On the Security of Nova Recursive Proof System

Hyeonbum Lee and Jae Hong Seo *

Department of Mathematics & Research Institute for Natural Sciences,
Hanyang University, Seoul 04763, Republic of Korea
{leehb3706, jaehongseo}@hanyang.ac.kr

Abstract. Nova is a new type of recursive proof system that uses a folding scheme as its core building block. This brilliant idea of folding relations can significantly reduce the recursion overhead. In this paper, we study some issues related to Nova's soundness proof, which relies on the soundness of the folding scheme in a recursive manner.

First, due to its recursive nature, the proof strategy inevitably causes the running time of the recursive extractor to expand polynomially for each additional recursive step. This constrains Nova's soundness model to only logarithmically bounded recursive steps. Consequently, the soundness proof in this limited model does not guarantee soundness for a linear number of rounds in the security parameter, such as 128 rounds for 128-bit security. On the other hand, there are no known attacks on the arbitrary depth recursion of Nova, leaving a gap between theoretical security guarantees and real-world attacks. We aim to bridge this gap in two opposite directions. In the negative direction, we present a recursive proof system that is unforgeable in a log-round model but forgeable if used in linear rounds. This shows that the soundness proof in the log-round model might not be applicable to real-world applications that require linearly long rounds. In a positive direction, we show that when Nova uses a specific group-based folding scheme, its knowledge soundness over polynomial rounds can be proven in the *Extended Algebraic Group Model* (EAGM), which is our novel computational model that lies between Algebraic Group Model (AGM) and the Generic Group Model (GGM). To the best of our knowledge, this is the first result to show Nova's polynomial rounds soundness.

Second, the folding scheme is converted non-interactively via the Fiat-Shamir transformation and then arithmetized into R1CS. Therefore, the soundness of Nova using the non-interactive folding scheme essentially relies on the heuristic random oracle instantiation in the standard model. In our new soundness proof for Nova in the EAGM, we replace this heuristic with a new computational assumption for a cryptographic hash function, called the *General Zero-Testing assumption*. We treat this hash assumption as an independent subject of interest and expect it to contribute to a deeper understanding of Nova's soundness.

* Corresponding Author.

1 Introduction

Incrementally Verifiable Computation (IVC) [55] and its generalization, Proof-Carrying Data (PCD) [26] are cryptographic primitives that facilitate the generation of proofs that convince 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) [7,41], succinct blockchains [12,27,11,39], and verifiable state machines [46].

VDF schemes are one of the key tools for Ethereum's consensus protocols, and several studies have incorporated the IVC scheme into VDF [41]. 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 [30], 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 [14,19,45] 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 [26,6,22], 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 brilliant 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 [45,50]. Nevertheless, there are no known attacks on arbitrary depth recursion, leaving a gap between theoretical security guarantees and real-world attacks.

Our Contribution. Our contribution is threefold, and we summarize them in the Table 1. First, we identify the gap between the theoretical security guarantees achievable in a limited IVC model with $O(\log \lambda)$ recursive rounds and the knowledge soundness in an unrestricted IVC model without log-round bounds. To address this, we introduce a variant of Nova, called Ephemeral-Nova, which satisfies knowledge soundness in the limited IVC model with $O(\log \lambda)$ recursive rounds, but becomes forgeable in the IVC model with a linear number of rounds in λ. Thus, Ephemeral-Nova demonstrates the necessity for a stronger security notion to account for poly-round bounds, leading us to propose a knowledge soundness for poly-round bounds, named *poly-depth knowledge soundness*.

The second contribution is a new security proof for the poly-depth knowledge soundness of Nova, derived from a group-based folding scheme, whereas the previous proof in [45] only covered at most logarithmic-round IVC. Notably, the folding scheme proposed in the Nova paper [45] is a group-based construction; therefore, we attempt to prove it using the algebraic group model (AGM) as defined in [32] for straight-line extraction [33]. However, to apply AGM to soundness proof, we need to clearly define the adversary's capabilities. To address this, we first introduce a new adversarial model, called *extended algebraic group model* (EAGM), in a reasonable manner and complete the proof for the poly-depth knowledge soundness of Nova. To the best of our knowledge, our security proof is the first to demonstrate the knowledge soundness of Nova for polynomial rounds and partially explains why there are no known attacks against Nova for arbitrary-depth recursion.

Our final contribution is the introduction of computational assumption for cryptographic hash function. This assumption is a more relaxed requirement than the random oracle instantiability. In Nova's construction, the non-interactive folding scheme (NIFS) is derived by applying the Fiat-Shamir transformation to its interactive version [45]. To construct Nova IVC from NIFS, it is arithmetized into R1CS, making the random oracle instantiation accessible to the adversary. In fact, many IVC schemes that use the Fiat-Shamir transformation rely on a similar heuristic assumption. We introduce a new computational problem and assumption for cryptographic hash functions, called a general zero-testing (GZT) hash problem and assumption, respectively. We then use the GZT assumption in our new soundness proof for Nova within the EAGM, without relying on random oracle instantiation.

Our Idea for Designing Ephemeral-Nova. Together with an execution function $F : \mathcal{Z} \times \mathcal{W} \to \mathcal{Z}$ and two values $z_0, z_n \in \mathcal{Z}$, a IVC prover generates a succinct proof that proves the knowledge of $\omega_0, \ldots, \omega_{n-1}$ that satisfy the relations $F(z_{i-1}, \omega_{i-1}) = z_i$ for $i = 1, \ldots, n$. Nova's idea for designing IVC is to use a folding scheme, which allows to fold two instance-witness pairs into one pair, on the R1CS relation for the augmented execution function F'. Here, the augmented function F' includes several necessary checks and computations, such as the execution of F and the folding procedure.

Although it is necessary for the augmented function F' to include the necessary procedures for soundness, such as the execution of F, we found that adding some redundant procedure may not harm the knowledge soundness of the IVC scheme. From this observation, we can try injecting a trigger into F' such that it only becomes activated after a sufficiently large number of rounds. For this

Security Notion		KS (Def. 2,[45])	Poly-depth KS (Def. 3)	
Model	Adversary	Standard		EAGM (Def.7)
	Hash	RO instantiation		GZT (Def.9)
Nova [45]		✓	✗	✓
Ephemeral-Nova (Sec.3)		✓	✗	✗

KS: knowledge soundness, EAGM: extended algebraic group model, GZT: general zero-testing hash assumption, RO instantiation: random oracle instantiation that is accessible to the adversary. The orange one represents a narrower and more limited notion compared to the green one. This table presents the provability of two IVC schemes under the given security notion in the model; ✓ indicates provability, while ✗ indicates non-provability.

Table 1. Comparision of Provability

purpose, such a trigger should be controllable for the timing of activation and also deterministic because the execution of F' should be arithmetized into R1CS. For Ephemeral-Nova, we found an appropriate trigger that can be summarized as the following recursive sequence:

$$Y_{n+1} := Y_n^{2\alpha} \cdot A_n \pmod q \text{ and } Y_0 := 1,$$

where q is a prime with form $\alpha \cdot 2^k + 1$, known as the Proth prime [13], for $k \geq \lambda$ and odd arbitrary integer α and there are sufficient large Proth primes used in the prime fields of elliptic curve parameter [21,16]. Suppose that each A_n is either 1 or chosen from a uniform distribution. If $n < k$, then $Y_{n+1} = 1$ is almost equivalent to the case in which all $A_0, \dots, A_n$ are ones. This equivalence is maintained until n is sufficiently smaller than k, but is suddenly broken if n exceeds k. This sequence contains a sudden transition in the equivalence, the timing of which can be controlled by selecting q, and the uniform distribution of A_n can be replaced with a deterministic procedure such as a cryptographic hash function. Using this special sequence, we can construct an Ephemeral-Nova whose behavior is almost equivalent to the original Nova before the linear round and satisfies the knowledge soundness in the constrained IVC model with a log-round bound, but is forgeable after the linear round due to the activated trigger.

The design of Ephemeral-Nova allows us to find that unnecessary steps in F' may cause a problem that cannot be captured by a general recursive proof strategy. Therefore, new knowledge soundness proof strategies are needed that can investigate all unexpected effects, including the above trigger.

Our Idea for New Knowledge Soundness Proof for Polynomial Rounds. Nova's soundness proof relies on the soundness of the underlying folding scheme and uses a recursive proof strategy to extract the witness ω_i in reverse order. Let $\mathcal{E}_i$ be an extractor to extract ω_i, $\tilde{\mathcal{A}}_i$ be an adversary for the folding scheme, and $\tilde{\mathcal{E}}_i$ be an extractor for the folding scheme. Then, the recursive proof strategy leads to an inequality between the running time:

$\mathtt{time}(\mathcal{E}_i) > \mathtt{time}(\tilde{\mathcal{E}}_i) + \mathtt{time}(\tilde{\mathcal{A}}_i) > 2 \cdot \mathtt{time}(\mathcal{E}_{i+1})$, where the right inequality holds if $\mathtt{time}(\tilde{\mathcal{E}}_i) > \mathtt{time}(\tilde{\mathcal{A}}_i)$. Therefore, the running time required to extract all ω_i increases exponentially in the final number of rounds.

To avoid recursive blowup, instead of relying on the extractor $\tilde{\mathcal{E}}_i$ for the folding scheme, we directly prove the soundness of the IVC scheme. This requires a direct procedure to extract all ω_i from the attacker's output $(F, (z_0, z, \Pi))$ only, where F is an execution function, z_0 is an initial input of F, z is the final output of F, and Π is a valid IVC proof. Indeed, the adversary's output is too limited to extract all intermediate ω_i without an additional resource such as a folding extractor.

Therefore, we move to an ideal model to observe a partial history of group-related operations performed by the adversary until the final result is output, where the underlying folding scheme is group-based. There are two well-established ideal models for handling group operations: the generic group model (GGM) [49,54,48] and algebraic group model (AGM) [32].

GGM is devised to demonstrate the hardness of group-based problems and the security of cryptographic schemes against attackers who are constrained not to use group descriptions. In the AGM, all group elements that the attack algorithm outputs are derived from known group elements via group operations.

In order to analyze the security of the Nova IVC scheme, both GGM and AGM have limitations. The GGM has the advantage of tracking the history of group operations because of its interactive feature. However, in the Nova IVC scheme, the folding verifier is arithmetized into R1CS, meaning that group operations should be instantiated in R1CS, which is not allowed in GGM. A similar situation occurs when we use the random oracle model in the analysis of the non-interactive folding scheme. That is, the cryptographic hash functions are modeled as the random oracle, but the hash function should also be instantiated in R1CS when the folding verifier is arithmetized into R1CS. Heuristically, one might assume that these are securely possible, but we avoid these heuristics as much as possible. (We will revisit the random oracle model later.)

In the AGM, the adversary should output a representation vector whenever a group element is output. Using the provided representation vector, we can construct a straight-line extrator [33] using the algebraic adversary. However, the AGM has other limitations.

First, the group elements for which the adversary should provide representation vectors are not clearly defined. What if the adversary outputs elements that are not part of the group but are encodable as group elements? In the knowledge soundness proof in Nova, the adversary provides an R1CS witness that contains elements that can be encoded as group elements, even though they are field elements, because the R1CS circuit includes group operations. In this case, the original AGM cannot ensure that the adversary provides representations for these group-encodable field elements.

Second, for direct extraction, we expect the representations provided by the adversary to form an R1CS witness. In other words, we require the algebraic adversary to provide a specific representation, but AGM does not restrict the form of the representation vectors provided by the adversary.

To circumvent these two limitations, we modify the AGM. We first let the algebraic adversary provide representation vectors of some group-encodable outputs, not only explicitly group elements. Depending on the situation, part of the adversary's output ensures group encodability. In this case, the adversary may obtain the group-encodable part by constructing the group element algebraically and then converting it to a non-group form. In this sense, the adversary knows the representation for the group-encodable part, so it is reasonable to let the adversary provide it, even if the group-encodable element does not form a group explicitly.

Second, we let the adversary output a representation satisfying specific conditions, e.g., a committed relaxed-R1CS (CR-R1CS) witness. We may assume that if the adversary can construct a CR-R1CS instance, it also knows the corresponding witness, which is a representation of the instance. This concept is similar to the knowledge of exponent (KOE) assumption [29], which is covered by the definition of AGM [32]. Similar to the KOE assumption, we require the adversary to provide a specific representation depending on the group element.

Our Idea for Nova construction without Random Oracle Instantiation. There are studies [23,22] that aim to remove heuristic instantiations of random oracles by introducing new variants of random oracles. We propose a different approach to avoid heuristic analysis, as we do not seek to modify the Nova IVC construction but rather to provide a new soundness analysis.

To this end, we introduce a new plausible computational assumption for cryptographic hash functions, such as SHA-256, that is sufficient for proving knowledge soundness in the AGM. This assumption alone cannot fully replace random oracles, as it lacks the ability to extract a witness by rewinding algorithms, which is a key feature of random oracles. However, we utilize this assumption in conjunction with the EAGM, which enables the extraction of certain values used by the adversary. This extracted values can then be analyzed using the hash function assumption to establish specific relations, ultimately proving that these values serve as a witness for the R1CS.

Additional Related Works. A well-known approach for IVC is to recursively utilize succinct non-interactive arguments of knowledge (SNARKs) [35,36] for arithmetic circuits. In this approach [4], at each incremental step i, the prover generates a SNARK proving the correct execution of F to the output of step i and that the SNARK verifier, represented as a circuit, has accepted the SNARK for step $i-1$. However, SNARK-based approaches are considered impractical because they require a cycle of pairing friendly elliptic curves. Furthermore, this approach requires a trusted setup that inherits from SNARKs. To address this issue, there are alternative approaches using NARKs [14,19] by deferring expensive verification circuit per each step.

Organization. The next section describes Nova IVC and its folding method, which is the core building block of Nova. In Section 3, we propose a new IVC scheme called Ephemeral-Nova that has knowledge soundness in log-bounded rounds but is forgeable in linear rounds. In Section 4, we review idealized models for group-based systems and propose a new idealized model, extended algebraic group model (EAGM). In Section 5, we introduce a new computational problem and assumption for hash functions and show how to use it in the AGM to replace random oracles. In Section 6, we present a new knowledge soundness proof for Nova from a group-based folding scheme in the EAGM. Finally, we provide concluding remarks in Section 7.

2 IVC from Folding Scheme

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 setup algorithm* Setup *and commitment algorithm* Com*. Let* M*,* R*, and* C *be message space, random space, and commitment space, respectively.* Setup *and* Com *are defined by:*

- $\mathsf{Setup}(1^\lambda, \ell) \to \mathsf{ck}$ *: 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* $m \in \mathsf{M}$*, and randomness* $r \in \mathsf{R}$*, output commitment* $C \in \mathsf{C}$

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

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

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

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

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

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

2.1 Definitions of IVC and (Refined) Folding Scheme

Definition 2 (IVC). *An incrementally verifiable computation (IVC) scheme is defined by four PPT algorithms: the generator* $\mathcal{G}$*, key generation* $\mathcal{K}$*, the prover* $\mathcal{P}$*, and the verifier* $\mathcal{V}$*. We say that an IVC scheme* $(\mathcal{G}, \mathcal{K}, \mathcal{P}, \mathcal{V})$ *satisfies perfect completeness if for any PPT adversary* $\mathcal{A}$

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

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

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

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

As mentioned in [45], IVC based recursive techniques [45,14,25,19,9,43,44,17,50] 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 cover knowledge soundness under the polynomial large number of applications n, we define *poly-depth knowledge soundness* by extending n to be bounded by a polynomial function. In addditon, we refer to an IVC scheme as log-bounded (poly-bounded) if the scheme satisfies knowledge soundness for logarithmic $n = O(\log \lambda)$ (polynomial $n = \mathsf{poly}(\lambda)$, respectively).

Definition 3 (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 arbitrary polynomial* $n = \mathsf{poly}(\lambda)$*, 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 in Eq.* (1)*.*

To define a folding scheme, we consider a special relation $\mathcal{R}$ over tuples consisting of public parameters pp_{FS}, structure s, instance u, and witness v. We use the notation $\mathcal{R}_{\mathsf{pp}_{FS},\mathsf{s}}$ to denote the subset consisting $(\mathsf{pp}_{FS}, \mathsf{s}, \cdot, \cdot) \in \mathcal{R}$ if pp_{FS} and s are fixed. Informally, the folding scheme has, beyond two interactive prover P and verifier V, additional algorithms G and K that specify the first two terms of $\mathcal{R}$, pp_{FS} and s. After fixing pp_{FS} and s, a folding scheme allows two instance-witness pairs $(\mathsf{u}_1, \mathsf{v}_1), (\mathsf{u}_2, \mathsf{v}_2) \in \mathcal{R}_{\mathsf{pp}_{FS},\mathsf{s}}$ to be folded into one pair $(\mathsf{u}, \mathsf{v}) \in \mathcal{R}_{\mathsf{pp}_{FS},\mathsf{s}}$ and the soundness of the folding scheme informally states that if two instances u_1 and u_2 are folded and the folded instance-witness pair (u, v) is included in $\mathcal{R}_{\mathsf{pp}_{FS},\mathsf{s}}$, then there are valid witness v_1 and v_2 satisfying $(\mathsf{u}_1, \mathsf{v}_1), (\mathsf{u}_2, \mathsf{v}_2) \in \mathcal{R}_{\mathsf{pp}_{FS},\mathsf{s}}$. The formal definition of folding scheme is given below.

