. Ph.D. thesis, University College London (UCL) (2014)
6. Boneh, D., Bünz, B., Fisch, B.: Batching techniques for accumulators with applications to iops and stateless blockchains. In: Proc. of the Annual International Cryptology Conference (CRYPTO). pp. 561–586. Springer (2019)
7. Bootle, J., Cerulli, A., Chaidos, P., Ghadafi, E., Groth, J., Petit, C.: Short Accountable Ring Signatures Based on DDH. In: Proc. of the European Symposium on Research in Computer Security (ESORICS). Springer (2015)
8. Bootle, J., Cerulli, A., Chaidos, P., Groth, J., Petit, C.: Efficient Zero-Knowledge Arguments for Arithmetic Circuits in the Discrete Log Setting. In: Proc. of the Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT). Springer (2016)
9. Bootle, J., Groth, J.: Efficient Batch Zero-Knowledge Arguments for Low Degree Polynomials. In: Proc. of the IACR International Conference on Practice and Theory in Public Key Cryptography (PKC). pp. 561–588. Springer (2018)
10. Bünz, B., Bootle, J., Boneh, D., Poelstra, A., Wuille, P., Maxwell, G.: Bulletproofs: Short Proofs for Confidential Transactions and More. In: Proc. of the IEEE Symposium on Security and Privacy (S&P). IEEE (2018)

11. Bünz, B., Maller, M., Mishra, P., Tyagi, N., Vesely, P.: Proofs for inner pairing products and applications. In: Proc. of the Annual International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT). pp. 65–97. Springer (2021)
12. Cramer, R.: Modular design of secure yet practical cryptographic protocols. Ph. D.-thesis, CWI and U. of Amsterdam **2** (1996)
13. Esgin, M.F., Zhao, R.K., Steinfeld, R., Liu, J.K., Liu, D.: Matrict: efficient, scalable and post-quantum blockchain confidential transactions protocol. In: Proc. of the ACM Conference on Computer & Communications Security (CCS). pp. 567–584 (2019)
14. Gorbunov, S., Reyzin, L., Wee, H., Zhang, Z.: Pointproofs: Aggregating proofs for multiple vector commitments. In: Proc. of the ACM Conference on Computer & Communications Security (CCS). pp. 2007–2023 (2020)
15. Groth, J.: Linear algebra with sub-linear zero-knowledge arguments. In: Proc. of the Annual International Cryptology Conference (CRYPTO). pp. 192–208. Springer (2009)
16. Groth, J., Kohlweiss, M.: One-Out-of-Many Proofs: Or How to Leak a Secret and Spend a Coin. In: Proc. of the Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT). Springer (2015)
17. Lai, R.W., Malavolta, G., Ronge, V.: Succinct arguments for bilinear group arithmetic: practical structure-preserving cryptography. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security. pp. 2057–2074 (2019)
18. Morais, E., Koens, T., Van Wijk, C., Koren, A.: A survey on zero knowledge range proofs and applications. SN Applied Sciences **1**(8), 1–17 (2019)
19. Nizzardo, L.: Incrementally aggregatable vector commitments and applications to verifiable decentralized storage. In: Proc. of the Annual International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT). Springer (2020)
20. Pedersen, T.P.: Non-interactive and information-theoretic secure verifiable secret sharing. In: Proc. of the Annual International Cryptology Conference (CRYPTO). pp. 129–140. Springer (1992)
21. Tomescu, A., Abraham, I., Buterin, V., Drake, J., Feist, D., Khovratovich, D.: Aggregatable subvector commitments for stateless cryptocurrencies. In: Proc. of International Conference on Security and Cryptography for Networks (SCN). pp. 45–64. Springer (2020)
22. Vaikuntanathan, V., Vasudevan, P.N.: Secret sharing and statistical zero knowledge. In: Proc. of the Annual International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT). pp. 656–680. Springer (2015)
23. Yuen, T.H., Sun, S.f., Liu, J.K., Au, M.H., Esgin, M.F., Zhang, Q., Gu, D.: Ringct 3.0 for blockchain confidential transaction: Shorter size and stronger security. In: Proc. of International Conference on Financial Cryptography and Data Security (FC). pp. 464–483. Springer (2020)
24. Zhang, M., Chen, Y., Yao, C., Wang, Z.: Sigma protocols from verifiable secret sharing and their applications. Cryptology ePrint Archive (2023)
25. Zheng, T., Gao, S., Song, Y., Xiao, B.: Leaking arbitrarily many secrets: Any-out-of-many proofs and applications to ringct protocols. In: 2023 IEEE Symposium on Security and Privacy (SP). pp. 2533–2550. IEEE (2023)

