On the Security of Nova Recursive Proof System: Limitations of and Alternatives to Bounded-Depth Analysis

Hyeonbum Lee[1] and Jae Hong Seo[2⋆]

[1] Seoul National University, Seoul
hyeonbumlee@snu.ac.kr
[2] Hanyang University, Seoul
jaehongseo@hanyang.ac.kr

Abstract. Standard definitions of Knowledge Soundness for Incrementally Verifiable Computation (IVC) are formulated to accommodate rewinding based analysis, effectively limiting their security guarantees to bounded (logarithmic) recursion depths. We demonstrate that this limitation is critical: we construct an artificial IVC scheme that satisfies this standard, bounded-depth definition but becomes efficiently forgeable when the recursion extends to a polynomial scale. This finding exposes a theoretical gap in Nova (CRYPTO '22), leaving its intended unbounded usage theoretically unfounded.

We bridge this gap by establishing the Polynomial-depth Knowledge Soundness of Nova. Our analysis framework is anchored in the Extended Algebraic Group Model (EAGM). Specifically, the EAGM extends the algebraic requirement to intermediate values involved in algebraic operations, allowing the extractor to reconstruct the recursive witness directly from the representations of these internal operations to bypass the exponential overhead of rewinding. Leveraging this extraction capability, we crucially depart from standard Fiat-Shamir analyses that rely on the Random Oracle Model (ROM) to guarantee that the extracted witnesses satisfy the intended relations. While the ROM is theoretically uninstantiable and necessitates heuristic realizations, we instead enforce the validity of the extracted witnesses using the General Zero-Testing (GZT) property. This property replaces the random oracle with a targeted requirement offering a plausible pathway to standard-model instantiation, providing the first rigorous security foundation for recursive proof systems in unbounded use cases.

Keywords: IVC, folding scheme, algebraic group model, random oracle model, correlation intractiblity

1 Introduction

A recursive proof composition is key idea of succinct proof for the large computation. It is widely used in incrementally Verifiable Computation (IVC) [58]

⋆ Corresponding authors

and its generalization, Proof-Carrying Data (PCD) [26], that are cryptographic primitives facilitating the generation of proofs that convincingly prove the accurate execution of lengthy computations. These proofs enable efficient verification by a verifier for any prefix of the computation. IVC schemes find applications in diverse domains, such as verifiable delay functions (VDF) [7,39], succinct blockchains [12,27,11,38], and verifiable state machines [48].

VDF schemes are one of the key tools for Ethereum's consensus protocols, and several studies have incorporated the IVC scheme into VDF [39]. VDF involves recursive computation, and IVC enables efficient verification even when the computation is computationally expensive.

There are also IVC-based succinct blockchain projects [12,27,11]. The IVC scheme allows for avoiding the need to download the full history for verification. Using the current state with IVC proof, a node can verify the validity of the current state and all previous states. If the IVC scheme is applied to Ethereum, which has a market capitalization of approximately hundreds of billions of dollars and provides approximately 13.4 seconds for block generation times [29], it would require approximately 6,000 recursive computations per day. Therefore, the IVC scheme for these applications should provide an appropriate level of security for large recursive steps.

Although many proposals for IVC/PCD schemes [13,17,45,44] offer provable security, their knowledge soundness is proven only in a limited model with at most $O(\log \lambda)$ recursive rounds, where λ is the security parameter. This is because the common proof strategy applied in those proposals is to construct a recursive extractor that blows up polynomially for each additional recursive step. Thus, recursion can be performed only for $O(\log \lambda)$ rounds before the extractor's running time becomes super-polynomial in λ. In fact, there are PCD schemes achieving polynomially-long chains [26,5,22], but those require additional strong assumptions such as hardware tokens or are relatively impractical compared to practical constructions such as Nova [45], a new type of recursive proof system.

Nova uses a folding scheme as its core building block. This novel idea of folding relations can significantly reduce the recursion overhead. Nova's soundness proof follows the common proof strategy of using a general recursive technique, and thus is also proven in the aforementioned limited model with $O(\log \lambda)$ rounds. Therefore, Nova's soundness proof does not guarantee soundness for linear rounds in the security parameter, for example, 128 rounds for 128 bit security, which is too short to be used in various aforementioned applications. This limitation of the current IVC model has been mentioned in several literature [45,53]. Nevertheless, there are no known attacks on arbitrary depth recursion, leaving a gap between theoretical security guarantees and real-world attacks.

1.1 Our Contribution.

We summarize our contributions in five key aspects:

Forgeable-but-Knowledge Sound IVC Scheme. First, we identify a critical gap between the security guaranteed by the existing Knowledge Soundness (KS)

Security Property	**Model**	Nova [43]	Ephemeral-Nova (Fig. 2)
Log-depth KS (Def. 2)	Standard+ROM	✓	✓
Relaxed Poly-depth KS (Def. 5)	Standard+ROM	✓	✗
Poly-depth KS (Def. 4)	Standard+ROM	?	✗
	EAGM+GZT	✓	✗

EAGM: Extended Algebraic Group Model (Def. 8) GZT: General Zero Testing property (Def. 10)

Table 1. Comparison between Nova and Ephemeral-Nova.

definition, which typically ensures extractability only for (at most) logarithmic depths $O(\log \lambda)$, and the security required for polynomial-depth IVC applications. To substantiate this, we introduce a variant of Nova, called *Ephemeral-Nova*. This construction satisfies the existing KS definition (implying validity for logarithmic depths), yet becomes efficiently forgeable when the depth extends to a polynomial scale $\mathsf{poly}(\lambda)$. Consequently, Ephemeral-Nova demonstrates that the existing KS notion is insufficient to guarantee security for polynomial-round bounds, necessitating a stronger security definition.

New Knowledge Soundness Definitions. The construction of Ephemeral-Nova reveals that the existing KS definition yields false positives in polynomial-depth scenarios. The primary cause of this issue is the computational gap regarding depth: while generating an IVC proof for polynomial depth is efficient, constructing an extractor for such depths is computationally intractable under the existing definition. To address this, we propose two novel definitions for IVC knowledge soundness: *Relaxed Poly-depth Knowledge Soundness* and *Poly-depth Knowledge Soundness*. As previously noted, it remains an open question whether Nova satisfies full Poly-depth KS under standard assumptions with random oracle instantiation. However, by relaxing the extraction condition, we demonstrate that Nova satisfies our proposed Relaxed Poly-depth KS, whereas Ephemeral-Nova does not under standard assumption. We describe the comparison between Nova and Ephemeral-Nova in Table 1.

New Computational Model: Extended Algebraic Group Model. To bridge this gap and prove the Poly-depth Knowledge Soundness of Nova, we introduce a new analysis framework using the *Extended Algebraic Group Model (EAGM)*. We adopt this model to address the limitations of existing algebraic models: the Generic Group Model (GGM) is unsuitable for arithmetized group operations used in Nova, and the standard Algebraic Group Model (AGM)—while resolving the exponential rewinding overhead via straight-line extraction—is insufficient as it guarantees extraction only for the final output, failing to capture the full recursive chain. The EAGM overcomes these issues by enforcing that the adversary provides algebraic representations not only for the final result but also for all intermediate group elements *necessarily related to the output*.

To formalize this necessity, we define a predicate to verify structural validity, a projection to identify component elements, and a *Linear-Algebraic Binding*

(LAB) function to enforce binding. In this framework, if the adversary produces a valid output satisfying the predicate, the algebraic representations of the pre-images (identified by the projection) must explicitly exist in the output trace due to the binding property of LAB. This enables the recursive extraction of the full witness chain without heuristics or rewinding.

Relaxing Random Oracles via General Zero-Testing Property. Most efficient recursive proof systems rely on the Random Oracle Model (ROM) to instantiate the Fiat-Shamir transformation. While the ROM facilitates the construction of knowledge extractors via rewinding, it acts as a theoretical idealization that cannot be realized in practice. To relax these strong assumptions, Correlation Intractable (CI) hash functions have been proposed. CI hashes guarantee the intractability of finding pre-images that satisfy specific relations and can be instantiated for sparse and efficiently sampleable relations. However, applying them to algebraic protocols without rewinding is non-trivial; specifically, there is no known strategy to achieve straight-line extraction for our target algebraic relations in the standard model.

To address this gap, we introduce the *General Zero-Testing (GZT)* property. We position GZT as a realistic middle ground between the ROM and instantiable CI hash. First, GZT is a significantly relaxed condition compared to the ROM, requiring only that the hash avoids specific algebraic roots rather than exhibiting perfect uniformity. Second, and most crucially, GZT enables straight-line extraction within the AGM, thereby avoiding the exponential overhead associated with rewinding in recursive proofs.

Proof of Poly-depth Knowledge Soundness. Finally, building on the EAGM framework and GZT property, we present a security proof of Nova. We first prove that the NIFS satisfies knowledge soundness in the EAGM with GZT. Then, leveraging the knowledge propagation in the EAGM, we construct a straight-line extractor for the Nova IVC scheme. This extractor successfully recovers the entire chain of local inputs for polynomial execution steps, confirming the security of Nova in the proposed model.

1.2 Technical Overview

Designing Ephemeral-Nova: Inject Trapdoor in Augmented Function. Existing knowledge soundness definitions guarantee security only for logarithmic depths due to the exponential blow-up of extractor complexity. Leveraging this theoretical blind spot, we design Ephemeral-Nova, a scheme that is *polynomially-faulty-but-logarithmically-provable*. Our core idea is to inject a state-dependent *trigger* into Nova's augmented function F', which remains benign for a polynomially large number of steps t (e.g., $t = \mathsf{poly}(\lambda)$) but deterministically breaks soundness at round t.

To achieve this without disrupting standard security guarantees, we extend F' to support two modes controlled by an auxiliary bit $\mathsf{b} \in \{0, 1\}$: *Normal Mode*

($\mathsf{b} = 1$) and *Trapdoor Mode* ($\mathsf{b} = 0$). The state transition is defined as:

$$z_{i+1} := \left[\mathsf{b} + (1 - \mathsf{b})(i - t)^{p-1}\right] \cdot F(z_i, \omega_i) \pmod{p}$$

By Fermat's Little Theorem, for all recursion depths $i < t$, the term $(i-t)^{p-1} \equiv 1 \pmod{p}$. Consequently, the term in the brackets simplifies to 1 regardless of b, making the trapdoor mode *algebraically equivalent* to the normal execution ($z_{i+1} = F(z_i, \omega_i)$). This ensures that Ephemeral-Nova preserves standard knowledge soundness within logarithmic depths. However, precisely at round $i = t$, the term vanishes (becomes 0), allowing the prover to force $z_{t+1} = 0$ and forge valid proofs for any subsequent invalid transitions.

The existence of Ephemeral-Nova highlights a critical gap: (1) It demonstrates that a scheme can be proven secure under current definitions (log-depth) while being totally insecure at polynomial depths. (2) It confirms that standard extraction techniques are insufficient for detecting vulnerabilities that manifest only at deep recursion layers, necessitating a stronger notion of security for recursive arguments.

Refining Definition of Knowledge Soundness. The first issue is that the standard knowledge soundness (KS) definition is too loose for polynomial depths; it technically considers Ephemeral-Nova secure because the extractor is only required to work for logarithmic depths (where the trigger is inactive). However, demanding a standard extractor for polynomial depths is too strict due to the rewinding overhead.

To bridge this gap, we propose a novel definition: *Relaxed Poly-depth Knowledge Soundness*. Instead of demanding the extraction of the entire witness chain from scratch, this new definition requires that if an adversary provides a proof for polynomial depth m along with the local inputs up to depth $m - n$ (where n is logarithmic), the extractor must be able to recover the witnesses for the remaining n steps. Under this refined definition, we can formally distinguish security: Nova satisfies Relaxed Poly-depth KS, whereas Ephemeral-Nova does not.

New Computation Model: Extended Algebraic Group Model. While relaxed KS helps classify schemes, our ultimate goal is to prove that Nova is secure in the strongest sense—*Poly-depth Knowledge Soundness*—where efficient extraction is guaranteed. We construct the EAGM through the following logical flow, starting with the limitations of existing models:

– **Limitation of Current Models: The Dilemma of Recursion.** In recursive proofs, the extractor acts on a much deeper level: it must recover the *entire chain* of intermediate witnesses, not just the final result. Existing models fail to support this dual requirement of tracking and concreteness.
  - **GGM (Lack of Group Description):** The Generic Group Model allows tracking all operations via oracle queries, but it models group elements as abstract handles (e.g. random strings). This black-box abstraction is

unsuitable for recursive systems like Nova, which require **concrete group arithmetic** to perform arithmetization (i.e., embedding group operations into a constraint system).

- **AGM (Incapability of Tracking Composite Structure):** The Algebraic Group Model ensures the adversary knows the representation of the final output relative to the setup. However, it treats the output as a **monolithic element**, ignoring how it was constructed. Even though the adversary *must* possess the representations of intermediate elements to perform algebraic operations (e.g., computing $Z = A + B$), the AGM does not enforce the disclosure of these intermediate representations. This creates a gap where the step-by-step witness chain required for IVC is known to the adversary but inaccessible to the extractor.

Solution: We introduce the *Extended Algebraic Group Model (EAGM)* to resolve these dilemmas. It extends the AGM to capture local algebraic dependencies (tracking intermediate steps similar to GGM) while retaining the concrete group descriptions of the AGM.

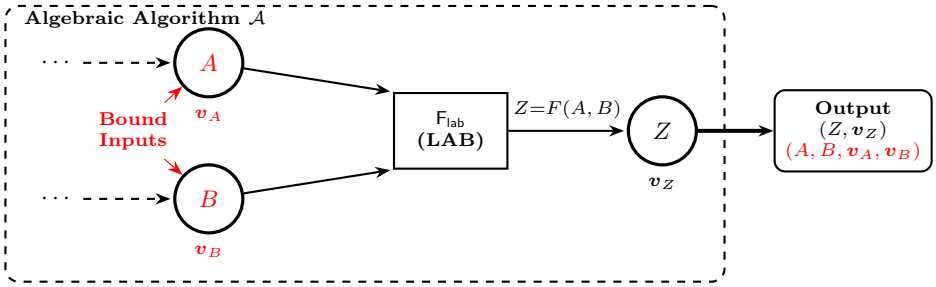
– **Defining Necessary Elements via LAB Function:** To formalize this, we define necessary elements based on the *Linear-Algebraic Binding (LAB)* function (Please refer Definition 6 for details). Specifically, if an output group element Z is the result of a LAB operation on inputs (A, B), then (A, B) are deemed algebraically necessary for Z.
Example: random linear combination. To visualize this, consider a function $\mathsf{F}_{\mathsf{lab}} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}$ defined as $\mathsf{F}_{\mathsf{lab}}(A, B) = A + \mathsf{H}(A, B) \cdot B$, where H is a random oracle. A collision $Z = \mathsf{F}_{\mathsf{lab}}(A, B) = \mathsf{F}_{\mathsf{lab}}(A', B')$ implies a linear relation:

$$A + c \cdot B = A' + c' \cdot B' \quad \Longleftrightarrow \quad c' = (A - A' + c \cdot B) \cdot (B')^{-1},$$

where $c = \mathsf{H}(A, B)$ and $c' = \mathsf{H}(A', B')$. Since c' is uniformly distributed over $\mathbb{Z}_p$, the probability that a randomly generated c' satisfies this equality for any fixed (A, B, c) is $1/p$, which is negligible. Thus, the binding property holds: an adversary cannot produce Z from different pre-images.

– **Enforcing Representations via Knowledge Propagation:** This binding property enforces a strict algebraic relationship: a valid representation of the output Z can only be constructed from the representations of its necessary inputs (A, B). This establishes the principle of *Knowledge Propagation*, possessing a valid representation for Z algebraically forces the adversary to also possess (and thus reveal) the representations of A and B. We illustrate this process in Figure 1.

– **Formalizing EAGM with Projection:** To apply knowledge propagation, we must capture the execution of the LAB function $\mathsf{F}_{\mathsf{lab}}$. Since monitoring the entire execution process of an algebraic algorithm is impractical, we utilize a projection Π that inspects the algorithm's *outputs*. Specifically, if the output satisfies *particular algebraic conditions* (e.g., including a valid R1CS witness for the operation), Π detects this to verify the execution of $\mathsf{F}_{\mathsf{lab}}$ and extracts its constituent inputs directly from the output structure. *In essence,* Π *serves to identify precisely which input representations are mandated by the structure*



The inputs A, B are structurally Bound by the Linear Algebraic Binding (LAB) function $\mathsf{F}_{\mathsf{lab}}$. Consequently, the extended model mandates the algorithm to output these inputs and their representations $(A, B, \boldsymbol{v}_A, \boldsymbol{v}_B)$ alongside the standard output $(Z, \boldsymbol{v}_Z)$.

Fig. 1. Illustration of Knowledge Propagation.

of the algorithm's output. Under the EAGM, this ensures that the algorithm explicitly provides these identified representations of the LAB function's inputs. Leveraging this capability, we can recursively retrieve the full witness chain for Nova proofs from step n down to the base case in a straight-line manner without rewinding.

Design Methodology: General Zero-Testing Hash. To realize straight-line extraction within the (Extended) AGM, we require a concrete hash property that facilitates algebraic extraction without relying on the rewinding capabilities of the Random Oracle Model (ROM). Our design methodology focuses on identifying the *minimal algebraic requirement* sufficient for straight-line extraction.

- **Abstraction: Zero-Testing as Specialized Correlation Intractability.** In algebraic protocols, security typically relies on the challenge γ being generic enough to preserve linear independence. A challenge is considered bad only if it satisfies a specific algebraic relation that allows a forgery to pass verification. This aligns with the notion of *Correlation Intractability (CI)*, where it should be hard to find an input whose hash output satisfies a specific relation. Methodologically, we characterize the set of bad challenges as the *roots of a polynomial* determined by the hash input. Thus, we formulate the *zero-testing* property as a specialized instance of CI: the hash output γ must not be a root of the non-zero polynomial associated with the input statement.
- **Generalization: Bridging the Gap for Hidden Representations.** However, standard CI is insufficient for algebraic protocols in the AGM. CI defines the relation based solely on the *explicit hash input* (e.g., a public group element), whereas the critical algebraic relation often depends on the adversary's *hidden internal representation* (the coefficients used to construct that element).
  Take Schnorr's NIZK for a relation $\{(G, H; x) : H = x \cdot G\}$ as a concrete example. The hash function inputs three group elements G, H, K, where K is generated by the prover. In the AGM, the adversary constructs K as a linear

combination $K = k_1G + k_2H$. The bad challenge that prevents extraction is the root of the polynomial $P(X) = k_2 + X$.

- **The Gap:** The hash function sees only the group element K, but the polynomial relation ($k_2 + \gamma = 0$) depends on the coefficient k_2, which is hidden from the public view.
- **Our Approach:** To address this, we extend the CI notion to consider the *representation* of the input. We require the hash to be intractable with respect to relations defined over the coefficients (k_1, k_2) rather than just the group element (K).

– **Enabling Straight-Line Extraction.** This design directly facilitates straight-line extraction within the (Extended) AGM. By guaranteeing that the hash output γ is not a root of the verification polynomial (e.g., $k_2+\gamma \neq 0$), we ensure that the critical algebraic terms are mathematically invertible. Consequently, the extractor can recover the witness components directly from the algebraic adversary's group representations (e.g., computing $x = (s - k_1) \cdot (k_2 + \gamma)^{-1}$) without the need for rewinding.

Poly-depth Knowledge Soundness in EAGM with GZT hash. Our ultimate goal is to prove that Nova is poly-depth knowledge soundness without relying on the ROM. We achieve this by combining our new tools: GZT enables efficient extraction and binding for a single step, while EAGM propagates this capability across the entire recursive chain. Our proof strategy proceeds in two main phases.

– **Phase 1: GZT for NIFS and Binding.** First, we analyze the underlying Non-Interactive Folding Scheme (NIFS). The **GZT property** plays a dual role, simultaneously establishing the soundness of the proof and the binding of the folding operation.

- **Role 1: NIFS Knowledge Soundness:** The Knowledge Soundness of the IVC scheme fundamentally relies on that of the underlying NIFS. To establish this in the AGM, we configure the GZT property to ensure that the challenge avoids the roots of the verification polynomial. This guarantees that satisfying the verification check strictly implies the validity of the folded witness relation, enabling straight-line extraction.
- **Role 2: Binding as a LAB Function:** To enable knowledge propagation in the EAGM, we leverage the NIFS verifier $\mathsf{NIFS.V}$, which is arithmetized within the Nova circuit, as a LAB function. To ensure the LAB property, the underlying GZT hash guarantees the intractability of finding collisions for the specific linear algebraic relations defined by the folding operation.

– **Phase 2: Recursive Extraction via Embedded LAB Logic.** Next, we lift these properties to the full Nova IVC system using the **EAGM**. The core insight is that the recursive step is secured by the structure of the verifier logic embedded within the circuit.

The extraction logic relies on identifying the $\mathsf{NIFS.V}$ as a LAB Function:

1. **The Embedded Verifier is the LAB Function:** The IVC verifier checks a proof for the augmented function F'. Inside F', $\mathsf{NIFS.V}$ is executed to update the running instance. Since this verifier computes a linear

Work	Primitive	Security Model	RO Hash
Chen et al. [22]	Micali's SNARK	Arithmetized ROM	Yes
Campanelli et al. [19]	SLE SNARK	AGM + Oracle	Yes
Maller et al. [49]	Multi Folding Scheme [44]	osROM	Yes
Ours	Folding Scheme [45]	**EAGM**	No (GZT)

Primitive denotes the base proof system used in the recursive construction. **RO Hash** indicates whether the security analysis relies on the Random Oracle Model.

Table 2. Comparison of IVC schemes targeting poly-depth Knowledge Soundness.

combination using a GZT-compliant hash (as established in Phase 1), the EAGM classifies the Folding Verifier itself as a LAB function.

2. **Triggering Knowledge Propagation:** The EAGM projection detects and extracts the execution of this LAB function within the trace. Because the function is binding (guaranteed by GZT), the output instance U_n is algebraically tied to the previous instance U_{n-1} and the fresh instance u_{n-1}. This compels the adversary to possess the representations of these inputs.
3. **Recursive Unrolling:** With the representations of $(\mathsf{U}_{n-1}, \mathsf{u}_{n-1})$ extracted, we invoke the NIFS extraction (Role 1 of GZT) to recover the witness v_{n-1}. Since this witness is valid for the previous step, it triggers the LAB logic again for step $n-1$. By inductively applying this logic, we unwind the recursion from step n down to 1 in a straight line.

1.3 Related Works

IVC from recursive proof composition. A well-known approach for IVC is to recursively utilize SNARKs [32,33] for arithmetic circuits. In this canonical approach [3], at each incremental step i, the prover generates a SNARK proving the correct execution of F and attesting that the SNARK verifier has accepted the proof for step $i-1$. However, these SNARK-based approaches often require a cycle of pairing-friendly elliptic curves and a trusted setup. To mitigate these overheads, alternative approaches utilizing *accumulation schemes* or *folding schemes* (often built upon NARKs) [13,17] have been proposed. These methods improve efficiency by deferring the expensive verification circuit to the very end of the recursion.

