zkDay Paris

ZK Conference and Open Hacker House

MAGGIE WU

Tech Lead

Foresight Ventures

DR. CATHIE SO

ZKML Researcher

PSE Team, EF

FORESIGHT VENTURES

TECH TALK

3:30 PM - 4:30 PM

TOPIC

Proof Recursion and Its Use in Practical ZKML

Presented By

MANTA NETWORK

COINTELEGRAPH
The future of money

Wednesday July 19th
11:00 AM - 08:00 PM

17 Rue de l'Aubrac,
75012 Paris, France

FORESIGHT VENTURES

THE FIRM

Foresight Ventures

A research-driven crypto investment & incubation powerhouse with premier media network.

foresightventures.com
@ForesightVen

Foresight New

The largest Multilingual Web3 Media in the Asia- Pacific.

foresightnews.pro
@Foresight_News

Foresight X

An accelerator that provides funding and dedicated support to fuel Web3 startups.

foresightx.net
@ForesightVen

OpenBuild

Bridging Web2 to Web3, Collaborating and Empowering Builders Worldwide.

build@openbuild.xyz
@OpenBuildxyz

FORESIGHT VENTURES

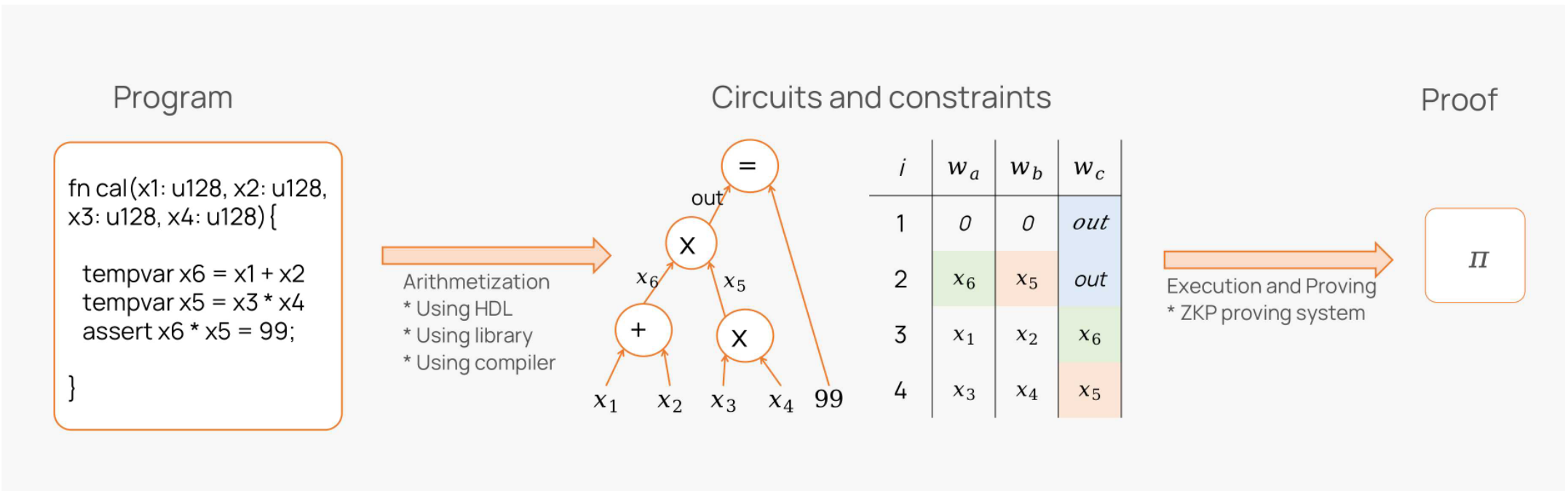
PROOF RECURSION

- Why do we care about proof recursion?
- An evolution of recursion and the folding scheme.
- More innovations based on folding schemes.

Motivation: Why Do We Care About Proof Recursion?

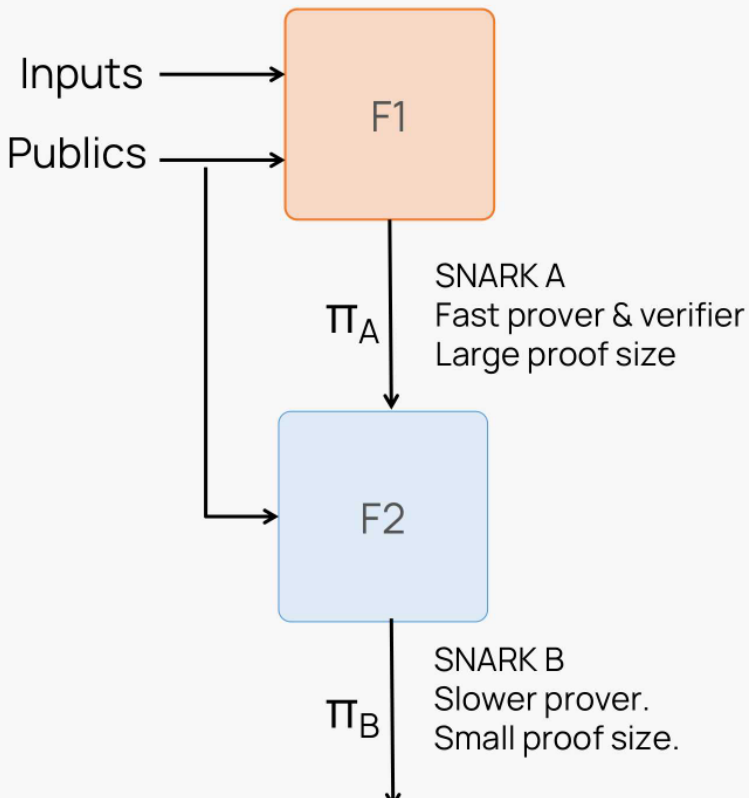
Challenges in Using Zero-Knowledge Proofs:

- Large and complex programs: Impractical to prove using a single monolithic zkp.
- Large proof sizes: Too big for blockchain verification.