A Formal Definitions

A.1 Commitment Schemes

Definition 5. (Binding Property [16]) *A non-interactive commitment scheme* $(\mathsf{Gen}, \mathsf{Com})$ *is* computationally binding *if for all PPT adversaries* $\mathcal{A}$*, there is a negligible function* $\mu(\lambda)$ *such that*

$$\Pr\left[\begin{array}{l}\mathsf{Com}(\mathsf{ck}; m_0; r_0)\\ = \mathsf{Com}(\mathsf{ck}; m_1; r_1)\\ \wedge\ m_0 \neq m_1\end{array}\middle|\begin{array}{l}\mathsf{ck} \leftarrow \mathsf{Gen}(1^\lambda);\\ (m_0, m_1, r_0, r_1) \leftarrow \mathcal{A}(\mathsf{ck})\end{array}\right] = \mu(\lambda).$$

where $\mathcal{A}$ *outputs* $m_0, m_1 \in \mathcal{M}_c$ *and* $r_0, r_1 \in \mathcal{R}_c$*. The scheme is* ***perfectly binding*** *when* $\mu(\lambda) = 0$*.*

Definition 6. (Hiding Property [16]) *A non-interactive commitment scheme* $(\mathsf{Gen}, \mathsf{Com})$ *is* computationally hiding *if for all PPT adversaries* $\mathcal{A}$*, there is an negligible function* $\mu(\lambda)$ *such that*

$$\left|\Pr\left[\mathcal{A}(c) = b \middle|\begin{array}{l}\mathsf{ck} \leftarrow \mathsf{Gen}(1^\lambda);\\ (m_0, m_1) \leftarrow \mathcal{A}(\mathsf{ck}),\\ b \leftarrow \{0,1\}, r \leftarrow\!\!\$\ \mathcal{R}_c,\\ c \leftarrow \mathsf{Com}(\mathsf{ck}; m_b; r)\end{array}\right] - \frac{1}{2}\right| = \mu(\lambda),$$

where $\mathcal{A}$ *outputs* $m_0, m_1 \in \mathcal{M}$*. The scheme is* ***perfectly hiding*** *when* $\mu(\lambda) = 0$*.*

A.2 Sigma Protocols

Definition 7. (Perfect completeness [16]) *A* Σ*-protocol* $(\mathsf{Setup}, \mathsf{P}, \mathsf{V})$ *is perfectly complete if for all PPT adversaries* $\mathcal{A}$

$$\Pr\left[\mathcal{V}(\mathsf{ck}, x, t, c, z) = 1 \middle|\begin{array}{l}\mathsf{ck} \leftarrow \mathsf{Setup}\left(1^\lambda\right); (x, w) \leftarrow \mathcal{A}(\mathsf{ck});\\ t \leftarrow \mathsf{P}(\mathsf{ck}, x, w); c \leftarrow\!\!\$\ \mathcal{C}_s; z \leftarrow \mathsf{P}(c)\end{array}\right] = 1,$$

where $\mathcal{A}$ *outputs* $(x; w)$ *and* $(\mathsf{ck}, x; w) \in R$*.*

Definition 8 (Computational Knowledge Soundness). *A* Σ*-protocol* $(\mathsf{Setup}, \mathsf{P}, \mathsf{V})$ *for relation* $\mathcal{R}$ *provides soundness with soundness error* σ *if for all deterministic polynomial time* P^* *with success probability* ϵ*, there exists an expected polynomial time extractor* $\mathcal{E}$ *such that for all PPT adversaries* $\mathcal{A}$

$$\Pr\left[(x; w) \in \mathcal{R} \middle|\begin{array}{l}\mathsf{ck} \leftarrow \mathsf{Setup}(1^\lambda); (x; w) \leftarrow \mathcal{A}(\mathsf{ck}); t \leftarrow \mathsf{P}^*(\mathsf{ck}, x, w);\\ c \leftarrow\!\!\$\ \mathcal{C}_s; z \leftarrow \mathsf{P}(c); \mathsf{V}(\mathsf{ck}, x, t, c, z) = 1; w \leftarrow \mathcal{E}^{\mathsf{P}^*}(\mathsf{ck}, x)\end{array}\right] \geq \frac{\epsilon - \kappa(|x|)}{\mathsf{poly}(|x|)}.$$

where $\kappa(|x|)$ *is negligible soundness error dependent on the statement length* $|x|$*.*

To define the special soundness property, we first present a denotation of the tree of the transcript.