Poly-depth Knowledge Soundness in IVC. Folding-based IVC schemes, most notably Nova [43], have achieved remarkable efficiency. However, proving knowledge soundness for these recursive protocols in the standard model typically necessitates *rewinding* techniques. This results in an extraction time complexity that grows exponentially with the recursion depth, creating a significant theoretical gap for deep computations [6,23]. To achieve knowledge soundness with *polynomial-time* extraction, recent works have adopted idealized models that support *Straight-Line Extraction (SLE)*.

Chen et al. [22] pioneered this direction by introducing the *Arithmetized Random Oracle Model (AROM)*. Unlike the standard ROM, the AROM captures the algebraic structure of oracle queries, enabling the construction of SNARKs (based on Micali's CS proofs) with SLE capability. However, instantiating the AROM requires heuristically replacing the random oracle with a concrete hash function and, crucially, *embedding its arithmetic circuit* into the proof system.

Building on this, Campanelli et al. [19] introduced a general methodology for composing primitives secure in the AGM. They identified that standard AGM extraction fails in recursive settings because the extractor's output may lack the necessary algebraic representations for the subsequent step [19]. By enforcing specific oracle queries, they constructed Relativized-SNARKs that maintain SLE capability, thereby achieving unbounded-depth IVC in the AGM+Oracle model.

Concurrently, Maller et al. [49] proposed the Open-and-Sign Random Oracle Model (osROM). The osROM explicitly captures the extractability of commitments via the oracle, enabling straight-line extraction without rewinding [49]. Leveraging this model, they formalized HyperNova and other folding schemes as Non-Interactive Reductions of Knowledge (NIROKs) and proved the security of IVC over cycles of elliptic curves for polynomial recursion depths. However, it is noted that realizing the osROM properties typically relies on trusted hardware assumptions [26].

Our work aligns with these approaches by employing the Extended Algebraic Group Model (EAGM) to ensure polynomial-depth knowledge soundness. Crucially, unlike previous works, our construction does not rely on random oracles for security analysis. We provide a detailed comparison of these approaches in Table 2.

Correlation Intractability. The notion of *Correlation Intractability (CI)* was introduced by Canetti et al. to address the fundamental limitations of the ROM, specifically its potential unsoundness and the impossibility of realizing random oracles in the standard model [21]. Formally, a hash family is said to be CI for a relation $\mathcal{R}$ if it is computationally hard to find an input x such that $(x, H(x)) \in \mathcal{R}$ [21]. While Canetti et al. proved that CI hash functions for *all* evasive relations do not exist, subsequent works have demonstrated their existence for restricted classes of relations, such as efficiently sampleable relations [36] and searchable relations [20]. However, utilizing CI in Arguments of Knowledge (AoK) is non-trivial. In the standard model, the lack of programmability rules out rewinding techniques, necessitating straight-line extraction. CI definitions, however, provide only soundness guarantees (difficulty of finding a correlation) and do not facilitate such witness extraction.

Security Analysis of Nova Implementation. Nguyen et al. [53] investigated security vulnerabilities within the concrete *implementation* of Nova using a *2-cycle of elliptic curves* to achieve non-native field arithmetic efficiency. They point out soundness vulnerability in the original implementation of the 2-cycle.

In contrast, our work highlights a *theoretical* limitation that is orthogonal to the specific choice of mathematical groups, such as elliptic curves. The vulnera-

bility demonstrated by Ephemeral-Nova arises from the definition of knowledge soundness regarding *linear recursion depth* in the standard model. Consequently, even if the 2-cycle implementation is perfectly patched, the theoretical gap we identify persists.

1.4 Organization

Section 2 provides an overview of the Nova IVC framework and its underlying folding scheme, which serves as the core building block. In Section 3, we introduce *Ephemeral-Nova*, a new IVC scheme that satisfies knowledge soundness for logarithmically bounded rounds but becomes forgeable at linear recursion depths. In Section 4, we review existing idealized models for group-based systems and propose a novel framework, the Extended Algebraic Group Model (EAGM). Section 5 details the instantiation of the Linear-Algebraic Binding (LAB) property using General Zero-Testing (GZT) hash functions. Subsequently, in Section 6, we present a formal knowledge soundness proof for Nova within the EAGM.

2 Preliminaries

Notation. We first define the notations used in this paper. $[m]$ denotes the set of the integers from 1 to m, i.e., $[m] := \{1, \cdots, m\}$.

Let $\mathbb{Z}_p$ be the ring of integers modulo p. Uniform sampling is denoted by $\xleftarrow{\$}$. For instance, $a \xleftarrow{\$} \mathbb{Z}_p$ indicates that a is uniformly chosen from $\mathbb{Z}_p$.

We use bold font to represent vectors such as $\boldsymbol{a}$. For two vectors $\boldsymbol{a} = (a_1, \dots a_\ell), \boldsymbol{b} = (b_1, \dots, b_\ell) \in \mathbb{Z}_p^\ell$, we define three binary operations: concatenation $\boldsymbol{a} \| \boldsymbol{b} = (a_1, \dots, a_\ell, b_1, \dots, b_\ell)$, Hadamard product $\boldsymbol{a} \circ \boldsymbol{b} = (a_1 b_1, \dots, a_\ell b_\ell)$, and inner product $\langle \boldsymbol{a}, \boldsymbol{b} \rangle = \sum_{i=1}^{\ell} a_i b_i$.

The symbol H denotes the cryptographic hash function whose range will be specified in the context.

Definition 1 (Commitment Scheme). *A commitment scheme is defined by two PPT algorithms: the key generation algorithm* Gen *and commitment algorithm* Com*. Let* M*,* R*, and* C *be message space, random space, and commitment space, respectively.* Gen *and* Com *are defined by:*

- $\mathsf{Gen}(1^\lambda, \ell) \to \mathsf{ck}$ *: On the input security parameter* λ *and dimension of message space* ℓ*, sample commitment key* ck
- $\mathsf{Com}(\mathsf{ck}, m; r) \to C$ *: Take commitment key* ck*, message* $m \in \mathsf{M}^\ell$*, and randomness* $r \in \mathsf{R}$*, output commitment* $C \in \mathsf{C}$

We call $(\mathsf{Gen}, \mathsf{Com})$ *a commitment scheme if the following two properties hold:*
[Binding]: For any expected PPT adversary $\mathcal{A}$*,*

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

[Hiding]: For any expected PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$

$$\left| \Pr\left[\mathsf{b} = \mathsf{b}' \;\middle|\; \begin{array}{c} \mathsf{ck} \leftarrow \mathsf{Gen}(1^\lambda, \ell), \\ (m_0, m_1, \mathsf{state}) \leftarrow \mathcal{A}_1(\mathsf{ck}), \\ \mathsf{b} \xleftarrow{\$} \{0,1\}, r \xleftarrow{\$} \mathcal{R}, C \leftarrow \mathsf{Com}(\mathsf{ck}, m_\mathsf{b}; r), \\ \mathsf{b}' \leftarrow \mathcal{A}_2(\mathsf{ck}, C, \mathsf{state}), \end{array} \right] - \tfrac{1}{2} \right| \leq \mathsf{negl}(\lambda)$$

Let M, R, *and* C *be efficiently computable (additive) groups. Then, we call a commitment scheme* $(\mathsf{Gen}, \mathsf{Com})$ homomorphic *if the* $(\mathsf{Gen}, \mathsf{Com})$ *satisfying the following homomorphic property.*
[Additive Homomorphic]: For any commitment key $\mathsf{ck} \leftarrow \mathsf{Gen}(1^\lambda, N)$ *and pairs of message-randomness* $(m_0, r_0), (m_1, r_1) \in \mathsf{M} \times \mathsf{R}$*, the following equation holds:*

$$\mathsf{Com}(\mathsf{ck}, m_0; r_0) + \mathsf{Com}(\mathsf{ck}, m_1; r_1) = \mathsf{Com}(\mathsf{ck}, m_0 + m_1; r_0 + r_1)$$

Definition 2 (IVC). *An incrementally verifiable computation (IVC) scheme for a function family* $\mathcal{F}$ *consists of four PPT algorithms* $(\mathcal{G}, \mathcal{K}, \mathcal{P}, \mathcal{V})$ *defined as follows:*

- $\mathcal{G}(1^\lambda) \to \mathsf{pp}$*: On input security parameter* λ*, samples public parameters* pp*.*
- $\mathcal{K}(\mathsf{pp}, F) \to (\mathsf{pk}, \mathsf{vk})$*: On input* pp *and a target function* $F \in \mathcal{F}$*, outputs a prover key* pk *and a verification key* vk*.*
- $\mathcal{P}(\mathsf{pk}, i, z_0, z_{i-1}, \omega_{i-1}, \Pi_{i-1}) \to z_i, \Pi_i$*: On input prover key* pk*, current step index* i*, initial input* z_0*, previous state* z_{i-1}*, witness* ω_{i-1}*, and previous proof* Π_{i-1}*, outputs a new IVC proof* Π_i*.*
- $\mathcal{V}(\mathsf{vk}, i, z_0, z_i, \Pi_i) \to 0/1$*: On input verification key* vk*, index* i*, initial input* z_0*, claimed output* z_i*, and proof* Π_i*, outputs reject* (0) *or accept* (1)*.*

We say that an IVC scheme $(\mathcal{G}, \mathcal{K}, \mathcal{P}, \mathcal{V})$ *satisfies perfect completeness if for any PPT adversary* $\mathcal{A}$

$$\Pr\left[\mathcal{V}(\mathsf{vk}, (i, z_0, z_i), \Pi_i) = 1 \;\middle|\; \begin{array}{r} \mathsf{pp} \leftarrow \mathcal{G}(1^\lambda), \\ F, (i, z_0, z_{i-1}, \omega_{i-1}, \Pi_{i-1}) \leftarrow \mathcal{A}(\mathsf{pp}), \\ (\mathsf{pk}, \mathsf{vk}) \leftarrow \mathcal{K}(\mathsf{pp}, F), \\ \mathcal{V}(\mathsf{vk}, (i-1, z_0, z_{i-1}), \Pi_{i-1}) = 1, \\ z_i, \Pi_i \leftarrow \mathcal{P}(\mathsf{pk}, (i, z_0, z_{i-1}), \omega_{i-1}, \Pi_{i-1}) \end{array} \right] = 1$$

where F *is a polynomially efficient computable function. We say that an IVC scheme satisfies knowledge-soundness if for any constant* n*, and expected polynomial time adversaries* $\mathcal{P}^*$*, there exists expected polynomial-time extractor* $\mathcal{E}$ *such that for any input randomness* ρ

$$\Pr\left[\begin{array}{c} z_n \neq z, \\ \text{where } z_i \leftarrow F(z_{i-1}, \omega_{i-1}) \\ \forall i \in [n], \\ \wedge \mathcal{V}(\mathsf{vk}, (n, z_0, z), \Pi) = 1 \end{array} \;\middle|\; \begin{array}{r} \mathsf{pp} \leftarrow \mathcal{G}(1^\lambda), \\ F, (z_0, z, \Pi) \leftarrow \mathcal{P}^*(\mathsf{pp}; \rho), \\ (\mathsf{pk}, \mathsf{vk}) \leftarrow \mathcal{K}(\mathsf{pp}, F), \\ (\omega_i)_{i=0}^{n-1} \leftarrow \mathcal{E}^{\mathcal{P}^*}(\mathsf{pp}, z_0, z; \rho) \end{array} \right] \leq \mathsf{negl}(\lambda) \quad (1)$$

Finally, we say that an IVC scheme satisfies succinctness if the size of the IVC proof Π *is independent from the number of applications* n*.*

As mentioned in [45], IVC based recursive techniques [45,13,25,17,9,43,44,16,53] can cover at most logarithmically large n, i.e., $n = O(\log \lambda)$. For a polynomial large n, e.g. $n = \mathsf{poly}(\lambda)$, the IVC schemes cannot provide PPT extractor $\mathcal{E}$ for knowledge soundness because of exponential blow-up. To emphasize depth limitation, we call the knowledge soundness in Definition 2 *log-depth knowledge soundness*.

Committed Relaxed R1CS. Committed Relaxed R1CS (CR-R1CS) is a variant of the R1CS constraint system, widely employed in recursive proof systems [56,18,24,17]. Specifically, CR-R1CS is a relation defined over a homomorphic commitment scheme $\mathcal{C} = (\mathsf{Gen}, \mathsf{Com})$ with message space $\mathsf{M} = \mathbb{Z}_p^k$ (where k depends on the vector length), randomness space R, and commitment space C.

Let $\mathsf{pp} = (\mathsf{ck}_{\boldsymbol{e}}, \mathsf{ck}_{\boldsymbol{w}})$ be the commitment keys, where $\mathsf{ck}_{\boldsymbol{e}} \leftarrow \mathsf{Gen}(1^\lambda, m)$ and $\mathsf{ck}_{\boldsymbol{w}} \leftarrow \mathsf{Gen}(1^\lambda, m - \ell - 1)$. Given an R1CS structure $\mathsf{s} = (A, B, C) \in (\mathbb{Z}_p^{m \times m})^3$, the CR-R1CS relation is defined as:

$$\mathcal{R}_{\mathsf{pp},\mathsf{s}} = \left\{ \Big((E, W, s, \mathsf{x}); (\boldsymbol{e}, r_{\boldsymbol{e}}, \boldsymbol{w}, r_{\boldsymbol{w}}) \Big) \ : \begin{array}{l} E = \mathsf{Com}(\mathsf{ck}_{\boldsymbol{e}}, \boldsymbol{e}; r_{\boldsymbol{e}}), \\ W = \mathsf{Com}(\mathsf{ck}_{\boldsymbol{w}}, \boldsymbol{w}; r_{\boldsymbol{w}}), \\ A\boldsymbol{z} \circ B\boldsymbol{z} = sC\boldsymbol{z} + \boldsymbol{e}, \\ \text{where } \boldsymbol{z} = \boldsymbol{w} \parallel \mathsf{x} \parallel s \end{array} \right\}. \tag{2}$$

Here, x denotes the public inputs and outputs. The instance (E, W, s, x) and witness $(\boldsymbol{e}, r_{\boldsymbol{e}}, \boldsymbol{w}, r_{\boldsymbol{w}})$ belong to $\mathsf{C} \times \mathsf{C} \times \mathbb{Z}_p \times \mathbb{Z}_p^\ell$ and $\mathbb{Z}_p^m \times \mathsf{R} \times \mathbb{Z}_p^{m-\ell-1} \times \mathsf{R}$, respectively.

Note that if we impose the conditions $\boldsymbol{e} = \mathbf{0}$ and $s = 1$, the relaxed constraint $A\boldsymbol{z} \circ B\boldsymbol{z} = sC\boldsymbol{z} + \boldsymbol{e}$ reduces to the standard R1CS constraint $A\boldsymbol{z} \circ B\boldsymbol{z} = C\boldsymbol{z}$ (assuming the last element of $\boldsymbol{z}$ corresponds to the constant 1). To distinguish instances that satisfy the original R1CS constraints, we define an *ordinary R1CS instance-witness pair*. A CR-R1CS pair $(\mathsf{u}, \mathsf{v}) \in \mathcal{R}_{\mathsf{pp},\mathsf{s}}$ is called ordinary if $\mathsf{u}.E = C_{\mathbf{0}}$ and $\mathsf{u}.s = 1$, where $C_{\mathbf{0}} := \mathsf{Com}(\mathsf{ck}_{\boldsymbol{e}}, \mathbf{0}; r_0)$ is a commitment to the zero vector using a publicly known canonical randomness $r_0 \in \mathsf{R}$ (e.g., $r_0 = 0$). Due to the binding property of Com, the condition $\mathsf{u}.E = C_{\mathbf{0}}$ implies that the underlying error vector is $\boldsymbol{e} = \mathbf{0}$.

Definition 3 (Non-Interactive Folding Scheme). *Consider a relation $\mathcal{R}$ over public parameters, structure, instance, and witness tuples. A non-interactive folding scheme* NIFS *for $\mathcal{R}$ consists of four PPT algorithms* $(\mathsf{G}, \mathsf{K}, \mathsf{P}, \mathsf{V})$ *as follows:*

- $\mathsf{G}(1^\lambda, N) \to \mathsf{pp}_{\mathsf{FS}}$*: On input security parameter λ and the maximum size of common structure N, samples public parameters* $\mathsf{pp}_{\mathsf{FS}}$
- $\mathsf{K}(\mathsf{pp}_{\mathsf{FS}}, \mathsf{s}) \to \mathsf{pk}_{\mathsf{FS}}$*: On input* $\mathsf{pp}_{\mathsf{FS}}$ *and a common structure* s*, of size N between instances to be folded, outputs a prover key* $\mathsf{pk}_{\mathsf{FS}}$*.*
- $\mathsf{P}(\mathsf{pk}_{\mathsf{FS}}, (\mathsf{u}_1, \mathsf{v}_1), (\mathsf{u}_2, \mathsf{v}_2)) \to (\mathsf{u}, \mathsf{v}, \Pi)$*: On input two instance-witness pairs* $(\mathsf{u}_1, \mathsf{v}_1)$ *and* $(\mathsf{u}_2, \mathsf{v}_2)$*, outputs a new witness* v *of the same size and folding proof Π to allow the verifier to update new instance.*

– $\mathsf{V}(\mathsf{pp}_{\mathsf{FS}}, \mathsf{u}_1, \mathsf{u}_2, \Pi) \to \mathsf{u}$*: On input two instances* u_1 *and* u_2*, outputs a new instance* u*.*

A folding scheme for $\mathcal{R}$ *satisfies the following requirements.*
1. Perfect Completeness: For all PPT adversaries $\mathcal{A}$*, we have that*

$$\Pr\left[(\mathsf{pp}_{\mathsf{FS}}, \mathsf{s}, \mathsf{u}, \mathsf{v}) \in \mathcal{R} \;\middle|\; \begin{array}{r} \mathsf{pp}_{\mathsf{FS}} \leftarrow \mathsf{G}(1^\lambda, N), \\ (\mathsf{s}, (\mathsf{u}_1, \mathsf{u}_2), (\mathsf{v}_1, \mathsf{v}_2)) \leftarrow \mathcal{A}(\mathsf{pp}_{\mathsf{FS}}), \\ (\mathsf{pp}_{\mathsf{FS}}, \mathsf{s}, \mathsf{u}_1, \mathsf{v}_1), (\mathsf{pp}_{\mathsf{FS}}, \mathsf{s}, \mathsf{u}_2, \mathsf{v}_2) \in \mathcal{R}, \\ \mathsf{pk}_{\mathsf{FS}} \leftarrow \mathsf{K}(\mathsf{pp}_{\mathsf{FS}}, \mathsf{s}), \\ (\mathsf{u}, \mathsf{v}, \Pi) \leftarrow \mathsf{P}(\mathsf{pk}_{\mathsf{FS}}, (\mathsf{u}_1, \mathsf{v}_1), (\mathsf{u}_2, \mathsf{v}_2)), \\ \mathsf{u} \leftarrow \mathsf{V}(\mathsf{pp}_{\mathsf{FS}}, \mathsf{u}_1, \mathsf{u}_2, \Pi) \end{array}\right] = 1.$$

2. Knowledge Soundness : For any expected PPT adversary $\tilde{\mathcal{A}} = (\mathcal{A}, \mathsf{P}^*)$*, there is an expected polynomial-time extractor* $\mathcal{E}$ *such that over all randomness* ρ

$$\Pr\left[\begin{array}{c} (\mathsf{pp}_{\mathsf{FS}}, \mathsf{s}, \mathsf{u}_1, \mathsf{v}_1) \in \mathcal{R}, \\ (\mathsf{pp}_{\mathsf{FS}}, \mathsf{s}, \mathsf{u}_2, \mathsf{v}_2) \in \mathcal{R} \end{array} \;\middle|\; \begin{array}{r} \mathsf{pp}_{\mathsf{FS}} \leftarrow \mathsf{G}(1^\lambda, N), \\ (\mathsf{s}, (\mathsf{u}_1, \mathsf{u}_2), \mathsf{st}) \leftarrow \mathcal{A}(\mathsf{pp}_{\mathsf{FS}}), \\ \mathsf{pk}_{\mathsf{FS}} \leftarrow \mathsf{K}(\mathsf{pp}_{\mathsf{FS}}, \mathsf{s}), \\ (\mathsf{v}_1, \mathsf{v}_2) \leftarrow \mathcal{E}^{\mathsf{P}^*(\mathsf{pk}_{\mathsf{FS}}, \cdot)}(\mathsf{pp}_{\mathsf{FS}}, \mathsf{u}_1, \mathsf{u}_2, \mathsf{st}) \end{array}\right] \approx^{\mathrm{c}}$$
$$\Pr\left[\begin{array}{c} \mathsf{u} = \mathsf{V}(\mathsf{pp}_{\mathsf{FS}}, \mathsf{u}_1, \mathsf{u}_2, \Pi) \\ (\mathsf{pp}_{\mathsf{FS}}, \mathsf{s}, \mathsf{u}, \mathsf{v}) \in \mathcal{R} \end{array} \;\middle|\; \begin{array}{r} \mathsf{pp}_{\mathsf{FS}} \leftarrow \mathsf{G}(1^\lambda), \\ (\mathsf{s}, (\mathsf{u}_1, \mathsf{u}_2), \mathsf{st}) \leftarrow \mathcal{A}(\mathsf{pp}_{\mathsf{FS}}), \\ \mathsf{pk}_{\mathsf{FS}} \leftarrow \mathsf{K}(\mathsf{pp}_{\mathsf{FS}}, \mathsf{s}), \\ (\mathsf{u}, \mathsf{v}, \Pi) \leftarrow \mathsf{P}^*(\mathsf{pk}_{\mathsf{FS}}, \mathsf{u}_1, \mathsf{u}_2, \mathsf{st}), \end{array}\right]$$

Nova: IVC from Folding Scheme. Given a function F, an IVC scheme iteratively invokes the computation of F for each round. Nova [45] is an IVC scheme built from a folding scheme such that the computation in each round is an augmented function F' that not only invokes F but also folds two CR-R1CS instances, where F' is represented by the CR-R1CS. We deferred details of Nova IVC scheme to Appendix B.

Theorem 1 (Nova-IVC [45]). *If the non-interactive folding scheme* NIFS *satisfies perfect completeness and knowledge soundness, then Nova in the Figure 7 is a log-bounded round IVC scheme satisfying perfect completeness and knowledge soundness in Definition 2.*

3 Ephemeral-Nova: Forgeable-but-Knowledge Sound IVC

In this section, we formally construct Ephemeral-Nova, a variant of Nova that satisfies the bounded-depth knowledge soundness (Definition 2) yet becomes forgeable when the recursion depth exceeds a linear bound t. This counterexample substantiates that satisfying log-bounded knowledge soundness does not imply security for linear recursion depths.

Design Rationale. To construct a *polynomially-faulty-but-logarithmically-provable* scheme, Ephemeral-Nova is designed to satisfy three key properties:

1. **Log-depth Equivalence:** For any recursion depth $i < t$, the verification logic must be algebraically equivalent to that of the original Nova, enforcing the correct state transition $z_{i+1} = F(z_i, \omega_i)$.
2. **Polynomial Trigger:** At a precise threshold $i = t$ (where $t = \mathsf{poly}(\lambda)$), the verification mechanism must deterministically permit a state reset, enabling proof forgery.
3. **R1CS Compatibility:** The trigger mechanism must be deterministically expressible within the CR-R1CS constraints.