Definition 4 ((Refined) Folding Scheme). *Consider a relation $\mathcal{R}$ over public parameters, structure, instance, and witness tuples. A folding scheme for $\mathcal{R}$ consists of three PPT algorithms, a generator G, a prover P and a verifier V, and a deterministic key generation algorithm K, all defined as follows.*

- $\mathsf{G}(1^\lambda, N) \rightarrow \mathsf{pp}_{FS}$*: On input security parameter λ and the maximum size of common structure N, samples public parameters pp_{FS}*
- $\mathsf{K}(\mathsf{pp}_{FS}, \mathsf{s}) \rightarrow \mathsf{pk}_{FS}$*: On input pp_{FS} and a common structure s, of size N between instances to be folded, outputs a prover key pk_{FS}.*
- $\mathsf{P}(\mathsf{pk}_{FS}, (\mathsf{u}_1, \mathsf{v}_1), (\mathsf{u}_2, \mathsf{v}_2)) \rightarrow (\mathsf{u}, \mathsf{v})$*: On input two instance-witness pairs $(\mathsf{u}_1, \mathsf{v}_1)$ and $(\mathsf{u}_2, \mathsf{v}_2)$, outputs a new instance-witness pair (u, v) of the same size and folding proof Π to allow the verifier to update new instance.*
- $\mathsf{V}(\mathsf{pp}_{FS}, \mathsf{u}_1, \mathsf{u}_2, \Pi) \rightarrow \mathsf{u}$*: On input two instances u_1 and u_2, outputs a new instance u.*

Although the final outputs of P and V are defined in the above description, both are interactive algorithms; thus, the interactive procedure and the corresponding transcript are denoted as follows.

$$(\mathsf{u}, \mathsf{v}) \leftarrow \langle \mathsf{P}(\mathsf{pk}_{FS}, \mathsf{v}_1, \mathsf{v}_2), \mathsf{V}(\mathsf{pp}_{FS}) \rangle (\mathsf{u}_1, \mathsf{u}_2)$$

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

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

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*

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

Definition 5 (Public Coin). *A folding scheme* $(\mathsf{G},\mathsf{K},\mathsf{P},\mathsf{V})$ *is called public coin if all the messages sent from* V *to* P *are sampled from a uniform distribution.*

Definitional Refinement for IVC design. In our refined definition of folding scheme, the verifier V takes pp_{FS} as input, unlike the prover P which takes pk_{FS} as input. In the original definition of folding scheme [45], V also takes vk_{FS} as input, where vk_{FS} is generated by both pp_{FS} and s. Our definition is a special case of the original definition since vk_{FS} can be set by pp_{FS}. We argue that our refinement is necessary if the folding scheme is used in the IVC design. Looking at the use of folding scheme in the IVC design in [45], the folding verifier should be a part of the augmented function F', which is arithmetized to the (committed relaxed) R1CS. That is, the description of V should be contained in s and thus V should not take s as input to avoid a circular contradiction. In particular, the concrete group-based construction of folding scheme in [45] satisfies our refined definition because its process does not require s.

Committed Relaxed R1CS. The committed relaxed R1CS is a variant of the R1CS constraints system, which is widely used in proof system [53,20,24,19]. In particular, the committed relaxed R1CS is a public parameter-dependent relation. Let us explain the committed relaxed R1CS in terms of the folding scheme. The public parameter generator of the folding scheme G takes the size parameter N as the input. We specify N to have two positive integers m and ℓ with $\ell+1<m$. G outputs public parameter pp_{FS} that consists of the commitment keys of the homomorphic commitment scheme Com for committing vectors over a finite field $\mathbb{Z}_p$. More precisely, $\mathsf{pp}_{FS}=(\mathsf{ck}_{\boldsymbol{w}},\mathsf{ck}_{\boldsymbol{e}})$, which are two commitment keys of Com with dimensions m and $m-\ell-1$, respectively. The structure s indicates the R1CS parameter matrices $A,B,C\in\mathbb{Z}_p^{m\times m}$, where there are at most $\Omega(m)$ non-zero entries in each matrix and they specify the R1CS relation $A\boldsymbol{x}\circ B\boldsymbol{x}=C\boldsymbol{x}$. Note that the dimensions of the matrices are already specified in N.

The committed relaxed R1CS relation is the relation with parameter $\mathsf{pp}_{FS}=(\mathsf{ck}_{\boldsymbol{w}},\mathsf{ck}_{\boldsymbol{e}})$ and structure $\mathsf{s}=(A,B,C)$ defined by

$$\mathcal{R}_{\mathsf{pp}_{FS},\mathsf{s}}=\left\{\big((E,W,s,\mathsf{x});(\boldsymbol{e},r_{\boldsymbol{e}},\boldsymbol{w},r_{\boldsymbol{w}})\big):\begin{array}{c}E=\mathsf{Com}(\mathsf{ck}_{\boldsymbol{e}},\boldsymbol{e};r_{\boldsymbol{e}})\\W=\mathsf{Com}(\mathsf{ck}_{\boldsymbol{w}},\boldsymbol{w};r_{\boldsymbol{w}})\\\boldsymbol{z}=(\boldsymbol{w},\mathsf{x},s)\\A\boldsymbol{z}\circ B\boldsymbol{z}=sC\boldsymbol{z}+\boldsymbol{e}\end{array}\right\},\quad(2)$$

where x is public inputs and outputs.

Note that if one adds conditions $\boldsymbol{e} = \mathbf{0}$ and $s = 1$ in the above relation, the resulting relation becomes equivalent to the R1CS relation specified by the structure s.[1]

Non-Interactive Folding Scheme. Given a public-coin interactive folding scheme can be transformed to a non-interactive folding scheme, defined below, in the random oracle model via the Fiat-Shamir transform [31].

Definition 6 (Non-Interactive). *We say that a folding scheme* $(\mathsf{G},\mathsf{K},\mathsf{P},\mathsf{V})$ *is non-interactive if the interaction between* P *and* V *consists of a single message* T *from* P *to* V*. To clearly indicate the single message interaction, the input and output of* P *and* V *can be rewritten as* $\mathsf{P}(\mathsf{pk}_{FS}, (\mathsf{u}_1, \mathsf{v}_1), (\mathsf{u}_2, \mathsf{v}_2)) \rightarrow (\mathsf{u}, \mathsf{v})$*,* T *and* $\mathsf{V}(\mathsf{pp}_{FS}, \mathsf{u}_1, \mathsf{u}_2, T) \rightarrow \mathsf{u}$*.*

In fact, the folding prover and verifier are implemented in the design of Nova IVC; therefore, we must heuristically instantiate the random oracle using a cryptographic hash function. Therefore, we can only heuristically argue for the security of the resulting non-interactive folding scheme in the standard model. Recent existing IVC proposals in the standard model rely on the same heuristics that require instantiating the random oracle with a cryptographic hash function [45,43,44,17,50].

2.2 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 committed relaxed R1CS instances, where F' is represented by the committed relaxed R1CS.

An informal description of the computation in each round is given in Figure 1, where H is a cryptographic hash function and $(\mathsf{u}_\perp, \mathsf{v}_\perp)$ is a trivial instance-witness pair such that $\mathsf{v}_\perp$ is set by zeros. In addition, we define the trivial proof $\Pi_0 = (\mathsf{u}_\perp, \mathsf{v}_\perp, \mathsf{u}_\perp, \mathsf{v}_\perp)$, which consists of two trivial instance-witness pairs.

Let $\mathsf{NIFS} = (\mathsf{G}, \mathsf{K}, \mathsf{P}, \mathsf{V})$ be the non-interactive folding scheme for the committed relaxed R1CS of F'. The formal descriptions of the augmented function F' and Nova from NIFS are, respectively, provided in Figure 2 and Figure 3. 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 committed relaxed 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 only the public IO of u_{i+1}, that is, $\mathsf{u}_{i+1}.\mathsf{x}$.

[1] In [45], the alphabet u is used instead of s in this paper. We changed it to avoid confusion because u_i is used to denote an instance of the relation. Similarly, we use v to denote witness.

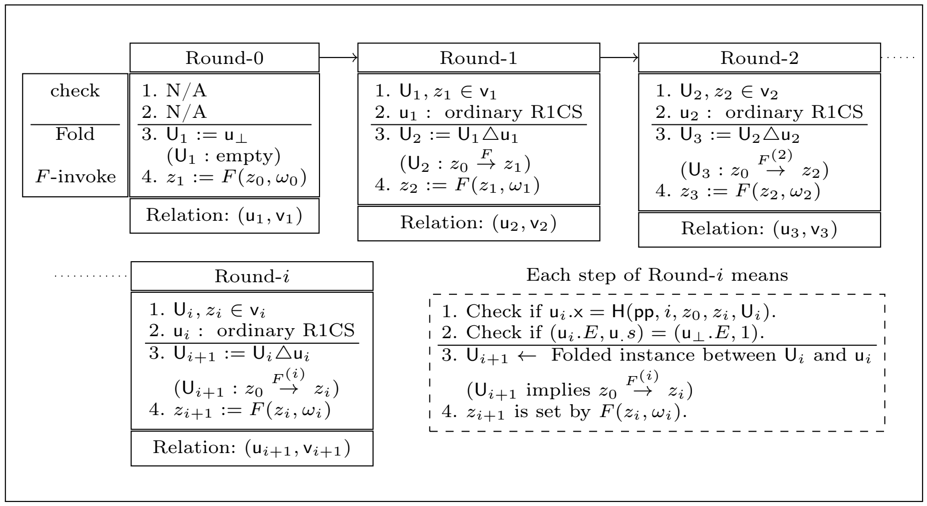

Fig. 1. Informal Description of Relation (u, v) for Each Round of Nova

$F'(\mathsf{pp}, \mathsf{U}_i, \mathsf{u}_i, (i, z_0, z_i), \omega_i, T) \rightarrow \mathsf{x}$:

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 $\mathsf{H}(\mathsf{pp}, i+1, z_0, F(z_i, \omega_i), \mathsf{U}_{i+1})$.

Fig. 2. Augmented Function F'

Theorem 1 (Nova-IVC [45]). *If the non-interactive folding scheme* NIFS *satisfies perfect completeness and knowledge soundness, then Nova in the Figure 3 is a log-bounded round IVC scheme satisfying perfect completeness and knowledge soundness in Definition 2.*

3 Ephemeral-Nova: A New Log-bounded round IVC

This section explores whether the security proof for the log-bounded round IVC scheme can provide an appropriate level of soundness guarantees for a linear number of rounds. In particular, we demonstrate that not all log-bounded round IVC schemes are knowledge-sound for a linear number of recursive rounds. To this end, we design a variant of Nova, called Ephemeral-Nova, that satisfies the knowledge soundness in Definition 2 but is forgeable when used more than a linearly large number of recursive rounds.

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

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

$\underline{\mathcal{P}(\mathsf{pk}, (i, z_0, z_i), \omega_i, \Pi_i) \to \Pi_{i+1}}$:
Parse Π_i as $((\mathsf{U}_i, \mathsf{V}_i), (\mathsf{u}_i, \mathsf{v}_i))$ and then
1. if i is 0, compute $(\mathsf{U}_{i+1}, \mathsf{V}_{i+1}, T) \leftarrow (\mathsf{u}_\bot, \mathsf{v}_\bot, \mathsf{u}_\bot.E)$;
   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 $\Pi_{i+1} \leftarrow ((\mathsf{U}_{i+1}, \mathsf{V}_{i+1}), (\mathsf{u}_{i+1}, \mathsf{v}_{i+1}))$.

$\underline{\mathcal{V}(\mathsf{vk}, (i, z_0, z_i), \Pi_i) \to \{0, 1\}}$:
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 $(\mathsf{u}_i.E, \mathsf{u}_i.s) = (\mathsf{u}_\bot.E, 1)$, 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'.

Fig. 3. Nova IVC

Our Idea for Ephemeral-Nova. Basically, the Ephemeral-Nova scheme should be knowledge sound in the log-bounded round model, and thus, we begin by looking at the original proof of knowledge-soundness of Nova. We first notice that the polynomial time extractor in the original proof of the knowledge soundness can extract the witness in the last $O(\log \lambda)$ number of rounds, where λ is the security parameter, because the running time of the extractor blows up exponentially at the number of rounds for each additional recursion round. From this observation, we find that to design a *linearly-faulty-and-logarithmically-provable* scheme, the verification procedure of the Ephemeral-Nova scheme should be in such a way of

- `[Faulty]` pardon for misbehavior before last log number of rounds, but
- `[Provable]` correctly checking the validity of the last log number of rounds.
- `[Compile]` deterministic to be compiled into the committed relaxed R1CS.

Designing an IVC satisfying the above requirements is somewhat challenging because the timing of the log number of rounds depends on the security parameter. Therefore, we need to devise a deterministic process of gradual change of (un)soundness in the security parameter. To this end, we first devise a recursive sequence with the above three features as follows.

$$Y_{n+1} := Y_n^{2\alpha} \cdot A_n \pmod{q} \text{ and } Y_0 := 1, \tag{3}$$

where q is a prime number of the form $\alpha \cdot 2^k + 1$, known as Proth prime [13], for some $k \geq \lambda$ and odd integer α and A_n is selected from one of two distributions, either a constant 1 or uniform distribution on $\mathbb{Z}_q$. For values A_i, we consider $A_i = 1$ normal and all other values abnormal. To provide verifiability of A_i, we use additional indicator Y_i to check the normality of all previous values $A_0, \ldots, A_{i-1}$. For example, for a given value A_i, if all intermediate values $A_{i+1}, \ldots, A_n$ are normal and the i-th indicator Y_i is 1, we obtain the following equation by the recurrences of Eq. (3).

$$Y_{n+1} = \prod_{i=0}^{n} A_i^{(2\alpha)^{n-i}} \pmod q. \tag{4}$$

For time step $n = O(\log \lambda)$, if all previous A_i $(i = 0, .., n)$ are normal, then we have $Y_{n+1} = 1$. If at least one A_i is abnormal, then $Y_{n+1} \neq 1$ except for the negligible probability in λ since A_i is uniformly distributed over $\mathbb{Z}_q$ with $q > 2^\lambda$ but n is logarithmic $O(\log \lambda)$. Therefore, checking $Y_{n+1} = 1$ is a good verification procedure for the normality of all previous A_i $(i = 0, .., n)$. However, when time step n becomes sufficiently large (e.g., $n \geq k$), $Y_{n+1} = 1$ does not guarantee the normality of all previous A_i. This is due to the shape of the prime number q and the Fermat's Little Theorem as follows.

$$\begin{aligned} Y_{n+1} = & \prod_{i=0}^{n} \quad A_i^{(2\alpha)^{n-i}} \pmod q \\ = & \prod_{i=n-k+1}^{n} A_i^{(2\alpha)^{n-i}} \pmod q \ \text{ (by Fermat's Little Theorem)} \end{aligned}$$

Therefore, checking $Y_{n+1} = 1$ is a good verification procedure for the normality of only the last $k-1$ values $A_n, \ldots, A_{n-k+1}$, so that using this sequence and the verification of $Y_{n+1} = 1$ is a good candidate having the first two conditions for linearly-faulty-and-logarithmically-provable scheme. A_i's uniform distribution can be replaced with the random oracle, and thus the above sequence satisfies the last "deterministic" condition heuristically by using a cryptographic hash function in the standard model.

Our Ephemeral-Nova Construction. Using the above idea for the recursive sequence and the verification procedure, we now construct the Ephemeral-Nova by slightly modifying the augmented function F' and the IVC procedure. As shown in Figure 1 and Figure 2, the execution of F' in each round consists of checking, folding, and invoking procedures. Although all these internal procedures of F' are necessary for proving knowledge-soundness, adding some redundant procedures may not affect knowledge-soundness. We extend F' to have two modes, which are determined by an additional input $\mathsf{b}_i \in \{0, 1\}$. We call this extended F' a *trapdoor augmented function* F' and sometimes use the notation F'_{b_i} to denote $F'(\cdot, \mathsf{b}_i)$. According to bit b_i, the trapdoor F'_{b_i} sets

$$z_{i+1} := \begin{cases} F(z_i, \omega_i) \text{ if } \mathsf{b}_i = 1 \\ \quad z_i \quad\;\; \text{ if } \mathsf{b}_i = 0 \end{cases} \quad // \ z_i \text{ can be replaced with any value except } F(z_i, \omega_i).$$

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

Compute $\boxed{z_{i+1}} := \begin{cases} F(z_i, \omega_i) & \text{if } \mathsf{b}_i = 1 \\ z_i & \text{if } \mathsf{b}_i = 0 \end{cases}$ // Any value except $F(z_i, \omega_i)$ can be used.

