



LatticeFold

Memory-Efficient Proving at Scale

Binyi Chen

Joint work with Dan Boneh

Stanford University



ZK-SNARKs: Advanced Applications

Applications

- zkVM/zkML
- Image/Video Provenance
- ZK Wallet/Passport
- Data Availability
- Decentralized Storage
-

Common Theme

Large-scale computation

VERIFIABLE COMPUTE LANDSCAPE

@dberenzon

| PRIVACY | STATE COMPRESSION | DATA INTEGRITY | COMPUTE COMPRESSION |
|--|---|---|---|
| Aleo Dark.fi FIRN PROTOCOL HINKAL IRON FISH MACI Mystiko Neptune nillion Oxbow PANTHER Personae PEKUMBA Polybase Labs PRIVASEA RAILGUN_ RENEGADE Vac zCloak Network zkBob | anoma Aztec Citrea Delphinus Lab DELTA Jolt Lighter Linea' MINA Mozak NEXUS =nil; Foundation ELA polygon Miden Scroll STARKWARE taiko VALIDA ZeroSync zkSync | Accountable blocksense Filecoin Holonym Jiri Maya Opacity Orochi reclaim SPACEANDTIME™ Terminal 3 WORLD COIN ZK EMAIL ZKON ZKP2P ZKPASS ZK Passport | AXIOM BREVIS lagrange Polyhedra RISC ZERO Succinct Sygma Union |
| PROOF GENERATION AND AGGREGATION | | | |
| ALIGNED LAYER | π² | | |
| Electron | Prover Network | | |
| GEVULOT | taralli labs | | |
| HORIZEN™ | Zero Computing | | |
| HYLE | ZKPOOL | | |
| NEBRA | | | |
| MACHINE LEARNING | | | |
| Aizel Network | GIZA | | |
| exo | HUNGRY CATS STUDIO | | |
| EZKL | Modulus | | |
| gensyn | Shinkai | | |
| | Vanna Labs | | |

ZK-SNARKs: Advanced Applications

Applications

- zkVM/zkML
- Image/Video Provenance
- ZK Wallet/Passport
- Data Availability
- Decentralized Storage
-