Construction of Ephemeral-Nova. We modify the original augmented function F' (from Figure 7) to include a conditional trapdoor. The new function, denoted as F'_{tr}, takes an additional auxiliary input bit $\mathsf{b} \in \{0,1\}$ and enforces the following state transition:

$$z_{i+1} := \left[\mathsf{b} + (1-\mathsf{b})(i-t)^{p-1}\right] \cdot F(z_i, \omega_i) \pmod p \tag{3}$$

where t is a polynomially bounded integer. This equation encapsulates two operating modes:

- **Algebraic Equivalence ($i < t$):** By Fermat's Little Theorem, $(i-t)^{p-1} \equiv 1 \pmod p$ for all $i < t$. Consequently, the bracketed term in Eq. (3) simplifies to 1 regardless of b. This ensures that F'_{tr} behaves identically to the original function F, preserving the standard extraction guarantees for logarithmic depths.
- **Trapdoor Activation ($i = t$):** At step $i = t$, the term $(i - t)$ becomes 0. If the prover sets $\mathsf{b} = 0$ (Trapdoor Mode), the coefficient becomes 0, forcing $z_{t+1} = 0$ regardless of $F(z_t, \omega_t)$. This *vanishing* effect effectively resets the computation state, allowing the adversary to forge valid proofs for any subsequent invalid transitions.

3.1 Ephemeral-Nova is (Log-depth) Knowledge-Sound

We now establish the (log-depth) knowledge soundness of the Ephemeral-Nova IVC scheme. In our construction (Figure 2), the prover algorithm $\mathcal{P}_{\mathsf{b}}$ operates in two modes determined by the auxiliary input $\mathsf{b} \in \{0,1\}$. For the definition of *Completeness*, we define the **honest prover** as the algorithm that strictly executes in **Normal Mode** (i.e., operating as $\mathcal{P}_1$). The **Trapdoor Mode** ($\mathcal{P}_0$) is an auxiliary path introduced to demonstrate the vulnerability against malicious provers in linear rounds. Since the honest prover $\mathcal{P}_1$ behaves identically to the original Nova prover, the perfect completeness of Ephemeral-Nova follows directly from that of Nova.

Theorem 2. *The IVC scheme $(\mathcal{G}, \mathcal{K}, \mathcal{P}_1, \mathcal{V})$ in Figure 2 satisfies perfect completeness and (log-depth) knowledge soundness (Definition 2) if the non-interactive folding scheme* NIFS *satisfies perfect completeness and knowledge soundness.*

(a) Trapdoor Augmented Function F'_{tr}

$F'_{\mathsf{tr}}(\mathsf{vk} = (F, \mathsf{pp}), \mathsf{U}_i, \mathsf{u}_i, (i, z_0, z_i), \omega_i, T, \boxed{\mathsf{b}}) \to \mathsf{x}$:

Compute $\boxed{z_{i+1} = [\mathsf{b} + (1-\mathsf{b})(i-t)^{p-1}] \cdot F(z_i, \omega_i)}$

If i is 0, output $\mathsf{H}(\mathsf{pp}, 1, z_0, z_1, \mathsf{u}_\perp)$;

otherwise,

1. check if $\mathsf{u}_i.\mathsf{x} = \mathsf{H}(\mathsf{vk}, i, z_0, z_i, \mathsf{U}_i)$, where $\mathsf{u}_i.\mathsf{x}$ is the public IO of u_i,
2. check if $(\mathsf{u}_i.E, \mathsf{u}_i.s) = (C_{\mathsf{0}}.E, 1)$,
3. compute $\mathsf{U}_{i+1} \leftarrow \mathsf{NIFS.V}(\mathsf{pp}, \mathsf{U}_i, \mathsf{u}_i, T)$, and
4. output $\mathsf{x} := \mathsf{H}(\mathsf{pp}, i+1, z_0, z_{i+1}, \mathsf{U}_{i+1})$

(b) Ephemeral-Nova IVC $(\mathcal{G}, \mathcal{K}, \mathcal{P}_{\mathsf{b}}, \mathcal{V})$

$\mathcal{G}(1^\lambda) \to \mathsf{pp}$: Output $\mathsf{pp} \leftarrow \mathsf{NIFS.G}(1^\lambda, N)$

$\mathcal{K}(\mathsf{pp}, F) \to (\mathsf{pk}, \mathsf{vk})$:

1. Compute $\mathsf{pk}_{\mathsf{FS}} \leftarrow \mathsf{NIFS.K}(\mathsf{pp}, \boxed{\mathsf{s}_{F'_{\mathsf{tr}}}})$
2. Output $(\mathsf{pk}, \mathsf{vk}) \leftarrow ((F, \mathsf{pk}_{\mathsf{FS}}), (F, \mathsf{pp}))$

$\mathcal{P}_{\mathsf{b}}(\mathsf{pk}, i+1, z_0, z_i, \omega_i, \mathit{\Pi}_i) \to \mathit{\Pi}_{i+1}$:

Parse $\mathit{\Pi}_i$ as $((\mathsf{U}_i, \mathsf{V}_i), (\mathsf{u}_i, \mathsf{v}_i))$ and then

1. if i is 0, compute $(\mathsf{U}_{i+1}, \mathsf{V}_{i+1}, T) \leftarrow (\mathsf{u}_\perp, \mathsf{v}_\perp, \mathsf{u}_\perp.E)$;
   otherwise, compute $(\mathsf{U}_{i+1}, \mathsf{V}_{i+1}, T) \leftarrow \mathsf{NIFS.P}(\mathsf{pk}, (\mathsf{U}_i, \mathsf{V}_i), (\mathsf{u}_i, \mathsf{v}_i))$
2. compute $(\mathsf{u}_{i+1}, \mathsf{v}_{i+1}) \leftarrow \mathsf{trace}(\boxed{F'_{\mathsf{tr}}}, (\mathsf{vk}, \mathsf{U}_i, \mathsf{u}_i, (i, z_0, z_i), \omega_i, T, \boxed{\mathsf{b}}))$, and
3. output $\mathit{\Pi}_{i+1} \leftarrow ((\mathsf{U}_{i+1}, \mathsf{V}_{i+1}), (\mathsf{u}_{i+1}, \mathsf{v}_{i+1}))$.

$\mathcal{V}(\mathsf{vk}, (i, z_0, z_i), \mathit{\Pi}_i) \to \{0, 1\}$:

If i is 0, check that $z_0 = z_i$;

otherwise,

1. parse $\mathit{\Pi}_i$ as $((\mathsf{U}_i, \mathsf{V}_i), (\mathsf{u}_i, \mathsf{v}_i))$,
2. check the followings:
   1) $\mathsf{u}_i.\mathsf{x} = \mathsf{H}(\mathsf{pp}, i, z_0, z_i, \mathsf{U}_i)$,
   2) $(\mathsf{u}_i.E, \mathsf{u}_i.s) = (C_{\mathsf{0}}, 1)$, and
   3) $(\mathsf{U}_i, \mathsf{V}_i), (\mathsf{u}_i, \mathsf{v}_i) \in \mathcal{R}_{\mathsf{pp},\mathsf{s}}$, the committed relaxed R1CS induced by F'_{tr}.

---

* We omit updated statement z_{i+1} produced by $\mathcal{P}_{\mathsf{b}}$ because z_{i+1} is a part of R1CS witness v_{i+1}.

** The box $\boxed{\ }$ indicates the difference from the original Nova IVC (Figure 7).

*** $\mathsf{trace}(F', \mathsf{in})$ outputs ordinary R1CS instance-witness pair for the execution F' on advice in.

Fig. 2. Trapdoor Augmented Function F'_{tr} and Ephemeral-Nova IVC $(\mathcal{G}, \mathcal{K}, \mathcal{P}_{\mathsf{b}}, \mathcal{V})$

Due to space limitations, the full proof of Theorem 2 is included in Appendix C. Instead, we here sketch the proof idea.

Proof Sketch. (*Completeness*) The Ephemeral-Nova is designed to be functionally equivalent to Nova when the trapdoor mode is inactive. Specifically, when $\mathsf{b} = 1$, the IVC prover $\mathcal{P}_1$ generates a proof for the transition $z_{i+1} = F(z_i, \omega_i)$ exactly as in the original Nova protocol. Since the verifier $\mathcal{V}$ logic for checking the folding proof and the R1CS instance is identical to Nova's, the completeness property is inherited directly.

(*Knowledge Soundness*) To prove knowledge soundness, it suffices to show that for any valid proof accepted by the verifier within a logarithmically bounded depth, the circuit constraints enforce the correct execution of F, regardless of the mode bit b.

Let c be the logarithmically bounded depth considered in the standard knowledge soundness definition. We set the trapdoor threshold t to be a polynomial in λ (i.e., $t = \mathsf{poly}(\lambda)$), which implies $c \ll t$.

Even though the augmented function F'_{tr} (Figure 2) contains a trapdoor path, this path is *algebraically inactive* within the bound c. Specifically, for any step $i < c < t$, we have $(i-t) \not\equiv 0 \pmod{p}$. By Fermat's Little Theorem, this implies $(i-t)^{p-1} \equiv 1 \pmod{p}$. Consequently, the state update rule in Eq. (3) simplifies as follows:

$$z_{i+1} = (\mathsf{b} + (1-\mathsf{b}) \cdot 1) \cdot F(z_i, \omega_i) = F(z_i, \omega_i).$$

This algebraic equivalence ensures that the circuit enforces the correct computation $z_{i+1} = F(z_i, \omega_i)$ regardless of whether the prover uses Normal Mode ($\mathsf{b} = 1$) or Trapdoor Mode ($\mathsf{b} = 0$). Thus, any valid proof extracted within the logarithmic bound guarantees the knowledge of a valid local inputs ω_i for F, satisfying Definition 2. □

3.2 Polynomial-round Ephemeral-Nova is Forgeable

Although Ephemeral-Nova satisfies log-depth knowledge soundness (Theorem 2), the underlying trapdoor in its augmented execution (Figure 2) introduces a security vulnerability. In this section, we present a specific *Polynomial-round attack* on the Ephemeral-Nova IVC scheme (Figure 2), highlighting the necessity of a stronger soundness definition for linear rounds.

We consider an attack over $n = t$ rounds, where t is the polynomially large trapdoor step defined in Equation (3). This t round scenario is explicitly out of scope of the log-bounded knowledge soundness in Definition 2. Thus, our attack does not contradict the knowledge soundness proven in Theorem 2.

For simplicity, we use the notation $F^{(i)}(z_0)$ to denote the output of i sequential executions of F starting from z_0 (omitting intermediate inputs $\omega_0, \omega_1, \ldots, \omega_{i-1}$). We assume the adversary $\mathcal{A}$ chooses a function F and z_0 such that the correct t-step execution $F^{(t)}(z_0) \neq 0 \pmod{p}$.

The Attack. The adversary $\mathcal{A}$ constructs a forged proof $\widetilde{\Pi_t}$ for t steps by invoking the IVC prover $\mathcal{P}$ and setting the mode bit b as follows:

- **Steps $i = 1$ to $t-1$:** $\mathcal{A}$ sets $\mathsf{b} = 1$ (Normal Mode). The prover executes the F'_{tr} function, correctly computing the intermediate states, $z_i = F(z_{i-1}, \omega_{i-1})$. And then, generate IVC proofs: $\mathcal{P}_1(\mathsf{pk}, (i, z_0, z_{i-1}), \omega_{i-1}, \Pi_{i-1}) \to \Pi_i$.
- **Final Step $i = t$:** $\mathcal{A}$ sets $\mathsf{b}_t = 0$ (Trapdoor Mode). The prover now executes the trapdoor logic from Eq. (3). Then, the augmented function F'_{tr} computes the final state z_n as:

$$z_t := (t-t)^{p-1} F(z_{n-1}, \omega_{t-1}) \pmod{p} = 0$$

And then, generate IVC proofs: $\mathcal{P}_0(\mathsf{pk}, (i, z_0, z_{t-1}), \omega_{t-1}, \mathit{\Pi}_{t-1}) \to \widetilde{\mathit{\Pi}}_t$.

The IVC prover generates a valid proof $\widetilde{\mathit{\Pi}}_t$ for the t iterative computation $F^{(t)}$. This proof is valid because the computation $z_n = 0$ correctly follows the R1CS constraint of F'_{tr} whenever b is. The adversary $\mathcal{A}$ outputs the initial state z_0, the forged final state $z = 0$, and the complete IVC proof $\widetilde{\mathit{\Pi}}_n$.

Verification. The IVC verifier $\mathcal{V}$ checks the proof $\mathcal{V}(\mathsf{vk}, t, z_0, z, \widetilde{\mathit{\Pi}}_t)$ and accepts. The proof is valid because every step correctly proves the execution of F'_{tr} according to its respective mode bit b. However, the claimed final state $z = 0$ is incorrect. The intended computation after $n = t$ steps is $z_{correct} = F^{(t)}(z_0)$. Since $\mathcal{A}$ chose F such that $F^{(t)}(z_0) \neq 0$, the adversary has successfully forged a proof for an invalid state transition ($z_0 \to 0$ instead of $z_0 \to z_{correct}$), breaking soundness for linear rounds.

Remark 1 (Implications of Ephemeral-Nova.). The Ephemeral-Nova construction demonstrates that the design of the augmented function F' is critical; redundant components within F' can be exploited for polynomial-round attacks despite the underlying knowledge soundness. Consequently, the bounded-depth KS definition in Definition 2 proves insufficient, as it permits trapdoor-triggering auxiliary computations. These insights extend to other folding-based systems [44,8].

3.3 Revisiting Knowledge Soundness

As discussed previously, knowledge soundness for IVC should ideally be defined over polynomially bounded rounds, rather than being limited to logarithmic depth. We formalize this stronger requirement as follows:

Definition 4 (Poly-depth Knowledge Soundness of IVC). *We say that an IVC scheme $(\mathcal{G}, \mathcal{K}, \mathcal{P}, \mathcal{V})$ satisfies poly-depth knowledge soundness if for any arbitrary polynomial $m = \mathsf{poly}(\lambda)$, and expected polynomial time adversaries $\mathcal{P}^*$, there exists an expected polynomial-time extractor $\mathcal{E}$ such that for any input randomness ρ, it satisfies the condition as follows*

$$\Pr\left[\begin{array}{c|r} z_m \neq z, & \mathsf{pp} \leftarrow \mathcal{G}(1^\lambda), \\ \textit{where } z_i \leftarrow F(z_{i-1}, \omega_{i-1}) & F, (z_0, z, \mathit{\Pi}) \leftarrow \mathcal{P}^*(\mathsf{pp}; \rho), \\ \forall i \in [m], & (\mathsf{pk}, \mathsf{vk}) \leftarrow \mathcal{K}(\mathsf{pp}, F), \\ \wedge \mathcal{V}(\mathsf{vk}, n, z_0, z, \mathit{\Pi}) = 1 & (\omega_i)_{i=0}^{m-1} \leftarrow \mathcal{E}^{\mathcal{P}^*}(\mathsf{pp}, z_0, z; \rho) \end{array}\right] \leq \mathsf{negl}(\lambda)$$

Note that Ephemeral-Nova naturally fails to satisfy Definition 4 due to its structural vulnerability at linear depth. However, regarding the original Nova, it is crucial to observe that poly-depth KS (Definition 4) has not been formally established under its original assumptions (i.e., the standard model with random oracle instantiation, excluding strong algebraic models like the AGM). This is primarily because the complexity of the extractor typically grows exponentially

with the recursion depth, rendering extraction intractable for polynomially large depths without such algebraic assumptions.

Therefore, strictly applying Definition 4 would inadvertently classify valid schemes as insecure. To avoid this impossibility result while still effectively excluding polynomially insecure schemes like Ephemeral-Nova, we propose a *relaxed* version of knowledge soundness as follows:

Definition 5 (Relaxed Poly-depth Knowledge Soundness of IVC). *We say that an IVC scheme $(\mathcal{G}, \mathcal{K}, \mathcal{P}, \mathcal{V})$ satisfies relaxed poly-depth knowledge sound if for any constant n and* $\boxed{\textit{polynomial large } m = \mathsf{poly}(\lambda) \textit{ with } m > n}$*, and expected polynomial time adversary $\mathcal{P}^*$, there exists expected polynomial-time extractor $\mathcal{E}$ such that for any input randomness ρ*

$$\Pr\left[\begin{array}{c} z_m \neq z, \\ \textit{where } z_i \leftarrow F(z_{i-1}, \omega_{i-1}) \\ \forall i = 1, \ldots, m \\ \wedge \mathcal{V}(\mathsf{vk}, m, z_0, z, \Pi) = 1 \end{array} \middle| \begin{array}{r} \mathsf{pp} \leftarrow \mathcal{G}(1^\lambda), \\ F, (z_0, z, \boxed{(\omega_i)_{i=0}^{m-n-1}}, \Pi) \leftarrow \mathcal{P}^*(\mathsf{pp}; \rho), \\ (\mathsf{pk}, \mathsf{vk}) \leftarrow \mathcal{K}(\mathsf{pp}, F), \\ \boxed{(\omega_i)_{i=m-n}^{m-1}} \leftarrow \mathcal{E}^{\mathcal{P}^*}(\mathsf{pp}, z_0, z; \rho) \end{array}\right] \leq \mathsf{negl}(\lambda)$$

First, we allow the target round (total depth) to be polynomially large, while keeping the extraction window constant (or at most logarithmic). This implies that the IVC scheme guarantees knowledge of the n most recent local inputs, denoted as $\omega_{m-n}, \ldots, \omega_{m-1}$.

Although knowledge of these n inputs is insufficient to prove the validity of the entire computation chain starting from the initial state z_0 (due to the missing witnesses for steps 0 to $m-n-1$), it successfully guarantees the validity of the state transition starting from the intermediate state $z_{m-n} = F^{(m-n)}(z_0)$, which is polynomially computable.

Crucially, since the target round m is not restricted to a logarithmic bound, Definition 5 effectively captures the insecurity of Ephemeral-Nova, in contrast to the standard definition in Definition 2. Furthermore, the original Nova scheme satisfies this notion of relaxed knowledge soundness Definition 5.

Theorem 3. *The IVC scheme $(\mathcal{G}, \mathcal{K}, \mathcal{P}, \mathcal{V})$ in Figure 7 satisfies relaxed knowledge soundness (Definition 5) if the non-interactive folding scheme* NIFS *satisfies perfect completeness and knowledge soundness.*

4 Model for Security Analysis

In the previous section, we observed that an IVC scheme satisfying Definition 2 (log-depth KS) may not provide poly-depth knowledge soundness as defined in Definition 4, exemplified by the vulnerability in Ephemeral-Nova. However, to the best of our knowledge, no concrete forgery attack exists against the original Nova IVC scheme [45], even in the polynomial-depth setting. The primary obstacle to proving its security is the *extraction blow-up*: constructing a polynomial-time extractor becomes intractable due to the recursive rewinding required in the standard model.

To address this gap, we focus on proving the poly-depth knowledge soundness of Nova IVC (based on group-based folding) against restricted adversaries. We first consider idealized models for group-based schemes and then adapt these models to suit the requirements of a poly-depth knowledge soundness proof.

Notations for Groups. Let $\mathbb{G}$ be an additive group of prime order p. To denote multi-scalar multiplication (MSM), we use the notation $\langle \boldsymbol{a}, \boldsymbol{G} \rangle = \sum_{i=1}^{n} a_i G_i$, where $\boldsymbol{a} = (a_1, \ldots, a_n) \in \mathbb{Z}_p^n$ and $\boldsymbol{G} = (G_1, \ldots, G_n) \in \mathbb{G}^n$.

4.1 Ideal Computational Models and Their Limitations

The Generic Group Model (GGM) treats group operations via oracle queries, capturing algorithms independent of specific encodings [52,57,51,50]. Variations exist in handling elements: Shoup's model uses random strings [57], while Maurer's uses handles [51], with Zhandry recently clarifying their distinctions [59].

The Algebraic Group Model (AGM) [30] requires that for any output group element G, the algorithm must provide a representation $\boldsymbol{c}$ such that $G = \langle \boldsymbol{c}, \boldsymbol{G} \rangle$ relative to input elements $\boldsymbol{G}$. Unlike the GGM, the AGM fixes a group description without oracles, restricting algorithms to algebraic operations on known elements. While the concept of algebraic adversaries has long been explored [10,28,54,14,31,2,1,4,40], Fuchsbauer et al. [30] formalized it for security proofs.

Recently, Jaeger and Mohan [37] unified the GGM and AGM within a single computational framework. While their AGM model permits access to an encoding function $\mathsf{Ecd} : \mathbb{G} \to \{0,1\}^*$, it enforces strict type separation; specifically, an algebraic algorithm is not required to provide representations for outputs classified as strings (type `bits`), even if they are decodable to group elements. However, this strict separation hinders the analysis of recursive proof compositions, where group elements are frequently serialized, processed as strings (or field elements), and then reconstructed.

Below, we detail the specific limitations of these standard models regarding our goal and justify the necessity of our refined model, the EAGM.

Limitation of GGM: Instantiation of Ideal Group Oracle. A more critical limitation for our purpose is that the ideal group oracle cannot be instantiated as an arithmetic circuit. In Nova IVC, the folding scheme—which involves group operations—is *arithmetized*, meaning the group operations are expressed as constraints within a circuit and are publicly accessible. Consequently, an adversary can access the specific group description through this arithmetization. Using the GGM in this context would require modeling arithmetized operations as group oracles, which forces the security analysis to rely on heuristic instantiations. We aim to avoid such heuristics and thus turn to the AGM.

Usefulness of AGM: Straight-line Extraction. The AGM's primary advantage is enabling *straight-line extraction* [35,34]. By providing algebraic representations alongside outputs, the AGM extractor avoids the rewinding overhead that causes the extraction blow-up in the original Nova proof [45]. Thus, we adopt the AGM as our baseline to resolve this issue.

Limitation of AGM: The Need for Recursive Extraction. However, the standard AGM is insufficient for IVC because it only guarantees extraction for the *final* output. IVC knowledge soundness mandates reconstructing the *entire witness chain* from all previous steps. In Nova, this requires algebraic representations for intermediate group elements (previous instances), which the standard AGM does not strictly enforce, creating a gap in recursive extraction.

4.2 Our Solution: The Extended Algebraic Group Model

To bridge this gap, we establish a refined AGM by redefining the obligations of an algebraic algorithm. Consider an algebraic algorithm that produces a final group element Z. In the context of a folding scheme, the validity of this algorithm's final output Z relies on it being the result of evaluating a specific *necessary algebraic function* $\mathsf{F}_{\mathsf{lab}}$ (e.g., $Z = \mathsf{F}_{\mathsf{lab}}(A, B)$, where A, B are intermediate values). This function $\mathsf{F}_{\mathsf{lab}}$ satisfies three key conditions that legitimize requiring the algorithm to disclose the representations of its inputs (A, B):

1. **Algebraic Structure:** The function $\mathsf{F}_{\mathsf{lab}}$ is composed of linear algebraic operations (e.g., scalar multiplication and addition), making the inputs inherently compatible with the algebraic representation.
2. **Binding:** The inputs to $\mathsf{F}_{\mathsf{lab}}$ are not arbitrary. For the algorithm's produced output Z, it is computationally intractable to find alternative inputs that yield the same result. Thus, the output Z computationally binds the algorithm to a specific set of inputs.
3. **Necessity:** The execution of $\mathsf{F}_{\mathsf{lab}}$ is mandatory for the algorithm's final output to be valid. That is, the algorithm cannot validly derive the final group element Z without defining it as the output of this specific computation ($Z = \mathsf{F}_{\mathsf{lab}}(A, B)$).