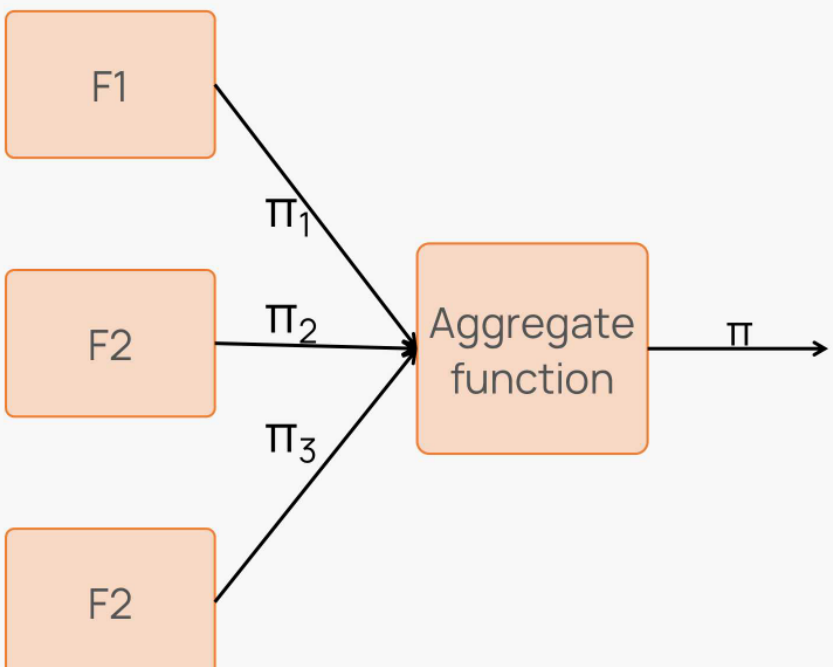


Motivation: Why Do We Care About Proof Recursion?

Proof Composition



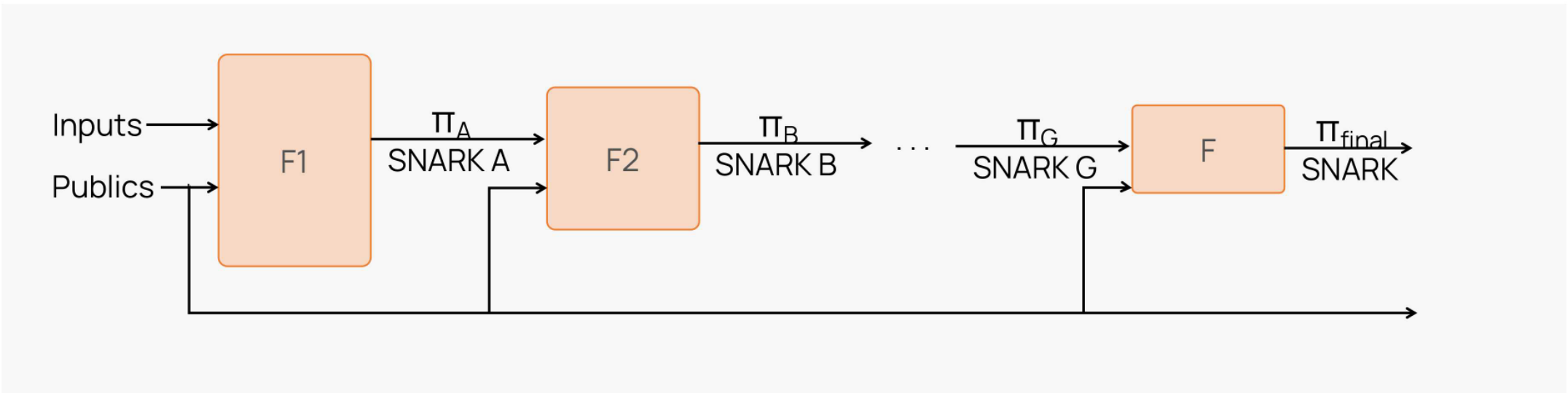
Proof Aggregation



Motivation: Why Do We Care About Proof Recursion?

Proof Recursion

At each step, the SNARK proof system will prove that the verifier of the former proof system would accept the witness of that step.

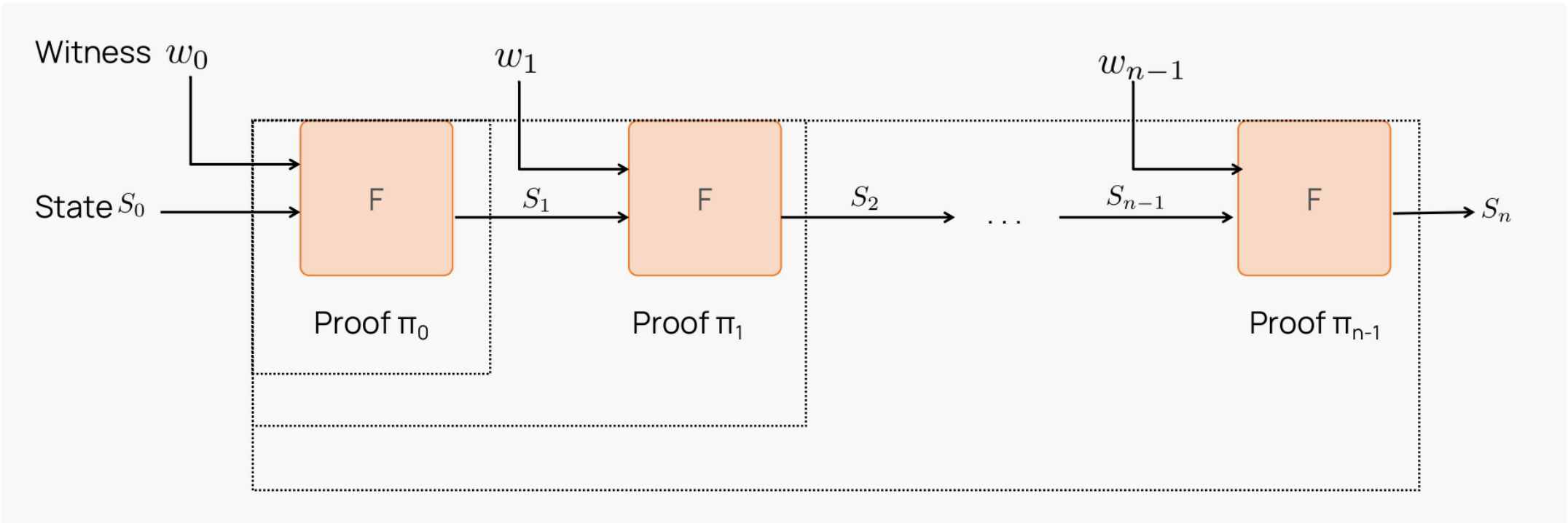


Motivation: Why Do We Care About Proof Recursion?