Common Theme

Large-scale computation

Design Requirements

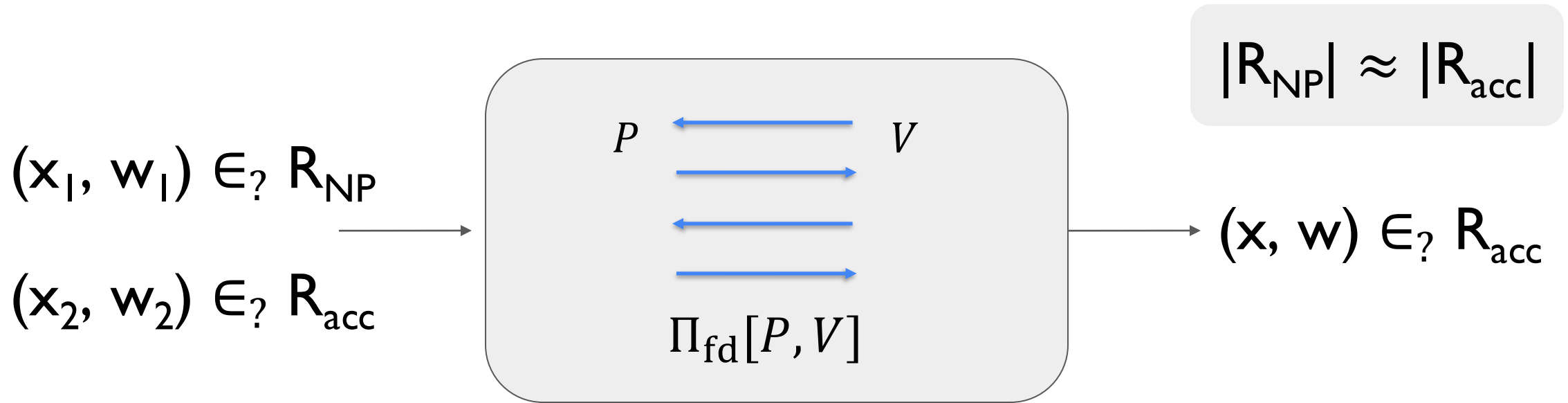
Scalable prover + post-quantum security

VERIFIABLE COMPUTE LANDSCAPE

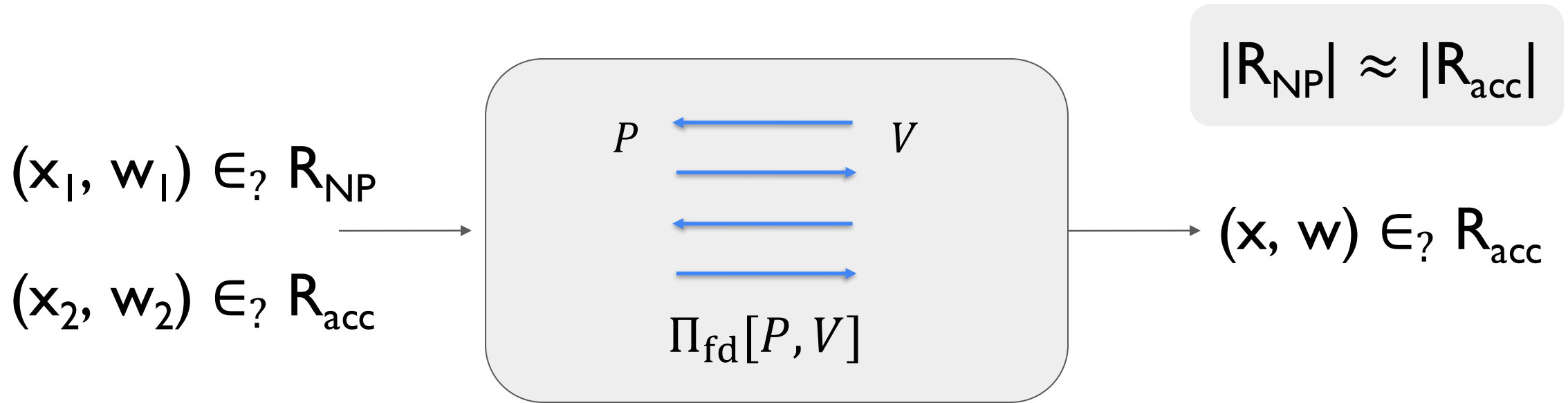
@dberenzon

| PRIVACY | STATE COMPRESSION | DATA INTEGRITY | COMPUTE COMPRESSION |
|--|---|---|---|
| Aleo Dark.fi FIRN PROTOCOL HINKAL IRON FISH MACI Mystiko Neptune nillion Oxbow PANTHER Personae PEKUMBA Polybase Labs PRIVASEA RAILGUN_ RENEGADE Vac zCloak Network zkBob | anoma Aztec Citrea Delphinus Lab DELTA Jolt Lighter Linea' MINA Mozak NEXUS =nil; Foundation ELA polygon Miden Scroll STARKWARE taiko VALIDA ZeroSync zkSync | Accountable blocksense Filecoin Holonym Jiri Maya Opacity Orochi reclaim SPACEANDTIME™ Terminal 3 WORLD COIN ZK EMAIL ZKON ZKP2P ZKPASS ZK Passport | AXIOM BREVIS lagrange Polyhedra RISC ZERO Succinct Sygma Union |
| PROOF GENERATION AND AGGREGATION | | | |
| ALIGNED LAYER | π² | | |
| Electron | Prover Network | | |
| GEVULOT | taralli labs | | |
| HORIZEN™ | Zero Computing | | |
| HYLE | ZKPOOL | | |
| NEBRA | | | |
| MACHINE LEARNING | | | |
| Aizel Network | GIZA | | |
| exo | HUNGRY CATS STUDIO | | |
| EZKL | Modulus | | |
| gensyn | Shinkai | | |
| | Vanna Labs | | |

Folding Schemes [BGH'19, BCLMS'20, KST'21]

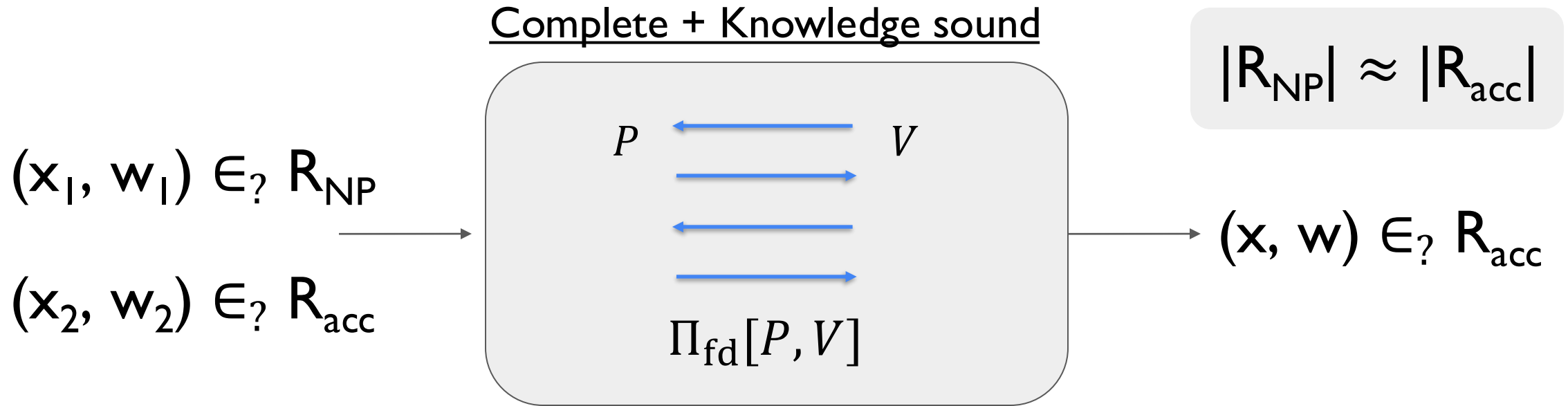


Folding Schemes [BGH'19, BCLMS'20, KST'21]



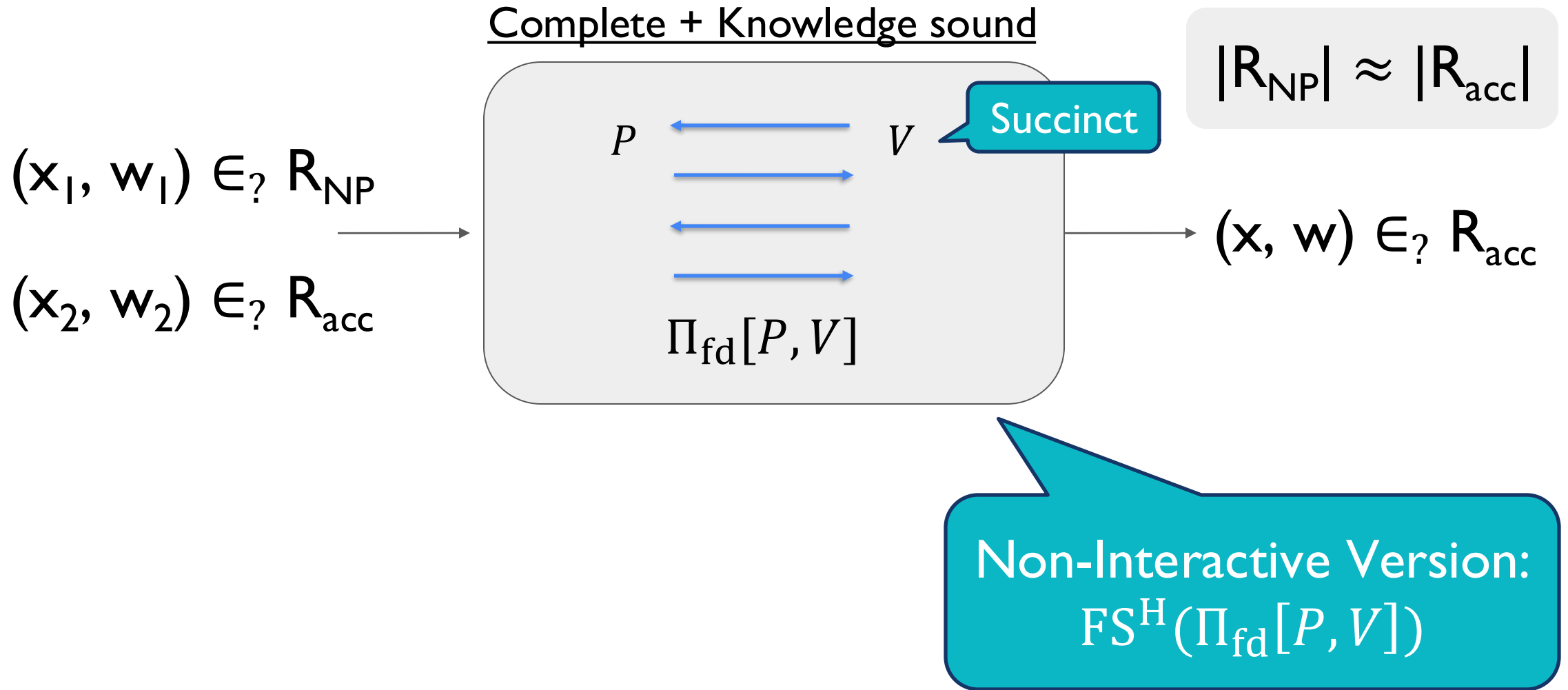
Non-Interactive Version:
 $FS^H(\Pi_{fd}[P, V])$