Definition 9 (Tree of transcript). *Let $\mu \in \mathbb{N}$ and $(k_1, ..., k_\mu) \in \mathbb{N}^\mu$. A $(k_1, ..., k_\mu)$-tree of transcripts constitutes a set of $\prod_{i=1}^{\mu} k_i$ transcripts of a tree-like structure. The edges within this tree represent the challenges of the verifier, while the vertices are the messages from the prover, which can be empty. Each node at depth i has exactly k_i child nodes, corresponding to k_i distinct challenges. Every transcript is uniquely represented by one path from the root node to a leaf node.*

Definition 10 ($(k_1, ..., k_\mu)$-special soundness). *Π provides $(k_1, ..., k_\mu)$-special soundness if there is an effective PPT extraction algorithm $\mathcal{E}$ that is capable of extracting the witness w given x and any $(k_1, \cdots, k_\mu)$-tree of accepting transcripts T [2] (defined above). Specifically, for all PPT adversaries $\mathcal{A}$*

$$\Pr\left[\, (x; w) \in \mathcal{R} \middle| \, \mathsf{ck} \leftarrow \mathsf{Setup}(1^\lambda), (x, T) \leftarrow \mathcal{A}(\mathsf{ck}), w \leftarrow \mathcal{E}(\mathsf{ck}, x, T) \,\right] \approx 1.$$

$(k_1, ..., k_\mu)$-special soundness is a generalization to the standard notion of special soundness. For example, a typical Σ-protocol given in [12] is a special type of the 3-move interactive proof satisfying k-special soundness, where P sends an initial message a, V issues with a random challenge $r \leftarrow\!\!\$\ \mathbb{F}$, and finally, P provides with a response z. Special soundness is typically easier to prove than the knowledge soundness property for an interactive proof. Although special soundness should be regarded as a weaker notion of knowledge soundness, [2] proves that $(k_1, ..., k_\mu)$-special-soundness tightly implies knowledge soundness as long as $K = \prod_{i=1}^{\mu} k_i$ is constant. Therefore, the protocols in this work are all arguments of knowledge under the DL assumption from the results of [2]. Note that most of the protocols mentioned in this paper only has 3-move, so it is sufficient to give the definition of k-special soundness. For generalization of the recursion techniques, e.g, the inner product arguments run logarithmic rounds, we present the $(k_1, \cdots, k_\mu)$-special soundness definition here.

Definition 11. (Special honest verifier zero-knowledge (SHVZK) [16]) $(\mathsf{Setup}, \mathsf{P}, \mathsf{V})$ *is SHVZK if there exists a PPT simulator $\mathcal{S}$ that can output an accepting transcript that is indistinguishable from real transcripts without knowing the witness when the verifier's challenge is known beforehand. Specifically, for all PPT adversaries $\mathcal{A}$,*

$$\Pr\left[\mathcal{A}(t, z) = 1 \middle| \begin{array}{l} \mathsf{ck} \leftarrow \mathsf{Setup}\left(1^\lambda\right); (x, w, c) \leftarrow \mathcal{A}(\mathsf{ck}); \\ t \leftarrow \mathcal{P}(\mathsf{ck}, x, w); z \leftarrow \mathcal{P}(c); \end{array}\right] \approx$$

$$\Pr\left[\mathcal{A}(t, z) = 1 \middle| \begin{array}{l} \mathsf{ck} \leftarrow \mathsf{Setup}\left(1^\lambda\right); (x, w, c) \leftarrow \mathcal{A}(\mathsf{ck}); \\ (t, z) \leftarrow \mathcal{S}(\mathsf{ck}, x, w); \end{array}\right].$$

where $\mathcal{A}$ outputs (x, w, c) such that $(\mathsf{ck}, x, w) \in R$.

B Security Proofs

B.1 Proof of Theorem 2

Completeness: In protocol Π_1, for any prover running his steps honestly, the verifier can finally obtain transcripts including $\{E_i\}_{i=3}^{2n-1}, x, m(x)$ and check the

following equation:

$$\mathsf{Com}(g(x); m(x)) = \sum_{i=1}^{n} A_i \cdot x^{2i} + \sum_{i=3}^{2n-1} E_i \cdot x^i.$$

If $\{A_i = \mathsf{Com}(ck; m_i)\}_{i=1}^{n}$ and $\{E_i\}_{i=3}^{2n-1}$ are computed correctly according to Figure 3, the above equation holds according to Equation (1).