If i is 0, check if $Y_0 = 1$. And then compute $\boxed{Y_1} := \mathsf{b}_i + (1 - \mathsf{b}_i) \cdot \mathsf{H}(\mathsf{pp}, 0, z_0, \mathsf{u}_\perp, Y_0) \pmod q$ and output $\mathsf{H}(\mathsf{pp}, 1, z_0, z_1, \mathsf{u}_\perp, \boxed{Y_1})$; otherwise,

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

$$\boxed{Y_{i+1}} := Y_i^{2\alpha} \cdot \big(\mathsf{b}_i + (1 - \mathsf{b}_i) \cdot \mathsf{u}_i.\mathsf{x}\big) \pmod q.$$

*We assume that appropriate *deterministic* encoding process is applied from the function value $\mathsf{H}(\cdot)$ into $\mathbb{Z}_q$ for computing Y_{i+1}.

** The box $\square$ indicates the difference from the original augmented function F'.

Fig. 4. Trapdoor Augmented Function F'

for some prime number q of the form $\alpha \cdot 2^k + 1$ for some $k \geq \lambda$ and odd integer α. If $\mathsf{b}_i = 1$, this process is equivalent to the original F'. Otherwise, F' skips the execution of F. Therefore, we call the cases of $\mathsf{b}_i = 1$ and $\mathsf{b}_i = 0$ a normal mode and a trapdoor mode, respectively. The trapdoor F'_{b_i} additionally takes Y_i as input and F'_{b_i} updates Y_i according to the following rule.

$$Y_{i+1} := Y_i^{2\alpha} \cdot \big(\mathsf{b}_i + (1 - \mathsf{b}_i) \cdot \mathsf{u}_i.\mathsf{x}\big) \pmod q \text{ and } Y_0 := 1.$$

Let $A_i = \big(\mathsf{b}_i + (1 - \mathsf{b}_i) \cdot \mathsf{u}_i.\mathsf{x}\big)$. If $\mathsf{b}_i = 1$, then we have $A_i = 1$. Otherwise, A_i has a uniform distribution heuristically since $\mathsf{u}_i.\mathsf{x}$ is a hash output. From the analysis of the recursive sequence in Eq. (3), we know that $Y_{i+1} = 1$ could be a good verification procedure for linearly-faulty-logarithmically-provable IVC scheme. We provide a concrete description of the trapdoor augmented function F' and the ephemeral-Nova in Figure 4 and Figure 5, respectively.

Choice of Prime Number q. The Proth prime $q = \alpha \cdot 2^k + 1$ is essential for constructing the ephemeral Nova. Using the prime number theorem, for fixed $k = O(\lambda)$, one can find $\alpha \cdot 2^k + 1$ prime by adjusting α in $O(\log \lambda)$ times.

3.1 Ephemeral-Nova is (Log-Bounded Round) Knowledge-Sound

Now, we provide an intuition for the (log-bounded round) knowledge soundness of the Ephemeral-Nova IVC scheme described in Figure 5. Notice that we regard $\mathcal{P}_1$ and $\mathcal{P}_0$ in Figure 5 as the normal IVC prover and the abnormal IVC prover

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

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

$\underline{\mathcal{P}_\mathsf{b}(\mathsf{pk}, (i, z_0, z_i), \omega_i, \Pi_i, \boxed{Y_i}) \to \Pi_{i+1}}$:

Parse Π_i as $((\mathsf{U}_i, \mathsf{V}_i), (\mathsf{u}_i, \mathsf{v}_i))$ and then
1. if i is 0, set $Y_0 = 1$ and 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 $(\mathsf{u}_{i+1}, \mathsf{v}_{i+1}) \leftarrow \mathsf{trace}(F'_{\mathsf{tr}}, (\mathsf{vk}, \mathsf{U}_i, \mathsf{u}_i, (i, z_0, z_i), \omega_i, T, \boxed{Y_i, \mathsf{b}}))$, and
3. output $\Pi_{i+1} \leftarrow ((\mathsf{U}_{i+1}, \mathsf{V}_{i+1}), (\mathsf{u}_{i+1}, \mathsf{v}_{i+1}))$.

$\underline{\mathcal{V}(\mathsf{vk}, (i, z_0, z_i), \Pi_i) \to \{0, 1\}}$:

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 $\mathsf{u}_i.\mathsf{x} = \mathsf{H}(\mathsf{pp}, i, z_0, z_i, \mathsf{U}_i, \boxed{1})$,
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'.

* The box $\square$ indicates the difference from the original augmented function F'

** The instance-witness pair $(\mathsf{u}_{i+1}, \mathsf{v}_{i+1})$ from the trace in $\mathcal{P}$ should follow the setting: $(\mathsf{u}_{i+1}.E, \mathsf{u}_{i+1}.s) = (\mathsf{u}_\perp.E, 1)$, .

Fig. 5. Ephemeral-Nova IVC

of Ephemeral-Nova, respectively. Although both IVC provers, $\mathcal{P}_1$ and $\mathcal{P}_0$, are described in Figure 5, the security proof focuses on the normal IVC prover, $\mathcal{P}_1$.

Theorem 2. *The IVC scheme $(\mathcal{G}, \mathcal{K}, \mathcal{P}_1, \mathcal{V})$ in Figure 5 satisfies perfect completeness and knowledge soundness (Definition 2) if the non-interactive folding scheme* NIFS *satisfies perfect completeness and knowledge soundness.*

Due to space limitations, the full proof of Theorem 2 is included in Appendix A. Instead, we here sketch the proof idea. The Ephemeral-Nova is designed to be equivalent to Nova if the trigger is not activated. In particular, if we set $b = 1$, the augmented function F'_1, the IVC prover $\mathcal{P}_1$, and verifier $\mathcal{V}$ are essentially identical to the original Nova IVC, so the Ephmeral-Nova IVC satisfies the completeness. For knowledge soundness, it would be sufficient to show that passing the IVC verification guarantees that the trigger has not been activated. If this is the case, all remaining proofs will be essentially equivalent to the original knowledge-soundness proof by the design of the Ephemeral Nova.

Let us provide a brief idea about proving non-activation of the trigger. We consider a log-round $n \leq \frac{\lambda}{2}$, where p is a λ-bit prime. We claim that if the IVC verifier $\mathcal{V}$ accepts the proof Π_n then the skipping trigger cannot be activated during n-times computation $F^{(n)}$. When the trigger is activated (that is, $b = 0$)

at i-th round, the additional indicator Y_i is changed to an arbitrary value because it is an output of H. On the other hand, to give an acceptance from $\mathcal{V}$, the final additional input Y_n, which is an element in Π_n, should be equal to 1. By the construction of F'_{b_i} and uniform distribution of H outputs, the additional value Y_i for all $i \in [n]$ should be equal to 1 without negligible probability. (Refer to Lemma 3 in Appendix A.) This means that the trigger has not been activated during n times computation; therefore, we can rule out the case $b = 0$, and the remaining soundness proof is equivalent to that of the original Nova.

3.2 Linear-round Ephemeral-Nova is Forgeable

Although Ephemeral-Nova satisfies knowledge soundness in Theorem 2, the underlying trapdoor augmented execution (Figure 4) may intuitively cause security issues. In this section, we present a specific *linear round attack* on the Ephemeral-Nova IVC scheme in Figure 5, highlighting the necessity of enhancing knowledge soundness for linear rounds.

We consider linearly large round, i.e. $n = O(\lambda)$. Of course, this scenario is out of scope in Definition 2 and thus our attack does not lead a contradiction with the log-round knowledge soundness in Theorem 2. We construct adversary which outputs function F, initial and final values (z_0, z), and IVC proof Π satisfying the following condition: $z_n \neq z$ and $\mathcal{V}(\mathsf{vk}, n, z_0, z, \Pi) = 1$, where $z_i \leftarrow F(z_{i-1}, \omega_{i-1})$. (Refer to the format of the adversary's outputs in Definition 2.)

For the sake of simplicity, we abuse the notation $F^{(t)}(z_i, \omega_i)$ to denote an output of t times F execution with t local inputs $\omega_i, \ldots, \omega_{i+t-1}$ sequentially, i.e., $F^{(t)}(z_i, \omega_i) = F(F(\cdots F(z_i, \omega_i), \omega_{i+1}), \cdots), \omega_{i+t-1})$. We consider the prime field $\mathbb{Z}_q$ where $k = O(\lambda)$ and $q = \alpha \cdot 2^k + 1$. (If k is sublinear of λ, the soundness error may not be negligible due to small field size.) Now we construct a PPT adversary algorithm that generates an IVC proof of $(k+1)$-th round, i.e., $n = k+1$. (For $n \geq k+1$, our approach can be used to forge the IVC proof.)

Let the adversary $\mathcal{A}$ choose the function F satisfying non-collision within $k+1$ steps. That is, for a given z_0 and $(\omega_0, \ldots, \omega_k)$, $F^{(k+1)}(z_0, \omega_0) \neq z_j$ for all $j \leq k+1$ where $z_i := F^{(i)}(z_0, \omega_0)$. And then, $\mathcal{A}$ creates a forgery $\widetilde{\Pi_n}$ by running $\mathcal{P}_{\mathsf{b}}$ for each round in the following order $\underbrace{(\mathcal{P}_0, \mathcal{P}_1, \ldots, \mathcal{P}_1)}_{n=k+1 \text{ steps}}$. That is, the trapdoor mode F'_0 is used only in the first step and the normal mode F'_1 is used in all the other steps. Finally, after the last step, the IVC verifier $\mathcal{V}$ verifies forged proof $\widetilde{\Pi_n}$ and the final output z.

It is clear that z is not equal to the correct value $F^{(n)}(z_0, \omega_0)$ since $\mathcal{P}_0$ skipped the execution of F. Concretely, $z = F^{(n-1)}(z_0, \omega_0) = F^{(k)}(z_0, \omega_0)$ is the output of k-th step. By the non-collision property of F, z cannot be the n-th outputs $F^{(n)}(z_0, \omega_0) = F^{(k+1)}(z_0, \omega_0)$. Nevertheless, we argue that the IVC verifier accepts the proof $\widetilde{\Pi_n} = ((\widetilde{\mathsf{U}_n}, \widetilde{\mathsf{V}_n}), (\widetilde{\mathsf{u}_n}, \widetilde{\mathsf{v}_n}))$. In fact, both the trapdoor mode and the normal mode of F are correct executions of the augmented function F'_{tr}. Therefore, both $(\widetilde{\mathsf{U}_k}, \widetilde{\mathsf{V}_k})$ and $(\widetilde{\mathsf{u}_k}, \widetilde{\mathsf{v}_k})$ are correct committed relaxed R1CS induced by F'_{tr}, where $(\widetilde{\mathsf{U}_n}, \widetilde{\mathsf{V}_n})$ are also corrected folded by the folding scheme for F'. This allows $\widetilde{\Pi_k}$ to pass the test in the third and fourth lines of the IVC verifier procedure in Figure 5. Next, we check whether Y_n is equal to 1 or not. By the k times sequentially running $\mathcal{P}_1$, we can confirm that $Y_n = Y_1^{(2\alpha)^k} = (Y_1^{\alpha \cdot 2^k})^{\alpha^{k-1}} = 1$ regardless of the value of Y_1. Hence, the second line of the IVC verifier procedure is passed.

Remark. From the construction of Ephemeral-Nova, we emphasize two points. First, designing the augmented execution function F' may affect the forgeability (in linear round) of the IVC scheme, even if the underlying folding scheme is sound. That is, some redundant part of F' can be utilized for linear-round attack as we did. Second, the definition of knowledge soundness described in Definition 2 is not sufficient to represent a *secure* linear-round IVC scheme. Therefore, the IVC knowledge soundness should cover polynomial bounded rounds, as defined in Definition 3.

4 Model for Security Analysis

In the previous section, we observed that an IVC satisfying Definition 2 may not provide poly-depth knowledge soundness in Definition 3. However, to the best of our knowledge, there is no known concrete forgery attack in the original Nova IVC scheme [45]. The main reason that Nova cannot provide poly-depth knowledge soundness is the construction of a polynomial time extractor.

To address this gap, we focus on how to prove the poly-depth knowledge soundness of Nova IVC against restricted adversaries. First, we consider an idealized model for a group-based scheme and then adapt the model on the poly-depth knowledge soundness proof.

We first briefly review the features of popular idealized models for group-based systems and then set up an appropriate model for security analysis of the Nova IVC scheme.

Notation. We define notations for groups. Let $\mathbb{G}$ be an additive cyclic group of prime order p. When the group generator G is fixed, we use the bracket notation $[a]_G$ for a scalar $a \in \mathbb{Z}_p$ to denote the group element $a \cdot G$. If the generator is clear from the context, we often omit the subscript G and write as $[b] \in \mathbb{G}$. For $\boldsymbol{a} = (a_1, \dots a_\ell) \in \mathbb{Z}_p^\ell$ and $[\boldsymbol{b}]_G = ([b_1]_G, \dots, [b_\ell]_G) \in \mathbb{Z}_p^\ell$, a multi-scalar addition between $\boldsymbol{a}$ and $[\boldsymbol{b}]_G$ is denoted by $\langle \boldsymbol{a}, [\boldsymbol{b}]_G \rangle = \sum_{i=1}^{\ell} a_i \cdot [b_i]_G$.

Let $\boldsymbol{h}_1, \dots, \boldsymbol{h}_n \in \mathbb{Z}_p^\ell$ be representations of each component of group elements $\boldsymbol{H} = (H_1, \dots, H_n) \in \mathbb{G}^n$ over the basis $\boldsymbol{G} \in \mathbb{G}^\ell$, i.e., $H_i = \langle \boldsymbol{h}_i, \boldsymbol{G} \rangle$ for all $i \in [n]$.

Two Candidates: Generic Group Model and Algebraic Group Model The generic group model (GGM) is an idealized model where all group operations are carried out by making oracle queries [49,54,48,47]. This model is designed to capture the behavior of natural general algorithms that operate independently of any particular group descriptions. In fact, this model is divided by a way to handle group elements. The adversary in Shoup's model [54] gets random-encoded values of the additive group $\mathbb{Z}_p$ which are considered as group elements, but the adversary in Maurer's model [48] cannot access the value directly but obtains pointers indicating the line number in the oracle's table. Recently, Zhandry demonstrated the difference between these two models [56].

The algebraic group model (AGM), another idealized model proposed by Fuchsbauer, Kiltz, and Loss, requires that whenever an algorithm outputs a group element G, it also outputs a representation $\boldsymbol{c}$ such that $\langle \boldsymbol{c}, \boldsymbol{G} \rangle = G$, where $\boldsymbol{G}$ is a vector of group elements the algorithm took as input [32]. In particular, a specific group description is fixed and known to all algorithms, and there is no oracle query for group operations in the AGM. The intuition of the AGM is to restrict algorithms to output a new group element G only by deriving it via group operations from known group elements. In fact, the concept of algebraic adversary has already been studied in several literature [10,28,51,15,34,2,1,5,42] and the AGM of Fuchsbauer, Kiltz, and Loss [32] is the first formal framework for security proofs with respect to algebraic adversaries.

GGM and AGM are the two most popular models for the analysis of group-based systems. We now present some limitations of the two models, which have been identified by either previous literature or our observations, and slightly refine the definitions for setting up an appropriate model for our purpose.

Limitation of GGM. From the definition of GGM, it might cover a smaller class of algorithms than those in the AGM because algorithms are not allowed to use group descriptions. Another limitation of GGM, which is more critical to our purpose, is that the ideal group oracle cannot be instantiated as an arithmetic circuit. In Nova IVC, which uses a group-based folding scheme, the folding process containing group operations is arithmetized, and the arithmetized group operations are publicly accessible to all algorithms. In other words, the adversary can access the specific group description from this arithmetization. In fact, the same issue occurs when we use the arithmetized cryptographic hash function, which is modeled as a random oracle. Then, the resulting security analysis should rely on the heuristic GGM instantiation in the standard model. We avoid heuristic analysis as much as possible so that we could move on to the next candidate, the AGM.

Usefulness of AGM. The AGM is proposed as a model lying between the standard model and the GGM, and it is one of main reasons why the AGM has received so much attention recently [38,3,40,8,37]. As mentioned above, the adversary should provide a representation of the output group elements.

AGM is a useful model for constructing a *straight-line extractor* that processes the output of an algebraic adversary [32,33]. In AGM, the extractor receives outputs along with their algebraic representations from the algebraic adversary and extracts a witness from both the outputs and their representations. In this scenario, the extractor does not need to rewind the adversary because the provided outputs and their representation are sufficient for extracting a witness.