Folding Schemes [BGH'19, BCLMS'20, KST'21]

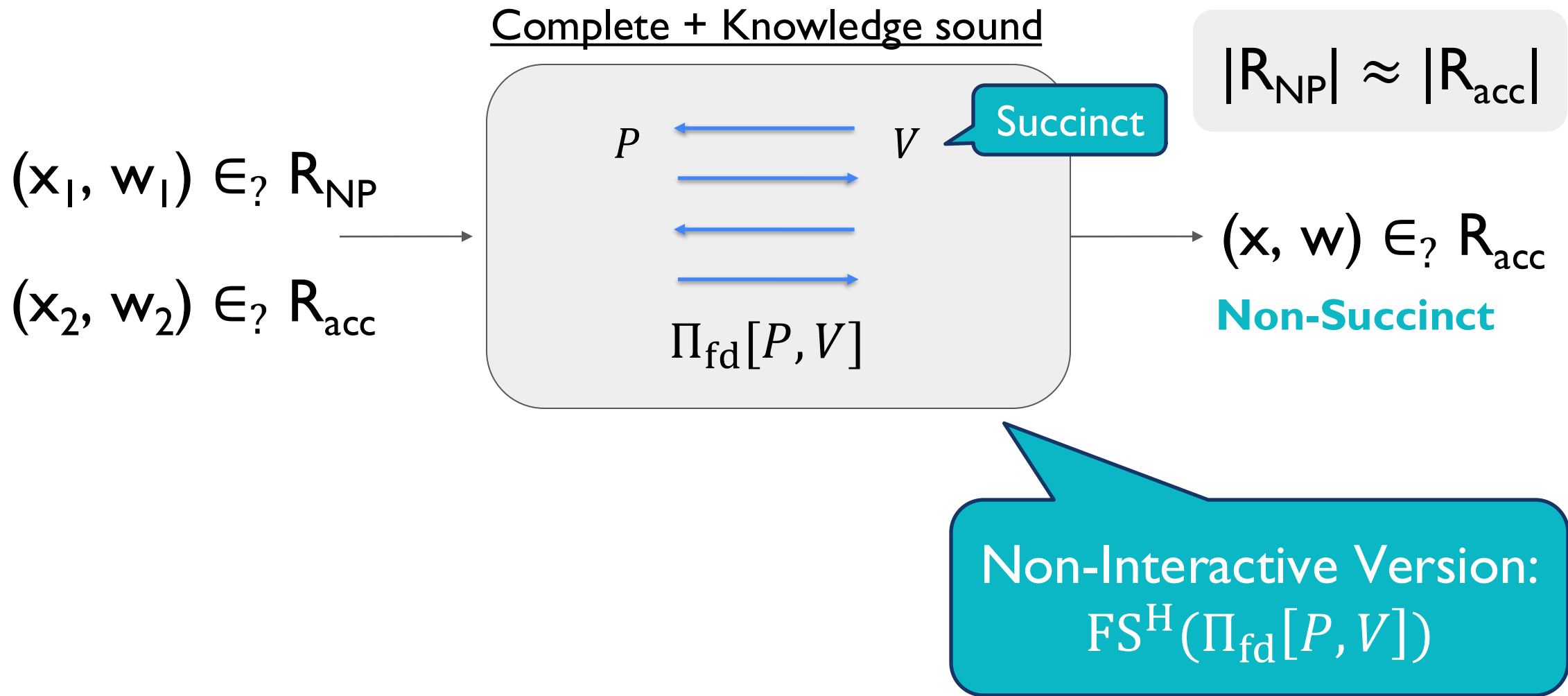


Non-Interactive Version:
 $FS^H(\Pi_{fd}[P, V])$

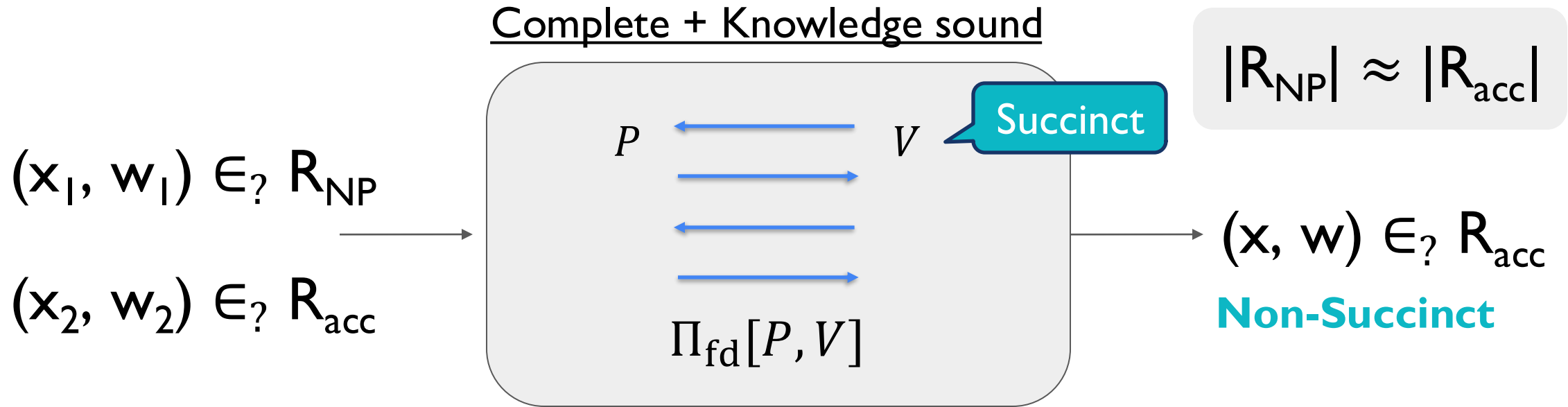
Folding Schemes [BGH'19, BCLMS'20, KST'21]