Special Soundness: To prove the special soundness property, we need to show that there exists an efficient extractor algorithm that outputs the witness for the statement in Π_1 with oracle access to a malicious prover. By the "polynomial amortization trick" mentioned in [2], if a prover can open the commitment $\mathsf{Com}(g(x), m(x))$ combined from $\{A_i\}_{i=1}^{n}$ and $\{E_i\}_{i=3}^{2n-1}$, then with high probability he can open all the A_i's and E_i's. Thus, we only need to show that it is feasible to build an extraction algorithm for commitment $\mathsf{Com}(g(x), m(x))$ on input n transcripts under different challenges. Upon receiving n points for the polynomial $m(X)$, an equation can be built accordingly as follows:

$$\boldsymbol{m} \cdot M_x = \begin{bmatrix} m_1 \\ m_2 \\ \vdots \\ m_n \end{bmatrix}^T \cdot \begin{bmatrix} 1 & 1 & \cdots & 1 \\ x_1 & x_2 & \cdots & x_n \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{n-1} & x_2^{n-1} & \cdots & x_n^{n-1} \end{bmatrix} = \begin{bmatrix} m(x_1) \\ m(x_2) \\ \vdots \\ m(x_n) \end{bmatrix}^T = \boldsymbol{z}$$

where $\boldsymbol{m} = [m_1, m_2, ..., m_n]$ is the vector of all secret values, $\boldsymbol{z} = [m(x_1), m(x_2), ..., m(x_n)]$ is the vector of all n opening values and $x_1, ..., x_n$ are n challenge values in each query. The above matrix equation represents the relation between secret values and the opening values, where the coefficient matrix constructed from $x_1, ..., x_n$ is denoted as M_x. Since M_x is a Vandermonde matrix, there exists an efficient algorithm for computing the inverse of M_x in polynomial time. Therefore, we can extract the secret values by computing $\boldsymbol{m} = \boldsymbol{z} \cdot M_x^{-1}$. And we say protocol Π_1 satisfies the n-special soundness.

B.2 Proof of Theorem 3

Completeness: In protocol Π_2, for any prover running his own steps honestly, the verifier can finally obtain transcripts including $\{E_{i,j}\}_{i=3,j=1}^{2n-1,k}, x, \boldsymbol{m}(x)$ and check the following equation:

$$\mathsf{Com}(g(x); m_j(x)) = \sum_{i=1}^{n} A_{i,j} \cdot x^{2i} + \sum_{i=3}^{2n-1} E_{i,j} \cdot x^i$$

If $\{A_i = \sum_{j=1}^{k} \mathsf{Com}(ck_i; m_{i,j})\}_{i=1}^{n}$ and $\{E_{i,j}\}_{i=3,j=1}^{2n-1,k}$ are correctly computed, the above equation holds according to Equation (1). Meanwhile, since $\boldsymbol{m}(x)$ satisfies $f(\boldsymbol{m}(x))$, if $\{F_l\}_{l=0}^{kn}$ are correctly computed, the following equation also holds,

$$\mathsf{Com}(g(x); f(\boldsymbol{m}(x))) = \sum_{l=0}^{kn} F_l \cdot x^l.$$

Special Soundness: To prove the special soundness property, we need to show that there exists an efficient algorithm that extracts the witness $\boldsymbol{m}(X)$ for the statement in Π_2 given by a malicious prover. Here we can follow the similar process we used for Π_1. For each j, the extractor uses the rewinding lemma to the algorithm to query the malicious prover with different challenge values. Therefore, we rewind Π_2 for n times under different challenges to obtain n transcripts. We get n points for each polynomial $m_j(X)$ with a degree at most n. Accordingly, a Vandermonde matrix can be built, and an efficient algorithm exists for computing the inverse of the matrix in polynomial time. Therefore, we can extract the secret values $\boldsymbol{m}_j$ for each $m_j(X)$ based on n transcripts in polynomial time. As a result, the extractor can compute $\boldsymbol{m}(X)$ from n transcripts in polynomial time. Ensured by the completeness of protocol Π_2, the extractor either (1) computes a satisfying witness $m(X)$ for the constraint function $f(\cdot)$ or (2) discovers another polynomial $f'(X)$ equals to $f(\boldsymbol{m}(X)$ on the challenge x (3) discovers a non-trivial instance for $\mathsf{Com}(g(x), f(x)) = \sum_{l=0}^{kn} F_l \cdot x^l$. The probability of case (2) is negligible due to the Schwartz-Zippel lemma. The probability of case (3) is also negligible due to the binding property of the commitment scheme Com.