Incrementally Verifiable Computation (IVC)

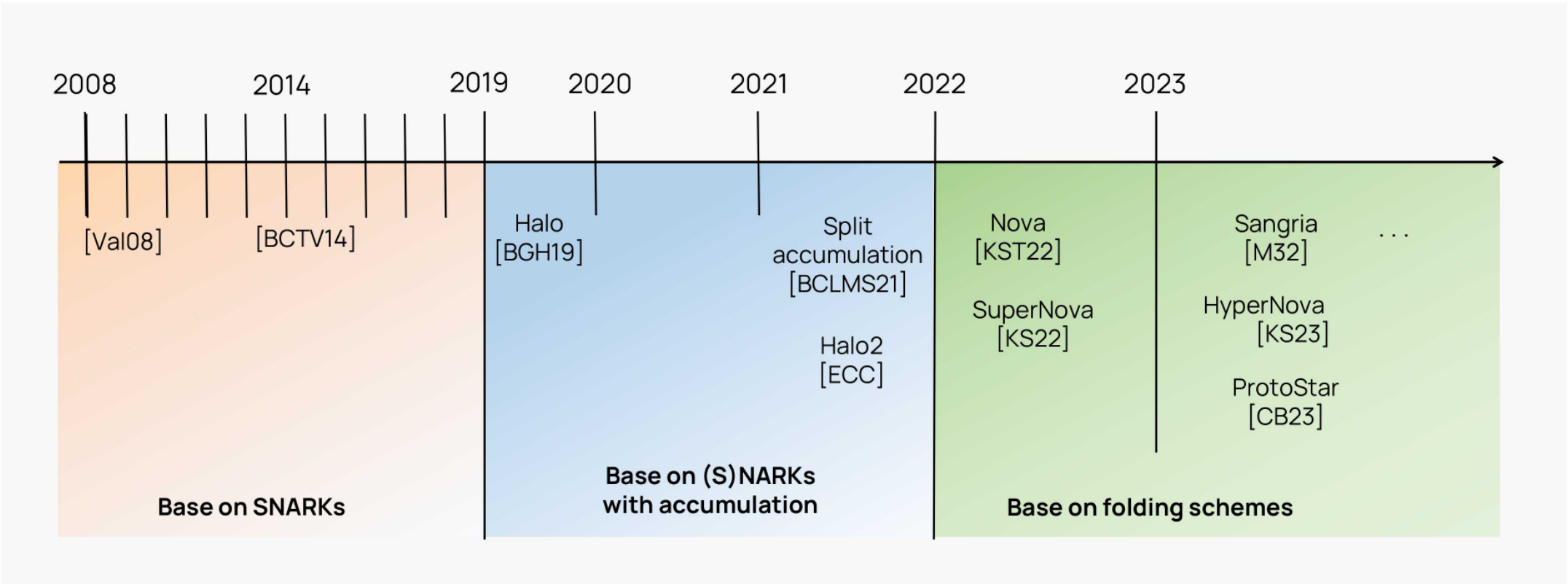
Enable long computation proving by break it into steps.

Verify state of chain by checking one recursive proof.



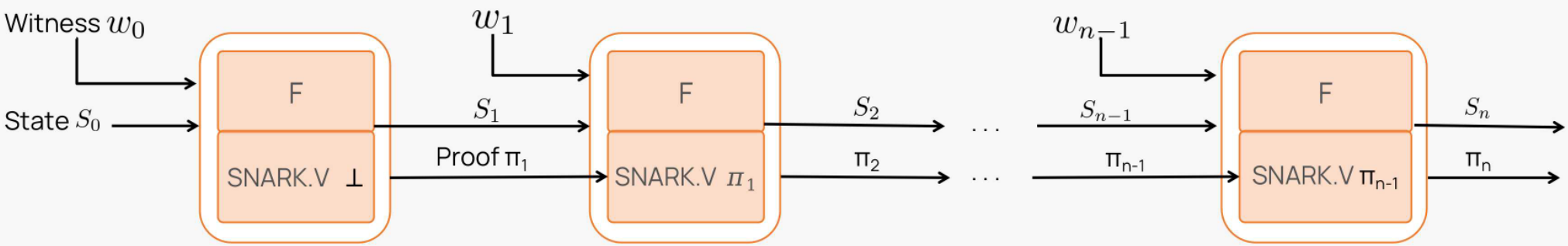
Used in succinct blockchains (e.g. Mina blockchain), zkML, VDF

An Evolution of IVC

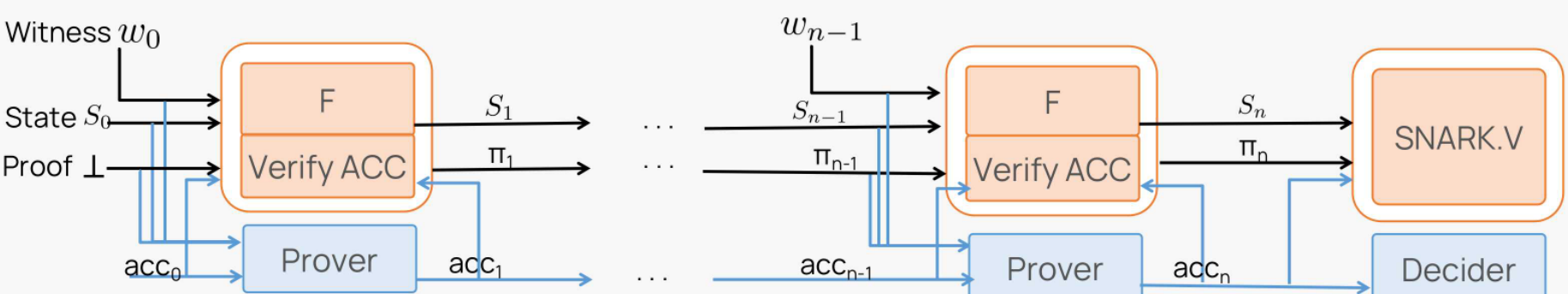


An Evolution of IVC

Base on SNARKs [Val08, BCTV14]

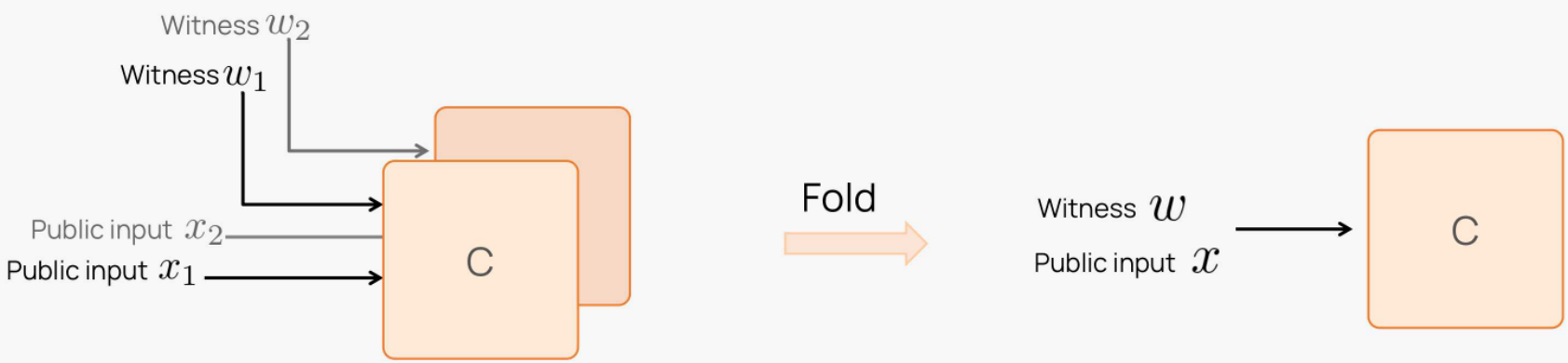


Base on (S)NARKs with accumulation [Halo, Halo2, ..]