Folding Schemes [BGH'19, BCLMS'20, KST'21]



Folding Schemes [BGH'19, BCLMS'20, KST'21]



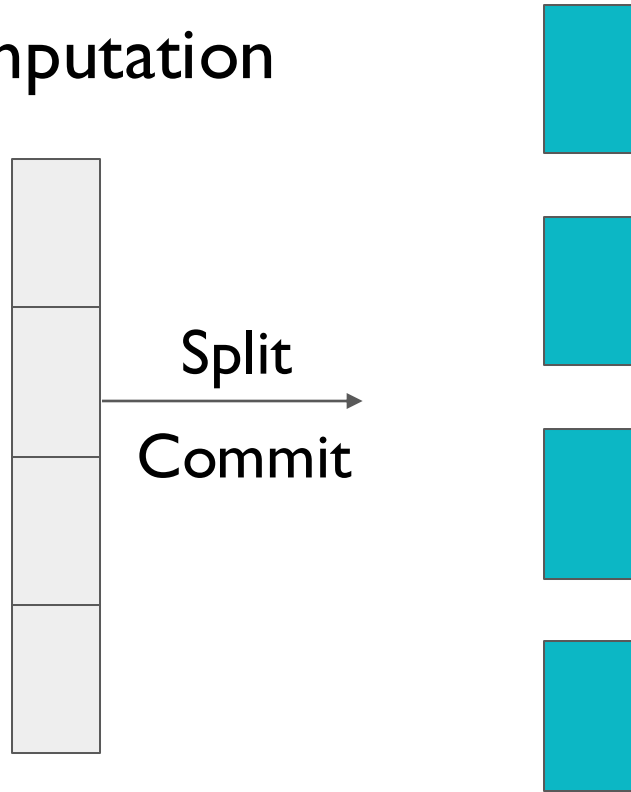
Advantages:

- Much faster than SNARKs
- Boost SNARK efficiency

Non-Interactive Version:
 $FS^H(\Pi_{fd}[P, V])$

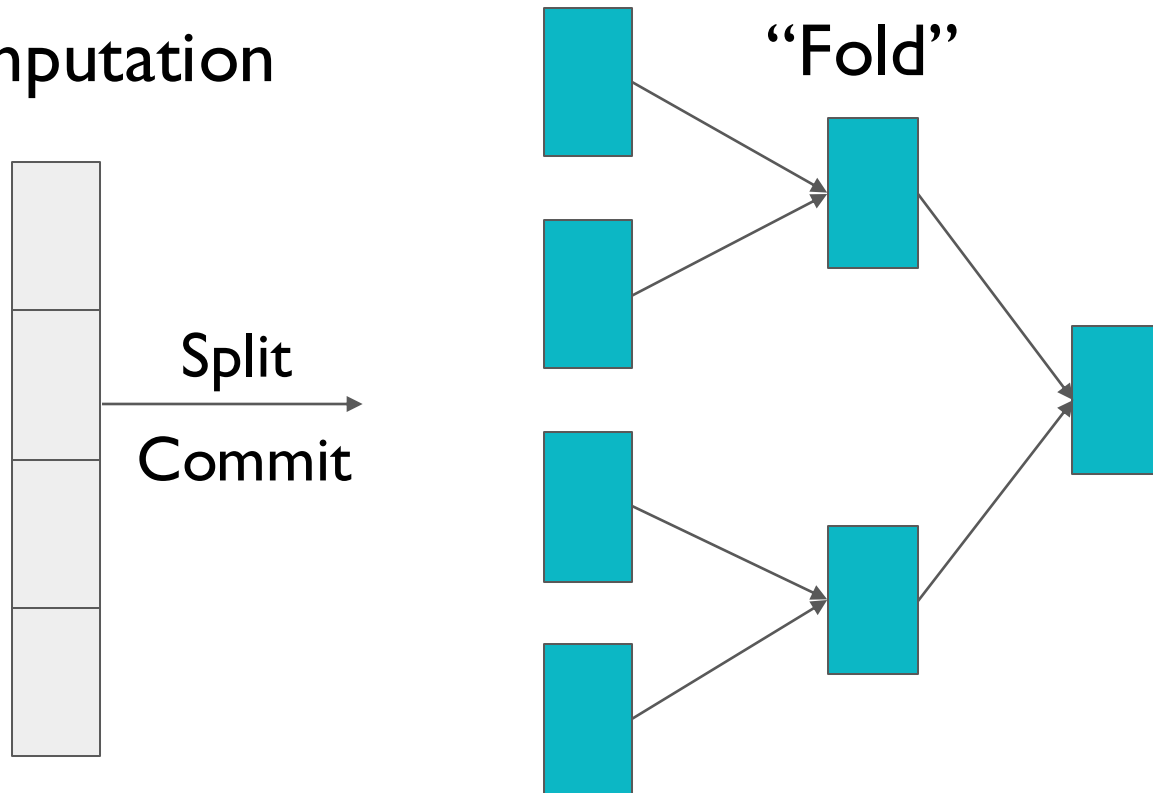
Folding-based SNARKs [BGH'19, COS'20, NDCTB'24...]