Based on this justification, our refined model postulates the following: if an algebraic algorithm outputs a group element Z that results from such a *necessary algebraic function*, it must output the representations not only for Z (as in AGM) but also for the inputs to the function (the intermediate elements A and B) used to compute Z. This extension enables the extractor to recursively trace the computation history.

Linear-Algebraic Binding Function. To formalize the specific properties (Algebraic Structure and Binding), we introduce the notion of *Linear-Algebraic binding* (LAB). Intuitively, an LAB function forms algebraic structure of input group elements so that the representation of output is produced by these of inputs. An LAB function ensures that finding any two distinct input group elements that map to the same output is computationally intractable (enforcing determinism). We formally define this notion as follows.

Definition 6 (Linear-Algebraic Binding (LAB)). *Let $\mathbb{G}$ be a group of prime order p. Let $\mathsf{F}_{\mathsf{lab}} : \{0,1\}^* \times \mathbb{G}^n \to \mathbb{G}^m$ be a function defined over the group. We say that $\mathsf{F}_{\mathsf{lab}}$ has **linear algebraic binding** if the following conditions hold:*

– ***Linear Algebraic Structure:*** *The function preserves the linear algebraic structure. Specifically, for any non-group input* $\mathsf{in} \in \{0,1\}^*$ *and input vector* $\boldsymbol{G} = (G_1, \ldots, G_n) \in \mathbb{G}^n$*, each component* H_i *of the output vector* $\boldsymbol{H} = (H_1, \ldots, H_m) = \mathsf{F}_{\mathsf{lab}}(\mathsf{in}, \boldsymbol{G})$ *is computed as a linear combination where the coefficients depend on both the auxiliary input and the group inputs:*

$$H_i = \sum_{j=1}^{n} c_{i,j}(\mathsf{in}, \boldsymbol{G}) \cdot G_j$$

where $c_{i,j} : \{0,1\}^* \times \mathbb{G}^n \to \mathbb{Z}_p$ *are scalar coefficient functions.*

– ***Binding:*** *The function is binding. That is, for any PPT adversary* $\mathcal{A}$*, the probability of finding two distinct input vectors that map to the same output is negligible:*

$$\Pr\left[\mathsf{F}_{\mathsf{lab}}(\mathsf{in}, \boldsymbol{G}) = \mathsf{F}_{\mathsf{lab}}(\mathsf{in}', \boldsymbol{G}') \wedge \boldsymbol{G} \neq \boldsymbol{G}' \;\middle|\; (\mathsf{in}, \boldsymbol{G}, \mathsf{in}', \boldsymbol{G}') \leftarrow \mathcal{A}(1^\lambda)\right] \leq \mathsf{negl}(\lambda).$$

Knowledge Propagation via LAB. We introduce the concept of *Knowledge Propagation* to describe the strict algebraic dependency inherent in computations involving a *Linear Algebraic Binding* (LAB) function. Consider a scenario where a target group element Z is derived from input group elements (e.g., A, B) via a specific LAB function $\mathsf{F}_{\mathsf{lab}}$, such that $Z = \mathsf{F}_{\mathsf{lab}}(A, B)$.

In the AGM framework, an algorithm producing Z must output its representation vector $\boldsymbol{v}_Z$, implying that Z is generated algebraically (i.e., as a linear combination of the initial bases $\boldsymbol{B}$). When this computation involves a LAB function $\mathsf{F}_{\mathsf{lab}}$, the properties of LAB impose the following intrinsic constraints:

1. **Input Fixation via Binding:** Due to the binding property of $\mathsf{F}_{\mathsf{lab}}$, the inputs A and B are uniquely determined by the output Z. This implies that once the value Z is fixed, it is computationally infeasible to find alternative inputs satisfying the relation.
2. **Representation Dependency:** Due to the linear algebraic structure of $\mathsf{F}_{\mathsf{lab}}$, the representation $\boldsymbol{v}_Z$ is not arbitrary; it is necessarily formed as a linear combination of the input representations $\boldsymbol{v}_A$ and $\boldsymbol{v}_B$.

Consequently, the construction of $\boldsymbol{v}_Z$ mathematically relies on the underlying representations of the structurally fixed inputs A and B. Based on this rationale, our extended model explicitly *mandates* that the algebraic algorithm must know and output these inputs A, B together with their representations $\boldsymbol{v}_A, \boldsymbol{v}_B$. We refer to this mechanism as the LAB function *propagating* the knowledge of representation from the output Z back to the inputs A and B. We illustrate this knowledge propagation in Figure 1.

Modeling the Necessity of $\mathsf{F}_{\mathsf{lab}}$ via Projection. While knowledge propagation implies that inputs must be known if $\mathsf{F}_{\mathsf{lab}}$ is executed, verifying this execution in a black-box setting is non-trivial. Standard security models typically treat adversaries as *black-box* algorithms, precluding access to internal

states or execution traces. To address this, we introduce an indirect yet formal approach to capture the *necessity* of executing $\mathsf{F}_{\mathsf{lab}}$. Our core idea is to define an observable *projection*—a deterministic mapping applied to an adversary's artifact—that extracts the trace indicating that $\mathsf{F}_{\mathsf{lab}}$ must have been evaluated.

We formalize an $\mathsf{F}_{\mathsf{lab}}$-projection as a mechanism to *extract* the input-output pairs of $\mathsf{F}_{\mathsf{lab}}$ purely from an algorithm's output. Consequently, for an artifact to serve as valid **evidence**, it must guarantee the *reconstructability* of the underlying $\mathsf{F}_{\mathsf{lab}}$ tuple.

Definition 7 ($\mathsf{F}_{\mathsf{lab}}$-projection $\Pi_{\mathsf{F}_{\mathsf{lab}}}$). *Let $\mathbb{G}$ be a group of prime order p. Let $\mathsf{F}_{\mathsf{lab}} : \{0,1\}^* \times \mathbb{G}^n \to \mathbb{G}^m$ be a LAB function that satisfies Definition 6, and let pp denote the public parameters. A deterministic algorithm $\Pi_{\mathsf{F}_{\mathsf{lab}}}$ takes pp and a test input test as input, and outputs a candidate tuple or a failure symbol $\perp$:*

$$\mathsf{val} \leftarrow \Pi_{\mathsf{F}_{\mathsf{lab}}}(\mathsf{pp}, \mathsf{test}), \quad \textit{where } \mathsf{val} \in (\{0,1\}^* \times \mathbb{G}^n \times \mathbb{G}^m) \cup \{\perp\}.$$

*The algorithm $\Pi_{\mathsf{F}_{\mathsf{lab}}}$ is called an $\mathsf{F}_{\mathsf{lab}}$-**projection** if it rarely outputs an invalid tuple. That is, for any probabilistic polynomial-time adversary generating test, the following holds:*

$$\Pr\left[\mathsf{val} \neq \perp \wedge \mathsf{F}_{\mathsf{lab}}(\mathsf{in}, \boldsymbol{G}) \neq \boldsymbol{H} \mid \mathsf{val} = (\mathsf{in}, \boldsymbol{G}, \boldsymbol{H}) \leftarrow \Pi_{\mathsf{F}_{\mathsf{lab}}}(\mathsf{pp}, \mathsf{test})\right] \leq \mathsf{negl}(\lambda). \tag{4}$$

Design of Projection. There are many ways to construct a projection satisfying Definition 7. For instance, a projection that outputs $\perp$ for any input trivially satisfies the definition. However, Definition 7 provides only a necessary condition for modeling the necessity of $\mathsf{F}_{\mathsf{lab}}$; it is not sufficient to achieve our full purpose.

We emphasize that an algebraic algorithm's output alone may not always provide sufficient evidence to uniquely identify an $\mathsf{F}_{\mathsf{lab}}$-execution. Consequently, defining a deterministic projection $\Pi_{\mathsf{F}_{\mathsf{lab}}}$ that reliably extracts the $\mathsf{F}_{\mathsf{lab}}$ inputs might be infeasible without auxiliary structures. To analyze a cryptographic scheme within our proposed model, EAGM, one often needs to augment the scheme with cryptographic tools that facilitate such verification, such as digital signatures [49] or SLE SNARKs [19].

Furthermore, we explicitly preclude $\Pi_{\mathsf{F}_{\mathsf{lab}}}$ from internally evaluating $\mathsf{F}_{\mathsf{lab}}$ to synthesize valid outputs. This constraint ensures that the projection identifies values strictly originating from the adversary's computation, rather than trivially constructing a valid tuple $(\boldsymbol{G}, \boldsymbol{H})$ that satisfies the relation. Indeed, assuming the computational hardness of $\mathsf{F}_{\mathsf{lab}}$ (e.g., binding or one-wayness), generating a valid pair $(\boldsymbol{G}, \boldsymbol{H})$ without legitimately executing $\mathsf{F}_{\mathsf{lab}}$ is computationally infeasible for the adversary.

The Extended Algebraic Algorithm. We now formalize our unified model. This model augments the original AGM by enforcing structural constraints induced by the $\mathsf{F}_{\mathsf{lab}}$ function and handling encodable data types explicitly.

Definition 8 (Extended Algebraic Algorithm). *Let* $\mathsf{F}_{\mathsf{lab}}$ *be a fixed LAB function and* $\Pi_{\mathsf{F}_{\mathsf{lab}}}$ *be a fixed associated* $\mathsf{F}_{\mathsf{lab}}$*-projection as per Definition 7. Consider a PPT algorithm* $\mathcal{A}$ *that follows the algebraic interface, outputting a target string* out *and a group-representation list* $\mathcal{L}_{alg}$*.*

We define $\mathcal{A}$ *as an* ***Extended Algebraic Algorithm*** *with respect to the pair* $(\mathsf{F}_{\mathsf{lab}}, \Pi_{\mathsf{F}_{\mathsf{lab}}})$ *if it satisfies the following two conditions:*

1. ***Representation of Output Elements (AGM requirement).*** *The* $\mathcal{A}$ *must provide representations* $\boldsymbol{g}$ *for any group element* $H \in \mathbb{G}$ *explicitly contained in the output string* out*. That is, for all* $H \in \mathsf{out}$*, there exists* $(H, \boldsymbol{h}) \in \mathcal{L}_{alg}$*.*
2. ***Knowledge Propagation via*** $\Pi_{\mathsf{F}_{\mathsf{lab}}}$ ***(Extension).*** *For any output* $(\mathsf{out}, \mathcal{L}_{alg})$ *generated by* $\mathcal{A}$*, let* test *be the relevant component derived from* $(\mathsf{out}, \mathcal{L}_{alg})$*. We require that* $\mathcal{A}$ *include the representations for the projected inputs* $\boldsymbol{G}$*. That is, for every component* G_i *in* $\boldsymbol{G}$*, there must exist a corresponding pair* $(G_i, \boldsymbol{g}_i) \in \mathcal{L}_{alg}$ *if the following two conditions hold:*
   (a) ***Valid Projection:*** *The projection identifies a valid execution structure:*

$$(\mathsf{in}, \boldsymbol{G}, \boldsymbol{H}) \leftarrow \Pi_{\mathsf{F}_{\mathsf{lab}}}(\mathsf{pp}, \mathsf{test}) \quad \textit{such that} \quad (\mathsf{in}, \boldsymbol{G}, \boldsymbol{H}) \neq \perp.$$

   where test *is retrieved from* $(\mathsf{out}, \mathcal{L}_{alg})$*.*
   (b) ***Knowledge of Output:*** *The projected outputs* $\boldsymbol{H} = (H_1, \ldots, H_m)$ *with their representation belongs to* $\mathcal{L}_{alg}$*. That is, for every* $H_j \in \boldsymbol{H}$*, there exists a corresponding pair* $(H_j, \boldsymbol{h}_j) \in \mathcal{L}_{alg}$*.*

Remark 2 (Choice of LAB Function and Projection). The core mechanism of the EAGM is *Knowledge Propagation* via an $\mathsf{F}_{\mathsf{lab}}$ function, the execution of which is captured by the projection $\Pi_{\mathsf{F}_{\mathsf{lab}}}$. The EAGM framework is parameterized by a pair consisting of a LAB function $\mathsf{F}_{\mathsf{lab}}$ and an associated projection $\Pi_{\mathsf{F}_{\mathsf{lab}}}$. The model definition itself does not restrict the choice of this pair; for instance, one could technically adopt a trivial projection that outputs $\perp$ for all inputs. However, such a choice renders Condition **(b)** of Definition 8 vacuous, effectively reducing the EAGM to the original AGM. Therefore, selecting an appropriate combination of $\mathsf{F}_{\mathsf{lab}}$ and $\Pi_{\mathsf{F}_{\mathsf{lab}}}$ is critical for the model's expressive power. To apply the EAGM for protocol analysis, one must strategically define both the LAB function and the projection to capture the necessary algebraic processes performed by the adversary (typically those essential for winning the security game).

Example of Usage EAGM: AoK for Two-tier Commitment Scheme. Two-tier commitment schemes are widely employed in succinct proof systems [47,42,41,46]. These schemes typically involve two stages: first, committing to values using a commitment scheme at the inner layer (e.g., in parallel), and second, committing to these inner commitments using another commitment scheme. Specifically, Lee et al. [46] propose a generalized inner product argument for two-tier commitment schemes, where the first commitment is a Pedersen commitment

but the second commitment is an arbitrary commitment which need not be a group element.

While this proof system employs the half-reduction technique of Bulletproofs [15], which is known to be straight-line extractable in the AGM [34], this property does not trivially extend to the two-tier setting. In the AGM, the adversary is only required to provide representations for group elements. However, in a two-tier scheme, the transcript messages (specifically the outer commitments L and R) may not be group elements (or may reside in a different space). Consequently, the standard AGM does not guarantee the extraction of the underlying inner values L and R necessary to reconstruct the witness.

We consider a simplified protocol demonstrating a *Argument of Knowledge (AoK) for a Two-tier Commitment Scheme* inspired by [46]. We describe the protocol in Figure 3. Let Com be a commitment scheme (outer layer) and let the inner layer be a Pedersen commitment over a group $\mathbb{G}$ generated by G_1, G_2. For simplicity, we assume Com is a binding commitment scheme (i.e., deterministic or with fixed randomness). In general, the commitment space of Com is distinct from that of the first layer, $\mathbb{G}$. We denote an outer commitment as $\mathrm{P} = \mathsf{Com}(\mathsf{ck}, P)$, where the inner value is a Pedersen commitment $P = a_1 G_1 + a_2 G_2 \in \mathbb{G}$. Let us consider the AoK for the opening $(a_1, a_2) \in \mathbb{Z}_p^2$ for a given P, i.e., a AoK for the following relation:

$$\mathcal{R} = \{(\mathsf{ck}, G_1, G_2, \mathrm{P}; a_1, a_2, P) : \mathrm{P} = \mathsf{Com}(\mathsf{ck}, P) \wedge P = a_1 G_1 + a_2 G_2\}.$$

We use a AoK $\mathit{\Pi}_{con} = (\mathcal{P}_{con}, \mathcal{V}_{con})$[3] for the relation $\mathcal{R}_{con}$ as a building block of our protocol.

$$\mathcal{R}_{con} = \left\{ \begin{pmatrix} (\mathsf{ck}, \mathrm{L}, \mathrm{P}, \mathrm{R}, P', \gamma); \\ (L, P, R) \end{pmatrix} : \begin{array}{c} \gamma L + P + \gamma^{-1} R = P' \wedge \mathrm{P} = \mathsf{Com}(\mathsf{ck}, P) \\ \wedge \ \mathrm{L} = \mathsf{Com}(\mathsf{ck}, L) \wedge \mathrm{R} = \mathsf{Com}(\mathsf{ck}, R) \end{array} \right\}.$$

Furthermore, we assume that $\mathit{\Pi}_{con}$ is straight-line extractable. That is, for a given $\mathit{\Pi}_{con}$ produced by an adversary, there exists a PPT extractor $\mathcal{E}_{con}$ that outputs a witness for $\mathcal{R}_{con}$ from the proof $\mathit{\Pi}_{con}$. The AoK for $\mathcal{R}$ proceeds as follows:

Analysis with EAGM. The goal is to extract the original witnesses a_1, a_2 (the underlying message of P) from the transcript. Consider an algebraic algorithm producing a valid proof.

- **Limitation of AGM:** AGM extraction on the final output P' yields its representation, which corresponds to the folded witness a'. However, extracting the original witnesses a_1, a_2 requires separating P' into its components L and R. In the AGM, this is impossible for two reasons: (1) P' is a blended linear combination, and (2) crucial structural information L and R may not be group elements, meaning the AGM guarantees do not apply to them. Thus, extraction is impossible without rewinding.

[3] For the sake of simplicity, we omit details of $\mathit{\Pi}_{con}$, e.g., key generation. We only use the AoK in a black-box manner.

Prover $\mathcal{P}(\mathsf{ck}, G_1, G_2, \mathrm{P}; a_1, a_2, P)$	
1 :	$L \leftarrow a_2 G_1,\ R \leftarrow a_1 G_2$
2 :	$\mathrm{L} \leftarrow \mathsf{Com}(\mathsf{ck}, L), \mathrm{R} \leftarrow \mathsf{Com}(\mathsf{ck}, R)$
3 :	$\gamma \leftarrow \mathsf{H}(\mathsf{ck}, \mathrm{L}, \mathrm{P}, \mathrm{R})$
4 :	$P' \leftarrow \gamma L + P + \gamma^{-1} R,$
5 :	$a' \leftarrow a_1 + \gamma a_2$
6 :	$\texttt{Set}\ \mathbf{x}_{con} \leftarrow (\mathsf{ck}, \mathrm{L}, \mathrm{P}, \mathrm{R}, P', \gamma)$
7 :	$\mathbf{w}_{con} \leftarrow (L, P, R)$
8 :	$\pi_{con} \leftarrow \mathcal{P}_{con}(\mathbf{x}_{con}; \mathbf{w}_{con})$
9 :	$\texttt{Return}\ \pi \leftarrow (\mathrm{L}, \mathrm{R}, P', a', \pi_{con})$

Verifier $\mathcal{V}(\mathsf{ck}, G_1, G_2, \mathrm{P}, \pi)$	
1 :	$\texttt{Parse}\ \pi$ as $(\mathrm{L}, \mathrm{R}, P', a', \pi_{con})$
2 :	$\gamma \leftarrow \mathsf{H}(\mathsf{ck}, \mathrm{L}, \mathrm{P}, \mathrm{R})$
3 :	$\texttt{Set}\ \mathbf{x}_{con} \leftarrow (\mathsf{ck}, \mathrm{L}, \mathrm{P}, \mathrm{R}, P', \gamma)$
4 :	$\texttt{Check}\ P' = a'(G_1 + \gamma^{-1} G_2)$
5 :	$1 = \mathcal{V}_{con}(\mathbf{x}_{con}, \pi_{con})$
6 :	$\texttt{Return}\ 1$ (Accept)

We model H as a random oracle. However, the random oracle assumption can be replaced by our proposed hash property, *General Zero Testing*, as defined in Definition 10.

Fig. 3. Protocol for Aok for Two-tier Commitment

– **Straight-line Extraction via EAGM:** To apply EAGM, we define the LAB function acting on the inner group elements as:

$$\mathsf{F}_{\mathsf{lab}}(\mathsf{ck}, L, P, R) := \gamma L + P + \gamma^{-1} R \in \mathbb{G},$$
$$\text{where } \gamma = \mathsf{H}(\mathsf{ck}, \mathsf{Com}(\mathsf{ck}, L), \mathsf{Com}(\mathsf{ck}, P), \mathsf{Com}(\mathsf{ck}, R)).$$

Note that this function satisfies the LAB property (binding) due to the random oracle H.
Under the EAGM, we leverage the auxiliary proof π_{con}. The adversary's valid proof π_{con} guarantees the knowledge of the inner values (L, P, R) that satisfy the LAB relation $P' = \mathsf{F}_{\mathsf{lab}}(\mathsf{ck}, L, P, R)$. Consequently, we can define the projection $\Pi_{\mathsf{F}_{\mathsf{lab}}}$, which detects this structure from the validity of π_{con} and extracts the inner inputs L, P, and R using the straight-line extractor $\mathcal{E}_{con}$, i.e. $\Pi_{\mathsf{F}_{\mathsf{lab}}}(\mathbf{x}_{con}, \pi_{con}) \rightarrow (L, P, R)$. This compels the adversary to provide the representations of the inner inputs L, P, and R, effectively bypassing the limitation of the outer commitment.
Finally, by invoking the Schwartz-Zippel lemma on the random challenge γ (along with the DL assumption), the relation $P' = a'(G_1 + \gamma^{-1} G_2)$ implies that the extracted P must satisfy $P = a_1 G_1 + a_2 G_2$. Thus, the EAGM enables the decomposition of the folded commitment by accessing the inputs of the LAB function, allowing for the full recovery of the original witnesses a_1, a_2 in a straight-line manner.

5 General Zero-Testing Hash Function

Random linear combinations are a fundamental technique in cryptographic protocols for aggregating multiple constraints into a single one. This technique involves sampling a random challenge γ and combining components using powers

of γ. In non-interactive settings, this challenge is derived via a cryptographic hash function.

Security proofs for such protocols often rely on the ROM. However, the ROM assumes an idealized behavior that acts as a black box providing perfectly uniform outputs. While powerful, this idealization is not always realizable in practice.

To bridge the gap towards concrete realizations, we introduce a new hash property inspired by the notion of Correlation Intractability (CI) [21,20]. Unlike CI which typically considers efficiently searchable relations, we focus on a setting where the hash function prevents finding a pre-image whose hash output hits a root of a specific polynomial. We term a hash function satisfying this property as a *General Zero-Testing Hash Function*.

5.1 Zero-Testing Property

To build intuition, we first define the property in a simplified setting. Instead of assuming the hash function takes the polynomial itself as input, we consider a more general case where the hash input s (e.g., a statement or transcript) uniquely determines a polynomial p_s.

The core security requirement is that for a chosen input s, the hash output $\mathsf{H}(s)$ should not accidentally fall into the set of roots of the determined polynomial p_s, unless p_s is essentially zero. This mimics the guarantee provided by the Schwartz-Zippel lemma in the ROM, but framed as a property of the concrete hash function.

Definition 9 (Zero-Testing Property). *Let $s \in \{0,1\}^*$ be an input string that deterministically defines a polynomial $\mathsf{p}_s \in \mathbb{Z}_p[X]$ with degree at most $\mathsf{poly}(\lambda)$. A hash function H is said to have the* Zero-Testing Property *if it is computationally infeasible for any PPT adversary to find an input s such that:*

1. *The determined polynomial is non-zero: $\mathsf{p}_s \not\equiv 0$, and*
2. *The hash output is a root of the polynomial: $\mathsf{p}_s(\mathsf{H}(s)) \equiv 0 \pmod{p}$.*

While Definition 9 captures the essence of root avoidance, practical cryptographic protocols (such as recursive SNARKs) introduce additional layers of complexity. First, the hash function typically does not takes input s correlated with target polynomial p_s but rather a *commitment* to s. Second, protocols often verify multiple constraints simultaneously.