A Folding Scheme: compress two instances into one

Let $C : \mathbb{F}_p^n \times \mathbb{F}_p^m \to \mathbb{F}_p$ be a circuit.



If $C(x_1, w_1) = C(x_2, w_2) = 0$ then $C(x, w) = 0$

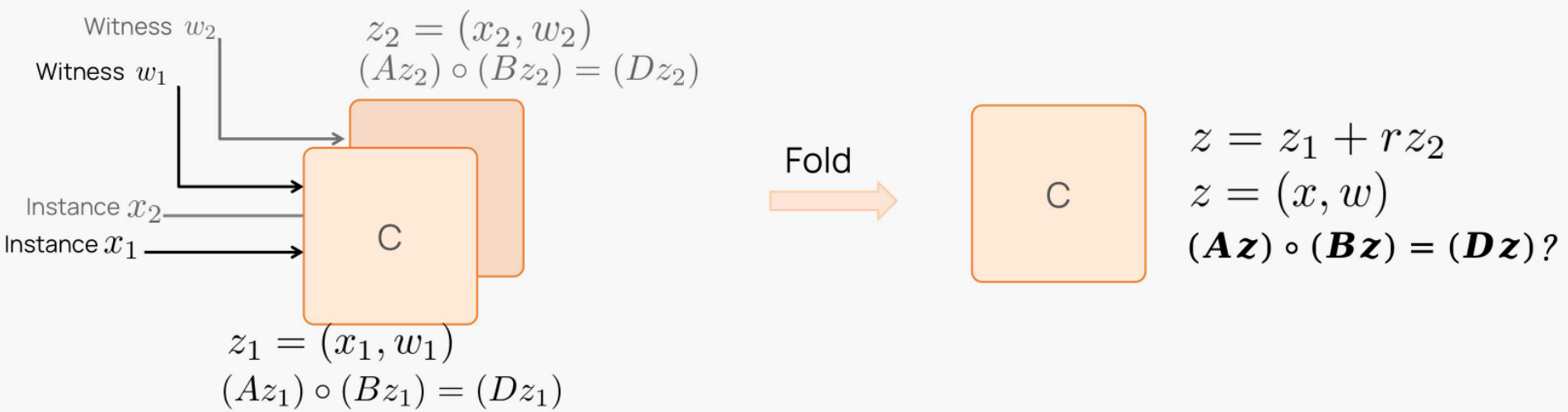
If $C(x, w) = 0$ is valid, the prover must know the valid $(x_1, w_1)(x_2, w_2)$

Nova - Folding Scheme

Attempt 1: Fold two R1CS instances into one.

R1CS Instance: $I = (x), A, B, D$

Witness: w , s.t. $(Az) \circ (Bz) = (Dz)$, $where\ z = (x, w)$

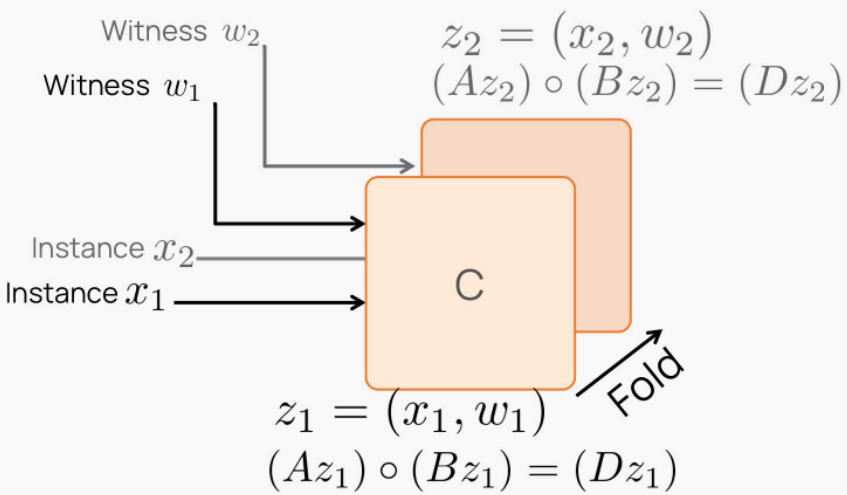


Nova - Folding Scheme

Attempt 1: Fold two R1CS instances into one.

R1CS Instance: $I = (x), A, B, D$

Witness: w , s.t. $(Az) \circ (Bz) = (Dz)$, $where\ z = (x, w)$



$z = z_1 + rz_2$, we want $(\boldsymbol{Az}) \circ (\boldsymbol{Bz}) = (\boldsymbol{Dz})$

$$
\begin{aligned}
&(Az) \circ (Bz) \\
&= A(z_1 + rz_2) \circ B(z_1 + rz_2) \\
&= Az_1 \circ Bz_1 + r^2(Az_2 \circ Bz_2) + \boxed{r(Az_1 \circ Bz_2 + Az_2 \circ Bz_1)} \\
&= Dz_1 + r^2 Dz_2 + E \\
&\neq Dz_1 + rDz_2 = Dz
\end{aligned}
$$

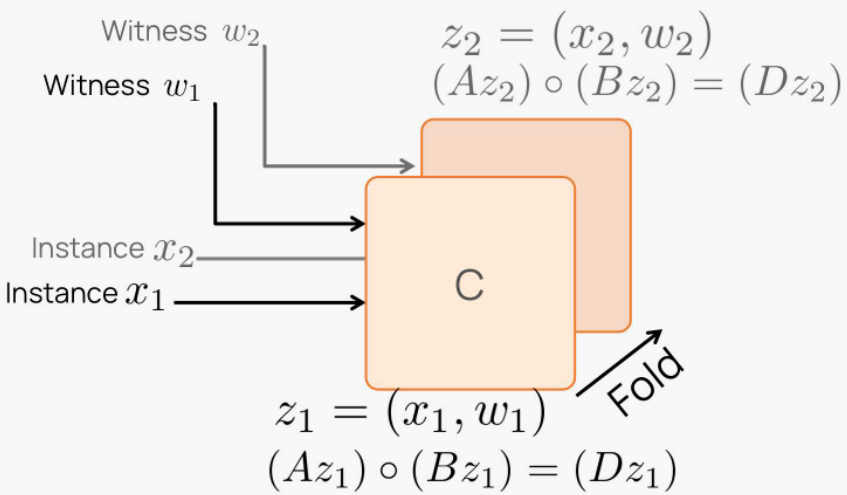
Issue: The folded instance is not valid!
$(\boldsymbol{Az}) \circ (\boldsymbol{Bz}) \neq (\boldsymbol{Dz})$

Nova - Folding Scheme

Attempt 1: Fold two R1CS instances into one.