Computation

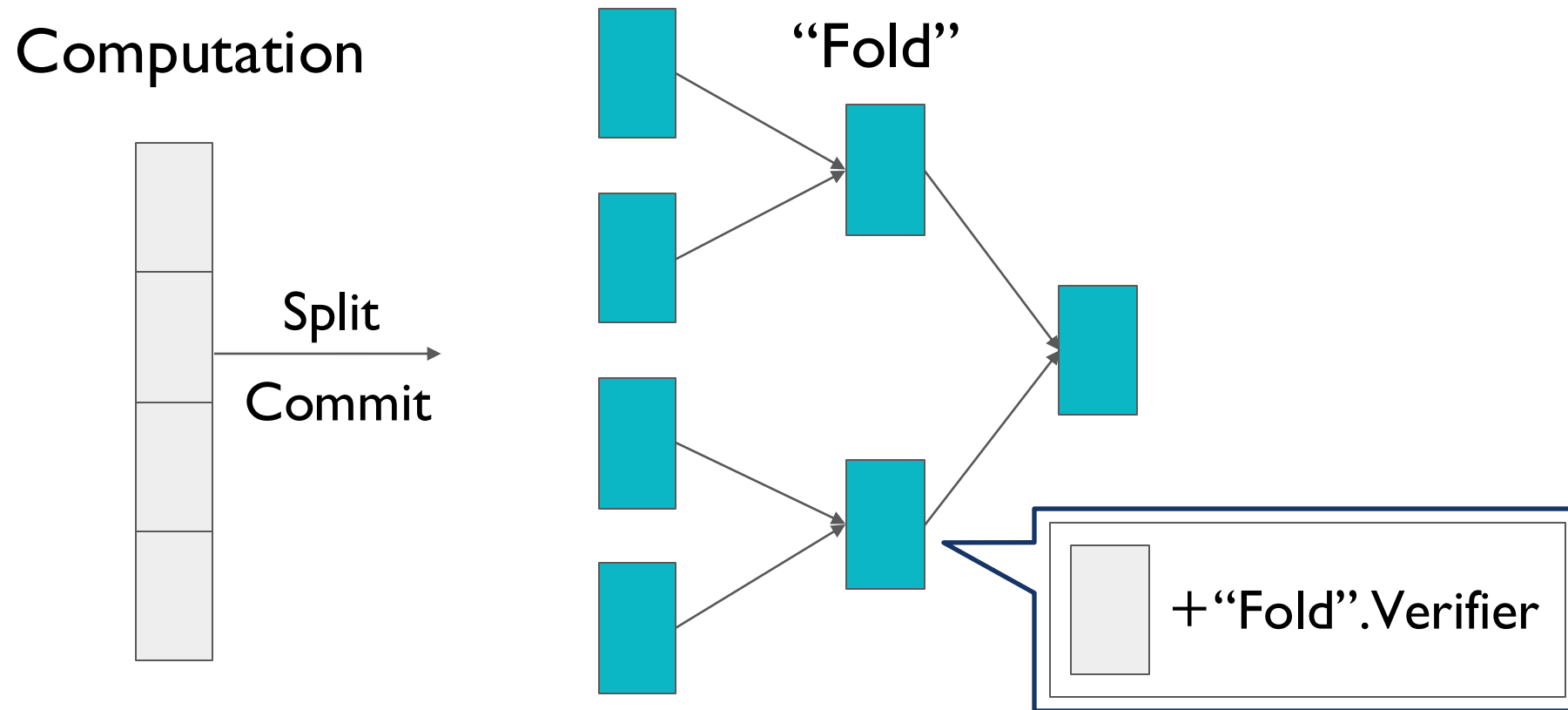


Folding-based SNARKs [BGH'19, COS'20, NDCTB'24...]

Computation

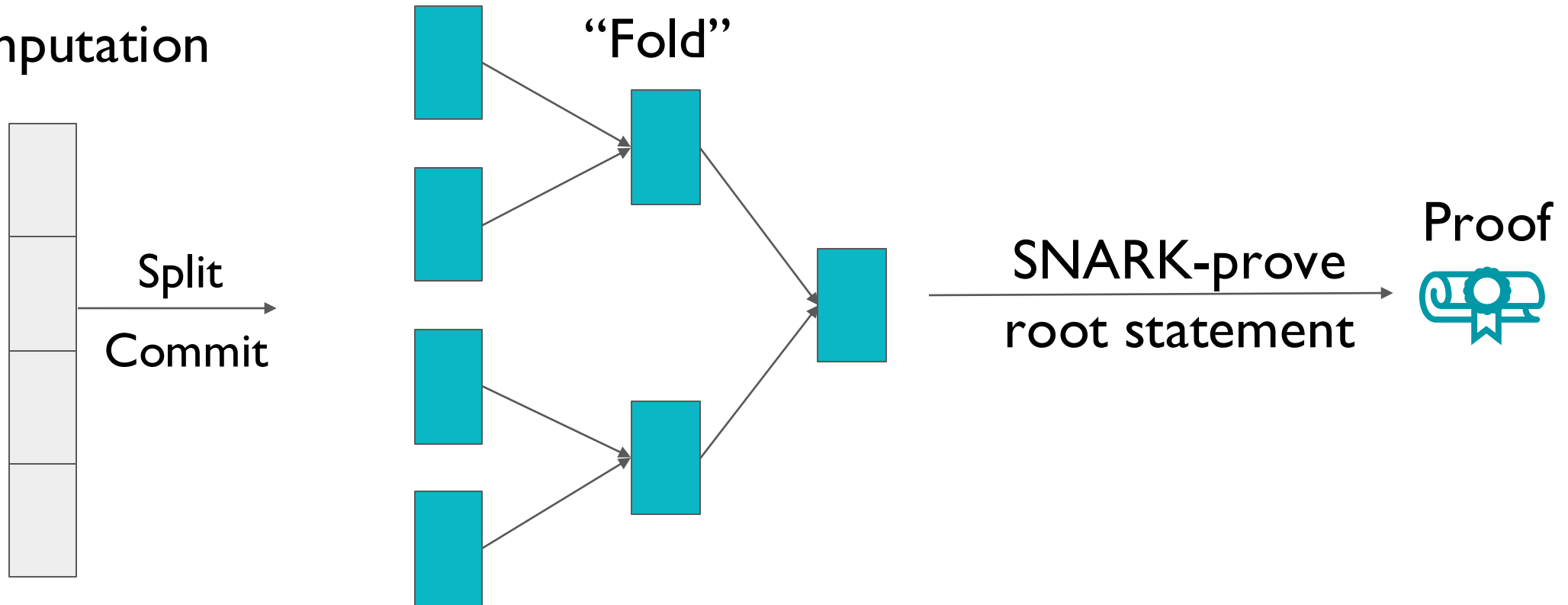


Folding-based SNARKs [BGH'19, COS'20, NDCTB'24...]