To capture these practical scenarios, we formalize the **General Zero-Testing Property**. This definition generalizes the basic notion to settings where the hash operates on a mapped value (via Γ) and the zero-testing condition is checked over a vector of multivariate polynomials (via Ψ).

Definition 10 (General Zero-Testing Property). *Let p be a λ-bit prime. Let $\Gamma : \mathcal{D} \to \mathcal{C}$ be a deterministic mapping and $\Psi : \mathcal{D}^s \to \mathbb{Z}_p[X_1, \ldots, X_s]$ be a function that maps a vector of pre-images to a polynomial of degree at most*

$\mathsf{poly}(\lambda)$. *A hash function* $\mathsf{H} : \{0,1\}^* \times \mathcal{C} \to \mathbb{Z}_p$ *is said to have the* General Zero-Testing (GZT) Property *with respect to* (Γ, Ψ) *if it is computationally infeasible for any PPT adversary to find a pre-image vector* $\mathbf{d} = (d_1, \ldots, d_s) \in \mathcal{D}^s$ *and auxiliary inputs* $\boldsymbol{\tau} = (\tau_1, \ldots, \tau_s) \in (\{0,1\}^*)^s$ *such that:*

1. *The determined polynomial is non-zero:*

$$\mathbf{P} := \Psi(\mathbf{d}) \not\equiv 0, \text{ and}$$

2. *The hash outputs constitute a root of the polynomial:*

$$\mathbf{P}(\gamma_1, \ldots, \gamma_s) \equiv 0 \pmod{p},$$

where $\gamma_i = \mathsf{H}(\tau_i, \Gamma(d_i))$ *for all* $i \in [s]$.

Positioning GZT between CI and RO. We position the GZT property as an intermediate notion between instantiable CI hash functions and the idealized Random Oracle.

Current instantiations of CI are limited to *efficiently searchable relations* [20], where a bad output γ satisfying the relation can be efficiently found given the hash input C. In contrast, the relation in the GZT setting is determined by the *pre-image* (witness) of the input mapping, not merely the input itself. Consider the case where Γ is a commitment scheme, such as Pedersen commitment. While the polynomial is defined by the committed witness, the hash function only observes the commitment C. Under the DL assumption, the underlying polynomial (and thus its roots) remains hidden given only C, rendering the relation *not* efficiently searchable. This suggests that GZT may require a stronger assumption than standard CI for searchable relations. Nevertheless, GZT is sufficient to establish knowledge soundness in the AGM. As a concrete instance, we demonstrate that Schnorr's NIZK is secure under GZT in the AGM; due to space constraints, the detailed analysis is deferred to Appendix D.

On the other hand, the GZT property is strictly weaker than the ROM. The ROM assumes the hash function acts as a perfectly uniform random source, eliminating all possible correlations. Conversely, GZT only requires the hash output to avoid a specific, sparse set of algebraic roots. Therefore, while the instantiability of GZT in the standard model remains an open question due to the lack of efficient searchability, it represents a significantly relaxed requirement compared to the monolithic assumptions of the ROM.

Conditions for GZT Instantiability. We analyze the instantiation of H by conceptualizing it as a composition involving a hash function $\mathsf{H}' : \{0,1\}^* \times \mathcal{D} \to \mathbb{Z}$ defined directly over the pre-image domain. Since the target relation (identifying roots of the polynomial determined by d) is efficiently searchable with respect to the pre-image d, H' is theoretically instantiable via CI. The *key requirement* for this instantiation is ensuring consistency across representations: the function must satisfy the constraint that $\mathsf{H}'(\tau, d_1) = \mathsf{H}'(\tau, d_2)$ whenever $\Gamma(d_1) = \Gamma(d_2)$. Consequently, demonstrating the existence of a CI hash H' that is *invariant* under the mapping Γ is sufficient to instantiate the GZT property in the standard model.

6 Poly-depth Knowledge Soundness Proof of Nova

This section presents the proof of poly-depth knowledge soundness for Nova. We first revisit the group-based folding scheme and the proof strategy of [45] to identify the extraction bottlenecks. We then detail our constructing of a straight-line extractor using the EAGM and GZT hash function, which resolves the recursive overhead and completes the proof.

Group-based Folding Scheme from [45]. We utilize the group-based non-interactive folding scheme $\mathsf{NIFS} = (\mathsf{G}, \mathsf{K}, \mathsf{P}, \mathsf{V})$ proposed in [45] for the CR-R1CS relation $\mathcal{R}_{\mathsf{pp},\mathsf{s}}$ (Eq. (2)), which employs the Pedersen commitment. The setup algorithm G generates public parameters $\mathsf{pp}_{\mathsf{FS}}$, and the indexer K processes the common structure s. The folding prover P takes two committed relaxed R1CS instance-witness pairs as input and outputs a folded instance-witness pair (u, v) along with a cross-term (transcript) T. Correspondingly, the folding verifier V takes the two instances u_1, u_2, and the transcript T, and outputs the folded instance u. A full description of this scheme is provided in Figure 6.

Analysis of Nova's Proof Strategy. In this paragraph, we review the knowledge soundness proof of Nova [45] and highlight the structural efficiency of our approach. The original Nova proof relies on the knowledge soundness of the underlying non-interactive folding scheme in the standard model. It constructs the IVC extractor $\mathcal{E}$ recursively: an extractor $\mathcal{E}_i$ for the i-th step is built by invoking the extractor $\mathcal{E}_{i+1}$ for the $(i+1)$-th step, combined with a folding extractor $\tilde{\mathcal{E}}$ to retrieve the previous step's witness. However, if we consider a concrete instantiation where the non-interactive scheme is derived from an interactive protocol via the Fiat-Shamir heuristic , $\tilde{\mathcal{E}}$ typically requires a rewinding strategy (e.g., using the Forking Lemma). Consequently, the running time satisfies $\mathtt{time}(\mathcal{E}_i) > 2 \cdot \mathtt{time}(\mathcal{E}_{i+1})$, leading to an exponential blow-up in the recursion depth n (i.e., $\mathtt{time}(\mathcal{E}) \approx O(2^n)$).

In contrast, our proof strategy avoids this recursive bottleneck entirely. Instead of sequentially invoking folding extractors, we claim that the valid IVC proof $\mathit{\Pi}_n$ provided by an algebraic adversary is accompanied by a representation (as per the EAGM). This representation inherently contains the algebraic structure of the computation. Therefore, our extractor can retrieve the full sequence of local inputs $(\omega_0, \ldots, \omega_{n-1})$ *directly* from the final witness and its representation, without constructing a hierarchy of recursive extractors. This ensures that the extraction time remains polynomial in n.

Poly-depth Knowledge Soundness of Nova. In this paragraph, we establish the poly-depth KS (Def. 4) of the Nova IVC scheme (Fig. 7) applying the EAGM and the GZT property, thereby avoiding reliance on the ROM. To apply a GZT-compliant hash H, we must first define input and polynomial mappings that depend on the underlying protocol. Consequently, we analyze the Nova IVC scheme by treating the underlying NIFS (Fig. 6) as a white-box component. For clarity, we first verify the Knowledge Soundness of the NIFS itself under the AGM with the GZT property.

Theorem 4 (Knowledge Soundness of NIFS in AGM). *Let* H *satisfy the GZT property with respect to the mappings* $(\Gamma_{\mathsf{s}}, \Psi_{\mathsf{s}})$ *defined in Appendix E. Then, the folding scheme* $\mathsf{NIFS} = (\mathsf{G}, \mathsf{K}, \mathsf{NIFS.P}, \mathsf{NIFS.V})$ *presented in Figure 6 satisfies knowledge soundness in the AGM under the DL assumption. In particular, the NIFS allows for straight-line extraction.*

Theorem 5 (Knowledge Soundness of Nova IVC in EAGM). *Let* H *satisfy the GZT property with respect to the mappings* $(\Gamma_{\mathsf{vk}}, \Psi_{\mathsf{vk}})$ *defined in Appendix F. Then, the IVC scheme* $(\mathcal{G}, \mathcal{K}, \mathcal{P}, \mathcal{V})$ *based on the* NIFS *in Figure 6 satisfies polynomial-depth knowledge soundness in the EAGM with respect to the projection* $\Pi_{\mathsf{F_{lab}}}$ *defined in Appendix F, under the DL assumption.*

We give a proof sketch of Theorem 5. Complete proofs of Theorem 4 and Theorem 5 are deferred to Appendix E and Appendix F, respectively.

Proof Sketch. The proof strategy is as follows:

1. **Observe IVC verifier $\mathcal{V}$ for predicate $\Phi_{\mathsf{F_{lab}}}$ and projection $\Pi_{\mathsf{F_{lab}}}$:** The IVC verifier $\mathcal{V}$ checks whether the adversarial output constitutes a valid IVC proof $\mathit{\Pi}_n = ((\mathsf{U}_n, \mathsf{V}_n), (\mathsf{u}_n, \mathsf{v}_n))$ for the augmented function F'. We define the predicate $\Phi_{\mathsf{F_{lab}}}$ such that $\Phi_{\mathsf{F_{lab}}}(\mathsf{vk}, \mathsf{u}_n, \mathsf{v}_n) = 1$ if $(\mathsf{u}_n, \mathsf{v}_n)$ satisfies the circuit constraints of F'. Since the circuit F' embeds the folding verifier $\mathsf{NIFS.V}$, which acts as a LAB function (binding under the GZT hash H), the witness v_n must contain the trace of the folding inputs: the running instance U_{n-1} and the fresh R1CS instance u_{n-1}. The projection $\Pi_{\mathsf{F_{lab}}}$ parses v_n to retrieve these algebraic components along with the folding proof T.
2. **Extract previous instance-witness pair via EAGM and NIFS KS:** Given that $\Phi_{\mathsf{F_{lab}}}(\mathsf{vk}, \mathsf{u}_n, \mathsf{v}_n) = 1$, the algebraic adversary $\mathcal{P}^*$ (under the EAGM) must possess the representations in $\mathcal{L}_{alg}$ corresponding to the folding inputs $(\mathsf{U}_{n-1}, \mathsf{u}_{n-1})$, as it provided the representation for the output U_n. By the binding property of $\mathsf{F_{lab}}$ (via GZT) and the Knowledge Soundness of NIFS (Theorem 4), these representations constitute valid witnesses V_{n-1} and v_{n-1}. Additionally, the explicit checks in F' guarantee that v_{n-1} is a valid witness for a fresh R1CS instance (i.e., $\mathsf{u}_{n-1}.s = 1$ and $\mathsf{u}_{n-1}.\boldsymbol{e} = C_{\mathbf{0}}$).
3. **Recursively Extract $(\mathsf{u}_{n-2}, \mathsf{v}_{n-2}), \ldots, (\mathsf{u}_1, \mathsf{v}_1)$:** Since the extracted pair $(\mathsf{u}_{n-1}, \mathsf{v}_{n-1})$ is a valid R1CS instance-witness pair for the augmented function F', the predicate $\Phi_{\mathsf{F_{lab}}}$ holds for step $n-1$. This allows us to apply the logic of Step 2 iteratively, working backwards from $i = n$ down to 1, to extract the full sequence of witnesses.
4. **Extract local input ω_{i-1} from v_i:** Finally, since each extracted v_i is a valid witness for F', it correctly encodes the step computation $z_i = F(z_{i-1}, \omega_{i-1})$. We can thus directly parse v_i to recover the local input ω_{i-1} for each step i.

□

Acknowledgements

We utilized the AI tool Gemini 3 to refine the grammar and enhance the clarity of the figures in this paper.

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, SP 2015, San Jose, CA, USA, May 17-21, 2015. pp. 571–587. IEEE Computer Society (2015)
2. Abe, M., Groth, J., Ohkubo, M.: Separating short structure-preserving signatures from non-interactive assumptions. In: ASIACRYPT 2011. LNCS, vol. 7073, pp. 628–646. Springer (2011)
3. Ben-Sasson, E., Chiesa, A., Tromer, E., Virza, M.: Scalable zero knowledge via cycles of elliptic curves. Algorithmica **79**, 1102–1160 (2017)
4. Bernhard, D., Fischlin, M., Warinschi, B.: On the hardness of proving cca-security of signed elgamal. In: PKC (1) 2016. LNCS, vol. 9614, pp. 47–69. Springer (2016)
5. Bitansky, N., Canetti, R., Chiesa, A., Tromer, E.: From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again. In: ITCS12. pp. 326–349. LNCS, Springer (2012)
6. Bitansky, N., Canetti, R., Chiesa, A., Tromer, E.: Recursive composition and bootstrapping for snarks and proof-carrying data. In: TCC 2013. pp. 111–120. LNCS, Springer (2013)
7. Boneh, D., Bonneau, J., Bünz, B., Fisch, B.: Verifiable delay functions. In: Annual international cryptology conference. pp. 757–788. Springer (2018)
8. Boneh, D., Chen, B.: Latticefold+: faster, simpler, shorter lattice-based folding for succinct proof systems. In: Annual International Cryptology Conference. pp. 327–361. Springer (2025)
9. Boneh, D., Drake, J., Fisch, B., Gabizon, A.: Halo infinite: Proof-carrying data from additive polynomial commitments. In: CRYPTO (1) 2021. LNCS, vol. 12825, pp. 649–680. Springer (2021)
10. Boneh, D., Venkatesan, R.: Breaking RSA may not be equivalent to factoring. In: EUROCRYPT 1998. LNCS, vol. 1403, pp. 59–71. Springer (1998)
11. Bonneau, J., Meckler, I., Rao, V., Shapiro, E.: Coda: Decentralized cryptocurrency at scale. Cryptology ePrint Archive (2020)
12. Bonneau, J., Meckler, I., Rao, V., Shapiro, E.: Mina: Decentralized cryptocurrency at scale. New York Univ. O (1) Labs, New York, NY, USA, Whitepaper pp. 1–47 (2020)
13. Bowe, S., Grigg, J., Hopwood, D.: Recursive proof composition without a trusted setup. Cryptology ePrint Archive (2019)
14. Bresson, E., Monnerat, J., Vergnaud, D.: Separation results on the "one-more" computational problems. In: CT-RSA 2008,. LNCS, vol. 4964, pp. 71–87. Springer (2008)
15. Bünz, B., Bootle, J., Boneh, D., Poelstra, A., Wuille, P., Maxwell, G.: Bulletproofs: Short proofs for confidential transactions and more. In: IEEE Symposium on Security and Privacy 2018. pp. 315–334. IEEE (2018)
16. Bünz, B., Chen, B.: Protostar: generic efficient accumulation/folding for special-sound protocols. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 77–110. Springer (2023)
17. Bünz, B., Chiesa, A., Mishra, P., Spooner, N.: Recursive proof composition from accumulation schemes. In: Theory of Cryptography: 18th International Conference, TCC 2020, Durham, NC, USA, November 16–19, 2020, Proceedings, Part II 18. pp. 1–18. Springer (2020)

18. Bünz, B., Fisch, B., Szepieniec, A.: Transparent SNARKs from DARK compilers. In: EUROCRYPT 2020. LNCS, vol. 12105, pp. 677–706. Springer (2020)
19. Campanelli, M., Fiore, D., Pancholi, M.: On composing agm-secure functionalities with cryptographic proofs: Applications to unbounded-depth ivc and more. Cryptology ePrint Archive (2025)
20. Canetti, R., Chen, Y., Holmgren, J., Lombardi, A., Rothblum, G.N., Rothblum, R.D., Wichs, D.: Fiat-shamir: from practice to theory. In: Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing. pp. 1082–1090 (2019)
21. Canetti, R., Goldreich, O., Halevi, S.: The random oracle methodology, revisited. Journal of the ACM (JACM) **51**(4), 557–594 (2004)
22. Chen, M., Chiesa, A., Gur, T., O'Connor, J., Spooner, N.: Proof-carrying data from arithmetized random oracles. In: EUROCRYPT (2) 2023. vol. 14005, pp. 379–404 (2022)
23. Chiesa, A., Guan, Z., Samocha, S., Yogev, E.: Security bounds for proof-carrying data from straightline extractors. In: Theory of Cryptography Conference. pp. 464–496. Springer (2024)
24. Chiesa, A., Hu, Y., Maller, M., Mishra, P., Vesely, N., Ward, N.: Marlin: Preprocessing zksnarks with universal and updatable srs. In: EUROCRYPT 2020. LNCS, vol. 12105, pp. 738–768. Springer (2020)
25. Chiesa, A., Ojha, D., Spooner, N.: Fractal: Post-quantum and transparent recursive proofs from holography. In: EUROCRYPT 2020 Part I. LNCS, vol. 12105, pp. 769–793. Springer (2020)
26. Chiesa, A., Tromer, E.: Proof-carrying data and hearsay arguments from signature cards. In: ICS. vol. 10, pp. 310–331 (2010)
27. Coda: `https://codaprotocol.com`
28. Coron, J.: Optimal security proofs for PSS and other signature schemes. In: EUROCRYPT 2002. LNCS, vol. 2332, pp. 272–287. Springer (2002)
29. Explorer, E.E.B.: `https://etherscan.io`
30. Fuchsbauer, G., Kiltz, E., Loss, J.: The algebraic group model and its applications. In: CRYPTO (2) 2018. LNCS, vol. 10992, pp. 33–62. Springer (2018)
31. Garg, S., Bhaskar, R., Lokam, S.V.: Improved bounds on security reductions for discrete log based signatures. In: CRYPTO 2008. LNCS, vol. 5157, pp. 93–107. Springer (2008)
32. Gennaro, R., Gentry, C., Parno, B., Raykova, M.: Quadratic span programs and succinct nizks without pcps. In: EUROCRYPT 2013. pp. 626–645. Springer (2013)
33. Gentry, C., Wichs, D.: Separating succinct non-interactive arguments from all falsifiable assumptions. In: Proceedings of the forty-third annual ACM symposium on Theory of computing. pp. 99–108 (2011)
34. Ghoshal, A., Tessaro, S.: Tight state-restoration soundness in the algebraic group model. In: CRYPTO (3) 2021. LNCS, vol. 12827, pp. 64–93. Springer (2021)
35. Gorbunov, S., Reyzin, L., Wee, H., Zhang, Z.: Pointproofs: Aggregating proofs for multiple vector commitments. In: CCS '20. pp. 2007–2023. ACM (2020)
36. Holmgren, J., Lombardi, A.: Cryptographic hashing from strong one-way functions (or: One-way product functions and their applications). In: 2018 IEEE 59th annual symposium on Foundations of Computer Science (FOCS). pp. 850–858. IEEE (2018)
37. Jaeger, J., Mohan, D.I.: Generic and algebraic computation models: When agm proofs transfer to the ggm. In: Annual International Cryptology Conference. pp. 14–45. Springer (2024)

38. Kattis, A., Bonneau, J.: Proof of necessary work: succinct state verification with fairness guarantees. In: International Conference on Financial Cryptography and Data Security. pp. 18–35. Springer (2023)
39. Khovratovich, D., Maller, M., Tiwari, P.R.: Minroot: Candidate sequential function for ethereum vdf. Cryptology ePrint Archive (2022)
40. Kiltz, E., Masny, D., Pan, J.: Optimal security proofs for signatures from identification schemes. In: CRYPTO (2) 2016. LNCS, vol. 9815, pp. 33–61. Springer (2016)
41. Kim, S., Lee, G., Lee, H., Seo, J.H.: Leopard: Sublinear verifier inner product argument under discrete logarithm assumption. IEEE Transactions on Information Forensics and Security **18**, 5332–5344 (2023)
42. Kim, S., Lee, H., Seo, J.H.: Efficient zero-knowledge arguments in discrete logarithm setting: Sublogarithmic proof or sublinear verifier. In: ASIACRYPT 2022. pp. 403–433. Springer (2023)
43. Kothapalli, A., Setty, S.: Supernova: Proving universal machine executions without universal circuits. Cryptology ePrint Archive (2022)
44. Kothapalli, A., Setty, S.: Hypernova: Recursive arguments for customizable constraint systems. In: Annual International Cryptology Conference. pp. 345–379. Springer (2024)
45. Kothapalli, A., Setty, S., Tzialla, I.: Nova: Recursive zero-knowledge arguments from folding schemes. In: CRYPTO (4) 2022. LNCS, vol. 13510, pp. 359–388. Springer (2022)
46. Lee, H., Paik, S., Son, H., Seo, J.H.: Cougar: Cubic root verifier inner product argument under discrete logarithm assumption. IEEE Access (2026)
47. Lee, J.: Dory: Efficient, transparent arguments for generalised inner products and polynomial commitments. In: TCC 2021, Proceedings, Part II. pp. 1–34. Springer (2021)
48. Lee, J., Nikitin, K., Setty, S.: Replicated state machines without replicated execution. In: 2020 IEEE Symposium on Security and Privacy (SP). pp. 119–134. IEEE (2020)
49. Maller, M., Mohnblatt, N., Zapico, A.: Ivc in the open-and-sign random oracle model. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 333–365. Springer (2025)
50. Maurer, U., Portmann, C., Zhu, J.: Unifying generic group models. Cryptology ePrint Archive, Report 2020/996 p. 996 (2020), `https://eprint.iacr.org/2020/996`
51. Maurer, U.M.: Abstract models of computation in cryptography (invited paper). In: 10th IMA International Conference on Cryptography and Coding. vol. 3796, pp. 1–12. Springer (2005)
52. Nechaev, V.I.: Complexiy of a determinate algorithm for the discrete logarithm. In: Mathematical Notes. vol. 55(2), pp. 165–172 (1994)
53. Nguyen, W., Boneh, D., Setty, S.: Revisiting the nova proof system on a cycle of curves. Cryptology ePrint Archive (2023)
54. Paillier, P., Vergnaud, D.: Discrete-log-based signatures may not be equivalent to discrete log. In: ASIACRYPT 2005. LNCS, vol. 3788, pp. 1–20. Springer (2005)
55. Schnorr, C.P.: Efficient signature generation by smart cards. Journal of cryptology **4**, 161–174 (1991)
56. Setty, S.: Spartan: Efficient and general-purpose zksnarks without trusted setup. In: CRYPTO 2020. LNCS, Springer (2020)
57. Shoup, V.: Lower bounds for discrete logarithms and related problems. In: EUROCRYPT 1997. LNCS, vol. 1233, pp. 256–266. Springer (1997)

58. Valiant, P.: Incrementally verifiable computation or proofs of knowledge imply time/space efficiency. In: TCC 2008. LNCS, vol. 4948, pp. 1–18. Springer (2008)
59. Zhandry, M.: To label, or not to label (in generic groups). In: CRYPTO 2022. vol. 13509, pp. 66–96 (2022)

A Deffered Definitions

Definition 11 (Discrete Logarithm Relation Assumption). *Let $\mathcal{G}$ be a group generator that takes security parameters λ and then outputs $\mathbb{G}$, describing a group of order p. We say that $\mathbb{G}$ satisfies the discrete logarithm relation (DLR) assumption if, for all non-uniform polynomial-time adversaries $\mathcal{A}$, the following inequality holds:*

$$\Pr\left[\begin{array}{l|l} \boldsymbol{a} \neq \mathbf{0} & (p, g, \mathbb{G}) \leftarrow \mathcal{G}(1^\lambda), \boldsymbol{g} \xleftarrow{\$} \mathbb{G}^n; \\ \wedge [\boldsymbol{a}]\boldsymbol{G} = 0_{\mathbb{G}} & \boldsymbol{a} \leftarrow \mathcal{A}(\boldsymbol{g}, p, g, \mathbb{G}) \end{array}\right] \leq \mathsf{negl}(\lambda)$$

It is well-known that the discrete logarithm relation (DLR) assumption is equivalent to the discrete logarithm (DL) assumption [15,42].