The main reason for the blow-up issue in the proof of Nova [45] is the necessity to rewind a folding adversary at each step. To avoid this issue, we construct a straight-line extractor using the algebraic adversary $\mathcal{P}^*$. Therefore, we modify AGM to suit our purposes more effectively.

Limitation 1: Ambiguity of Group Elements. Fuchsbauer et al. pointed out that the output group elements should be distinguishable from other inputs syntactically [32]. However, in terms of the adversary against KS of Nova, the adversary provides R1CS witness v, which contains *group-convertible* $\mathbb{Z}_p$-elements, that correspond to $\mathsf{NIFS.V}$ inputs u, U. Syntactically, the group-convertible elements in $\mathbb{Z}_p$ are not group elements but can be regarded as group elements following a publicly known conversion process. According to the AGM definition in [32], it is unclear whether the algebraic adversary provides a representation of group-convertible elements or not.

Modification 1: Representation of all Group-convertible Elements. Let us consider group-convertible elements in R1CS for the augmented function F' (Figure 2). To construct F', one should instantiate the group operation over $\mathbb{G}$ into a $\mathbb{Z}_p$-arithmetic circuit for the instantiation of the non-interactive folding scheme $\mathsf{NIFS.V}$. In this phase, the input and output group elements of $\mathsf{NIFS.V}$ should be converted to $\mathbb{Z}_p$ elements. Specifically, $\mathsf{NIFS.V}$ takes 4 group elements $\mathsf{U}_{n-1}.E$, $\mathsf{U}_{n-1}.W$, $\mathsf{u}_{n-1}.E$, and $\mathsf{u}_{n-1}.W$, and outputs 2 group elements $\mathsf{U}_n.E$ and $\mathsf{U}_n.W$. To instantiate $\mathsf{NIFS.V}$, one should convert these 6 group elements to field elements.

If an algebraic algorithm outputs group-convertible elements, we let it provide representations of each group-convertible element, which are indeed group elements generated from algebraic operations.

Limitation 2: Extracting Intermediate Representations. When using the model, the AGM is rather cumbersome compared to the GGM because the AGM allows us to extract only limited information—the representation of the group element in the final output of the algorithm. In contrast, the GGM enables tracking all group operation queries made by the adversary.

In the proof of IVC schemes, even if the adversary forges a proof at a particular time period, the definition of knowledge soundness requires extracting all witnesses from previous time periods. Therefore, in the context of IVC, the AGM is not suitable for extracting intermediate values computed by the adversary.

Notably, the GGM appears to be a better choice than the AGM in this case because the GGM allows the extraction of intermediate values. However, as we mentioned above, the GGM has another issue—the circuitization of group operations. Consequently, we need a new model that lies somewhere between the AGM and the GGM, one that enables the extraction of intermediate values while avoiding the circuitization problem of group operations.

Modification 2: Extended Algebraic Algorithm with respect to Verification. In many analyses using the GGM, the concept of an oracle is necessary to track all intermediate group operations. In contrast, a key benefit of the AGM is that it hides all intermediate group-related operations, except for the representation of the final output.

Our goal is to preserve this benefit of the AGM as much as possible while still enabling the tracking of certain intermediate group operations that are necessarily related to the final output—particularly without relying on oracles.

To achieve this, we first introduce the concept of verification to check whether group elements appear in the adversary's outputs, whether intermediate or final, and refine the original AGM accordingly. Next, we develop a methodology to determine whether an adversary's intermediate results are necessarily related to the final output. To establish this necessary relatedness, we use a conditional probability that holds with overwhelming probability.

We provide a formal definition of Extended Algebraic Algorithm with respect to verification as follows:

Definition 7 ((Extended) Algebraic Algorithm w.r.t. Verification). *Let* pp *be a public parameter and* $\mathtt{V}(\mathsf{pp}, \cdot) \to 1/0$ *be an algorithm, which we call verification, taking* pp *as input. We consider PPT algorithms* $\mathcal{A}^{\mathtt{V}(\mathsf{pp},\cdot)}$ *that takes* pp *as input and its goal is to output* out *such that* $\mathtt{V}(\mathsf{pp}, \mathsf{out}) \to 1$.[2] *We call* $\mathcal{A}$ *an* (Extended) Algebraic Algorithm with respect to $\mathtt{V}(\mathsf{pp}, \cdot)$ *if it satisfies the following algebraic requirements.*

Basic *If* $\mathtt{V}$ *specifies group elements included in* out *in the sense that* $\mathtt{V}$ *checks group memberships of them, then* $\mathcal{A}$ *should additionally output the corresponding representations* rep*.*

Extended *Let* $\overline{\mathcal{A}}$ *be an extended algorithm, which is naturally induced from* $\mathcal{A}$*, to output* $\overline{\mathsf{out}} := (\mathsf{out}, \mathsf{rep})$*. If* $\mathtt{V}$ *implies another PPT verification algorithm* $\overline{\mathtt{V}}(\mathsf{pp}, \cdot)$ *in the sense that*

$$\Pr[\overline{\mathtt{V}}(\mathsf{pp}, \overline{\mathsf{out}}) = 1 | \mathtt{V}(\mathsf{pp}, \mathsf{out}) = 1] \geq 1 - \mathsf{negl}(\lambda),$$

where λ *is a security parameter and the probability goes over the randomness used in generation of* pp *and* $\overline{\mathsf{out}}$*, then* $\overline{\mathcal{A}}^{\overline{\mathtt{V}}(\mathsf{pp},\cdot)}$ *should satisfy the above basic requirement.*

Additionally, we call an extended algebraic group model (EAGM) if all adversaries in it are modeled as extended algebraic algorithms.

It is clear that Definition 7 encompasses the original algebraic algorithm by setting the verification algorithm to specify only the description of the input and output of algebraic algorithms.

It is reasonable to claim that the representation rep is an intermediate value that the basic algebraic algorithm computed to create the final output out. That is, it is necessarily related to the final output.

$\overline{\mathtt{V}}$ in the extended algebraic algorithm is designed to specify which part of the intermediate value rep consists of group elements. Let us briefly consider

[2] It does not require for $\mathcal{A}$ to successfully output 1 with high probability, but just specifies its descriptions of input and output.

what algorithms can be $\overline{\mathsf{V}}$. In fact, $\overline{\mathsf{V}}$ can be a constant algorithm that constantly outputs 1 since it satisfies the overwhelming conditional probability requirement. However, the constant algorithm does not include any group membership test, so it does not differ from basic algebraic algorithms.

Therefore, any meaningful $\overline{\mathsf{V}}$ should include group membership tests. If rep contains a group element, $\overline{\mathsf{V}}$ can be the group membership test for it. Thus, $\overline{\mathsf{V}}$ serve as the group membership test for the intermediate values rep, and our definition of extended algebraic algorithms requires the adversary to output the representation of intermediate value rep along with the final output.

Similarly, we can repeatedly apply this approach to track any sequence of intermediate values necessarily related to the final outputs. Note that this is not a recursive process to extract the intermediate values the adversary computed, but rather an analysis to determine what should be included in the representations the adversary outputs along with out.

5 Zero-Testing Assumption

The group-based folding scheme in [45] is knowledge-sound under the DL assumption and can be made non-interactive in the random oracle model using the Fiat-Shamir transformation [31]. However, to use the non-interactive folding scheme in Nova IVC, the folding verifier must be arithmetized, requiring the random oracle to be instantiated in the standard model using a hash function.

There are studies [23,22] that aim to remove the heuristic instantiations of random oracles by introducing new variants. We propose a different approach to avoid heuristic analysis, as we do not seek to modify the Nova IVC construction but rather to provide a new soundness analysis.

To this end, we introduce a new plausible computational assumption for cryptographic hash functions, such as SHA-256, that is sufficient for proving knowledge soundness in the AGM. This hash assumption alone cannot fully replace random oracles, as it lacks the ability to extract a witness by rewinding algorithms. However, our new assumption can be combined with the EAGM to completely replace random oracles in the proof of Nova IVC.

5.1 Zero-Testing Problem over Hash Functions

In the context of proof systems, a polynomial is often used to prove several relations at the same time. For example, to prove three equality $a_i = b_i$ for $i = 0, 1, 2$, one can claim that the polynomial $\mathsf{p}(X) = \sum_i (a_i - b_i) X^i$ is identical to zero. In interactive protocols, the Schwartz-Zippel lemma enables to statistically verify it; (1) Prover commits to the polynomial $\mathsf{p}(X)$, (2) a random challenge r is chosen by the verifier, (3) check $\mathsf{p}(r) \stackrel{?}{=} 0$. In non-interactive protocols, the Fiat-Shamir transformation is applied. The second step can be changed with H evaluation and check if $\mathsf{p}(\mathsf{H}(\mathsf{p})) \stackrel{?}{=} 0$, where H is considered as the random oracle. In the random oracle model, we can rewind the prover multiple times with a fixed commitment. Therefore, p passing the test implies that p vanishes at multiple points larger than the degree of p, so that it is identical to zero. Although this argument in the non-interactive protocol is well analyzed in the random oracle model, we believe that even without the random oracle model, it is still reasonable to expect that the cryptographic hash function also guarantees this method of testing zero polynomial. We formalize this belief in Definition 8. Let λ be the security parameter and H be a cryptographic hash function that maps to $\mathbb{Z}_p$, where p is a prime of length $O(\lambda)$.

Definition 8. *(Zero-Testing Problem) Let* H *be a hash function whose output length is of size* λ*, and* p *is a prime of size* λ*. We define the problem of finding a nonzero polynomial* $\mathsf{p} \in \mathbb{Z}_p[X]$ *of degree at most* $\mathsf{poly}(\lambda)$ *that satisfies* $\mathsf{p}(\mathsf{H}(\mathsf{p})) = 0 \pmod p$ *as the zero-testing problem over* H*.*

Based on the Definition 8, we define a computational assumption called the *zero-testing assumption* (over H). The zero-testing assumption states that there is no PPT algorithms to solve the zero-testing problem over H.

In fact, the above zero-testing assumption is too simple to apply directly to various cryptosystems. We provide this information to help readers understand the intuition behind the following generalization of the zero testing problem

Definition 9. *(General Zero-Testing problem) Let* $\mathsf{C} : \mathcal{D} \to \mathcal{C}$ *be a binding commitment and* $\mathsf{D} : \mathcal{D} \to \mathbb{Z}_p[X]$ *be an arbitrary deterministic function where* $\mathcal{D}$ *is a domain set and* $\mathbb{Z}_p[X]$ *is a set of polynomials of degree at most* $\mathsf{poly}(\lambda)$*. For a hash function* H*, we define the problem of finding* $d \in \mathcal{D}$ *and auxiliary input* τ *such that* $D(d)$ *is a non-zero polynomial and* $\mathsf{D}(d)(\mathsf{H}(\mathsf{C}(d), \tau)) = 0 \pmod p$*.*

In the similar manner in the zero testing assumption, we define the *general zero-testing (GZT) assumption* (over H, C, and D) as follows: there is no PPT algorithms to solve the general zero-testing problem over H, C, and D.

Note that the general zero-testing problem is equivalent to the zero-testing problem if we set $\mathcal{D} = \mathbb{Z}_p[X]$, both C and D to be identity maps, and $\tau = \emptyset$.

We expect that cryptographic hash functions such as SHA-256 satisfy the zero-testing assumption. Although we do not provide a concrete analysis of this new assumption for hash functions, we can at least demonstrate that the (general) zero-testing problem over a random oracle hash is hard, as shown in the following lemmas. In particular, we emphasize that the proofs of these lemmas do not rely on the programmability of the random oracle but instead use only the uniform and independent distribution of the random oracle outputs. Therefore, the zero-testing assumption is weaker than the random oracle model.

Lemma 1. *Let* p *be a* λ*-bit prime and* $\mathsf{H} : \{0,1\}^* \to \mathbb{Z}_p$ *be a hash function. If* H *is random oracle, then the zero-testing problem over* H *is infeasible.*

Proof. In the random oracle model, one can obtain hash outputs by querying the random oracle. For each hash query p, the hash output $\mathsf{H}(\mathsf{p})$ is uniformly random, so that the probability $\mathsf{p}(\mathsf{H}(\mathsf{p})) = 0 \pmod p$ holds is at most $\deg(\mathsf{p})/p$. For $q \leq \mathsf{poly}(\lambda)$ distinct queries, all query results are mutually independent; thus the probability that at least one equality holds is bounded by the sum probability

$\frac{q\deg(\mathsf{p})}{p} \leq \frac{\mathsf{poly}(\lambda)}{2^\lambda}$, which is still negligible in λ. That is, the probability of solving the zero-testing problem is negligible. □

Lemma 2. *Let p be a λ-bit prime and $\mathsf{H} : \{0,1\}^* \rightarrow \mathbb{Z}_p$ be a hash function. Let C be a Pedersen commitment with binding property and $\mathcal{D}$ be an arbitrary deterministic function. Then, if H is random oracle, the general zero-testing problem over H, C, and D is infeasible.*

Proof. The basic proof strategy is identical to Lemma 1, except that we additionally require the ability that for each query (c, τ). In the similar manner in the proof of Lemma 1, the probability that the random oracle output $\mathsf{H}(c, \tau)$ as a root of the polynomial $\mathsf{D}(d)$ is negligible. Furthermore, by the binding property, it is infeasible to find collision pair d_1, d_2 such that $c = \mathsf{C}(d_1) = \mathsf{C}(d_2)$. That is, for a query (c, τ), the adversary know at most one d such that $(c, \tau) = (\mathsf{C}(d), \tau)$. In other words, the probability to find the solution (d, τ) is at most the probability of finding a random oracle query (c, τ) that satisfies $\mathsf{D}(d)(\mathsf{H}(c, \tau))$. The probability of successfully finding such a query is negligible.

□

Remark. In [18], a similar concept, called *zero-finding game*, to our general zero-testing problem was introduced. There is noticeably obvious differences between two concepts because of the goal of proposals. Lemma 3.3 of [18] assumes a *perfect* binding commitment and analyzes, from an information-theoretic perspective, the probability of an adversary successfully winning the zero-finding game, given access to *random oracle* and the ability to rewind. In contrast, we analyze the difficulty of breaking the general zero-testing problem under the assumptions of *computational* binding commitments and the *AGM*. Based on these assumptions, we provide a tighter bound on the adversary's success probability compared to the one derived in [18]. In particular, the (general) zero-testing assumption is proposed not to rely on the random oracles, contrary to [18].

5.2 Schnorr's NIZK in the AGM

As a warm-up example to show the effectiveness of the zero-testing assumption, we present a new knowledge-soundness proof of Schnorr's NIZK protocol [52], which is one of the simplest proof knowledge protocol; it proves that (G, H) is an instance of the relation $\mathcal{R} = \{(G, [x]_G; x \in \mathbb{Z}_p)\}$, i.e., $H = [x]_G$.

Prover
1. chooses $k \xleftarrow{\$} \mathbb{Z}_p$ and computes $K := [k]_G$.
2. computes $e \leftarrow \mathsf{H}(G, H, K)$.
3. computes $s = k + ex \bmod p$ and outputs (s, K).

Verifier accepts if,

given (s, K), $[s]_G \stackrel{?}{=} K + [e]_H$ holds, where $e \leftarrow \mathsf{H}(G, H, K)$.

Using the general zero-testing assumption, we can prove that Shnorr's non-interactive protocol is knowledge sound in the AGM. In particular, the extraction is tight and random oracles are not required.

Theorem 3. *Under the general zero-testing assumption over* H*, the Schnorr's non-interactive protocol satisfies the knowledge soundness in the AGM. In particular, the running time of the extractor is equivalent to that of the algebraic prover, except for constant operations.*

Proof. Given an arbitrary algebraic prover $\mathcal{P}^*$, we construct an extractor $\mathcal{E}$ that extracts the witness x. $\mathcal{P}^*$ begins with taking a pair of (G, H) as input. Suppose that $\mathcal{P}^*$ outputs a proof (s, K) that passes verification; that is, the equality $[s]_G = K + [e]_H$ holds where $e \leftarrow \mathsf{H}(G, H, K)$. Since $\mathcal{P}^*$ is an algebraic adversary, it should output the representation (k_1, k_2) of the group element K such that $K = [k_1]_G + [k_2]_H$. Thus, we have $[s - k_1]_G = [k_2 + e]_H$, so we obtain the discrete logarithm of H as $x = (s - k_1) \cdot (k_2 + e)^{-1} \pmod p$ unless $k_2 + e = 0 \pmod p$.

Now, we argue that $k_2 \neq -e \pmod p$. Suppose that $k_2 = -e \pmod p$. Then, $\mathsf{H}(G, H, [k_1]_G - [e]_H)$ is a solution of a polynomial $e - X = 0 \pmod p$. Using the notations d, C, and D in the general zero-testing problem in Definition 9, we can set $d = (k_1, -e)$, where (G, H) is the commitment key of C, and $\mathsf{D}(k_1, -e) = e - X \in \mathbb{Z}_p[X]$, where D discards k_1. Therefore, no PPT algorithm can find $d = (k_1, -e)$ that satisfies $\mathsf{D}(d)(\mathsf{H}(G, H, \mathsf{C}(d))) = 0 \pmod p$ by the general zero-testing assumption, so that $k_2 \neq -e \pmod p$.