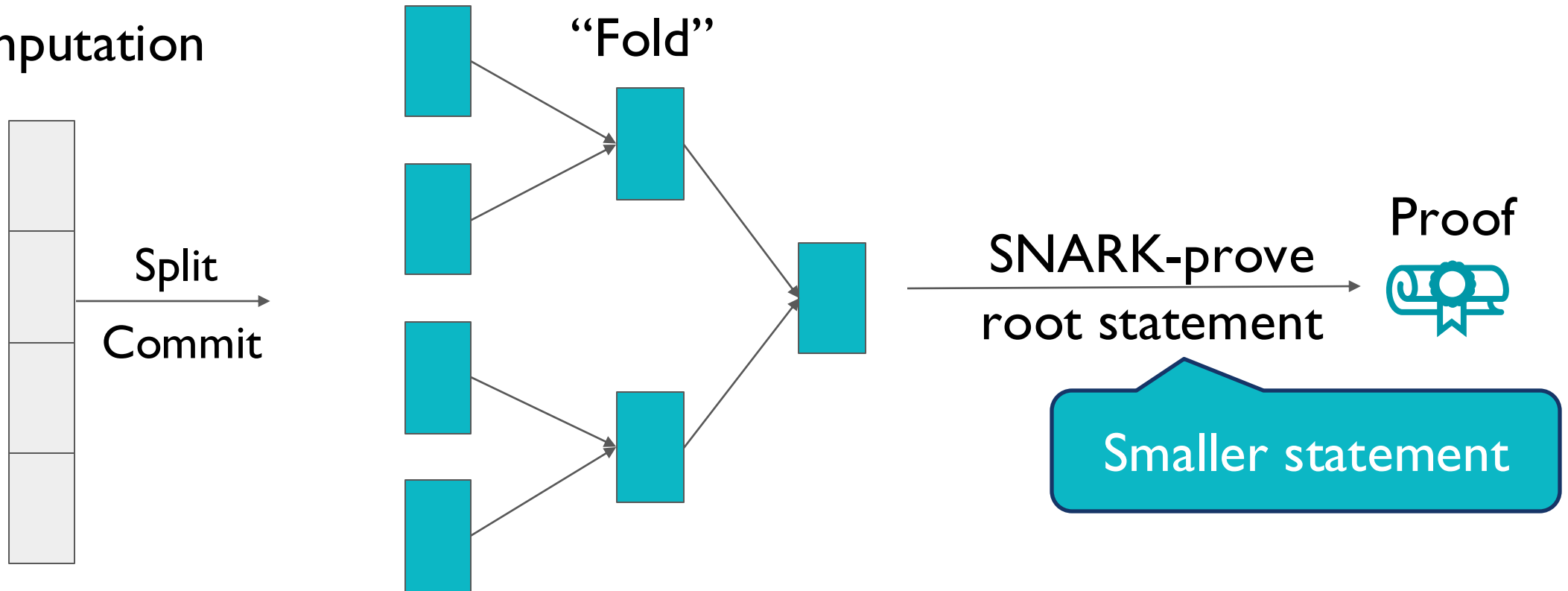
Folding-based SNARKs [BGH'19, COS'20, NDCTB'24...]

Computation



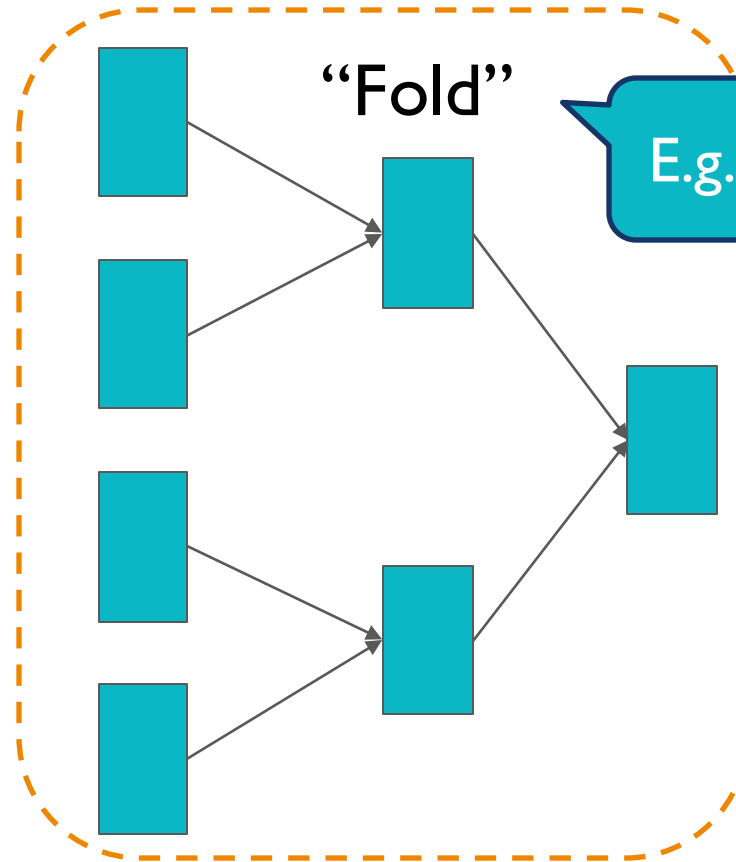
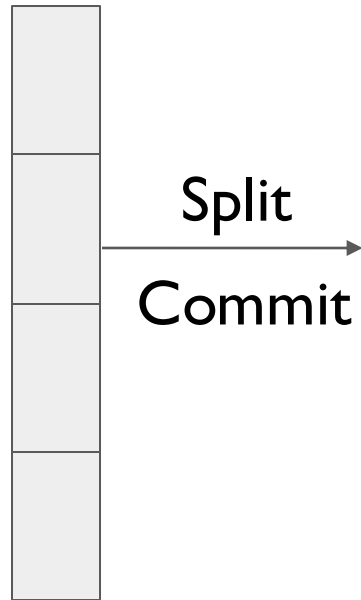
Folding-based SNARKs [BGH'19, COS'20, NDCTB'24...]

Computation



Folding-based SNARKs [BGH'19, COS'20, NDCTB'24...]