Pedersen Commitment for Vectors. The Pedersen commitment scheme is a homomorphic commitment scheme that satisfies perfect hiding and computational binding properties under the DL assumption. The scheme consists of two algorithms $(\mathsf{Gen}_{\mathsf{Ped}}, \mathsf{Com}_{\mathsf{Ped}})$ as detailed in Figure 4. Given a message vector $\boldsymbol{x} \in \mathbb{Z}_p^\ell$ and a random scalar $r \in \mathbb{Z}_p$, the commitment is computed via a multi-scalar multiplication (MSM). The homomorphic property is naturally induced by the linear structure of the cyclic group $\mathbb{G}$, allowing for efficient aggregation of commitments.

Pedersen Vector Commitment $(\mathsf{Gen}_{\mathsf{Ped}}, \mathsf{Com}_{\mathsf{Ped}})$ over message space $\mathbb{Z}_p^\ell$:

- $\underline{\mathsf{Gen}_{\mathsf{Ped}}(1^\lambda, \ell) \to \mathsf{ck}}$:
  1. Sample a vector of generators $\boldsymbol{G} \xleftarrow{\$} \mathbb{G}^\ell$ and a blinding generator $H \xleftarrow{\$} \mathbb{G}$.
  2. Output $\mathsf{ck} := (\boldsymbol{G}, H) \in \mathbb{G}^\ell \times \mathbb{G}$.
- $\underline{\mathsf{Com}_{\mathsf{Ped}}(\mathsf{ck}, \boldsymbol{x}; r) \to C}$:
  1. Parse ck as $(\boldsymbol{G}, H)$.
  2. Compute $C := \langle \boldsymbol{x}, \boldsymbol{G} \rangle + r \cdot H$.
  3. Output $C \in \mathbb{G}$.

Fig. 4. Description of Pedersen Commitment Scheme for Vectors

Argument of Knowledge. Let $\mathcal{R}$ be a polynomial-time verifiable relation consisting of common reference string (CRS), statement, and witness, denoted by σ, x, and w respectively. An interactive argument system for relation $\mathcal{R}$

consists of three probabilistic polynomial-time algorithms (PPTs) $(\mathcal{K}, \mathcal{P}, \mathcal{V})$. The $\mathcal{K}$ algorithm takes the security parameter λ and outputs CRS σ, which is the input of $\mathcal{P}$ and $\mathcal{V}$. $\mathcal{P}$ and $\mathcal{V}$ generate a transcript interactively, denoted by $tr \leftarrow \langle \mathcal{P}(\sigma, x, w), \mathcal{V}(\sigma, x) \rangle$. At the end of the transcript, $\mathcal{V}$ outputs a bit, 0(reject) or 1(accept). An argument of knowledge (AoK) is a special case of an argument system that satisfies the properties of completeness and witness extractability.

Definition 12 (Argument of Knowledge). *Let $\mathcal{R}$ be a polynomial-time verifiable relation. We call that the argument system $(\mathcal{K}, \mathcal{P}, \mathcal{V})$ for the relation $\mathcal{R}$ an Argument of knowledge if the following properties hold:*
[Completeness]: For every PPT adversary $\mathcal{A}$, the following inequality holds:

$$\Pr\left[\begin{array}{l} tr \leftarrow \langle \mathcal{P}(\sigma, x, w), \mathcal{V}(\sigma, x) \rangle \\ tr \text{ is accepting} \end{array} \middle| \begin{array}{r} \sigma \leftarrow \mathcal{K}(1^\lambda); \\ (x, w) \leftarrow \mathcal{A}(\sigma) \\ \wedge (\sigma, x; w) \in \mathcal{R} \end{array}\right] > 1 - \mathsf{negl}(\lambda)$$

[Witness Extended Emulation]: For every deterministic polynomial prover $\mathcal{P}^*$, which may not follow $\mathcal{P}$, and all pairs of PPT adversaries $(\mathcal{A}_1, \mathcal{A}_2)$, there exists a PPT emulator $\mathcal{E}$, the following inequality holds:

$$\left| \begin{array}{l} \Pr\left[\mathcal{A}_1(tr) = 1 \middle| \begin{array}{r} \sigma \leftarrow \mathcal{K}(1^\lambda); \\ (x, s) \leftarrow \mathcal{A}_2(\sigma); \\ tr \leftarrow \langle \mathcal{P}^*(\sigma, x, s), \mathcal{V}(\sigma, x) \rangle \end{array}\right] \\ - \Pr\left[\begin{array}{l} tr \text{ is accepting} \\ \wedge\ \mathcal{A}_1(tr) = 1 \\ \wedge\ (\sigma, w, x) \in \mathcal{R} \end{array} \middle| \begin{array}{r} \sigma \leftarrow \mathcal{K}(1^\lambda); \\ (x, s) \leftarrow \mathcal{A}_2(\sigma); \\ (tr, w) \leftarrow \mathcal{E}^{\mathcal{O}}(\sigma, x) \end{array}\right] \end{array} \right| < \mathsf{negl}(\lambda)$$

The emulator $\mathcal{E}$ can access the oracle $\mathcal{O} = \langle \mathcal{P}^*(\sigma, x, s), \mathcal{V}(\sigma, x) \rangle$, which outputs the transcript between $\mathcal{P}^*$ and $\mathcal{V}$. $\mathcal{E}$ permits to rewind $\mathcal{P}^*$ at a specific round and rerun $\mathcal{V}$ using fresh randomness. s can be considered as the state of $\mathcal{P}^*$ If $\mathcal{E}$ accesses $\mathcal{P}^*$ only one time without rewinding, we call $\mathcal{E}$ is straight-line extractor.

B Nova IVC: Recursive Proof System from Folding Scheme

Nova IVC [45] is an IVC scheme constructed by recursive proof composition via a folding scheme. For each round, the IVC prover produces an execution of F and folds instances concurrently. An informal description of the computation in each round is given in Figure 5, where H is a cryptographic hash function and $(\mathsf{u}_\perp, \mathsf{v}_\perp)$ is a trivial instance-witness pair, i.e., $\mathsf{u}_\perp := (C_{\mathbf{0}}, C_{\mathbf{0}}, 1, 0)$ and $\mathsf{v}_\perp := (\mathbf{0}, r_0, \mathbf{0}, r_0)$ [4]. In addition, we define the trivial proof $\mathit{\Pi}_0 = (\mathsf{u}_\perp, \mathsf{v}_\perp, \mathsf{u}_\perp, \mathsf{v}_\perp)$, which consists of two trivial instance-witness pairs.

Let $\mathsf{NIFS} = (\mathsf{G}, \mathsf{K}, \mathsf{NIFS.P}, \mathsf{NIFS.V})$ be the non-interactive folding scheme for the CR-R1CS of F'. The formal descriptions of the augmented function F' and

[4] Strictly speaking, the two zero vectors have distinct dimensions (m and $m - \ell - 1$, respectively). However, for simplicity, we abuse notation by using $\mathbf{0}$ and $C_{\mathbf{0}}$ regardless of the dimension.

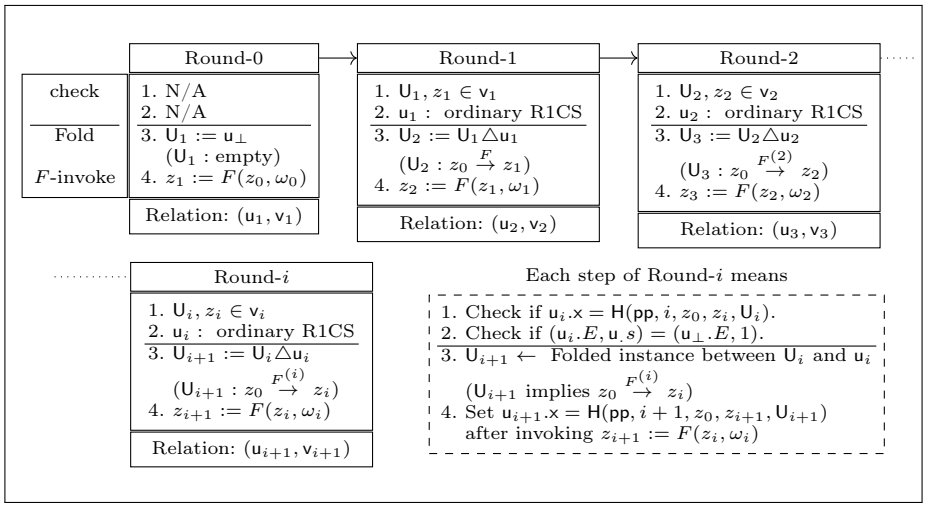


Fig. 5. Informal Description of Relation (u, v) for Each Round of Nova

Nova from NIFS are provided in Figure 7. Here, trace is a compiler that converts an execution of F' on non-deterministic advice $(\mathsf{pp}, \mathsf{U}_i, \mathsf{u}_i, (i, z_0, z_i), \omega_i, T)$ to the corresponding CR-R1CS instance-witness pair $(\mathsf{u}_{i+1}, \mathsf{v}_{i+1})$, where the advice is a part of v_{i+1} and the output hash value of F' is the only public IO of u_{i+1}, that is, $\mathsf{u}_{i+1}.\mathsf{x}$.

We recall the details of the Group-based Non-Interactive Folding Scheme (NIFS) and the original Nova IVC construction employing the augmented function F' [45]. These protocols are presented in Figure 6 and Figure 7, respectively.

$\underline{\mathsf{G}(1^\lambda, N=(m,\ell)) \to \mathsf{pp}_{\mathsf{FS}}}$: Output commitment keys $\mathsf{ck}_{\boldsymbol{e}} \xleftarrow{\$} \mathbb{G}^m$ and $\mathsf{ck}_{\boldsymbol{w}} \xleftarrow{\$} \mathbb{G}^{m-\ell-1}$.

$\underline{\mathsf{K}(\mathsf{pp}_{\mathsf{FS}}, \mathsf{s}=(A,B,C)) \to \mathsf{pk}_{\mathsf{FS}}}$: Output $\mathsf{pk}_{\mathsf{FS}} \leftarrow (\mathsf{pp}_{\mathsf{FS}}, \mathsf{s})$.

$\underline{\mathsf{NIFS.P}(\mathsf{pk}_{\mathsf{FS}}, (\mathsf{u}_1, \mathsf{v}_1), (\mathsf{u}_2, \mathsf{v}_2)) \to (\mathsf{u}, \mathsf{v}), T}$:

1. For $i = 1, 2$, parse $(\mathsf{u}_i, \mathsf{v}_i) = ((E_i, s_i, W_i, \mathsf{x}_i), (\boldsymbol{e}_i, r_{\boldsymbol{e}_i}, \boldsymbol{w}_i, r_{\boldsymbol{w}_i}))$ and then set $\boldsymbol{z}_i = (\boldsymbol{w}_i, \mathsf{x}_i, s_i)$
2. Compute $\boldsymbol{t} = A\boldsymbol{z}_1 \circ B\boldsymbol{z}_2 + A\boldsymbol{z}_2 \circ B\boldsymbol{z}_1 - s_1 \cdot C\boldsymbol{z}_2 - s_2 \cdot C\boldsymbol{z}_1$.
3. Pick $r_{\boldsymbol{t}} \xleftarrow{\$} \mathbb{Z}_p$ and compute $T = \langle (\boldsymbol{t}, r_{\boldsymbol{t}}), \mathsf{ck}_{\boldsymbol{e}} \rangle$.
4. $\gamma \leftarrow \mathsf{H}(s_1, s_2, \mathsf{x}_1, \mathsf{x}_2, E_1, E_2, W_1, W_2, T)$.
5. Output $T, \mathsf{u} := (E, s, W, \mathsf{x})$ and $\mathsf{v} := (\boldsymbol{e}, r_{\boldsymbol{e}}, \boldsymbol{w}, r_{\boldsymbol{w}})$ where

$$E \leftarrow E_1 + \gamma \cdot T + \gamma^2 \cdot E_2 \qquad W \leftarrow W_1 + \gamma \cdot W_2 \tag{5}$$

$$s \leftarrow s_1 + \gamma \cdot s_2 \qquad \mathsf{x} \leftarrow \mathsf{x}_1 + \gamma \cdot \mathsf{x}_2 \tag{6}$$

$$\boldsymbol{e} \leftarrow \boldsymbol{e}_1 + \gamma \cdot \boldsymbol{t} + \gamma^2 \cdot \boldsymbol{e}_2 \qquad \boldsymbol{w} \leftarrow \boldsymbol{w}_1 + \gamma \cdot \boldsymbol{w}_2 \tag{7}$$

$$r_{\boldsymbol{e}} \leftarrow r_{\boldsymbol{e}_1} + \gamma \cdot r_{\boldsymbol{t}} + \gamma^2 \cdot r_{\boldsymbol{e}_2} \qquad r_{\boldsymbol{w}} \leftarrow r_{\boldsymbol{w}_1} + \gamma \cdot r_{\boldsymbol{w}_2} \tag{8}$$

$\underline{\mathsf{NIFS.V}(\mathsf{pp}_{\mathsf{FS}}, \mathsf{u}_1, \mathsf{u}_2, T) \to \mathsf{u}}$:

1. For $i = 1, 2$, parse $\mathsf{u}_i = (E_i, s_i, W_i, \mathsf{x}_i)$
2. $\gamma \leftarrow \mathsf{H}(s_1, s_2, \mathsf{x}_1, \mathsf{x}_2, E_1, E_2, W_1, W_2, T)$
3. Output $\mathsf{u} := (E, s, W, \mathsf{x})$ satisfying Eq. (5) and Eq. (6).

* The blue text denotes randomness for hiding.

Fig. 6. Group-based Non-Interactive Folding Scheme in [43]

(a) Augmented Function F'

$F'(\mathsf{pp}, \mathsf{U}_i, \mathsf{u}_i, (i, z_0, z_i), \omega_i, T) \to \mathsf{x}$:
If i is 0, output $\mathsf{H}(\mathsf{pp}, 1, z_0, F(z_0, \omega_i), \mathsf{u}_\perp)$;
otherwise,

1. check that $\mathsf{u}_i.\mathsf{x} = \mathsf{H}(\mathsf{pp}, i, z_0, z_i, \mathsf{U}_i)$, where $\mathsf{u}_i.\mathsf{x}$ is the public IO of u_i
2. check that $(\mathsf{u}_i.E, \mathsf{u}_i.s) = (\mathsf{u}_\perp.E, 1)$
3. compute $\mathsf{U}_{i+1} \leftarrow \mathsf{NIFS.V}(\mathsf{pp}, \mathsf{U}_i, \mathsf{u}_i, T)$, and
4. output $\mathsf{H}(\mathsf{pp}, i+1, z_0, F(z_i, \omega_i), \mathsf{U}_{i+1})$.

(b) Nova IVC Scheme $(\mathcal{G}, \mathcal{K}, \mathcal{P}, \mathcal{V})$

$\mathcal{G}(1^\lambda) \to \mathsf{pp}$: Output $\mathsf{pp} \leftarrow \mathsf{NIFS.G}(1^\lambda, N)$.

$\mathcal{K}(\mathsf{pp}, F) \to (\mathsf{pk}, \mathsf{vk})$:

1. Run $\mathsf{pk}_{\mathsf{FS}} \leftarrow \mathsf{NIFS.K}(\mathsf{pp}, \mathsf{s})$
2. Output $(\mathsf{pk}, \mathsf{vk}) \leftarrow ((F, \mathsf{s}, \mathsf{pk}_{\mathsf{FS}}), (F, \mathsf{s}, \mathsf{pp}))$

$\mathcal{P}(\mathsf{pk}, (i, z_0, z_i), \omega_i, \Pi_i) \to \Pi_{i+1}$:
Parse Π_i as $((\mathsf{U}_i, \mathsf{V}_i), (\mathsf{u}_i, \mathsf{v}_i))$ and then

1. if i is 0, compute $(\mathsf{U}_{i+1}, \mathsf{V}_{i+1}, T) \leftarrow (\mathsf{u}_\perp, \mathsf{v}_\perp, \mathsf{u}_\perp.E)$;
   otherwise, compute $(\mathsf{U}_{i+1}, \mathsf{V}_{i+1}, T) \leftarrow \mathsf{NIFS.P}(\mathsf{pk}, (\mathsf{U}_i, \mathsf{V}_i), (\mathsf{u}_i, \mathsf{v}_i))$
2. compute $(\mathsf{u}_{i+1}, \mathsf{v}_{i+1}) \leftarrow \mathsf{trace}(F', (\mathsf{vk}, \mathsf{U}_i, \mathsf{u}_i, (i, z_0, z_i), \omega_i, T))$, and
3. output $\Pi_{i+1} \leftarrow ((\mathsf{U}_{i+1}, \mathsf{V}_{i+1}), (\mathsf{u}_{i+1}, \mathsf{v}_{i+1}))$.

$\mathcal{V}(\mathsf{vk}, (i, z_0, z_i), \Pi_i) \to \{0, 1\}$:
If i is 0, check that $z_0 = z_i$;
otherwise,

1. parse Π_i as $((\mathsf{U}_i, \mathsf{V}_i), (\mathsf{u}_i, \mathsf{v}_i))$,
2. check if $\mathsf{u}_i.\mathsf{x} = \mathsf{H}(\mathsf{vk}, i, z_0, z_i, \mathsf{U}_i)$,
3. check if $(\mathsf{u}_i.E, \mathsf{u}_i.s) = (\mathsf{u}_\perp.E, 1)$, and
4. check if $(\mathsf{U}_i, \mathsf{V}_i), (\mathsf{u}_i, \mathsf{v}_i) \in \mathcal{R}_{\mathsf{pp},\mathsf{s}}$, the committed relaxed R1CS induced by F'.

---

* We omit updated statement z_{i+1} produced by $\mathcal{P}_\mathsf{b}$ because z_{i+1} is a part of R1CS witness v_{i+1}. $\mathsf{trace}(F', \mathsf{in})$ outputs ordinary R1CS instance-witness pair for the execution F' on advice in.

Fig. 7. Augmented function F' and Nova IVC Scheme $(\mathcal{G}, \mathcal{K}, \mathcal{P}, \mathcal{V})$.

C Log-depth Knowledge Soundness of Ephemeral Nova (Proof for Theorem 2)

Proof. (*Completeness*): We argue that the $\mathcal{P}_1$'s output Π_{i+1} from the execution F with $(i+1, z_0, z_{i+1}, \Pi_i)$ is valid proof if the i-th proof Π_i is valid. Let $\mathsf{pp} \leftarrow \mathcal{G}(1^\lambda)$, F, and $(\mathsf{pk}, \mathsf{vk}) \leftarrow \mathcal{K}(\mathsf{pp}, F')$ be the public parameters, an IVC execution, and prover/verifier key, respectively. Now, we claim that the IVC proof, which satisfies $\mathcal{V}(\mathsf{vk}, i, z_0, z_i, \Pi_i) = 1$ and $\mathcal{P}(\mathsf{pk}, i, z_0, z_i, \omega_i, \Pi_i) \to \Pi_{i+1}$, implies $\mathcal{V}(\mathsf{vk}, i, z_0, z_{i+1}, \Pi_{i+1}) = 1$, where $z_{i+1} = F(z_i, \omega_i)$. We consider two cases in which the step index i is equal to 0 or not.

Case $(i = 0)$: According to our premise, we know that Π_0 is a trivial valid proof $((\mathsf{u}_\perp, \mathsf{v}_\perp), (\mathsf{u}_\perp, \mathsf{v}_\perp))$, which is accepted by IVC verifier $\mathcal{V}$. Now, we consider the validity of the updated proof Π_1. Let $\mathcal{P}_1$ take the input $(\mathsf{pk}, (1, z_0, z_1, \omega_0, \Pi_0)$ and then get Π_1. From the $\mathcal{P}_1$ in Figure 2, we obtain

$$\Pi_1 = ((\mathsf{u}_\perp, \mathsf{v}_\perp), (\mathsf{u}_1, \mathsf{v}_1))$$

where $(\mathsf{u}_1, \mathsf{v}_1)$ is R1CS instance-witness pair for F'_{tr} execution with input $\mathsf{b}_1 = 1$. Following the execution F'_{tr} in Figure 2, we know

$$\mathsf{u}_1.\mathsf{x} = \mathsf{H}(\mathsf{vk}, 1, z_0, F(z_0, \omega_0), \mathsf{u}_\perp) \tag{9}$$

$$(\mathsf{u}_1.E, \mathsf{u}_1.s) = (\mathsf{u}_\perp.E, 1) \tag{10}$$

From Eq. (9) and Eq. (10), the first and second verifier conditions in Figure 2 hold. To check the third condition, we only consider $(\mathsf{u}_1, \mathsf{v}_1) \in \mathcal{R}_{\mathsf{pp}_{\mathsf{FS}}, \mathsf{s}_{\mathsf{tr}}}$ because $(\mathsf{U}_1, \mathsf{V}_1) = (\mathsf{u}_\perp, \mathsf{v}_\perp)$ is already belong in the relation. From the tracing of F'_{tr}, $(\mathsf{u}_1, \mathsf{v}_1)$ should belong to the committed relaxed R1CS relation. Therefore, we can conclude that the IVC verifier accepts the following proof Π_1, $\mathcal{V}(\mathsf{pp}, 1, z_0, z, \Pi_1) = 1$.

Case $(i \geq 1)$: Suppose that Π_i is a valid IVC proof for verification $\mathcal{V}$ and Π_{i+1} be a proof generated by $\mathcal{P}_1$ with input $(\mathsf{pk}, i, z_0, z_i, \omega_i, \Pi_i)$

Based on the completeness of the underlying folding scheme and the premise that $(\mathsf{u}_i, \mathsf{v}_i)$ and $(\mathsf{U}_i, \mathsf{V}_i)$ are satisfying instance-witness pairs of the relation, we have $(\mathsf{U}_{i+1}, \mathsf{V}_{i+1})$ is a satisfying instance-witness pair of the relation, i.e. $(\mathsf{U}_{i+1}, \mathsf{V}_{i+1}) \in \mathcal{R}_{\mathsf{pp}_{\mathsf{FS}}, \mathsf{s}_{\mathsf{tr}}}$.

From the tracing of F'_{tr} execution with input $(\mathsf{U}_i, \mathsf{u}_i, (i, z_0, z_i), \omega_i, T, 1)$, we have that $\mathsf{u}_{i+1}.\mathsf{x} = \mathsf{H}(\mathsf{pp}, i+1, z_0, z_{i+1}, \mathsf{U}_{i+1})$ and $(\mathsf{u}_{i+1}.E, \mathsf{u}_{i+1}.s) = (\mathsf{u}_\perp.E, 1)$. Therefore, the verifier $\mathcal{V}$ accepts the IVC proof $\Pi_{i+1} = ((\mathsf{U}_{i+1}, \mathsf{V}_{i+1}), (\mathsf{u}_{i+1}, \mathsf{v}_{i+1}))$.