R1CS Instance: $I = (x), A, B, D$

Witness: w , s.t. $(Az) \circ (Bz) = (Dz)$, $where\ z = (x, w)$



$z = z_1 + rz_2$, we want $(\boldsymbol{Az}) \circ (\boldsymbol{Bz}) = (\boldsymbol{Dz})$

$$(Az) \circ (Bz)$$
$$= A(z_1 + rz_2) \circ B(z_1 + rz_2)$$
$$= Az_1 \circ Bz_1 + r^2(Az_2 \circ Bz_2) + \boxed{r(Az_1 \circ Bz_2 + Az_2 \circ Bz_1)}$$
$$= Dz_1 + r^2 Dz_2 + E$$
$$\neq Dz_1 + rDz_2 = Dz$$

Let allow some noice.

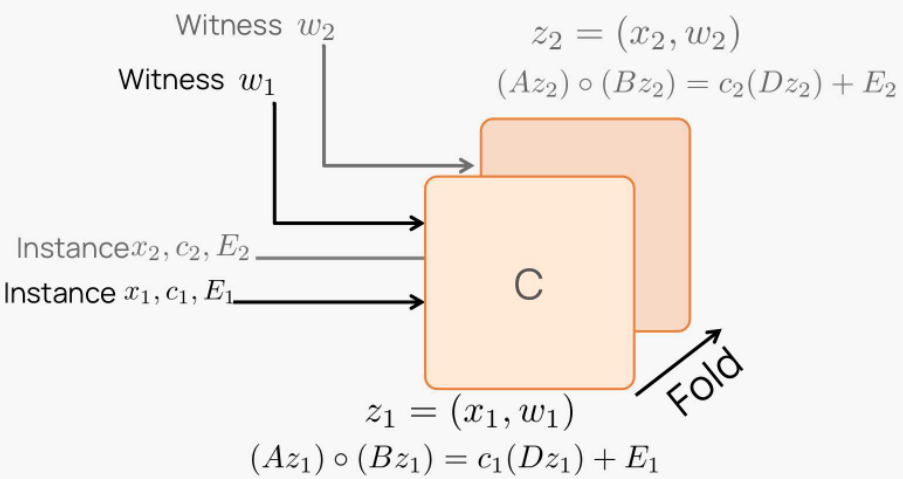
$$(Az) \circ (Bz) = c(Dz) + E$$

Nova - Folding Scheme

Attempt 2: Fold two relaxed R1CS instances into one

Relaxed R1CS Instance: $I = (x, c, E), A, B, D$

Witness: w , s.t. $(Az) \circ (Bz) = c(Dz) + E$, $where\ z = (x, w)$



$z = z_1 + rz_2$, we want $(\boldsymbol{Az}) \circ (\boldsymbol{Bz}) = \boldsymbol{c}(\boldsymbol{Dz}) + \boldsymbol{E}$

$$
\begin{aligned}
&(Az) \circ (Bz) \\
&= A(z_1 + rz_2) \circ B(z_1 + rz_2) \\
&= Az_1 \circ Bz_1 + r^2(Az_2 \circ Bz_2) + r(Az_1 \circ Bz_2 + Az_2 \circ Bz_1) \\
&= (c_1 Dz_1 + E_1) + r^2(c_1 Dz_2 + E_2) + r(Az_1 \circ Bz_2 + Az_2 \circ Bz_1) \\
&= (c_1 + rc_2) \circ D(z_1 + rz_2) + E \\
&= cDz + E \ \checkmark
\end{aligned}
$$

E is the error term, T includs all the cross terms

$E = E_1 + r^2 E_2 + rT$

$T = Az_1 \circ Bz_2 + Az_2 \circ Bz_1 - c_1 Dz_2 - c_2 Dz_1$

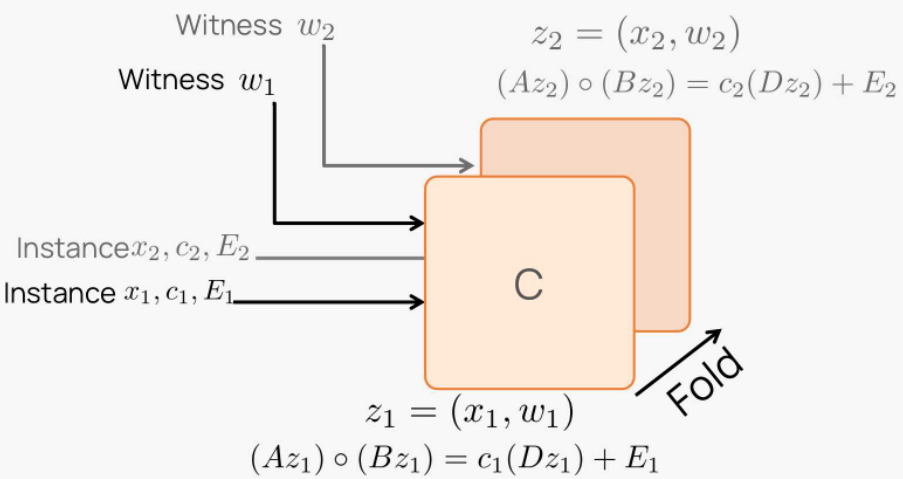
$c = c_1 + rc_2$

Nova - Folding Scheme

Attempt 2: Fold two relaxed R1CS instances into one

Relaxed R1CS Instance: $I = (x, c, E), A, B, D$

Witness: w , s.t. $(Az) \circ (Bz) = c(Dz) + E$, $where\ z = (x, w)$



$$z = z_1 + rz_2,\ \boldsymbol{(Az) \circ (Bz) = c(Dz) + E}$$

$$(Az) \circ (Bz)$$

$$= A(z_1 + rz_2) \circ B(z_1 + rz_2)$$

Issue: E extracts from witnesses. Not zk and not non-trivial!