What the extractor did except running $\mathcal{P}^*$ is only to compute constant operations $x = (s - k_1) \cdot (k_2 + e)^{-1} \pmod p$. □

6 New Soundness Analysis of Nova IVC with Group-based Folding Scheme

Pedersen Commitment for Vectors. Pedersen commitment scheme is a homomorphic commitment scheme with perfect hiding and computational binding properties under the discrete logarithm assumption. The setup algorithm $\mathsf{Setup}(1^\lambda, \ell)$ takes the dimension variable ℓ and outputs the commitment key ck consisting of a $(\ell+1)$-dimensional vector $\mathbb{G}^{\ell+1}$. The message $\boldsymbol{x}$ is an ℓ-dimensional vector in $\mathbb{Z}_p^\ell$. The commitment to $\boldsymbol{x}$ with a random scalar $r \xleftarrow{\$} \mathbb{Z}_p$ is computed as a multi-scalar addition $\langle \boldsymbol{x} \| r, \mathsf{ck} \rangle \leftarrow \mathsf{Com}(\mathsf{ck}, \boldsymbol{x}; r)$. The homomorphic property is naturally induced by the characteristics of the cyclic group $\mathbb{G}$.

Group-based Folding Scheme from [45]. In [45], the group-based non-interactive folding scheme $\mathsf{NIFS} = (\mathsf{G}, \mathsf{K}, \mathsf{P}, \mathsf{V})$ for the committed relaxed R1CS relation $\mathcal{R}_{\mathsf{pp}_{FS}, \mathsf{s}}$ in Eq. (2) is proposed, where the public parameter pp_{FS} is generated by G and the common structure s is taken as an input of K. The folding prover $\mathsf{NIFS.P}$ takes two committed relaxed R1CS instance-witness pairs and outputs a folded instance-witness pair (u, v), with the prover's transcript T. The folding verifier $\mathsf{NIFS.V}$ takes two committed relaxed R1CS instances u_1, u_2,

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

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

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

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. $r \leftarrow \mathsf{H}(\mathsf{u}_1, \mathsf{u}_2, T)$.
5. Output $T, \mathsf{u} := (E, s, W, \mathsf{x})$ and $\mathsf{v} := (\boldsymbol{e}, r_{\boldsymbol{e}}, \boldsymbol{w}, r_{\boldsymbol{w}})$ where

$$E \leftarrow E_1 + r \cdot T + r^2 \cdot E_2 \qquad W \leftarrow W_1 + r \cdot W_2 \tag{5}$$
$$s \leftarrow s_1 + r \cdot s_2 \qquad \mathsf{x} \leftarrow \mathsf{x}_1 + r \cdot \mathsf{x}_2 \tag{6}$$
$$\boldsymbol{e} \leftarrow \boldsymbol{e}_1 + r \cdot \boldsymbol{t} + r^2 \cdot \boldsymbol{e}_2 \qquad \boldsymbol{w} \leftarrow \boldsymbol{w}_1 + r \cdot \boldsymbol{w}_2 \tag{7}$$
$$r_{\boldsymbol{e}} \leftarrow r_{\boldsymbol{e}_1} + r \cdot r_{\boldsymbol{t}} + r^2 \cdot r_{\boldsymbol{e}_2} \qquad r_{\boldsymbol{w}} \leftarrow r_{\boldsymbol{w}_1} + r \cdot r_{\boldsymbol{w}_2} \tag{8}$$

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

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

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

and T and then outputs a folded instance u. We have provided a full description of this group-based folding scheme in Figure 6.

Looking at the Knowledge Soundness Proof of Nova [45]. In this paragraph, we briefly review the knowledge soundness proof of Nova [45]. The premise of the proof is the knowledge soundness of the internal non-interactive folding scheme in the standard model, which assumes the existence of the extractor $\tilde{\mathcal{E}}$ satisfying condition in Definition 4. To construct the IVC extractor $\mathcal{E}$, which outputs $(\omega_0, \ldots, \omega_{n-1})$, the proof follows a general recursive proof strategy. That is, $\mathcal{E}$ inductively generates $\mathcal{E}_i$ that, given $\mathcal{E}_{i+1}$, outputs $(z_i, \ldots, z_{n-1})$, $(\omega_i, \ldots, \omega_{n-1})$ and Π_n. In fact, $\mathcal{E}_{i+1}$ directly implies an adversarial folding prover $\tilde{\mathcal{A}}_i$ for the i-th round and $\mathcal{E}_i$ can be constructed from $\tilde{\mathcal{A}}_i$. In the procedure of $\mathcal{E}_i$, the folding extractor $\tilde{\mathcal{E}}_i$ of $\tilde{\mathcal{A}}_i$ is additionally called, so that the inequality between the running times of the algorithms is as follows:

$$\mathtt{time}(\mathcal{E}_i) > \mathtt{time}(\tilde{\mathcal{E}}_i) + \mathtt{time}(\tilde{\mathcal{A}}_i) > 2 \cdot \mathtt{time}(\mathcal{E}_{i+1})$$

if $\mathtt{time}(\tilde{\mathcal{E}}_i) > \mathtt{time}(\tilde{\mathcal{A}}_i)$. Then, $\mathtt{time}(\mathcal{E})$ increases exponentially in n. The soundness proof of the Nova paper relies on the assumption of the knowledge soundness of the non-interactive folding scheme in the standard model when the random oracle is instantiated with a cryptographic hash function. Considering the corresponding interactive folding scheme (or non-interactive scheme in the random oracle model), $\tilde{\mathcal{E}}_i$ uses the rewinding strategy with the forking lemma so that $\texttt{time}(\tilde{\mathcal{E}}_i) > \texttt{time}(\tilde{\mathcal{A}}_i)$ holds. To avoid exponential growth, we do not apply the folding scheme extractor to construct the IVC extractor, ensuring that $\texttt{time}(\mathcal{E}_i)$ increases only incrementally without growing exponentially.

6.1 Knowledge Soundness of NIFS in the AGM

Before the proof of knowledge soundness Nova IVC based on the NIFS scheme in Figure 6, we prove that NIFS with a general zero-testing hash satisfies knowledge soundness in Definition 4. Although we avoid using the folding extractor as a subroutine to construct the IVC extractor, we use the fact that the NIFS scheme Figure 6 satisfies knowledge soundness in AGM to prove the knowledge soundness of Nova IVC. In a nutshell, a representation provided by an extended algebraic adversary is indeed a witness of the instance. Concretely, by the knowledge soundness of NIFS, an IVC adversary outputting a valid pair u, v should know the original pairs $\mathsf{u}_1, \mathsf{v}_1$ and $\mathsf{u}_2, \mathsf{v}_2$ beforehand. In the view of the adversary, u_1 and u_2 are group-convertible elements; therefore, it should output their representation, but it may not witness for the instance. However, knowledge soundness guarantees that the adversary knows a witness so that if the adversary can obtain a representation different from the witness, the adversary can know the discrete relation of the CRS, which contradicts the DL assumption. Now, we prove the knowledge soundness of NIFS under the AGM with DL assumption.

Theorem 4 (Knowledge Soundness of NIFS in AGM). *Let* H *be a general zero-testing hash function. Then, the group-based non-interactive folding scheme* $\mathsf{NIFS} = (\mathsf{G}, \mathsf{K}, \mathsf{NIFS.P}, \mathsf{NIFS.V})$ *in Figure 6 satisfies knowledge soundness in AGM with DL assumption.*

Proof Sketch. For the knowledge soundness proof, we construct an extractor that outputs witnesses for the given folded instances u_1 and u_2 using an algebraic adversary. The extractor is designed to output algebraic representations from the adversary. Note that the general zero-testing hash assumption guarantees that these representations are indeed valid witnesses without rewinding the adversary. The complete proof is deferred to Appendix B.

6.2 Poly-depth Knowledge Soundness of Nova in EAGM

In this section, we prove the poly-depth KS (Definition 3) of the Nova IVC scheme (Figure 3) in the EAGM (Definition 7).

Theorem 5. *Let* H *be a cryptographic hash function. The Nova IVC scheme* $(\mathcal{G}, \mathcal{K}, \mathcal{P}, \mathcal{V})$ *in Figure 3 combined with the group-based folding scheme* NIFS *based on* $\mathbb{G}$ *and* H *in Figure 6 satisfies poly-depth knowledge soundness Definition 3 in the EAGM (Definition 7) under the DL assumption and general zero-test assumption over* H.

Proof Outline. We consider extended algebraic adversaries $\mathcal{A}$ with respect to the IVC verifier $\mathcal{V}$. That is, $\mathcal{A}$ takes the time period n of polynomial size in the security parameter λ and the public parameters pp of the Nova IVC, and then outputs a tuple (F, z_0, z_n, Π_n) of the target function F, a genesis value z_0, an n-th output z_n, and a proof Π_n. We construct an expected polynomial time extractor $\mathcal{E}(\mathsf{pp}, z_0, z_n; \rho) \rightarrow (\omega_i)_{i=0}^{n-1}$, where ρ is the randomness used by the adversary. We will show that if the adversarial output passes the IVC verifier $\mathcal{V}$, then the extractor's output, $(\omega_i)_{i=0}^{n-1}$, satisfies $\forall i \in [n], z_i \leftarrow F(z_{i-1}, \omega_{i-1})$.

By Definition 7, the extended algebraic adversaries $\mathcal{A}^{\mathcal{V}(\mathsf{pp},\cdot)}$ should output the representations defined by not only $\mathcal{V}$ but also induced verification algorithms $\overline{\mathcal{V}}$. That is, the extended algebraic adversary has a duty to output the representations corresponding to the valid outputs passing $\mathcal{V}$ and $\overline{\mathcal{V}}$.

Our proof outline is as follows.

1. *Look into adversarial outputs in terms of passing $\mathcal{V}$:* The IVC verifier $\mathcal{V}$ checks the adversarial output out_n is a valid pair of instance and witness of $\mathcal{R}_{\mathsf{Nova}}$ that consists of several Pedersen commitments and openings. Since every group operation necessarily checks group membership checks, it is natural to suppose that $\mathcal{V}$ specifies group-convertible elements. Therefore, $\mathcal{A}$ additionally outputs the corresponding representations rep_n. Let $\mathsf{out}_{n-1} := (\mathsf{out}_n, \mathsf{rep}_n)$. Looking into out_{n-1}, we can find a candidate of the previous round proof Π_{n-1} that contains ω_{n-1}.

2. *Define another verification algorithms $\overline{\mathcal{V}}$:* We can define additional verification algorithm $\overline{\mathcal{V}}$ as checking the correctness of the extractor's finding Π_{n-1}. That is, Π_{n-1} belongs to $\mathcal{R}_{\mathsf{Nova}}$.

3. *Prove that $\overline{\mathcal{V}}$ satisfies the extension requirement in Definition 7:* By our hypothesis given in the theorem statement, H is a general zero-testing hash function and NIFS is the group-based folding scheme over the groups under the DL assumption. Applying Theorem 4, NIFS has the knowledge soundness under our hypothesis. We prove that the knowledge soundness of NIFS implies an inequality

$$\Pr[\overline{\mathcal{V}}(\mathsf{pp}, \mathsf{out}_{n-1}) = 1 | \mathcal{V}(\mathsf{pp}, \mathsf{out}_n) = 1] \geq 1 - \mathsf{negl}(\lambda).$$

4. *Repeat the processes* **1**. - **3**. *with $\overline{\mathcal{V}}$, instead of $\mathcal{V}$.* $\mathcal{V}$ checks the validity of the proof Π_n. That is, Π_n is a belonging of $\mathcal{R}_{\mathsf{Nova}}$. Similarly, $\overline{\mathcal{V}}$ checks the validity of Π_{n-1}, a belonging of $\mathcal{R}_{\mathsf{Nova}}$. Therefore, due to a similar structure between two algorithms, we can repeats the processes **1**. - **3**., by using $\overline{\mathcal{V}}$ instead of $\mathcal{V}$. Note that these repeats are not a recursive process, but a process of finding what $\mathcal{A}$ should include in the representations by finding appropriate additional verification functions. That is, these repeats are just to define and analyze additional verification algorithms and there is no recursive process in $\mathcal{E}$.

Full proof is deferred to Appendix C.

Ephemeral Nova does not satisfy poly-depth knowledge soundness. Using the above idea, we can construct extractor that covers linearly many rounds. Furthermore, in Section 3.2, we provide a concrete linear-round attack on Ephemeral Nova. In this case, the knowledge soundness adversary can forge an IVC proof, making the adversary's advantage (the probability in Eq.(1)) non-negligible. By extending the definition of knowledge soundness, we can conclude that Ephemeral Nova is not sound.

7 Concluding Remarks

In this paper, we showed that an unnecessary redundant procedure in the augmented function F' may serve as a trigger for attacks that activate only at a predetermined time. To investigate this type of attack on the Nova IVC scheme, it is necessary to prove knowledge soundness for polynomially many rounds.

We presented the first provable security analysis of Nova IVC's knowledge soundness for polynomial rounds. In particular, our proof does not rely on heuristic random oracle instantiation but instead on a newly introduced computational assumption for hash functions, the general zero-testing assumption.

There are several interesting open questions. Many other IVC schemes have soundness proofs only for logarithmic rounds, and studying their security in the polynomial round setting would be valuable. In particular, our proposed EAGM may be useful for group-based schemes. Additionally, finding an alternative security proof for Nova IVC in the standard model would be another interesting direction for future research.

References

1. Michel Abdalla, Fabrice Benhamouda, and Philip MacKenzie. 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*, pages 571–587. IEEE Computer Society, 2015.
2. Masayuki Abe, Jens Groth, and Miyako Ohkubo. Separating short structure-preserving signatures from non-interactive assumptions. In *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 628–646. Springer, 2011.
3. Balthazar Bauer, Georg Fuchsbauer, and Julian Loss. A classification of computational assumptions in the algebraic group model. In *CRYPTO (2) 2020*, volume 12171 of *LNCS*, pages 121–151. Springer, 2020.
4. Eli Ben-Sasson, Alessandro Chiesa, Eran Tromer, and Madars Virza. Scalable zero knowledge via cycles of elliptic curves. *Algorithmica*, 79:1102–1160, 2017.
5. David Bernhard, Marc Fischlin, and Bogdan Warinschi. On the hardness of proving cca-security of signed elgamal. In *PKC (1) 2016*, volume 9614 of *LNCS*, pages 47–69. Springer, 2016.
6. Nir Bitansky, Ran Canetti, Alessandro Chiesa, and Eran Tromer. From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again. In *ITCS12*, LNCS, pages 326–349. Springer, 2012.

7. Dan Boneh, Joseph Bonneau, Benedikt Bünz, and Ben Fisch. Verifiable delay functions. In *Annual international cryptology conference*, pages 757–788. Springer, 2018.
8. Dan Boneh, Justin Drake, Ben Fisch, and Ariel Gabizon. Efficient polynomial commitment schemes for multiple points and polynomials. *Cryptology ePrint Archive, Report 2020/81*, page 81, 2020.
9. Dan Boneh, Justin Drake, Ben Fisch, and Ariel Gabizon. Halo infinite: Proof-carrying data from additive polynomial commitments. In *CRYPTO (1) 2021*, volume 12825 of *LNCS*, pages 649–680. Springer, 2021.
10. Dan Boneh and Ramarathnam Venkatesan. Breaking RSA may not be equivalent to factoring. In *EUROCRYPT 1998*, volume 1403 of *LNCS*, pages 59–71. Springer, 1998.
11. Joseph Bonneau, Izaak Meckler, Vanishree Rao, and Evan Shapiro. Coda: Decentralized cryptocurrency at scale. *Cryptology ePrint Archive*, 2020.
12. Joseph Bonneau, Izaak Meckler, Vanishree Rao, and Evan Shapiro. Mina: Decentralized cryptocurrency at scale. *New York Univ. O (1) Labs, New York, NY, USA, Whitepaper*, pages 1–47, 2020.
13. Bertalan Borsos, Attila Kovács, and Norbert Tihanyi. Tight upper and lower bounds for the reciprocal sum of proth primes. *The Ramanujan Journal*, 59(1):181–198, 2022.
14. Sean Bowe, Jack Grigg, and Daira Hopwood. Recursive proof composition without a trusted setup. *Cryptology ePrint Archive*, 2019.
15. Emmanuel Bresson, Jean Monnerat, and Damien Vergnaud. Separation results on the "one-more" computational problems. In *CT-RSA 2008,*, volume 4964 of *LNCS*, pages 71–87. Springer, 2008.
16. Daniel RL Brown. Cm55: special prime-field elliptic curves almost optimizing den boer's reduction between diffie-hellman and discrete logs. *Cryptology ePrint Archive, Report 2014/877*, page 877, 2014.
17. Benedikt Bünz and Binyi Chen. Protostar: generic efficient accumulation/folding for special-sound protocols. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 77–110. Springer, 2023.
18. Benedikt Bünz, Alessandro Chiesa, Pratyush Mishra, and Nicholas Spooner. Proof-carrying data from accumulation schemes. 2020. `https://eprint.iacr.org/2020/499.pdf`.
19. Benedikt Bünz, Alessandro Chiesa, Pratyush Mishra, and Nicholas Spooner. 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*, pages 1–18. Springer, 2020.
20. Benedikt Bünz, Ben Fisch, and Alan Szepieniec. Transparent SNARKs from DARK compilers. In *EUROCRYPT 2020*, volume 12105 of *LNCS*, pages 677–706. Springer, 2020.
21. Timothy Joseph Casey. *Searching for large Proth primes.* Central Missouri State University, 2003.
22. Megan Chen, Alessandro Chiesa, Tom Gur, Jack O'Connor, and Nicholas Spooner. Proof-carrying data from arithmetized random oracles. In *EUROCRYPT (2) 2023*, volume 14005, pages 379–404, 2022.
23. Megan Chen, Alessandro Chiesa, and Nicholas Spooner. On succinct non-interactive arguments in relativized worlds. In *EUROCRYPT (2) 2022*, volume 13276, pages 336–366, 2022.