Computation



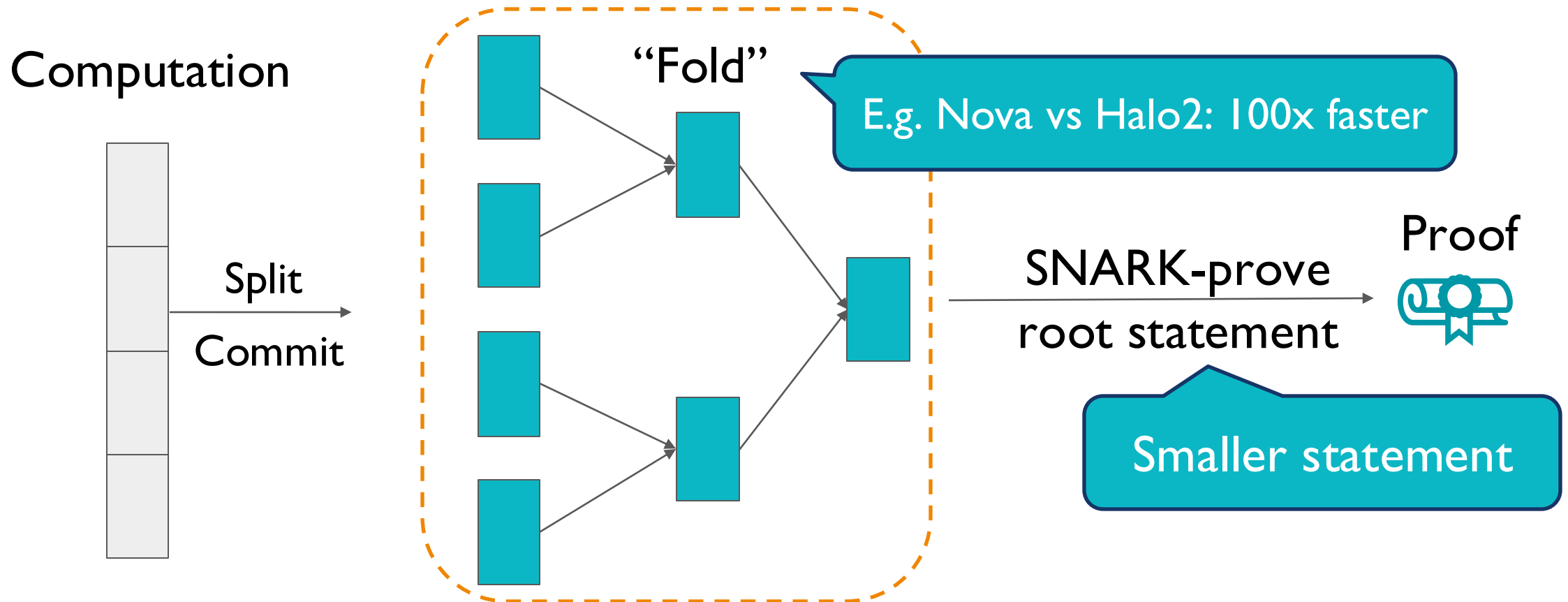
E.g. Nova vs Halo2: 100x faster

SNARK-prove
root statement

Proof

Smaller statement

Folding-based SNARKs [BGH'19, COS'20, NDCTB'24...]



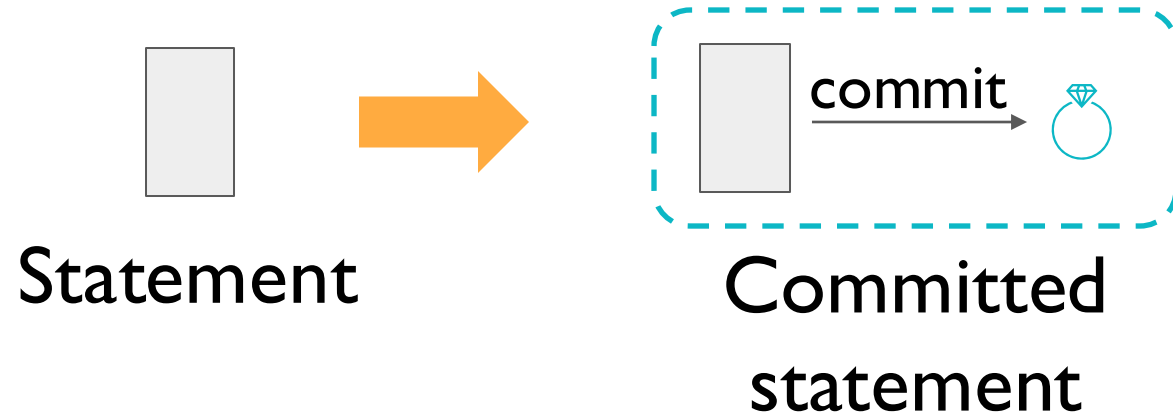
Qs: Which commitment should we use?