$$= cDz + E \quad \surd$$

E is the error term, T includs all the cross terms

$$E = E_1 + r^2 E_2 + rT$$

$$T = Az_1 \circ Bz_2 + Az_2 \circ Bz_1 - c_1 Dz_2 - c_2 Dz_1$$

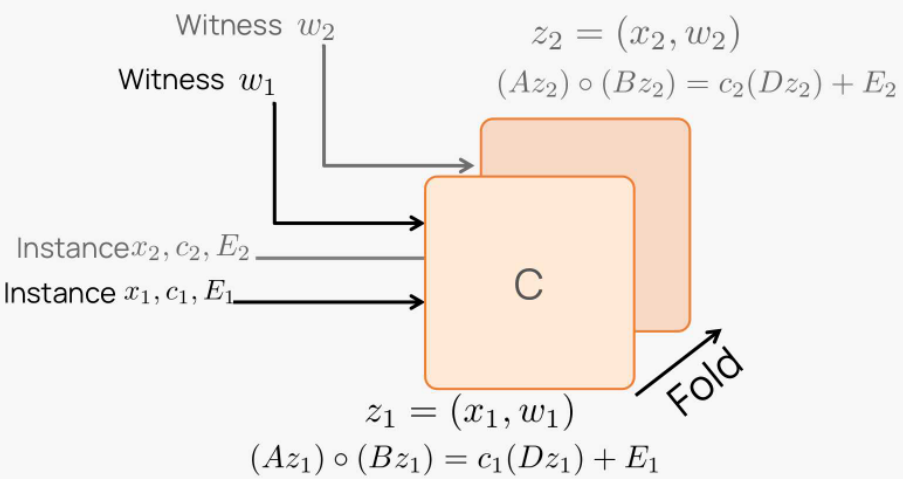
$$c = c_1 + rc_2$$

Nova - Folding Scheme

Attempt 2: Fold two relaxed R1CS instances into one

Relaxed R1CS Instance: $I = (x, c, E), A, B, D$

Witness: w , s.t. $(Az) \circ (Bz) = c(Dz) + E$, $where\ z = (x, w)$



$z = z_1 + rz_2$, $(\boldsymbol{Az}) \circ (\boldsymbol{Bz}) = \boldsymbol{c}(\boldsymbol{Dz}) + \boldsymbol{E}$

$(Az) \circ (Bz)$

$= A(z_1 + rz_2) \circ B(z_1 + rz_2)$

Let's use commitments of T and E when folding the instances.

$= cDz + E$ ✓

E is the error term, T includs all the cross terms

$E = E_1 + r^2 E_2 + rT$

$T = Az_1 \circ Bz_2 + Az_2 \circ Bz_1 - c_1 Dz_2 - c_2 Dz_1$

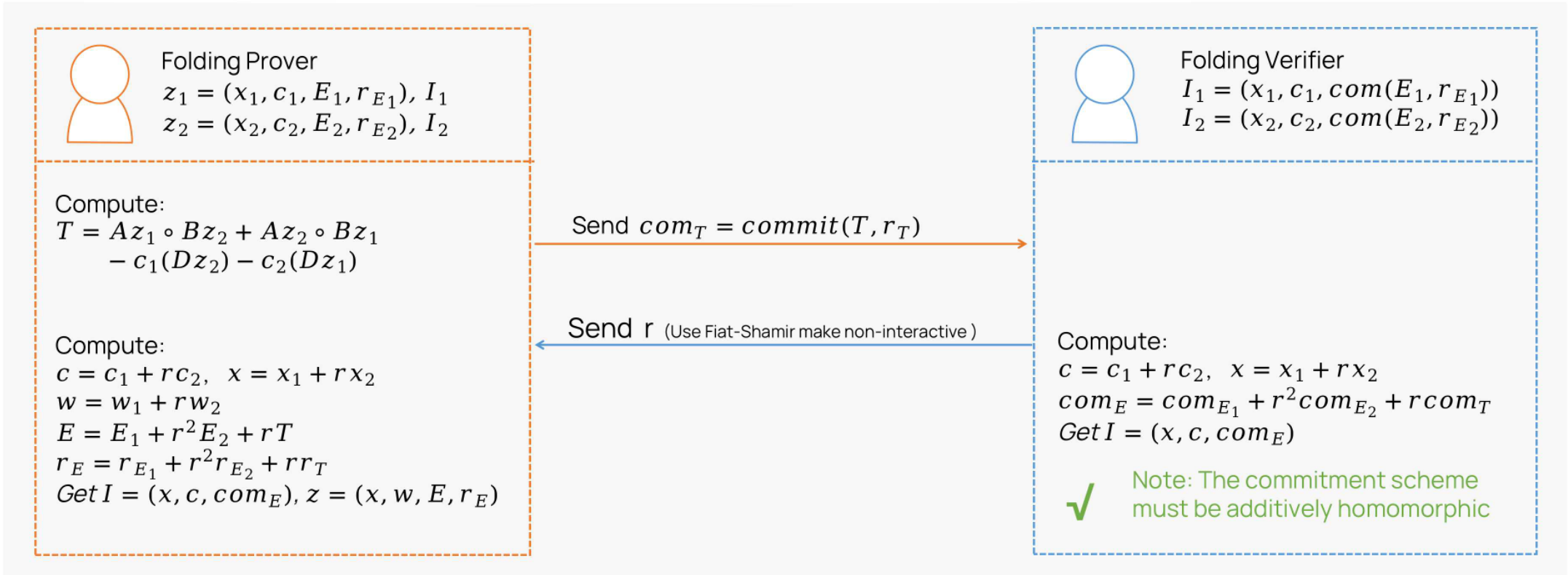
$c = c_1 + rc_2$

Nova - Folding Scheme

Final Proposal (simplified): Fold two committed relaxed R1CS instances into one.

Committed Relaxed R1CS Instance: $I = (x, c, com(E, r_E))$ (Simplified)

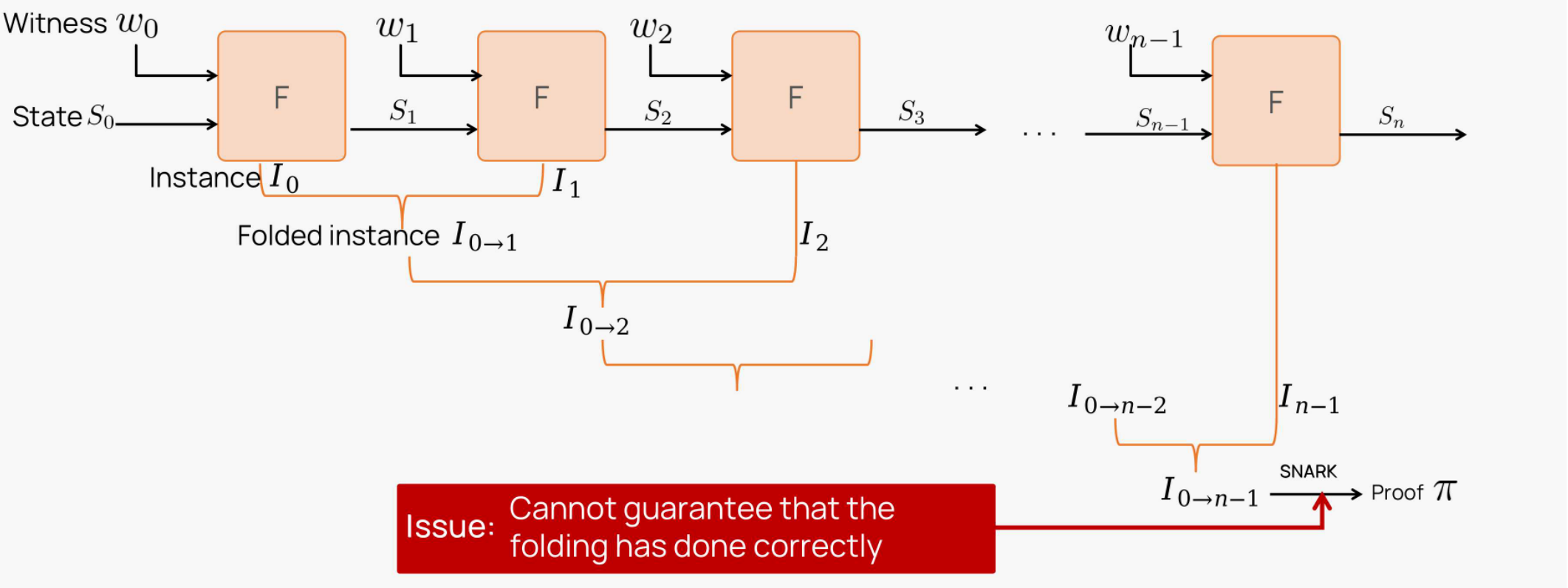
Witness: (w, E, r_E) s.t. $(Az) \circ (B) = c(Dz) + E$, $com_E = commit(E, r_E)$,where $z = (x, w, E, r_E)$