24. Alessandro Chiesa, Yuncong Hu, Mary Maller, Pratyush Mishra, Noah Vesely, and Nicholas Ward. Marlin: Preprocessing zksnarks with universal and updatable srs. In *EUROCRYPT 2020*, volume 12105 of *LNCS*, pages 738–768. Springer, 2020.
25. Alessandro Chiesa, Dev Ojha, and Nicholas Spooner. Fractal: Post-quantum and transparent recursive proofs from holography. In *EUROCRYPT 2020 Part I*, volume 12105 of *LNCS*, pages 769–793. Springer, 2020.
26. Alessandro Chiesa and Eran Tromer. Proof-carrying data and hearsay arguments from signature cards. In *ICS*, volume 10, pages 310–331, 2010.
27. Coda. `https://codaprotocol.com`.
28. Jean-Sébastien Coron. Optimal security proofs for PSS and other signature schemes. In *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 272–287. Springer, 2002.
29. Ivan Damgård. Towards practical public key systems secure against chosen ciphertext attacks. In *Advances in Cryptology—CRYPTO'91: Proceedings 11*, pages 445–456. Springer, 1992.
30. Ethereum (ETH) Blockchain Explorer. `https://etherscan.io`.
31. Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO 1986*, volume 263 of *LNCS*, pages 186–194. Springer, 1987.
32. Georg Fuchsbauer, Eike Kiltz, and Julian Loss. The algebraic group model and its applications. In *CRYPTO (2) 2018*, volume 10992 of *LNCS*, pages 33–62. Springer, 2018.
33. Georg Fuchsbauer, Antoine Plouviez, and Yannick Seurin. Blind schnorr signatures and signed elgamal encryption in the algebraic group model. In *Advances in Cryptology–EUROCRYPT 2020: 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10–14, 2020, Proceedings, Part II 30*, pages 63–95. Springer, 2020.
34. Sanjam Garg, Raghav Bhaskar, and Satyanarayana V. Lokam. Improved bounds on security reductions for discrete log based signatures. In *CRYPTO 2008*, volume 5157 of *LNCS*, pages 93–107. Springer, 2008.
35. Rosario Gennaro, Craig Gentry, Bryan Parno, and Mariana Raykova. Quadratic span programs and succinct nizks without pcps. In *EUROCRYPT 2013*, pages 626–645. Springer, 2013.
36. Craig Gentry and Daniel Wichs. Separating succinct non-interactive arguments from all falsifiable assumptions. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 99–108, 2011.
37. Ashrujit Ghoshal and Stefano Tessaro. Tight state-restoration soundness in the algebraic group model. In *CRYPTO (3) 2021*, volume 12827 of *LNCS*, pages 64–93. Springer, 2021.
38. Sergey Gorbunov, Leonid Reyzin, Hoeteck Wee, and Zhenfei Zhang. Pointproofs: Aggregating proofs for multiple vector commitments. In *CCS '20*, pages 2007–2023. ACM, 2020.
39. Assimakis Kattis and Joseph Bonneau. Proof of necessary work: succinct state verification with fairness guarantees. In *International Conference on Financial Cryptography and Data Security*, pages 18–35. Springer, 2023.
40. Jonathan Katz, Julian Loss, and Jiayu Xu. On the security of time-lock puzzles and timed commitments. In *TCC (3) 2020*, volume 12552 of *LNCS*, pages 390–413. Springer, 2020.
41. Dmitry Khovratovich, Mary Maller, and Pratyush Ranjan Tiwari. Minroot: Candidate sequential function for ethereum vdf. *Cryptology ePrint Archive*, 2022.

42. Eike Kiltz, Daniel Masny, and Jiaxin Pan. Optimal security proofs for signatures from identification schemes. In *CRYPTO (2) 2016*, volume 9815 of *LNCS*, pages 33–61. Springer, 2016.
43. Abhiram Kothapalli and Srinath Setty. Supernova: Proving universal machine executions without universal circuits. *Cryptology ePrint Archive*, 2022.
44. Abhiram Kothapalli and Srinath Setty. Hypernova: Recursive arguments for customizable constraint systems. In *Annual International Cryptology Conference*, pages 345–379. Springer, 2024.
45. Abhiram Kothapalli, Srinath Setty, and Ioanna Tzialla. Nova: Recursive zero-knowledge arguments from folding schemes. In *CRYPTO (4) 2022*, volume 13510 of *LNCS*, pages 359–388. Springer, 2022.
46. Jonathan Lee, Kirill Nikitin, and Srinath Setty. Replicated state machines without replicated execution. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 119–134. IEEE, 2020.
47. Ueli Maurer, Christopher Portmann, and Jiamin Zhu. Unifying generic group models. *Cryptology ePrint Archive, Report 2020/996*, page 996, 2020.
48. Ueli M. Maurer. Abstract models of computation in cryptography (invited paper). In *10th IMA International Conference on Cryptography and Coding*, volume 3796, pages 1–12. Springer, 2005.
49. V. I. Nechaev. Complexiy of a determinate algorithm for the discrete logarithm. In *Mathematical Notes*, volume 55(2), pages 165–172, 1994.
50. Wilson Nguyen, Dan Boneh, and Srinath Setty. Revisiting the nova proof system on a cycle of curves. *Cryptology ePrint Archive*, 2023.
51. Pascal Paillier and Damien Vergnaud. Discrete-log-based signatures may not be equivalent to discrete log. In *ASIACRYPT 2005*, volume 3788 of *LNCS*, pages 1–20. Springer, 2005.
52. Claus-Peter Schnorr. Efficient signature generation by smart cards. *Journal of cryptology*, 4:161–174, 1991.
53. Srinath Setty. Spartan: Efficient and general-purpose zksnarks without trusted setup. In *CRYPTO 2020*, LNCS. Springer, 2020.
54. Victor Shoup. Lower bounds for discrete logarithms and related problems. In *EUROCRYPT 1997*, volume 1233 of *LNCS*, pages 256–266. Springer, 1997.
55. Paul Valiant. Incrementally verifiable computation or proofs of knowledge imply time/space efficiency. In *TCC 2008*, volume 4948 of *LNCS*, pages 1–18. Springer, 2008.
56. Mark Zhandry. To label, or not to label (in generic groups). In *CRYPTO 2022*, volume 13509, pages 66–96, 2022.

A Proof for Theorem 2

Proof. (*Completeness*): We argue that the $\mathcal{P}$'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 $\mathcal{P}(\mathsf{pk}, i, z_0, z_i, \omega_i, \Pi_i) \rightarrow \Pi_{i+1}$, 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 $((\mathsf{u}_\perp, \mathsf{v}_\perp), (\mathsf{u}_\perp, \mathsf{v}_\perp))$. Now, we consider the validity of the updated proof Π_1. Let

$\mathcal{P}_1$ take the input $(\mathsf{pk}, (1, z_0, z_1, \omega_0, \Pi_0, Y_0)$ and then get Π_1. From the $\mathcal{P}_1$ in Figure 3, 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'_1 execution. Following the execution F'_1 in Figure 4, we know

$$\mathsf{u}_1.\mathsf{x} = \mathsf{H}(\mathsf{vk}, 1, z_0, F(z_0, \omega_0), \mathsf{u}_\perp, Y_1) \text{ where } Y_1 = Y_0^2 = 1 \quad (9)$$

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

From Eq. (9) and Eq. (10), second and third verifier conditions in Figure 5 hold. To check the fourth condition, we only consider $(\mathsf{u}_1, \mathsf{v}_1) \in \mathcal{R}_{\mathsf{pp}_{FS}, \mathsf{s}_{\mathsf{F}'}}$ because $(\mathsf{U}_1, \mathsf{V}_1) = (\mathsf{u}_\perp, \mathsf{v}_\perp)$ is already belong in the relation. From the tracing of F', $(\mathsf{u}_1, \mathsf{v}_1)$ 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 $(\mathsf{pk}, (i, z_0, z_i, \omega_i, \Pi_i, Y_i)$

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. $(\mathsf{U}_{i+1}, \mathsf{V}_{i+1}) \in \mathcal{R}_{\mathsf{pp}_{FS}, \mathsf{s}_{\mathsf{F}'}}$.

From the tracing of F' execution with input $(\mathsf{U}_i, \mathsf{u}_i, (i, z_0, z_i), \omega_i, T, Y_i, 1)$, we have that $\mathsf{u}_{i+1}.\mathsf{x} = \mathsf{H}(\mathsf{pp}, i+1, z_0, z_{i+1}, \mathsf{U}_{i+1}, Y_{i+1})$ where $Y_{i+1} = Y_i^2 = 1$ and $(\mathsf{u}_{i+1}.E, \mathsf{u}_{i+1}.s) = (\mathsf{u}_\perp.E, 1)$. Therefore, the verifier $\mathcal{V}$ should accept 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 fixed step n, let the security parameter λ satisfy the following inequality: $\frac{\lambda}{2} \geq n$ and p be a λ-bit prime number. First, we claim that if the IVC verifier accepts the proof Π_n of n times execution F'_{b_i}, then all execution types of i-th step should be F'_1 with high probability.

Let $j-1$ be the latest step of execution F' with the choice bit $\mathsf{b} = 0$. In this case, $Y_j = Y_{j-1}^{2\alpha} \cdot \mathsf{u}_i.\mathsf{x}$ can be viewed as a uniform random sample from $\mathbb{Z}_p^*$ because $\mathsf{u}_i.\mathsf{x}$ is an image of H. From our hypothesis regarding the latest step, Y_n can be described by the following equation:

$$Y_n = Y_j^{(2\alpha)^{n-j}} \quad (11)$$

Due to the premise of acceptance by $\mathcal{V}$ in Figure 5, the following relation holds: $\mathsf{u}_n.\mathsf{x} = \mathsf{H}(\mathsf{pp}, n, z_0, z_n, \mathsf{U}_n, 1)$. On the other hand, the R1CS relation for F' constrains that the last input of H is Y_n. Therefore, $Y_n = 1$ holds with overwhelming probability. To claim that the probability of $Y_j^{(2\alpha)^{n-j}} = 1$ is $\mathsf{negl}(\lambda)$, let us consider the following Lemma 3.

Lemma 3. *Let $p = \alpha \cdot 2^\lambda$ be a prime with odd integer α. If integer n satisfies $\frac{\lambda}{2} \geq n$, then the following probability equation holds.*

$$\Pr_{x \xleftarrow{\$} \mathbb{Z}_p^*} [x^{(2\alpha)^n} = 1] \leq 2^{-\frac{\lambda}{2}} \quad (12)$$

Proof. Since the multiplicative group $\mathbb{Z}_p^*$ has order $\alpha \cdot 2^k$, the α^n-power subgroup $H := \{x^{\alpha^n} | x \in \mathbb{Z}_p^*\}$ has 2^k distinct elements. From the subgroup H, we can describe the probability as:

$$\Pr_{x \xleftarrow{\$} \mathbb{Z}_p^*}[x^{(2\alpha)^n} = 1] = \Pr_{x \xleftarrow{\$} \mathbb{Z}_p^*}[(x^{\alpha^n})^{2^n} = 1] = \Pr_{y \xleftarrow{\$} H}[y^{2^n} = 1]$$

To get upper bound of the probability $\Pr_{y \xleftarrow{\$} H}[y^{2^n} = 1]$, let us consider the upper bound of total number of $y \in H$ such that $y^{2^n} = 1$. If $y \in H$ satisfies $y^{2^n} = 1$, y should be a root of the polynomial $X^{2^n} - 1 \in \mathbb{Z}_p[X]$. By the fundamental theorem of algebra, $X^{2^n} - 1 \in \mathbb{Z}_p[X]$ has at most 2^n distinct roots, which means that the number of ys is at most 2^n. Therefore, the probability $\Pr_{y \xleftarrow{\$} H}[y^{2^n} = 1]$ is at most $\frac{2^n}{2^\lambda} = \frac{1}{2^{\lambda - n}} \leq 2^{-\frac{\lambda}{2}}$ □

By Lemma 3 and our premise $\frac{\lambda}{2} \geq n$, we can conclude that the probability of $Y_j^{(2\alpha)^{n-j}} = 1$ is negligible. For this reason, the probability of the case $\mathsf{b} = 0$ for any i-step is $\mathsf{negl}(\lambda)$. Then, we can consider that all execution types of i-th step should be F_1' with the exception of negligible probability.

Now, we only consider that augmented execution is F_1'. The following process is similar to soundness proof of Nova-IVC [45].

Let $\mathsf{pp} \leftarrow \mathcal{G}(1^\lambda)$. Consider an expected polynomial-time adversary $\mathcal{P}^*$ that outputs a function F on input pp, and let $(\mathsf{pk}, \mathsf{vk}) \leftarrow \mathcal{K}(\mathsf{pp}, F)$. Suppose that, for a constant $n \leq \lambda$, $\mathcal{P}^*$ outputs (z_0, z, Π) such that

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

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

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

and $z_n = z$ with the exception of the probability $\mathsf{negl}(\lambda)$.

We show inductively that $\mathcal{E}$ can run an expected polynomial-time extractor $\mathcal{E}_i(\mathsf{pp})$ that 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

$$\mathcal{V}(\mathsf{vk}, i, z_0, z_i, \Pi_i) = 1 \tag{13}$$

for $z_n = z$ with the exception of the probability $\mathsf{negl}(\lambda)$.

$\mathcal{E}$ run $\mathcal{E}_n$ first, and then using $\mathcal{E}_n$, construct $\mathcal{E}_{n-1}$ and repeat this process until reaching $\mathcal{E}_0$.

First, $\mathcal{E}_n(\mathsf{pp}, \rho)$ outputs $(\perp, \perp, \Pi_n)$, where Π_n is the output of $\mathcal{P}^*(\mathsf{pp}, \rho)$. Assume that $\mathcal{E}_n$ succeeds to get valid proof Π_n from IVC adversary $\mathcal{P}^*$.