(*Knowledge Soundness*): For a fixed step $n = O(\log \lambda)$, we construct an extractor $\mathcal{E}$ that outputs the sequence of witnesses $(\omega_i)_{i=0}^{n-1}$. Since the activation step $t = \mathsf{poly}(\lambda)$ is polynomially large, we have $n \ll t$. For any step $i < n$, the execution of F'_{tr} successfully updates the state $z_{i+1} = F(z_i, \omega_i)$ regardless of whether b_i is 0 or 1 because the trigger in F'_{tr} cannot activate less than step t. Building on this fact and the soundness proof strategy in [45], we construct $\mathcal{E}$ as follows.

Let $\mathsf{pp} \leftarrow \mathcal{G}(1^\lambda)$. Consider an expected polynomial-time adversary $\mathcal{P}^*$ that, on input pp, outputs a function F. Let $(\mathsf{pk}, \mathsf{vk}) \leftarrow \mathcal{K}(\mathsf{pp}, F)$. Suppose that for a

constant $n \ll t$, $\mathcal{P}^*$ outputs (z_0, z, Π) such that

$$\mathcal{V}(\mathsf{vk}, n, z_0, z, \Pi) = 1.$$

We construct an expected polynomial-time extractor $\mathcal{E}$ which, on input (pp, z_0, z), outputs $(\omega_0, \ldots, \omega_{n-1})$ such that for all $i \leq n$,

$$z_i \leftarrow F(z_{i-1}, \omega_{i-1})$$

holds, where $z_n = z$, with all but negligible probability.

We show that $\mathcal{E}$ can be constructed by inductively running an expected polynomial-time partial extractor $\mathcal{E}_i(\mathsf{pp})$. This partial extractor outputs $((z_i, \ldots, z_{n-1}), (\omega_i, \ldots, \omega_{n-1}), \Pi_i)$ such that for all $j \in \{i+1, \ldots, n\}$,

$$z_j = F(z_{j-1}, \omega_{j-1})$$

and

$$\mathcal{V}(\mathsf{vk}, i, z_0, z_i, \Pi_i) = 1 \tag{11}$$

where $z_n = z$, with all but negligible probability.

The extractor $\mathcal{E}$ operates by backward induction: it runs $\mathcal{E}_n$ first, and then uses the output of $\mathcal{E}_n$ to construct $\mathcal{E}_{n-1}$, repeating this process until reaching $\mathcal{E}_0$.

Base Case ($i = n$): $\mathcal{E}_n(\mathsf{pp}, \rho)$ simply outputs $(\perp, \perp, \Pi_n)$, where Π_n is the proof output by $\mathcal{P}^*(\mathsf{pp}, \rho)$. By the premise, $\mathcal{E}_n$ succeeds in obtaining a valid proof Π_n from the IVC adversary $\mathcal{P}^*$.

Inductive Step ($i \to i-1$): For $i \geq 1$, suppose $\mathcal{E}$ can construct an expected polynomial-time extractor $\mathcal{E}_i$ that outputs $((z_i, \ldots, z_{n-1}), (\omega_i, \ldots, \omega_{n-1}), \Pi_i)$ satisfying the inductive hypothesis. To construct the extractor $\mathcal{E}_{i-1}$, $\mathcal{E}$ first constructs an adversary $\tilde{\mathcal{A}}_{i-1}$ against the underlying non-interactive folding scheme as follows:

$\underline{\tilde{\mathcal{A}}_{i-1}(\mathsf{pp}, \rho)}$:

1. Run $((z_i, \ldots, z_{n-1}), (\omega_i, \ldots, \omega_{n-1}), \Pi_i) \leftarrow \mathcal{E}_i(\mathsf{pp}, \rho)$.
2. Parse Π_i as $((\mathsf{U}_i, \mathsf{V}_i), (\mathsf{u}_i, \mathsf{v}_i))$.
3. Parse v_i to retrieve $(\mathsf{U}_{i-1}, \mathsf{u}_{i-1}, T_{i-1})$.
4. Output $(\mathsf{U}_{i-1}, \mathsf{u}_{i-1})$ and $((\mathsf{U}_i, \mathsf{V}_i), T_{i-1})$.

By the inductive hypothesis, we have $\mathcal{V}(\mathsf{vk}, i, z_0, z_i, \Pi_i) = 1$, where $\Pi_i \leftarrow \mathcal{E}_i(\mathsf{pp})$, with all but negligible probability. Therefore, the verifier's checks ensure that $(\mathsf{u}_i, \mathsf{v}_i)$ and $(\mathsf{U}_i, \mathsf{V}_i)$ are satisfying instance-witness pairs. Specifically, it holds that:

$$\mathsf{u}_i.\mathsf{x} = \mathsf{H}(\mathsf{vk}, i, z_0, z_i, \mathsf{U}_i).$$

Since $\mathcal{V}$ ensures that $(\mathsf{u}_i.E, \mathsf{u}_i.u) = (\mathsf{u}_\perp.E, 1)$, the pair $(\mathsf{u}_i, \mathsf{v}_i)$ is indeed a valid R1CS instance-witness pair for the augmented circuit F'_{tr}. By the construction of F'_{tr} (which contains the NIFS verification logic), we have:

$$\mathsf{U}_i = \mathsf{NIFS.V}(\mathsf{vk}, \mathsf{U}_{i-1}, \mathsf{u}_{i-1}, T_{i-1})$$

with all but negligible probability. Consequently, $\tilde{\mathcal{A}}_{i-1}$ succeeds in producing an accepting folded instance-witness pair $(\mathsf{U}_i, \mathsf{V}_i)$ for the instances $(\mathsf{U}_{i-1}, \mathsf{u}_{i-1})$ in expected polynomial time.

By the knowledge soundness of the underlying folding scheme NIFS, there exists a folding extractor $\mathcal{E}_{i-1}$. Given the expected polynomial-time adversary $\tilde{\mathcal{A}}_{i-1}$ and the folding extractor $\tilde{\mathcal{E}}_{i-1}^{\tilde{\mathcal{A}}_{i-1}}$, $\mathcal{E}$ constructs an expected polynomial-time extractor $\mathcal{E}_{i-1}$ as follows:

$\underline{\mathcal{E}_{i-1}(\mathsf{pp}, \rho)}$:

1. Run $((\mathsf{U}_{i-1}, \mathsf{u}_{i-1}), (\mathsf{U}_i, \mathsf{V}_i), T_{i-1}) \leftarrow \tilde{\mathcal{A}}_{i-1}(\mathsf{pp}, \rho)$.
2. Retrieve $((z_i, \ldots, z_{n-1}), (\omega_i, \ldots, \omega_{n-1}), \mathit{\Pi}_i)$ from the internal state of $\tilde{\mathcal{A}}_{i-1}$.
3. Parse $\mathit{\Pi}_i.\mathsf{v}_i$ to retrieve z_{i-1} and ω_{i-1}.
4. Let $(\mathsf{v}_{i-1}, \mathsf{V}_{i-1}) \leftarrow \tilde{\mathcal{E}}_{i-1}(\mathsf{pp}, \rho)$.
5. Set $\mathit{\Pi}_{i-1} \leftarrow ((\mathsf{U}_{i-1}, \mathsf{V}_{i-1}), (\mathsf{u}_{i-1}, \mathsf{v}_{i-1}))$.
6. Output $((z_{i-1}, \ldots, z_{n-1}), (\omega_{i-1}, \ldots, \omega_{n-1}), \mathit{\Pi}_{i-1})$.

We first reason that the extracted values $(z_{i-1}, \ldots, z_{n-1})$ and $(\omega_{i-1}, \ldots, \omega_{n-1})$ are valid. By the inductive hypothesis, for all $j \in \{i+1, \ldots, n\}$, it holds that $z_j = F(z_{j-1}, \omega_{j-1})$, and $\mathcal{V}(\mathsf{vk}, i, z_0, z_i, \mathit{\Pi}_i) = 1$ with all but negligible probability.

Due to the knowledge soundness of the folding scheme NIFS, $\mathcal{E}_{i-1}$ extracts a valid R1CS instance-witness pair $(\mathsf{u}_{i-1}, \mathsf{v}_{i-1})$ for F'_{tr}. The satisfiability of F'_{tr} implies the following relations:

$$\begin{aligned} z_i &= F(z_{i-1}, \omega_{i-1}) \\ \mathsf{u}_{i-1}.\mathsf{x} &= \mathsf{H}(\mathsf{vk}, i-1, z_0, z_{i-1}, \mathsf{U}_{i-1}) \\ (\mathsf{u}_{i-1}.E, \mathsf{u}_{i-1}.u) &= (\mathsf{u}_\perp.E, 1) \end{aligned}$$

From the second and third conditions, it follows that $\mathcal{V}(\mathsf{vk}, i-1, z_0, z_{i-1}, \mathit{\Pi}_{i-1}) = 1$. Thus, the output of $\mathcal{E}_{i-1}$ is valid. This completes the construction of the partial extractor $\mathcal{E}_{i-1}$.

By inductively running the extraction from $\mathcal{E}_n$ down to $\mathcal{E}_0$, $\mathcal{E}$ outputs $((z_0, \ldots, z_{n-1}), (\omega_0, \ldots, \omega_{n-1}))$ satisfying $z_{i+1} = F(z_i, \omega_i)$ for all $i = 0, \ldots, n-1$ with all but negligible probability. □

D Straight-line Extraction of Schnorr's NIZK

As a warm-up example to demonstrate the effectiveness of the GZT property, we present a new knowledge-soundness proof for Schnorr's NIZK protocol [55]. This

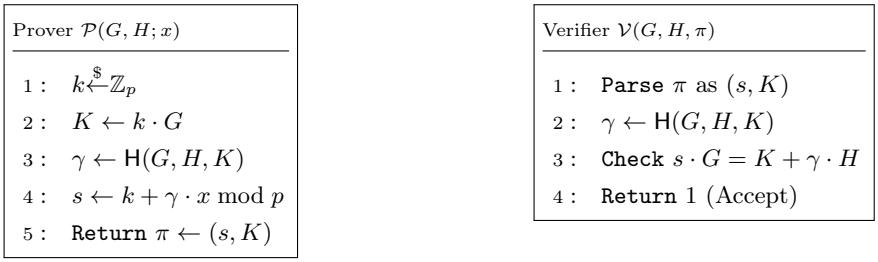


Fig. 8. Schnorr's NIZK

serves as a concrete demonstration of how the GZT property enables straight-line extraction in the AGM without rewinding.

The protocol proves knowledge of the discrete logarithm x for a statement (G, H) such that $H = x \cdot G$.

Using the GZT property, we prove that Schnorr's NIZK protocol is knowledge sound in the AGM. Crucially, the extraction is straight-line and tight.

Theorem 6. *Let* H *be a hash function satisfying the* ***General Zero-Testing Property*** *with respect to the mappings* (Γ, Ψ) *defined as follows:*

- $\Gamma(k_1, k_2) := K$*, where* $K = k_1 \cdot G + k_2 \cdot H$*.*
- $\Psi(k_1, k_2) := P(X) = k_2 + X$*.*

Then, Schnorr's NIZK protocol satisfies knowledge soundness in the AGM. Specifically, the running time of the extractor is equivalent to that of the algebraic prover, except for a constant number of operations.

Proof. Given an arbitrary algebraic prover $\mathcal{P}^*$, we construct an extractor $\mathcal{E}$ that extracts the witness x. $\mathcal{P}^*$ takes a pair (G, H) as input and outputs a proof (s, K) that passes verification. Since the proof is valid, the equality $s \cdot G = K + e \cdot H$ holds, where $e \leftarrow \mathsf{H}(G, H, K)$. Note that we regards G, H as auxiliary input of H.

Since $\mathcal{P}^*$ is an algebraic adversary in the AGM, it must output the representation $(k_1, k_2) \in \mathbb{Z}_p^2$ of the group element K such that $K = k_1 \cdot G + k_2 \cdot H$ alongside K. Substituting this representation into the verification equation:

$$s \cdot G = (k_1 \cdot G + k_2 \cdot H) + \gamma \cdot H = k_1 \cdot G + (k_2 + \gamma) \cdot H.$$

Rearranging the terms, we obtain:

$$(s - k_1) \cdot G = (k_2 + \gamma) \cdot H.$$

To extract the valid witness, we compute $x = (s - k_1)(k_2 + \gamma)^{-1} \bmod p$. This computation is valid only if the denominator $(k_2 + \gamma)$ is non-zero.

Consider the polynomial $P(X) = \Psi(k_1, k_2) = k_2 + X$. Since the polynomial is linear and monic, it is not identically zero ($P(X) \not\equiv 0$). If $k_2 + \gamma \equiv 0 \pmod p$,

then $P(\gamma) = 0$, which means the adversary has found a pre-image (k_1, k_2) whose hash output is a root of the non-zero polynomial $P(X)$.

By the GZT property of H, finding such inputs is computationally infeasible. Therefore, with overwhelming probability, $k_2 + \gamma \not\equiv 0 \pmod{p}$, and the extractor successfully recovers the witness x. □

E Knowledge Soundness of NIFS (Proof for Theorem 4)

Proof. For simplicity, we analyze the NIFS instantiated with a non-hiding Pedersen commitment scheme; that is, we omit the randomness factor in the commitments, that are corresponding to blue part in Figure 6. However, due to the homomorphic property of Pedersen commitments, the core logic for witness extraction remains identical for the hiding Pedersen-based NIFS.

To prove the knowledge soundness of NIFS, we construct an extractor $\mathcal{E}$ that outputs valid witnesses from the algebraic adversary $\tilde{\mathcal{A}} = (\mathcal{A}, \mathsf{P}^*)$'s output $(\mathsf{u}_1, \mathsf{u}_2, \mathsf{u}, \mathsf{v}, T)$. Before proceeding, we recall the notation for instances and witnesses:

$$\begin{aligned}\mathsf{u}_i &= (E_i, s_i, W_i, \mathsf{x}_i) \in \mathbb{G} \times \mathbb{Z}_p \times \mathbb{G} \times \mathbb{Z}_p^{\ell},\\ \mathsf{v}_i &= (\boldsymbol{e}_i, \boldsymbol{w}_i) \in \mathbb{Z}_p^{m} \times \mathbb{Z}_p^{m-\ell-1}, \quad \text{for } i = 1, 2,\end{aligned}$$

where m and ℓ are pre-designated parameters of G.

Now, we define mappings for the GZT property. For a given R1CS structure s, let the input domain be $\mathcal{D} = \{0,1\}^* \times (\mathbb{Z}_p^m \times \mathbb{Z}_p^{m-\ell-1})^5$. We parse the input vector as $(\mathsf{in}, \tilde{\boldsymbol{e}}_1, \tilde{\boldsymbol{w}}_1, \tilde{\boldsymbol{e}}_2, \tilde{\boldsymbol{w}}_2, \tilde{\boldsymbol{t}})$, where $\tilde{\boldsymbol{e}}_i = (\boldsymbol{e}_i, \bar{\boldsymbol{e}}_i)$ and $\tilde{\boldsymbol{w}}_i = (\bar{\boldsymbol{w}}_i, \boldsymbol{w}_i)$.

We define the mappings $(\Gamma_{\mathsf{s}}, \Psi_{\mathsf{s}})$ as follows.

First, the inner mapping $\Gamma_{\mathsf{s}} : \mathcal{D} \to \mathbb{Z}_p^4 \times \mathbb{G}^5$ computes the instance and commitment elements:

$$\Gamma_{\mathsf{s}}(\mathsf{in}, \tilde{\boldsymbol{e}}_1, \tilde{\boldsymbol{w}}_1, \tilde{\boldsymbol{e}}_2, \tilde{\boldsymbol{w}}_2, \tilde{\boldsymbol{t}}) = (\mathsf{in}, E_1, W_1, E_2, W_2, T), \tag{12}$$

where $E_i = \langle \tilde{\boldsymbol{e}}_i, \boldsymbol{B} \rangle$, $W_i = \langle \tilde{\boldsymbol{w}}_i, \boldsymbol{B} \rangle$, and $T = \langle \tilde{\boldsymbol{t}}, \boldsymbol{B} \rangle$ with the basis $\boldsymbol{B} := \mathsf{ck}_{\boldsymbol{e}} \| \mathsf{ck}_{\boldsymbol{w}}$ (folding scheme parameter).

Second, the polynomial mapping $\Psi_{\mathsf{s}} : \mathcal{D} \to (\mathbb{Z}_p[X])^3$ outputs a vector of three polynomials $(P_{\bar{\boldsymbol{e}}}, P_{\bar{\boldsymbol{w}}}, P_{\mathsf{stmt}})$ defined as:

$$P_{\bar{\boldsymbol{e}}}(X) := \bar{\boldsymbol{e}}_1 + \bar{\boldsymbol{t}} X + \bar{\boldsymbol{e}}_2 X^2, \tag{13a}$$

$$P_{\bar{\boldsymbol{w}}}(X) := \bar{\boldsymbol{w}}_1 + \bar{\boldsymbol{w}}_2 X, \tag{13b}$$

$$\begin{aligned}P_{\mathsf{stmt}}(X) := {} & (A\boldsymbol{z}_1 \circ B\boldsymbol{z}_1 - s_1 C\boldsymbol{z}_1 - \boldsymbol{e}_1) \\ & + (A\boldsymbol{z}_1 \circ B\boldsymbol{z}_2 + A\boldsymbol{z}_2 \circ B\boldsymbol{z}_1 - s_1 C\boldsymbol{z}_2 - s_2 C\boldsymbol{z}_1 - \boldsymbol{t})X \\ & + (A\boldsymbol{z}_2 \circ B\boldsymbol{z}_2 - s_2 C\boldsymbol{z}_2 - \boldsymbol{e}_2)X^2.\end{aligned} \tag{13c}$$

Here, $\mathsf{in} = (s_1, s_2, \mathsf{x}_1, \mathsf{x}_2)$ and $\boldsymbol{z}_i = (\boldsymbol{w}_i, \mathsf{x}_i, 1)$. Note that the relation is sparse because the maximum degree of the polynomials is at most 2.

$\mathcal{A}$ takes the public parameter $\mathsf{pp}_{FS} = (\mathsf{ck}_{\boldsymbol{e}} \| \mathsf{ck}_{\boldsymbol{w}})$ and outputs a pair of instances $(\mathsf{u}_1, \mathsf{u}_2)$. Since $\mathcal{A}$ is algebraic, it also outputs representations $(\tilde{\boldsymbol{e}}_1, \tilde{\boldsymbol{e}}_2, \tilde{\boldsymbol{w}}_1, \tilde{\boldsymbol{w}}_2)$ relative to the base $\boldsymbol{B}$. P^* takes $\mathsf{pp}_{FS}, (\mathsf{u}_1, \mathsf{u}_2), \mathsf{st}$ as input and outputs a folded instance-witness pair $(\mathsf{u}, \mathsf{v}) = ((E, s, W, \mathsf{x}), (\boldsymbol{e}, \boldsymbol{w}))$ along with a folding proof T. Being algebraic, P^* also provides the representation $\tilde{\boldsymbol{t}}$ for T.

Consider the case where the verification holds: $\mathsf{V}(\mathsf{pp}_{FS}, \mathsf{u}_1, \mathsf{u}_2, T) = \mathsf{u}$. We claim that the provided representations $(\tilde{\boldsymbol{e}}_1, \tilde{\boldsymbol{e}}_2, \tilde{\boldsymbol{w}}_1, \tilde{\boldsymbol{w}}_2)$ yield valid witnesses $\mathsf{v}_1, \mathsf{v}_2$.

From the folding verification equations, the following relations hold for the folded instance:

$$E = E_1 + \gamma T + \gamma^2 E_2, \quad W = W_1 + \gamma W_2, \tag{14}$$

$$E = \langle \boldsymbol{e}, \mathsf{ck}_{\boldsymbol{e}} \rangle, \quad W = \langle \boldsymbol{w}, \mathsf{ck}_{\boldsymbol{w}} \rangle \tag{15}$$

where $\gamma = \mathsf{H}(s_1, s_2, \mathsf{x}_1, \mathsf{x}_2, E_1, E_2, W_1, W_2, T)$. Note that the witness $\mathsf{v} = (\boldsymbol{e}, \boldsymbol{w})$ for the folded instance must be well-formed (i.e., defined over the correct sub-bases).

The algebraic adversary provides representations $\tilde{\boldsymbol{e}}_i, \tilde{\boldsymbol{w}}_i, \tilde{\boldsymbol{t}}$ for E_i, W_i, T over the full base $\boldsymbol{B} = \mathsf{ck}_{\boldsymbol{e}} \| \mathsf{ck}_{\boldsymbol{w}}$. We parse these vectors as:

$$\tilde{\boldsymbol{e}}_i = (\boldsymbol{e}_i, \bar{\boldsymbol{e}}_i), \quad \tilde{\boldsymbol{w}}_i = (\bar{\boldsymbol{w}}_i, \boldsymbol{w}_i), \quad \tilde{\boldsymbol{t}} = (\boldsymbol{t}, \bar{\boldsymbol{t}}),$$

where $\boldsymbol{e}_i, \boldsymbol{t} \in \mathbb{Z}_p^m$ and $\bar{\boldsymbol{e}}_i, \bar{\boldsymbol{t}}, \bar{\boldsymbol{w}}_i, \boldsymbol{w}_i \in \mathbb{Z}_p^{m-\ell-1}$. Substituting these representations into Eq. (14) and comparing with Eq. (15), and relying on DL assumption (which implies unique representations relative to the basis), we obtain the following linear relations in $\mathbb{Z}_p$:

$$\boldsymbol{e} = \boldsymbol{e}_1 + \gamma \boldsymbol{t} + \gamma^2 \boldsymbol{e}_2, \qquad \boldsymbol{0} = \bar{\boldsymbol{e}}_1 + \gamma \bar{\boldsymbol{t}} + \gamma^2 \bar{\boldsymbol{e}}_2,$$
$$\boldsymbol{0} = \bar{\boldsymbol{w}}_1 + \gamma \bar{\boldsymbol{w}}_2, \qquad \boldsymbol{w} = \boldsymbol{w}_1 + \gamma \boldsymbol{w}_2.$$

Specifically, for the garbage components (the parts of the witness that should be zero), we have:

$$P_{\bar{\boldsymbol{e}}}(\gamma) = 0 \quad \text{and} \quad P_{\bar{\boldsymbol{w}}}(\gamma) = 0.$$

By the GZT property of H (Eq. (13a), Eq. (13b)), since γ is a root, the polynomials $P_{\bar{\boldsymbol{e}}}(X)$ and $P_{\bar{\boldsymbol{w}}}(X)$ must be identically zero. This implies $\bar{\boldsymbol{e}}_i = \boldsymbol{0}$ and $\bar{\boldsymbol{w}}_i = \boldsymbol{0}$. Therefore, the extracted representations are well-formed witnesses with respect to the correct commitment keys $\mathsf{ck}_{\boldsymbol{e}}$ and $\mathsf{ck}_{\boldsymbol{w}}$.