Nova - Folding-Based IVC

Represent steps as committed relaxed R1CS instance-witness pairs and fold them.

$F: S_{i+1} = F(S_i)$ is a iterate function.

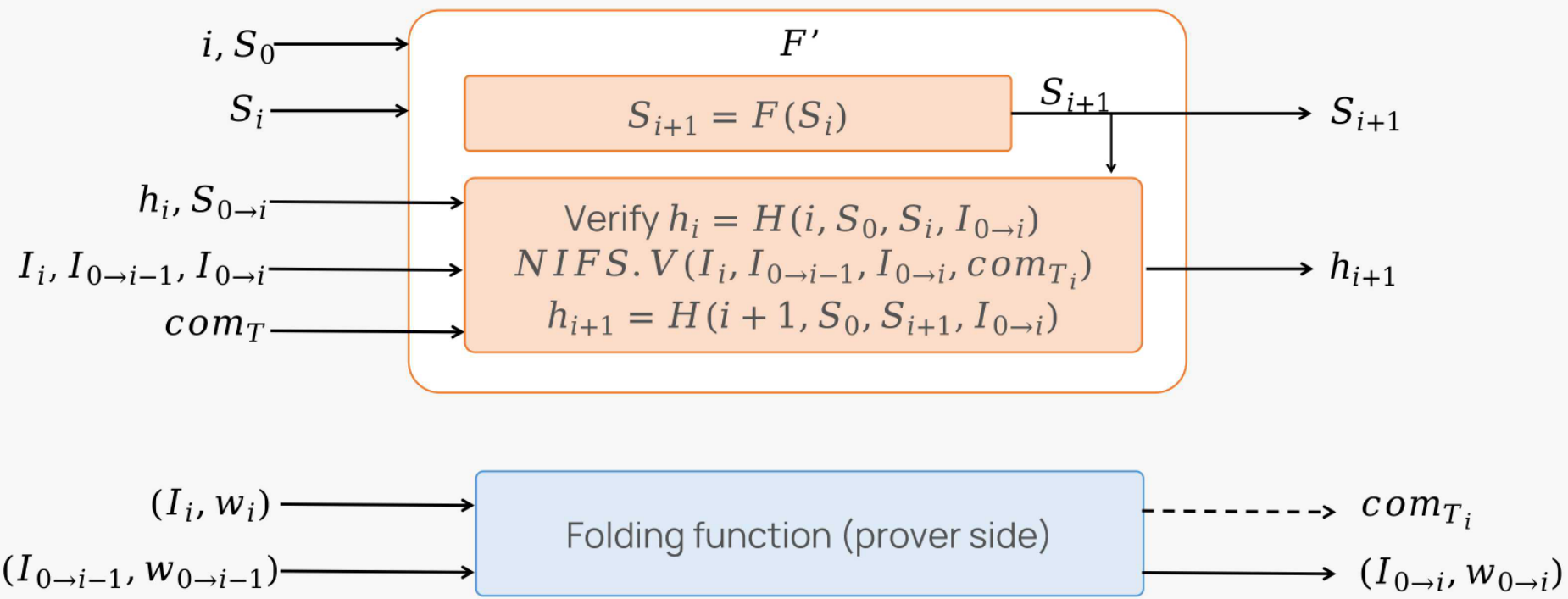


Nova - Folding-Based IVC

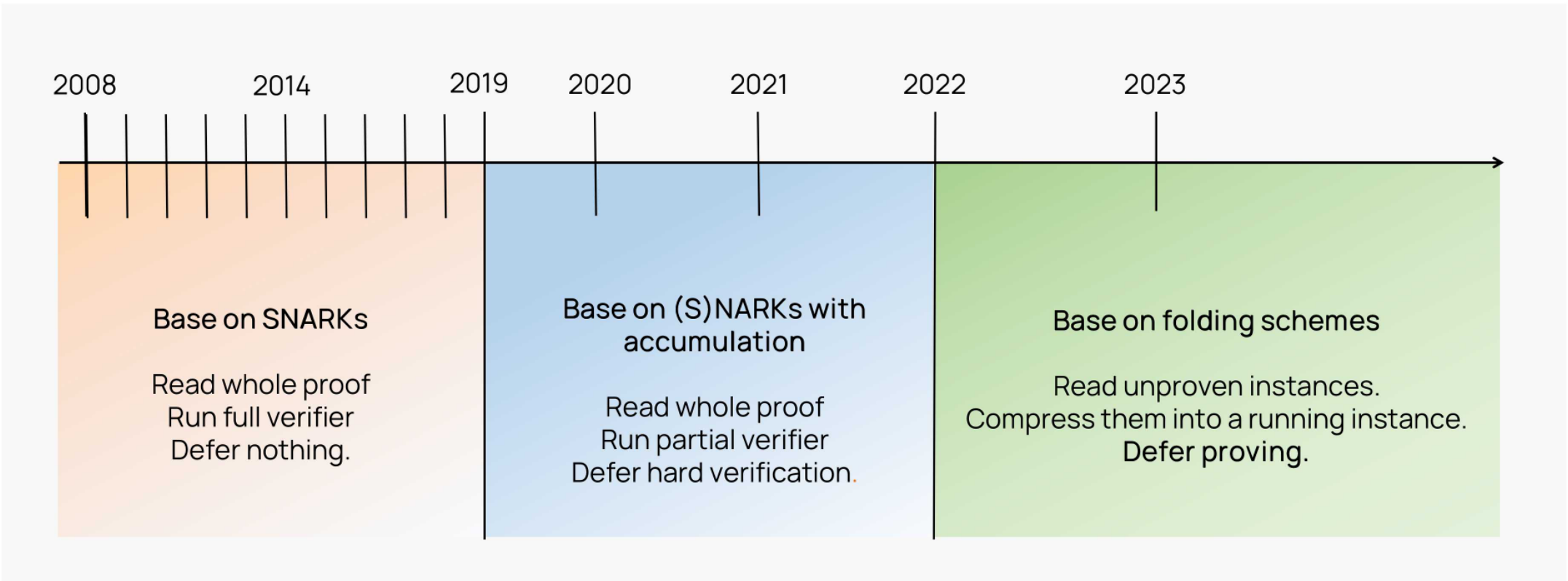
Verify folding in the folding-based IVC function F'.