For $i \geq 1$, suppose $\mathcal{E}$ can construct an expected polynomial-time extractor $\mathcal{E}_i$ that outputs $((z_i, \ldots, z_{n-1}), (\omega_i, \ldots, \omega_{n-1}))$, and Π_i that satisfies the inductive hypothesis. To construct an extractor $\mathcal{E}_{i-1}$, $\mathcal{E}$ first constructs an adversary $\mathcal{A}_{i-1}$ for the non-interactive folding scheme as follows:
$\underline{\tilde{\mathcal{A}}_{i-1}(\mathsf{pp}, \rho)}$:

1. Let $((z_i, \ldots, z_{n-1}), (\omega_i, \ldots, \omega_{n-1}), \Pi_i) \leftarrow \mathcal{E}_i(\mathsf{pp}, \rho)$.
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 that $\mathcal{V}(\mathsf{vk}, i, z_0, z_i, \Pi_i) = 1$, where $\Pi_i \leftarrow \mathcal{E}_i(\mathsf{pp})$ with the exception of negligible probability $\mathsf{negl}(\lambda)$. Therefore, by the verifier's checks we have that $(\mathsf{u}_i, \mathsf{v}_i)$ and $(\mathsf{U}_i, \mathsf{V}_i)$ are satisfying instance-witness pairs, and that

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

Because $\mathcal{V}$ ensures that $(\mathsf{u}_i.E, \mathsf{u}_i.u) = (\mathsf{u}_\perp.E), 1)$, we have that v_i is indeed a satisfying assignment for F'. Then, by the construction of F' and the binding property of the hash function, we have that

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

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

Given an expected polynomial-time $\tilde{\mathcal{A}}_{i-1}$ and an expected polynomial-time folding scheme extractor $\tilde{\mathcal{E}}_{i-1}, \mathcal{E}$ constructs an expected polynomial time $\mathcal{E}_{i-1}$ as follows
$\underline{\mathcal{E}_{i-1}(\mathsf{pp}, \rho)}$:

1. $((\mathsf{U}_{i-1}, \mathsf{u}_{i-1}), (\mathsf{U}_i, \mathsf{V}_i), T_{i-1}) \leftarrow \tilde{\mathcal{A}}_{i-1}(\mathsf{pp}, \rho)$
2. Retrieve $((z_i, \ldots, z_{n-1}), (\omega_i, \ldots, \omega_{n-1}), \Pi_i)$ from the internal state of $\mathcal{A}_{i-1}$
3. Parse $\Pi_i.\mathsf{v}_i$ 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. Let $\Pi_{i-1} \leftarrow ((\mathsf{U}_{i-1}, \mathsf{V}_{i-1}), (\mathsf{u}_{i-1}, \mathsf{v}_{i-1}))$
6. Output $((z_{i-1}, \ldots, z_{n-1}), (\omega_{i-1}, \ldots, \omega_{n-1}), \Pi_{i-1})$

We first reason that the output $(z_{i-1}, \ldots, z_{n-1})$, and $(\omega_{i-1}, \ldots, \omega_{n-1})$ are valid. By the inductive hypothesis, we already have that for all $j \in \{i+1, \ldots, n\}$,

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

and that $\mathcal{V}(\mathsf{vk}, i, z_0, z_i, \Pi_i) = 1$ with the exception of $\mathsf{negl}(\lambda)$. Because $\mathcal{V}$ additionally checks that

$$\mathsf{u}_i.\mathsf{x} = \mathsf{H}(\mathsf{vk}, i, z_0, z_i, \mathsf{U}_i, Y_i) \tag{14}$$

by the construction of F_1' and the binding property of the hash function, we have

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

with the exception of $\mathsf{negl}(\lambda)$. Next, we argue that Π_{i-1} is valid. Because $(\mathsf{u}_i, \mathsf{v}_i)$ satisfies F', and $(\mathsf{U}_{i-1}, \mathsf{u}_{i-1})$ were retrieved from v_i, by the binding property of the hash function, and by Eq. (14), we have that

$$\mathsf{u}_{i-1}.\mathsf{x} = \mathsf{H}(\mathsf{vk}, i-1, z_0, z_{i-1}, \mathsf{U}_{i-1}, Y_{i-1})$$
$$(\mathsf{u}_{i-1}.E, \mathsf{u}_{i-1}.s) = (\mathsf{u}_{\perp}.E, 1)$$

Additionally, in the case where $i = 1$, by the base case check of F_1', we have that $z_{i-1} = z_0$. Because $\tilde{\mathcal{E}}_{i-1}$ succeeds with the exception of $\mathsf{negl}(\lambda)$, we have that

$$\mathcal{V}(\mathsf{vk}, i-1, z_0, z_{i-1}, \Pi_{i-1}) = 1$$

with the exception of at most $\mathsf{negl}(\lambda)$. □

B Proof for Theorem 4

Proof. To prove knowledge soundness of NIFS, we construct extractor $\mathcal{E}$ which outputs valid witnesses from the adversary output. Before the proof, we remind the notation of instance and witness as following:

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

where m and ℓ is pre-designated input of G.

Let $(\mathcal{A}, \mathcal{P}^*)$ be a pair of algebraic adversaries, that take $\mathsf{pp}_{FS} = (\mathsf{ck}_{\boldsymbol{e}} \| \mathsf{ck}_{\boldsymbol{w}})$ outputted by G, against folding knowledge soundness.

Assume that $\mathcal{A}$ outputs structure s and two instance $\mathsf{u}_1, \mathsf{u}_2$ with representations $\tilde{\boldsymbol{e}}_1, \tilde{\boldsymbol{e}}_2, \tilde{\boldsymbol{w}}_1, \tilde{\boldsymbol{w}}_2$ for the group elements $\mathsf{u}_1.E, \mathsf{u}_2.E, \mathsf{u}_1.W, \mathsf{u}_2.W$ respectively. And $\mathcal{P}^*$ outputs updated witness v and folding proof T with a representation $\tilde{\boldsymbol{t}}$.

Let the tuple $(\mathsf{u}_1, \mathsf{u}_2, T, \mathsf{u}, \mathsf{v})$ be a valid input and output of $\mathsf{NIFS.V}$ circuit. That is, $\mathsf{NIFS.V}(\mathsf{pp}_{FS}, \mathsf{u}_1, \mathsf{u}_2, T) = \mathsf{u}$, $(\mathsf{u}, \mathsf{v}) \in \mathcal{R}_{\mathsf{pp}_{FS}, \mathsf{s}}$, and $\mathsf{u}_1, \mathsf{u}_2 \in \mathcal{L}(\mathcal{R}_{\mathsf{s}, \mathsf{pp}_{FS}})$. We construct an extractor $\mathcal{E}$ that outputs 4 representations $\tilde{\boldsymbol{e}}_1, \tilde{\boldsymbol{e}}_2, \tilde{\boldsymbol{w}}_1, \tilde{\boldsymbol{w}}_2$ outputted by $\mathcal{A}$.

Now we claim that $(\mathsf{u}_1, \tilde{\boldsymbol{e}}_1, \tilde{\boldsymbol{w}}_1), (\mathsf{u}_2, \tilde{\boldsymbol{e}}_2, \tilde{\boldsymbol{w}}_2) \in \mathcal{R}_{\mathsf{pp}_{FS}, \mathsf{s}}$. By the Eq. (5) and Eq. (6) in Figure 6, we know that the following relation holds.

$$E = E_1 + rT + r^2 E_2,\ s = s_1 + rs_2,\ W = W_1 + rW_2,\ \mathsf{x} = \mathsf{x}_1 + r\mathsf{x}_2 \quad (15)$$
$$E = \langle \boldsymbol{e} \parallel r_{\boldsymbol{e}}, \mathsf{ck}_{\boldsymbol{e}} \rangle,\ W = \langle \boldsymbol{w} \parallel r_{\boldsymbol{w}}, \mathsf{ck}_{\boldsymbol{w}} \rangle \text{ where } \mathsf{v} = (\boldsymbol{e}, r_{\boldsymbol{e}}, \boldsymbol{w}, r_{\boldsymbol{w}}) \quad (16)$$

By algebraic relation between outputted group elements and their representations, we get the following equations:

$$\begin{aligned} &E_1 = \langle \tilde{\boldsymbol{e}}_1, \mathsf{ck}_{\boldsymbol{e}} \| \mathsf{ck}_{\boldsymbol{w}} \rangle, W_1 = \langle \tilde{\boldsymbol{w}}_1, \mathsf{ck}_{\boldsymbol{e}} \| \mathsf{ck}_{\boldsymbol{w}} \rangle, \\ &E_2 = \langle \tilde{\boldsymbol{e}}_2, \mathsf{ck}_{\boldsymbol{e}} \| \mathsf{ck}_{\boldsymbol{w}} \rangle, W_2 = \langle \tilde{\boldsymbol{w}}_2, \mathsf{ck}_{\boldsymbol{e}} \| \mathsf{ck}_{\boldsymbol{w}} \rangle \\ &T = \langle \tilde{\boldsymbol{t}}, \mathsf{ck}_{\boldsymbol{e}} \| \mathsf{ck}_{\boldsymbol{w}} \rangle \end{aligned} \quad (17)$$

We denote $\mathsf{v}_1 := (\tilde{\boldsymbol{e}}_1, \tilde{\boldsymbol{w}}_1), \mathsf{v}_2 := (\tilde{\boldsymbol{e}}_2, \tilde{\boldsymbol{w}}_2)$. Now we claim that the extracted witnesses v_1 and v_2 are valid witness for the instances u_1 and u_2 respectively. Combining Eq. (15), Eq. (16) with Eq. (17). By DL assumption, we obtain the following linear relations.

$$\langle \boldsymbol{e} \parallel r_{\boldsymbol{e}}, \mathsf{ck}_{\boldsymbol{e}} \rangle \overset{(16)}{=} E \overset{(15)\&(17)}{=} \langle \tilde{\boldsymbol{e}}_1 + r\tilde{\boldsymbol{t}} + r^2\tilde{\boldsymbol{e}}_2, \mathsf{ck}_{\boldsymbol{e}} \| \mathsf{ck}_{\boldsymbol{w}} \rangle \overset{\mathrm{DL}}{=} \langle \tilde{\boldsymbol{e}}_1 + r\tilde{\boldsymbol{t}} + r^2\tilde{\boldsymbol{e}}_2, \mathsf{ck}_{\boldsymbol{e}} \rangle,$$
$$\langle \boldsymbol{w} \parallel r_{\boldsymbol{w}}, \mathsf{ck}_{\boldsymbol{w}} \rangle \overset{(16)}{=} W \overset{(15)\&(17)}{=} \langle \tilde{\boldsymbol{w}}_1 + r\tilde{\boldsymbol{w}}_2, \mathsf{ck}_{\boldsymbol{e}} \| \mathsf{ck}_{\boldsymbol{w}} \rangle \overset{\mathrm{DL}}{=} \langle \tilde{\boldsymbol{w}}_1 + r\tilde{\boldsymbol{w}}_2, \mathsf{ck}_{\boldsymbol{w}} \rangle$$

Let the representation vectors parse to two parts as follows:

$$\tilde{\boldsymbol{e}}_i = \bar{\boldsymbol{e}}_i \| \bar{r}_{\boldsymbol{e}_i}, \tilde{\boldsymbol{t}} = \bar{\boldsymbol{t}} \| \bar{r}_{\boldsymbol{t}} \in \mathbb{Z}_p^m \times \mathbb{Z}_p,\ \tilde{\boldsymbol{w}}_i = \bar{\boldsymbol{w}}_i \| \bar{r}_{\boldsymbol{w}_i} \in \mathbb{Z}_p^{m-\ell-1} \times \mathbb{Z}_p \tag{18}$$

Then, we can rewrite Eq. (17) as the commitment forms:

$$E_1 = \mathsf{Com}(\mathsf{ck}_{\boldsymbol{e}}, \bar{\boldsymbol{e}}_1; \bar{r}_{\boldsymbol{e}_1}),\ W_1 = \mathsf{Com}(\mathsf{ck}_{\boldsymbol{w}}, \bar{\boldsymbol{w}}_1; \bar{r}_{\boldsymbol{w}_1}),$$
$$E_2 = \mathsf{Com}(\mathsf{ck}_{\boldsymbol{e}}, \bar{\boldsymbol{e}}_2; \bar{r}_{\boldsymbol{e}_2}),\ W_2 = \mathsf{Com}(\mathsf{ck}_{\boldsymbol{w}}, \bar{\boldsymbol{w}}_2; \bar{r}_{\boldsymbol{w}_2}).$$

To complete the claim $(\mathsf{u}_1, \mathsf{v}_1), (\mathsf{u}_2, \mathsf{v}_2) \in \mathcal{R}_{\mathsf{pp},\mathsf{s}}$, we showed the opening-checks and the R1CS-like relation is remained. From the hypothesis $(\mathsf{u}, \mathsf{v}) \in \mathcal{R}_{\mathsf{pp},\mathsf{s}}$, we can derive the following equality.

$$\begin{aligned} 0 &= A\boldsymbol{z} \circ B\boldsymbol{z} - sC\boldsymbol{z} - \boldsymbol{e} \\ &= A(\bar{\boldsymbol{z}}_1 + r\bar{\boldsymbol{z}}_2) \circ B(\bar{\boldsymbol{z}}_1 + r\bar{\boldsymbol{z}}_2) - (s_1 + rs_2)C(\bar{\boldsymbol{z}}_1 + r\bar{\boldsymbol{z}}_2) - (\bar{\boldsymbol{e}}_1 + r\bar{\boldsymbol{t}} + r^2\bar{\boldsymbol{e}}_2) \\ &= A\bar{\boldsymbol{z}}_1 \circ B\bar{\boldsymbol{z}}_1 - s_1C\bar{\boldsymbol{z}}_1 - \bar{\boldsymbol{e}}_1 + r^2(A\bar{\boldsymbol{z}}_2 \circ B\bar{\boldsymbol{z}}_2 - s_2C\bar{\boldsymbol{z}}_2 - \bar{\boldsymbol{e}}_2) + r\delta(\bar{\boldsymbol{z}}_1, \bar{\boldsymbol{z}}_2, A, B) \end{aligned}$$

where $\boldsymbol{z} = (\boldsymbol{w}, \mathsf{x}, s), \bar{\boldsymbol{z}}_i = (\bar{\boldsymbol{w}}_i, \mathsf{x}_i, s_i)$ for $i \in \{1, 2\}$ and $\delta(\bar{\boldsymbol{z}}_1, \bar{\boldsymbol{z}}_2, A, B)$ is a redundant term consisting $\bar{\boldsymbol{z}}_1$, $\bar{\boldsymbol{z}}_2$, A, and B. We argue that the general zero-testing assumption over H guarantees that each coefficient of r^j-term should be zero without negligible probability; The last term of the above equation can be considered as a degree-2 polynomial in r whose coefficients are determined by $d := (\bar{\boldsymbol{z}}_1, \bar{\boldsymbol{e}}_1, \bar{\boldsymbol{z}}_2, \bar{\boldsymbol{e}}_2, \bar{\boldsymbol{t}})$ with A, B, C. We also know that r is the hash value of $\mathsf{u}_{i-1}, \mathsf{U}_{i-1}$ and T_{i-1}, which can be considered as commitments to d with binding property.

Therefore, we finally obtain the following equation:

$$A\bar{\boldsymbol{z}}_1 \circ B\bar{\boldsymbol{z}}_1 - s_1C\bar{\boldsymbol{z}}_1 - \bar{\boldsymbol{e}}_1 = 0 = A\bar{\boldsymbol{z}}_2 \circ B\bar{\boldsymbol{z}}_2 - s_2C\bar{\boldsymbol{z}}_2 - \bar{\boldsymbol{e}}_2$$

and we can conclude $(\mathsf{u}_1, \mathsf{v}_1), (\mathsf{u}_2, \mathsf{v}_2) \in \mathcal{R}_{\mathsf{pp},\mathsf{s}}$. □

C Proof for Theorem 5

Proof. To show poly-depth knowledge soundness (KS), for polynomially large step $n = \mathsf{poly}(\lambda)$, we construct an extractor $\mathcal{E}$ that derives local inputs $(\omega_i)_{i=0}^{n-1}$ from the valid outputs with representations provided by the extended algebraic adversary $\mathcal{P}^*$ with respect to IVC verifier $\mathsf{V}_n := \mathcal{V}$. The verification V_n takes $\mathsf{out}_n := (z_0, z_n, \mathit{\Pi}_n)$ and checks IVC proof $\mathit{\Pi}_n = ((\mathsf{U}_n, \mathsf{V}_n), (\mathsf{u}_n, \mathsf{v}_n))$ belongs to the relation $\mathcal{R}_{\mathsf{Nova}} = \mathcal{R}_{\mathsf{pp,s}} \times \mathcal{R}^*_{\mathsf{pp,s}}$, outer product of the CR-R1CS relation $\mathcal{R}_{\mathsf{pp,s}}$ and the R1CS relation $\mathcal{R}^*_{\mathsf{pp,s}}$. Since we consider (extended) algebraic adversary, $\mathcal{P}^*$ additionally outputs the corresponding representation rep_n of out_n. Let $\mathsf{out}_{n-1} := (\mathsf{out}_n, \mathsf{rep}_n)$.

From the definition of $\mathtt{V}_n$, we can see that a part of its input consists of group elements. That is, $\mathtt{V}_n$ checks the structure of Π_n; it first specifies group elements not only in the instances U_n and u_n but also in the witness v_n. Note that v_n is the witness of $\mathcal{R}^*_{\mathsf{pp,s}}$, the ordinary R1CS of F', where F' contains group operations defined $\mathsf{NIFS.V}$, so that $\mathtt{V}_n$ definitely checks group membership of group-converted elements in v_n. In addition, $\mathtt{V}_n$ also checks if Π_n belongs to $\mathcal{R}_{\mathsf{Nova}} = \mathcal{R}_{\mathsf{pp,s}} \times \mathcal{R}^*_{\mathsf{pp,s}}$, so that V_n and v_n have openings of Pedersen commitments.