Finally, we verify that these witnesses satisfy the R1CS relation. From the hypothesis that the folded witness is valid (i.e., the folded R1CS constraints are satisfied):

$$\begin{aligned} 0 &= A\boldsymbol{z} \circ B\boldsymbol{z} - sC\boldsymbol{z} - \boldsymbol{e} \\ &= P_{\mathsf{stmt}}(\gamma). \end{aligned}$$

This implies that the challenge γ is a root of $P_{\mathsf{stmt}}(X)$. Again, by the GZT property (Eq. (13c)), it is computationally infeasible for the adversary to force γ

to be a root unless $P_{\mathsf{stmt}}(X) \equiv 0$. The condition $P_{\mathsf{stmt}}(X) \equiv 0$ implies that each coefficient is zero:

$$\text{Constant term: } A\boldsymbol{z}_1 \circ B\boldsymbol{z}_1 - s_1 C\boldsymbol{z}_1 - \boldsymbol{e}_1 = 0,$$
$$\text{Quadratic term: } A\boldsymbol{z}_2 \circ B\boldsymbol{z}_2 - s_2 C\boldsymbol{z}_2 - \boldsymbol{e}_2 = 0.$$

These conditions correspond exactly to the R1CS satisfaction checks for $(\mathsf{u}_1, \mathsf{v}_1)$ and $(\mathsf{u}_2, \mathsf{v}_2)$. Thus, $\mathcal{E}$ extracts valid witnesses.

The extraction failure probability is bounded by the probability that γ is a root of non-zero polynomials $P_{\bar{\boldsymbol{e}}}, P_{\bar{\boldsymbol{w}}}, P_{\mathsf{stmt}}$. Thus, the knowledge soundness error is equivalent to the GZT failure probability. □

F Proof of Poly-Depth Knowledge Soundness for Nova IVC (Proof for Theorem 5)

Proof. To establish the polynomial-depth Knowledge Soundness (KS) of the Nova IVC scheme for a polynomial number of steps $n = \mathsf{poly}(\lambda)$, we construct an extractor $\mathcal{E}$ that derives the sequence of local inputs $(\omega_i)_{i=0}^{n-1}$ from the valid outputs provided by the extended algebraic adversary $\mathcal{P}^*$. Let $\mathsf{pp} = \mathsf{pp}_{FS} = (\mathsf{ck}_{\boldsymbol{e}}, \mathsf{ck}_{\boldsymbol{w}})$ be the IVC public parameters generated by the $\mathcal{G}$ algorithm, and let pk, vk denote the proving and verification keys generated by $\mathcal{K}$ (as defined in Figure 7). Note that pk and vk contain the R1CS structure s for the augmented function F'.

1. Setup: LAB Function and GZT Hash. We define the Linear Algebraic Binding (LAB) function $\mathsf{F}_{\mathsf{lab}}$ as the restriction of the in-circuit folding verifier $\mathsf{NIFS.V}$ to its group element outputs. Specifically, while the full verifier $\mathsf{NIFS.V}$ outputs a complete folded instance $\mathsf{u} = (\mathsf{in}, E, W)$ containing non-group elements in (e.g., scalar s and public IO x), we define $\mathsf{F}_{\mathsf{lab}} := \mathsf{NIFS.V}_{\mathbb{G}}$ to capture only the algebraic group components (E, W). Concretely, the algorithm $\mathsf{NIFS.V}_{\mathbb{G}}$ is defined as follows:

$\underline{\mathsf{NIFS.V}_{\mathbb{G}}(\mathsf{pp}_{\mathsf{FS}}, s_1, s_2, \mathsf{x}_1, \mathsf{x}_2, E_1, W_1, E_2, W_2, T) \to (E, W)}$:

1. $\gamma \leftarrow \mathsf{H}(s_1, s_2, \mathsf{x}_1, \mathsf{x}_2, E_1, E_2, W_1, W_2, T)$
2. Output (E, W) satisfying $E \leftarrow E_1 + \gamma \cdot T + \gamma^2 \cdot E_2,\ W \leftarrow W_1 + \gamma \cdot W_2$.

The linear algebraic property of $\mathsf{NIFS.V}_{\mathbb{G}}$ can be verified directly, as the output consists of linear combinations of the input group elements. To ensure that $\mathsf{NIFS.V}_{\mathbb{G}}$ satisfies the binding property (thereby qualifying as a LAB function for the EAGM), we enforce specific GZT conditions on the hash function H.

We require that H satisfies the following GZT properties:

1. **Binding of $\mathsf{F}_{\mathsf{lab}}$:** To uniquely determine the inputs of $\mathsf{F}_{\mathsf{lab}} = \mathsf{NIFS.V}_{\mathbb{G}}$ from its output, the function must be collision-resistant. We prove this using GZT over *two* sets of inputs (via 2-variable polynomials).

2. **Validity of NIFS:** To ensure the folded witness satisfies the relation, we require the GZT property over *one* set of inputs (via 1-variable polynomials).

Now, we define mappings $(\Gamma_{\mathsf{vk}}, \Psi_{\mathsf{vk}})$. Both mappings takes IVC verifier key $\mathsf{vk} = (F, \mathsf{s}, \mathsf{pp})$, which containing R1CS structure of F' and commitment key $\mathsf{pp} = (\mathsf{ck}_{\boldsymbol{e}}, \mathsf{ck}_{\boldsymbol{w}})$. We define base groups $\boldsymbol{B} = \mathsf{ck}_{\boldsymbol{e}} \parallel \mathsf{ck}_{\boldsymbol{w}}$.

Inner Mapping Γ_{vk}: We define the input domain $\mathcal{D}$ for the hash. The mapping $\Gamma_{\mathsf{vk}} : \mathcal{D} \to \mathcal{C}$ processes a *single* input tuple to produce the commitment elements used in the hash:

$$\Gamma_{\mathsf{vk}}(\tilde{\boldsymbol{e}}_1, \tilde{\boldsymbol{w}}_1, \tilde{\boldsymbol{e}}_2, \tilde{\boldsymbol{w}}_2, \tilde{\boldsymbol{t}}) = (E_1, W_1, E_2, W_2, T), \tag{16}$$

where $E_i = \langle \tilde{\boldsymbol{e}}_i, \boldsymbol{B} \rangle$, $W_i = \langle \tilde{\boldsymbol{w}}_i, \boldsymbol{B} \rangle$, and $T = \langle \tilde{\boldsymbol{t}}, \boldsymbol{B} \rangle$. For binding analysis, we consider two distinct input sets denoted by $\mathsf{Gp.in}^{(k)} = (\tilde{\boldsymbol{e}}_1^{(k)}, \tilde{\boldsymbol{w}}_1^{(k)}, \tilde{\boldsymbol{e}}_2^{(k)}, \tilde{\boldsymbol{w}}_2^{(k)}, \tilde{\boldsymbol{t}}^{(k)})$ for $k \in \{1, 2\}$.

Polynomial Mapping Ψ_{vk}: The mapping outputs a vector of polynomials including 1-variable polynomials (for validity checks) and 2-variable polynomials (for binding checks).

$$P_{\mathsf{valid}}^{(k)}(X) := (P_{\tilde{\boldsymbol{e}}}, P_{\tilde{\boldsymbol{w}}}, P_{\mathsf{stmt}}) \quad \text{(defined for } \mathsf{Gp.in}^{(k)} \text{ as in App. E)}, \tag{17a}$$

$$P_{\mathsf{bind},\boldsymbol{e}}(X, Y) := \underbrace{(\tilde{\boldsymbol{e}}_1^{(1)} + \tilde{\boldsymbol{t}}^{(1)} X + \tilde{\boldsymbol{e}}_2^{(1)} X^2)}_{\text{Folded } \boldsymbol{e} \text{ from } \mathsf{Gp.in}^{(1)}} - \underbrace{(\tilde{\boldsymbol{e}}_1^{(2)} + \tilde{\boldsymbol{t}}^{(2)} Y + \tilde{\boldsymbol{e}}_2^{(2)} Y^2)}_{\text{Folded } \boldsymbol{e} \text{ from } \mathsf{Gp.in}^{(2)}}, \tag{17b}$$

$$P_{\mathsf{bind},\boldsymbol{w}}(X, Y) := \underbrace{(\tilde{\boldsymbol{w}}_1^{(1)} + \tilde{\boldsymbol{w}}_2^{(1)} X)}_{\text{Folded } \boldsymbol{w} \text{ from } \mathsf{Gp.in}^{(1)}} - \underbrace{(\tilde{\boldsymbol{w}}_1^{(2)} + \tilde{\boldsymbol{w}}_2^{(2)} Y)}_{\text{Folded } \boldsymbol{w} \text{ from } \mathsf{Gp.in}^{(2)}}. \tag{17c}$$

Lemma 1. *The function $\mathsf{F}_{\mathsf{lab}} = \mathsf{NIFS.V}_{\mathbb{G}}$ satisfies the binding property (as defined in Def. 6) provided that the underlying hash function H satisfies the GZT property with respect to $(\Gamma_{\mathsf{vk}}, \Psi_{\mathsf{vk}})$ and the DL assumption holds in the AGM.*

Proof. Suppose that an algebraic adversary $\mathcal{A}_{bind}$ outputs a collision for $\mathsf{NIFS.V}_{\mathbb{G}}$. That is, it produces two distinct inputs $(\mathsf{in}^{(1)}, \mathsf{Gp.in}^{(1)}) \neq (\mathsf{in}^{(2)}, \mathsf{Gp.in}^{(2)})$ that yield the same output (E, W), where $\mathsf{Gp.in}^{(k)} := (E_1^{(k)}, W_1^{(k)}, E_2^{(k)}, W_2^{(k)}, T^{(k)})$ for $k \in \{1, 2\}$.

Since the adversary is algebraic, it must output the representations (preimages) of the group elements in $\mathsf{Gp.in}^{(k)}$ with respect to the basis $\boldsymbol{B}$. Let us denote these representations as $\mathsf{Rep}^{(k)} := (\tilde{\boldsymbol{e}}_1^{(k)}, \tilde{\boldsymbol{w}}_1^{(k)}, \tilde{\boldsymbol{e}}_2^{(k)}, \tilde{\boldsymbol{w}}_2^{(k)}, \tilde{\boldsymbol{t}}^{(k)})$, which satisfy $\Gamma_{\mathsf{vk}}(\mathsf{Rep}^{(k)}) = \mathsf{Gp.in}^{(k)}$. Assuming the binding property of the Pedersen commitment (guaranteed by the DL assumption), the distinctness of the inputs $\mathsf{Gp.in}^{(1)} \neq \mathsf{Gp.in}^{(2)}$ implies the distinctness of their representations $\mathsf{Rep}^{(1)} \neq \mathsf{Rep}^{(2)}$.

Let the challenges be $\gamma_k = \mathsf{H}(\mathsf{in}^{(k)}, \mathsf{Gp.in}^{(k)})$. The collision condition on the outputs implies:

$$E_1^{(1)} + \gamma_1 T^{(1)} + \gamma_1^2 E_2^{(1)} = E_1^{(2)} + \gamma_2 T^{(2)} + \gamma_2^2 E_2^{(2)},$$
$$W_1^{(1)} + \gamma_1 W_2^{(1)} = W_1^{(2)} + \gamma_2 W_2^{(2)}.$$

Crucially, by the homomorphic property of the inner mapping (Pedersen commitment), these group elements are linear combinations of the representation vectors relative to the basis $\boldsymbol{B}$. We can rewrite the first equation in terms of the representations:

$$\langle(\tilde{\boldsymbol{e}}_1^{(1)} + \gamma_1\tilde{\boldsymbol{t}}^{(1)} + \gamma_1^2\tilde{\boldsymbol{e}}_2^{(1)}) - (\tilde{\boldsymbol{e}}_1^{(2)} + \gamma_2\tilde{\boldsymbol{t}}^{(2)} + \gamma_2^2\tilde{\boldsymbol{e}}_2^{(2)}), \boldsymbol{B}\rangle = 0.$$

Using the binding polynomial $P_{\mathsf{bind},\boldsymbol{e}}$ defined in Eq. (17b), this is equivalent to $\langle P_{\mathsf{bind},\boldsymbol{e}}(\gamma_1, \gamma_2), \boldsymbol{B}\rangle = 0$.

Under the DL assumption, finding a non-trivial relation between the basis elements is computationally hard. Therefore, the scalar coefficient vector must be zero:

$$P_{\mathsf{bind},\boldsymbol{e}}(\gamma_1, \gamma_2) = 0.$$

This implies that the pair (γ_1, γ_2) is a root of the polynomial $P_{\mathsf{bind},\boldsymbol{e}}(X, Y)$. By the GZT property of H, finding such a pair is computationally infeasible unless the polynomial is identically zero, i.e., $P_{\mathsf{bind},\boldsymbol{e}}(X, Y) \equiv 0$.

If $P_{\mathsf{bind},\boldsymbol{e}}(X, Y) \equiv 0$, the coefficients for each power of X and Y must match independently. This implies $\tilde{\boldsymbol{e}}_j^{(1)} = \tilde{\boldsymbol{e}}_j^{(2)}$ and $\tilde{\boldsymbol{t}}^{(1)} = \tilde{\boldsymbol{t}}^{(2)}$. Applying the same logic to the witness component W via Eq. (17c) implies $\tilde{\boldsymbol{w}}_j^{(1)} = \tilde{\boldsymbol{w}}_j^{(2)}$. Consequently, $\mathsf{Rep}^{(1)} = \mathsf{Rep}^{(2)}$, which contradicts the assumption that the inputs were distinct. Thus, $\mathsf{F}_{\mathsf{lab}}$ is binding. □

2. Define $\mathsf{F}_{\mathsf{lab}}$-Projection for EAGM. With $\mathsf{F}_{\mathsf{lab}}$ defined and its binding property established, we define the projection map as follows: **Projection $\Pi_{\mathsf{F}_{\mathsf{lab}}}$:**

- **Validity Check:** We define $\Pi_{\mathsf{F}_{\mathsf{lab}}}(\mathsf{vk}, \mathsf{u}, \mathsf{v}) = \perp$ if and only if (u, v) is NOT a valid committed R1CS instance-witness pair for the augmented circuit F'. This entails: 1) (u, v) does not satisfy the CR-R1CS relation, i.e., $(\mathsf{u}, \mathsf{v}) \in \mathcal{R}_{\mathsf{pp},\mathsf{s}}$. 2) The structure is correct for a fresh R1CS instance: $\mathsf{u}.E = C_{\mathbf{0}}$ and $\mathsf{u}.u = 1$. If (u, v) os valid committed R1CS instance-witness pair, follows next extraction step.
- **Extraction:** This function acts as a semantic parser for the execution trace. Since the augmented circuit F' explicitly embeds the folding verifier $\mathsf{NIFS.V}$, a valid witness v inherently contains the assignments for the folding inputs and outputs. The projection $\Pi_{\mathsf{F}_{\mathsf{lab}}}$ performs the following steps:
  1. **Witness Retrieval:** It inspects the witness v to locate the sub-vectors corresponding to the inputs and outputs of the embedded $\mathsf{NIFS.V}$.
  2. **Component Assignment:** It extracts these values and assigns them to the algebraic components of the running instance (U_{i-1}), the fresh R1CS instance (u_{i-1}), and the resulting folded instance (U_i):

$$\begin{aligned} E_1 &\leftarrow \mathsf{U}_{i-1}.E, & W_1 &\leftarrow \mathsf{U}_{i-1}.W, \\ E_2 &\leftarrow \mathsf{u}_{i-1}.E, & W_2 &\leftarrow \mathsf{u}_{i-1}.W, \\ E &\leftarrow \mathsf{U}_i.E, & W &\leftarrow \mathsf{U}_i.W, \end{aligned}$$

  along with the folding proof T and non-group instructions in.

Consequently, the output is the tuple $(\mathsf{in}, E_1, W_1, E_2, W_2, T, E, W)$. The projection holds the following inequality due to the committed R1CS constraints of the circuit F':

$$\Pr\left[\begin{matrix}\mathsf{val} = (\mathsf{in}, E_1, W_1, E_2, W_2, T, E, W) \neq \bot \wedge \\ \mathsf{NIFS.V}_{\mathbb{G}}(\mathsf{in}, E_1, W_1, E_2, W_2, T) \neq (E, W)\end{matrix} \;\middle|\; \mathsf{val} \leftarrow \Pi_{\mathsf{F_{lab}}}(\mathsf{vk}, \mathsf{u}, \mathsf{v})\right] \leq \mathsf{negl}(\lambda). \tag{18}$$

where $\mathsf{in} = (\mathsf{pp}, \mathsf{U}_{i-1}.s, \mathsf{u}_{i-1}.s, \mathsf{U}_{i-1}.\mathsf{x}, \mathsf{u}_{i-1}.\mathsf{x})$.

3. Recursive Extraction via EAGM. We construct the extractor $\mathcal{E}$ by working backwards from step n down to 1 and we illustrate extraction process in Figure 9. Let the algebraic adversary $\mathcal{P}^*$ output a valid IVC proof $\mathit{\Pi}_n := ((\mathsf{U}_n, \mathsf{V}_n), (\mathsf{u}_n, \mathsf{v}_n))$ and the representation list $\mathcal{L}_{alg}$. Since $\mathit{\Pi}_n$ is accepted by the IVC verifier $\mathcal{V}$, which incorporates the $\Phi_{\mathsf{F_{lab}}}$ check, it holds that $\Phi_{\mathsf{F_{lab}}}(\mathsf{vk}, \mathsf{u}_n, \mathsf{v}_n) = 1$.

- **Recursive Step (from $i = n$ down to $i = 1$):** Assume we have a valid folded witness V_i (corresponding to U_i) and a valid R1CS instance-witness pair $(\mathsf{u}_i, \mathsf{v}_i)$ for the augmented function F'.
  - **Step 1: Input Extraction (ω_{i-1}):** Since v_i is verified as a valid R1CS witness, it correctly encodes the execution of the step function $z_i = F(z_{i-1}, \omega_{i-1})$. Consequently, v_i explicitly contains the local input ω_{i-1}. The extractor parses v_i to retrieve this local input ω_{i-1}.
  - **Step 2: Parse via Projection.** Using the projection $\Pi_{\mathsf{F_{lab}}}$, we identify the underlying group elements involved in the folding step from the current witness v_i:

    $$(E_1, W_1, E_2, W_2, T, E, W) \leftarrow \Pi_{\mathsf{F_{lab}}}(\mathsf{vk}, \mathsf{u}_i, \mathsf{v}_i).$$

    Here, (E, W) correspond to the algebraic components of the folded instance U_i for the current step i, while (E_1, W_1) and (E_2, W_2) correspond to the previous accumulator U_{i-1} and the fresh instance u_{i-1}, respectively.
  - **Step 3: Extraction via EAGM.** The folding verification constraint $\mathsf{NIFS.V}(\mathsf{vk}, \mathsf{U}_{i-1}, \mathsf{u}_{i-1}) = \mathsf{U}_i$ is satisfied within the circuit since $\Phi_{\mathsf{F_{lab}}}(\mathsf{vk}, \mathsf{u}_i, \mathsf{v}_i) = 1$. According to the knowledge propagation property of the EAGM (Definition 8), since the algebraic adversary holds the representation V_i for the output (E, W), they must also possess the representations for the inputs in the list $\mathcal{L}_{alg}$. Thus, the extractor locates the representations corresponding to (E_1, W_1) and (E_2, W_2) in $\mathcal{L}_{alg}$. Let us denote these extracted representations as V_{i-1} and v_{i-1}.
  - **Step 4: Validity Analysis (Execution Trace).** We now demonstrate that these extracted representations correspond to a valid execution of F'.

1. **Preservation of Satisfiability (NIFS KS):** Since the folding verifier $\mathsf{NIFS.V}$ is instantiated with a GZT-compliant hash H (which renders finding pre-images satisfying Eq. (17a) computationally infeasible), the Knowledge Soundness of the NIFS is established. This guarantees that if V_i is a valid witness for the folded instance, the extracted pre-images V_{i-1} and v_{i-1} are valid witnesses for the CR-R1CS relation $\mathcal{R}_{\mathsf{pp,s}}$.
2. **Fresh Instance Check (Standard R1CS):** The augmented function F' contains explicit constraints enforcing that the fresh instance u_{i-1} is a standard R1CS instance (i.e., $\mathsf{u}_{i-1}.E = C_{\mathbf{0}}$ and $\mathsf{u}_{i-1}.u = 1$). Since $(\mathsf{u}_i, \mathsf{v}_i)$ satisfies the circuit constraints of F', these conditions must hold for the extracted components.

- **Step 5: Update and Recurse.** Proceeds to the next iteration with the pair $\mathit{\Pi}_{i-1} = ((\mathsf{U}_{i-1}, \mathsf{V}_{i-1}), (\mathsf{u}_{i-1}, \mathsf{v}_{i-1}))$.

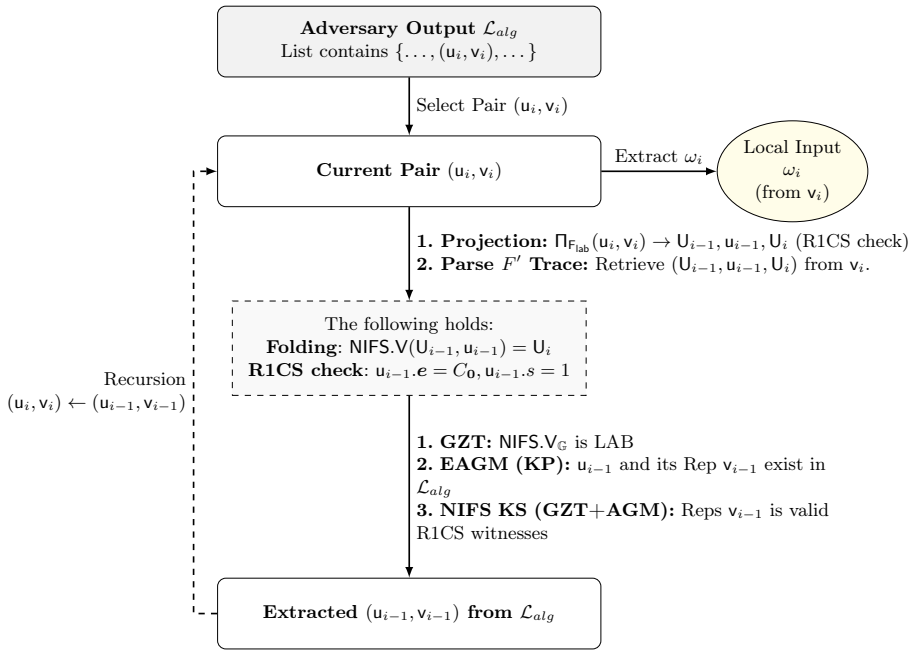


Fig. 9. Visual flow of the Extractor $\mathcal{E}$. The extractor parses the local input ω_{i-1} from the pair $(\mathsf{u}_i, \mathsf{v}_i)$. KP stands for knowledge propagation. It then inspects the embedded folding trace. Using GZT and EAGM, it locates the previous step's witness $(\mathsf{u}_{i-1}, \mathsf{v}_{i-1})$ in $\mathcal{L}_{alg}$ to proceed with recursion.

4. Error and Computation Analysis.

– **Computation:** Unlike rewinding-based approaches that incur exponential overhead, our extractor $\mathcal{E}$ runs in time linear in the recursion depth n (i.e., $T_{\mathcal{E}} \approx O(n)$), as it processes the trace sequentially without rewinding.
– **Soundness Error:** The failure probability at each step is bounded by the sum of the GZT error ϵ_{GZT} and the DL advantage ϵ_{DL}. Specifically, ϵ_{GZT} governs the algebraic checks for NIFS and LAB, while ϵ_{DL} underpins the binding property of the underlying commitments. By applying a union bound over n steps, the total soundness error is bounded by $n \cdot (\epsilon_{\mathsf{GZT}} + \epsilon_{\mathsf{DL}})$. Since both terms are negligible, the total error remains negligible for any polynomial depth n.

□