$F: S_{i+1} = F(S_i)$ is a iterate function.

(I, w) is the committed relaxed R1CS instance-witness pair of function F.

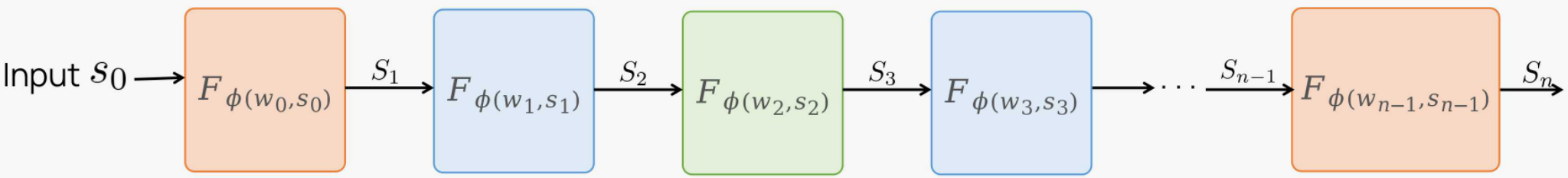


Folding-Based IVCs Incur the Lowest Recursion Overhead



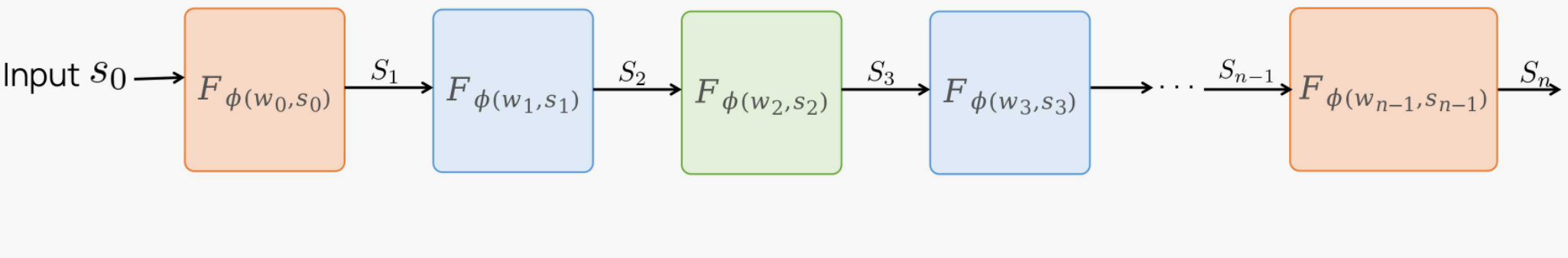
SuperNova: A Nova Generalization to Semi-structured Circuits

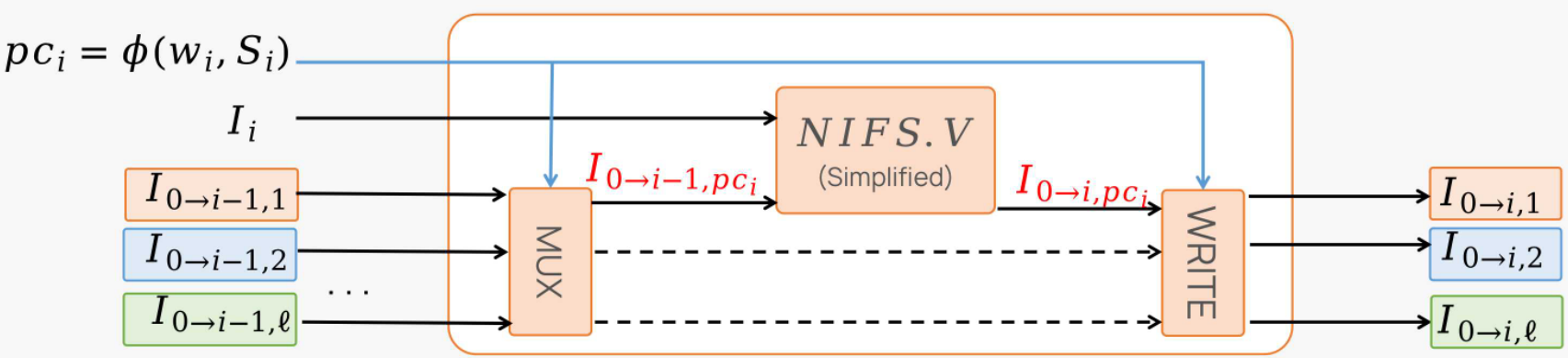
SuperNova supports multiple types of functions in the IVC chain (Non-uniform IVC).



SuperNova: A Nova Generalization to Semi-structured Circuits

SuperNova applies Nova to each set of *Fi* separately.





Sangria: A Folding Scheme for PLONK

Sangria expands the folding scheme to PLONKish arithmetic system

PLONKish arithmetic format

- Supports addition and multiplication constraints, copy constraints, and custom constraints including lookup constraints.
- Very flexible to meet different circuit needs.
- Lookup constraints enable zk-unfriendly functions to be precomputed for better performance.

Sangria supports addition and multiplication constraints, copy constraints, and degree 2 custom constraints.

Summary

Folding-based IVCs are very efficient in proving long computations.

	Introduction	Arithmetic circuit formats	Lookup Constraints	Non-uniform
Nova	A folding scheme for R1CS	R1CS	✗	✗
SuperNova	A Nova generalization to semi-structured circuits	R1CS	✗	✓
Sangria	A folding scheme for PLONK	PLONKish (Degree 2)	✗	✗
HyperNova	A multi-folding scheme for CCS	CCS (Degree d)	✓	✗
ProtoStar	A non-uniform IVC scheme for Plonk that supports high-degree gates and (vector) lookups.	PLONKish/CCS (Degree d)	✓	✓

FORESIGHT VENTURES

@ForesightVen

@Foresight_News

@0xmaggie5

\ CONNECT WITH US! \