We will design another verification algorithm $\mathtt{V}_{n-1}(\mathsf{pp}, \mathsf{out}_{n-1}) \to 0\backslash 1$ as follows. Before describing $\mathtt{V}_{n-1}$, let us first look into the shape of $\mathsf{out}_{n-1} = (\mathsf{out}_n, \mathsf{rep}_n)$, where $\mathsf{out}_n = ((\mathsf{U}_n, \mathsf{u}_n), (\mathsf{V}_n, \mathsf{v}_n))$. We argue that if $\mathtt{V}_n$ outputs $1(accept)$, then out_{n-1} includes $\Pi_{n-1} := ((\mathsf{U}_{n-1}, \mathsf{V}_{n-1}), (\mathsf{u}_{n-1}, \mathsf{v}_{n-1}))$, with overwhelming probability. As aforementioned, $\mathcal{R}^*_{\mathsf{pp,s}}$ includes the folding process $\mathsf{NIFS.V}$ and v_n is the witness of this relation if $\mathtt{V}_n \to 1$. Thus, v_n contains both the instances U_{n-1} and u_{n-1}, the inputs of the folding verifier. Since rep_n contains the representation of v_n, it also contains the representation of U_{n-1} and u_{n-1}, which are exactly equal to V_{n-1} and v_{n-1}. Therefore, $\Pi_{n-1} := ((\mathsf{U}_{n-1}, \mathsf{V}_{n-1}), (\mathsf{u}_{n-1}, \mathsf{v}_{n-1}))$ is included in out_{n-1}.

Now we are ready to describe $\mathtt{V}_{n-1}$. As we shown, Π_{n-1} is included in the input of out_{n-1}. $\mathtt{V}_{n-1}$ checks whether Π_{n-1} belongs to $\mathcal{R}_{\mathsf{Nova}} = \mathcal{R}_{\mathsf{pp,s}} \times \mathcal{R}^*_{\mathsf{pp,s}}$. Next, we argue that the following inequality holds.

$$\Pr[\mathtt{V}_{n-1}(\mathsf{pp}, \mathsf{out}_{n-1}) = 1 | \mathtt{V}_n(\mathsf{pp}, \mathsf{out}_n) = 1] \geq 1 - \mathsf{negl}(\lambda). \tag{19}$$

The input of $\mathtt{V}_n$ contains $(\mathsf{U}_n, \mathsf{V}_n)$ and the IVC verifier $\mathtt{V}_n$ checks the validity of $(\mathsf{U}_n, \mathsf{V}_n)$ for the relation $\mathcal{R}_{\mathsf{pp,s}}$. If $\mathtt{V}_n$'s output is 1, then the KS of NIFS guarantees $(\mathsf{U}_{n-1}, \mathsf{V}_{n-1})$ and $(\mathsf{u}_{n-1}, \mathsf{v}_{n-1})$ are valid for the same relation $\mathcal{R}_{\mathsf{pp,s}}$. Otherwise, using the extended algebraic IVC adversary, we can construct an adversary against the KS of NIFS that generates the valid folded witness V_n from the IVC adversary but neither V_{n-1} nor v_{n-1} is the valid witness. Therefore, we show that Π_{n-1} belongs to $\mathcal{R}_{\mathsf{pp,s}} \times \mathcal{R}_{\mathsf{pp,s}}$. Note that KS of NIFS holds by Theorem 4 due to DL assumption. Finally, it is remained to show that $(\mathsf{u}_{n-1}, \mathsf{v}_{n-1})$ is a valid pair of not only $\mathcal{R}_{\mathsf{pp,s}}$ but also $\mathcal{R}^*_{\mathsf{pp,s}}$. The IVC verifier V_n additionally checks $(\mathsf{u}_n, \mathsf{v}_n)$ for the relation $\mathcal{R}^*_{\mathsf{pp,s}}$, which is the R1CS of the function F' described in Fig. 2. We can see that u_{n-1} is the input of F' and its shape is checked in the second step in Fig. 2. Combining with the fact that u_{n-1} is an instance of $\mathcal{R}_{\mathsf{pp,s}}$, this process exactly guarantees that the relation is the ordinary R1CS $\mathcal{R}^*_{\mathsf{pp,s}}$. This proves that the equation 19 holds, where the negligible errors occurs due to the KS error of NIFS.

Similarly, we can repeat to extract Π_{n-2} and define $\mathtt{V}_{n-2}$. Let us review the process until now; we showed that

1. algebraic adversary should output rep_n.

2. if $\mathtt{V}_n(\mathsf{pp}, \mathsf{out}_n) \rightarrow 1(accept)$, then, $\mathtt{V}_{n-1}(\mathsf{pp}, \mathsf{out}_{n-1}) \rightarrow 1(accept)$, except negligible probability.

First, we argue that the algebraic adversary should output the representation of out_{n-1}, denoted by rep_{n-1}. From Eq. (19), we know that the requirement for extended algebraic adversary in Def. 7 satisfies.

Next, suppose that $\mathtt{V}_{n-1}(\mathsf{pp}, \mathsf{out}_{n-1}) \rightarrow 1(accept)$. Looking at the input out_{n-1}, we know that it contains $\Pi_{n-1} = (\mathsf{U}_{n-1}, \mathsf{V}_{n-1}, \mathsf{u}_{n-1}, \mathsf{v}_{n-1})$ and v_{n-1} has U_{n-2} and u_{n-2}. In fact, U_{n-2} and u_{n-2} should be interpreted as arbitrary strings unless $\mathtt{V}_{n-1}$ recognizes it as group elements. $\mathtt{V}_{n-1}$ checks if $(\mathsf{U}_{n-1}, \mathsf{V}_{n-1})$ and $(\mathsf{u}_{n-1}, \mathsf{v}_{n-1})$ belong to the (CR-)R1CS relations, which includes group membership test for group operations. Therefore, on our assumption that $\mathtt{V}_{n-1}(\mathsf{pp}, \mathsf{out}_{n-1}) \rightarrow 1(accept)$, U_{n-2} and u_{n-2} are recognized as group elements, so that the corresponding representations, V_{n-2} and v_{n-2}, can be found from rep_{n-1}. Therefore, $\mathsf{out}_{n-2} := (\mathsf{out}_{n-1}, \mathsf{rep}_{n-1})$ contains $\Pi_{n-2} := ((\mathsf{U}_{n-2}, \mathsf{V}_{n-2}), (\mathsf{u}_{n-2}, \mathsf{v}_{n-2}))$. Similar to $\mathtt{V}_{n-1}$, $\mathtt{V}_{n-2}$ is defined to check if Π_{n-2} belongs to $\mathcal{R}_{\mathsf{Nova}}$.

Since v_{n-1} is the witness of original R1CS relation $\mathcal{R}^*_{\mathsf{pp,s}}$ and NIFS has the knowledge soundness, $\mathtt{V}_{n-2}$ outputs 1, only except negligible error. In other words, we can represent this situation as the following inequality.

$$\Pr[\mathtt{V}_{n-2}(\mathsf{pp}, \mathsf{out}_{n-2}) = 1 | \mathtt{V}_{n-1}(\mathsf{pp}, \mathsf{out}_{n-1}) = 1] \geq 1 - \mathsf{negl}(\lambda). \tag{20}$$

However, this inequality does not guarantee that $\mathtt{V}_{n-2}$ is induced from $\mathtt{V}_n$ following the Definition 7. More concretely, to let algbraic adversary provide rep_2 by the verification $\mathtt{V}_{n-2}$ induced from $\mathtt{V}_n$, we need to show the following inequality:

$$\Pr[\mathtt{V}_{n-2}(\mathsf{pp}, \mathsf{out}_{n-2}) = 1 | \mathtt{V}_n(\mathsf{pp}, \mathsf{out}_n) = 1] \geq 1 - \mathsf{negl}(\lambda). \tag{21}$$

To show Eq. (21), we introduce a useful statement regarding a probability inequality.

Lemma 4. *Let λ be a security parameter and $n = \mathsf{poly}(\lambda)$ be a polynomially large integer. If $(E_i)_{i\in[n]}$ be events satisfying $\Pr[E_{i-1}|E_i] \geq 1 - \epsilon_{i-1}(\lambda)$ and $\Pr[E_n]$ is non-negligible in λ, where ϵ_{i-1} is a negligible function of λ, for $i = 2, \ldots, n$, then $\Pr[E_1|E_n] \geq 1 - \mathsf{negl}(\lambda)$.*

Before proving Lemma 4, we first introduce another useful lemma about probability inequality.

Lemma 5. *Let E_1, E_2, and E_3 be events satisfying $\Pr[E_{i-1}|E_i] \geq 1 - \epsilon_{i-1}$ for $i = 2, 3$, where ϵ_i's are positive. Then, $\Pr[E_1|E_3] \geq 1 - \frac{\epsilon_1+\epsilon_2}{\Pr[E_3]}$. Moreover, if ϵ_i's are negligible functions in a parameter λ and $\Pr[E_3]$ is non-negligible in λ, then we have $\Pr[E_1|E_3] \geq 1 - \epsilon$ for some negligible function ϵ in λ.*

Proof. We begin with showing the following inequalities:

$$\Pr[\neg E_{i-1} \wedge E_i] \leq \epsilon_{i-1} \tag{22}$$

$$\Pr[E_{i-1} \wedge E_i] \geq \Pr[E_i] - \epsilon_{i-1} \tag{23}$$

These inequalities can be derived as follows:

$$\Pr[E_{i-1} \wedge E_i] = \Pr[E_{i-1}|E_i] \cdot \Pr[E_i] \geq \Pr[E_i](1-\epsilon_{i-1}) \geq \Pr[E_i] - \epsilon_{i-1},$$

where the first inequality comes from the hypothesis. This result directly shows that Eq. (23) holds. Using the equality $\Pr[E_i] - \Pr[E_{i-1} \wedge E_i] = \Pr[\neg E_{i-1} \wedge E_i]$, we obtain Eq. (22).

Now we show the main statement $\Pr[E_1|E_3] \geq 1 - \frac{\epsilon_1+\epsilon_2}{\Pr[E_3]}$. It is sufficient to show that $\Pr[E_1 \wedge E_3] \geq \Pr[E_3] - \epsilon_1 - \epsilon_2$.

$$\begin{aligned}
\Pr[E_1 \wedge E_3] &= \Pr[E_1 \wedge E_3 \wedge E_2] + \Pr[E_1 \wedge E_3 \wedge \neg E_2] \text{ discards right factor } \Pr[E_1 \wedge E_3 \wedge \neg E_2] \\
&\geq \Pr[E_2 \wedge E_3] - \Pr[E_2 \wedge E_3 \wedge \neg E_1] \text{ Eq. (23)} \\
&\geq \Pr[E_3] - \epsilon_2 - \Pr[E_2 \wedge E_3 \wedge \neg E_1] \text{ discards event } E_3 \\
&\geq \Pr[E_3] - \epsilon_2 - \Pr[\neg E_1 \wedge E_2] \text{ Eq. (22)} \\
&\geq \Pr[E_3] - \epsilon_2 - \epsilon_1
\end{aligned}$$

Therefore, $\Pr[E_1|E_3] \geq 1 - \frac{\epsilon_1+\epsilon_2}{\Pr[E_3]}$ holds. This directly implies that if ϵ_i's are negligible in λ and $\Pr[E_3]$ is non-negligible in λ, then we have $\Pr[E_1|E_3] \geq 1 - \epsilon$ for some negligible function ϵ. □

By Lemma 5, $\Pr[E_{n-2}|E_n] \geq 1 - \mathsf{negl}(\lambda)$ holds if $\Pr[E_n]$ is non-negligible. By the mathematical induction, we can derive $\Pr[E_1|E_n] \geq 1 - \mathsf{negl}(\lambda)$ for any polynomially large integer n. More precisely, combining $\Pr[E_{n-2}|E_n] \geq 1 - \mathsf{negl}(\lambda)$ and $\Pr[E_{n-3}|E_{n-2}] \geq 1 - \mathsf{negl}(\lambda)$, applying Lemma 5 results $\Pr[E_{n-3}|E_n] \geq 1 - \mathsf{negl}(\lambda)$. We repeat this process polynomially many n-times, we finally have $\Pr[E_1|E_n] \geq 1 - \mathsf{negl}(\lambda)$. This completes the proof of Lemma 4 □

We come back for proving Eq. (21). Before applying Lemma 4, let us consider the scale of $\Pr[\mathtt{V}_n(\mathsf{pp}, \mathsf{out}_n) = 1]$. Assume that $\Pr[\mathtt{V}_n(\mathsf{pp}, \mathsf{out}_n) = 1]$ is negligible. It means that adversary's output out_n is accepted by IVC verifier $\mathcal{V}$ with negligible probability. In this case, constructing extractor is meaningless; for any strategy of extraction, the probability in Eq. (1) is negligible. For this reason, we assume that $\Pr[\mathtt{V}_n(\mathsf{pp}, \mathsf{out}_n) = 1]$ is not negligible.

With non-negligible $\Pr[\mathtt{V}_n(\mathsf{pp}, \mathsf{out}_n) = 1]$ and two inequalities, Eq. (19) and Eq. (20), we directly have the inequality Eq (21) holds by applying Lemma 4. Furthermore, Lemma 4 can be applied in general case for $\mathtt{V}_i$. Thus, recursively designed verifications $\mathtt{V}_i$ can be induced from the original verification $\mathtt{V}_n$.

We now define $(\mathtt{V}_i)_{i\in[n]}$ recursively as follows:

1. If $\mathtt{V}_{i+1}(\mathsf{pp}, \mathsf{out}_{i+1}) = 1$, then derive a pair of instances $(\mathsf{U}_i, \mathsf{u}_i)$ and a pair of witness $(\mathsf{V}_i, \mathsf{v}_i)$ from out_{i+1} and rep_{i+1}, respectively. Otherwise, output 0.
2. Set $\mathsf{out}_i := (\mathsf{out}_{i+1}, \mathsf{rep}_{i+1})$ and check $\Pi_i = ((\mathsf{U}_i, \mathsf{V}_i), (\mathsf{u}_i, \mathsf{v}_i)) \in \mathcal{R}_{\mathsf{Nova}} = \mathcal{R}_{\mathsf{pp,s}} \times \mathcal{R}^*_{\mathsf{pp,s}}$.

Upon reaching $\mathtt{V}_1$, the IVC proof Π_1 obtained from out_1 does not contain any meaningful instances. We no longer extend the verification $\mathtt{V}_0$. Specifically, the first instance u_1 is an artificial instance for the null-fold, where the input instances to $\mathsf{NIFS.V}$ are empty.

Now we claim that $\mathtt{V}_1$ is an induced verification from $\mathtt{V}_n$. By design of $\mathtt{V}_i$, we get

$$\Pr[\mathtt{V}_i(\mathsf{pp}, \mathsf{out}_i) = 1 | \mathtt{V}_{i+1}(\mathsf{pp}, \mathsf{out}_{i+1}) = 1] \geq 1 - \mathsf{negl}(\lambda).$$

for all $i = 1, \dots, n-1$. By applying Lemma 4, we obtain

$$\Pr[\mathtt{V}_1(\mathsf{pp}, \mathsf{out}_1) = 1 | \mathtt{V}_n(\mathsf{pp}, \mathsf{out}_n) = 1] \geq 1 - \mathsf{negl}(\lambda).$$

Therefore, $\mathtt{V}_1$ is the verification induced from $\mathtt{V}_n$, and the extended algebraic adversary $\mathcal{P}^*$ must output out_1. Note that all the above process about recursive design of $\mathtt{V}_i$'s are not about the actual behavior of the extractor, but just reasoning for what $\mathcal{A}$ should output along with the original output out_n.

Now we construct the extractor $\mathcal{E}$. The extended algebraic adversary $\mathcal{P}^*$'s overall output is out_1. Furthermore, out_1 contains all sequential out_i for $i = 2, \dots n$. Let us feed out_1 on $\mathcal{E}$ and define $\mathcal{E}$ as following:

$\underline{\mathcal{E}^{\mathcal{P}^*(\mathsf{pp},\rho)}(\mathsf{pp}, z_0, z; \rho)}$

1. Receive out_1 from $\mathcal{P}^*$
   For $i = 1, \dots, n-1$
   (a) Derive v_i from out_i.
   (b) Extract ω_{i-1} from v_i.
   (c) Parse out_i to $(\mathsf{out}_{i+1}, \mathsf{rep}_{i+1})$.
2. Extract ω_{n-1} from v_n in out_n.
3. Output $(\omega_i)_{i=0}^{n-1}$

$\mathcal{E}$ extracts local inputs $(\omega_i)_{i=0}^{n-1}$ by deriving ω_{i-1} from i-th R1CS witness v_i, which can be obtained by tracing out_i. The fact that v_i is a witness of R1CS relation $\mathcal{R}^*_{\mathsf{pp},\mathsf{s}}$ guarantees $F(z_{i-1}, \omega_{i-1}) = z_i$. Furthermore, since $\mathcal{E}$ performs at most $O(n)$ times searching, the running time of $\mathcal{E}$ is $\mathsf{poly}(\lambda)$. Therefore, we successfully construct extractor $\mathcal{E}$ from the extended algebraic adversary $\mathcal{P}^*$ with non-negligible success probability. □