The Choice of Commitments

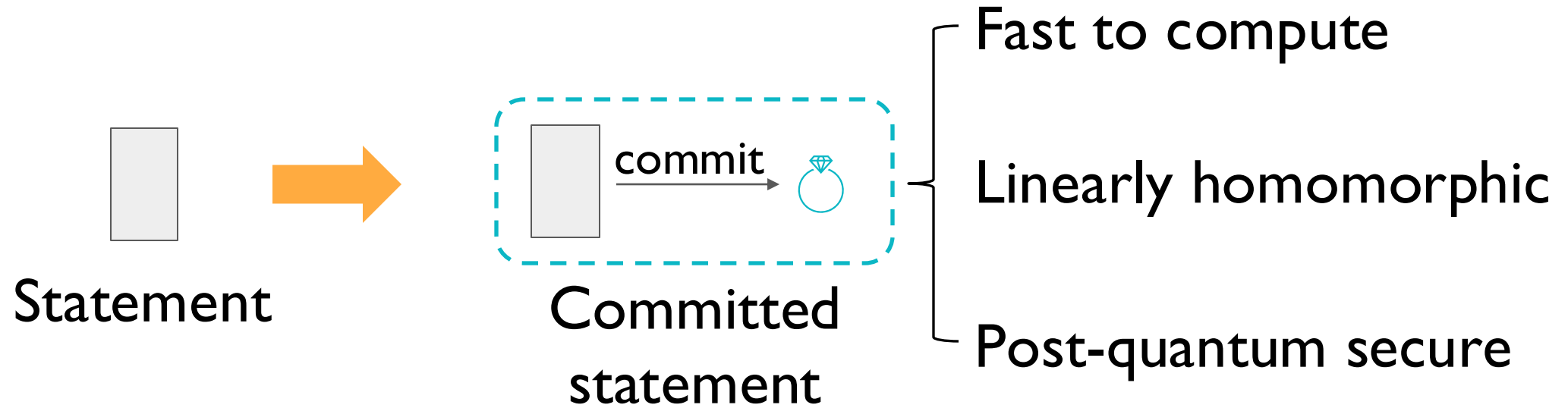


Statement

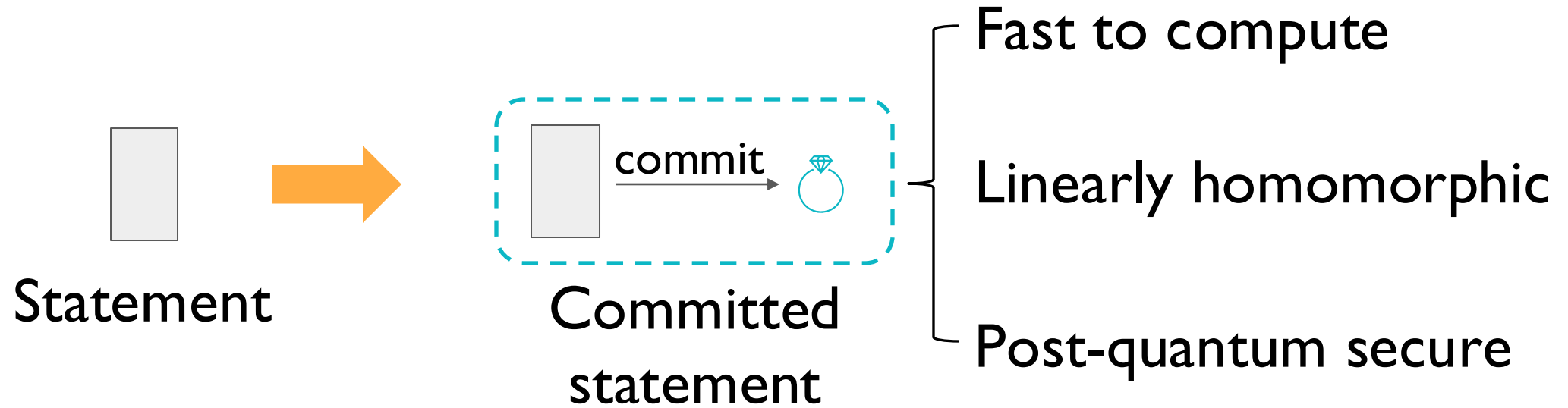
■ The Choice of Commitments



■ The Choice of Commitments



The Choice of Commitments

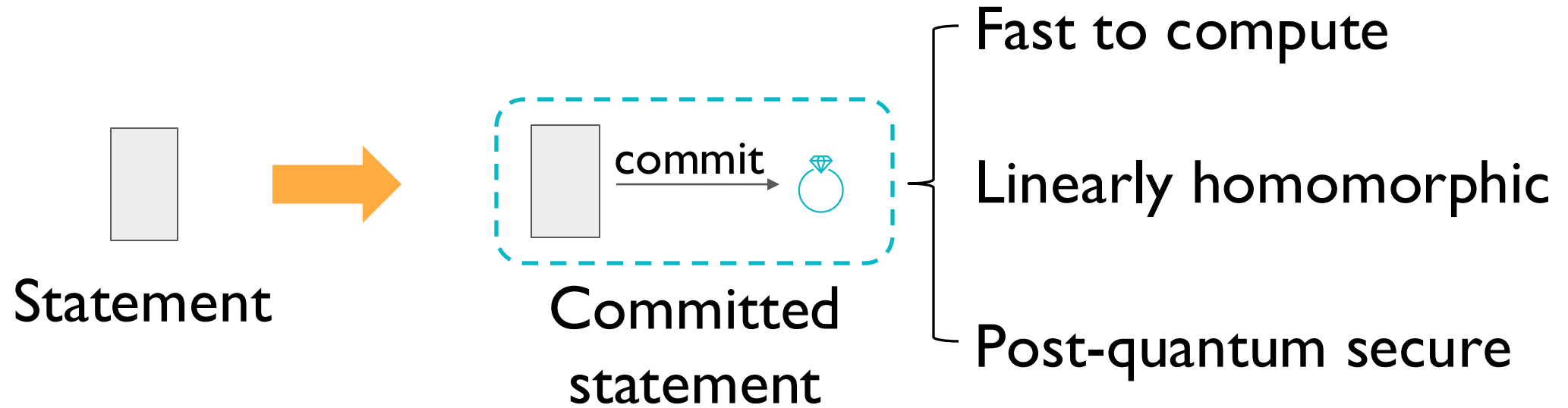


Before: Pedersen commitments [KST'21, BCLMS'21, BC'23, KS'23...]

Pros: 

- Linearly homomorphic

The Choice of Commitments



Before: Pedersen commitments [KST'21, BCLMS'21, BC'23, KS'23...]

Pros: 

- Linearly homomorphic

Cons: 

- Pre-quantum
- Require expensive cycle-curves

**Q: A folding scheme from a faster
and PQ-secure commitment?**

Our Contribution

Main Result

- The **first** plausibly PQ-secure folding scheme from Ajtai hashes
- Commitment space \approx Witness space \Rightarrow Native verifier circuit

Our Contribution

Main Result

- The **first** plausibly PQ-secure folding scheme from Ajtai hashes
- Commitment space \approx Witness space \Rightarrow Native verifier circuit

Follow-up Works:

- LF+ [BC'25], Symphony [C'25]
- Neo [NS'25]
- RoK-and-Roll [KLNO'25]
- Lova [FKNP'24]

Our Contribution

Main Result

- The **first** plausibly PQ-secure folding scheme from Ajtai hashes
- Commitment space \approx Witness space \Rightarrow Native verifier circuit

Follow-up Works:

- LF+ [BC'25], Symphony [C'25]
- Neo [NS'25]
- RoK-and-Roll [KLNO'25]
- Lova [FKNP'24]

Hash-based Schemes:

[BMNW'25a, BMNW'25b, BCFW25, BMMS'25...]

- Based on Merkle-commitments
- PQ-secure and fast committing
- More expensive verifier circuit

■ Lattice-based Commitments [Ajtai'96]

random matrix

$$A \in \mathbb{Z}_q^{L \times n}$$

*



mod q =

$$\text{cm} \in \mathbb{Z}_q^L$$



$$f \in \mathbb{Z}^n$$
$$\|f\| < B$$

Lattice-based Commitments [Ajtai'96]

random matrix

$$A \in \mathbb{Z}_q^{L \times n}$$

*



mod q =

$$\text{cm} \in \mathbb{Z}_q^L$$



$$f \in \mathbb{Z}^n$$
$$\|f\| < B$$

Support arbitrary w : decomp w
to f and check $w = Gf$

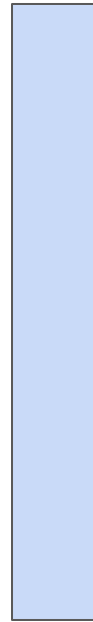
■ Lattice-based Commitments [Ajtai'96]

“Semi”-linearly homomorphic

random matrix

$$A \in \mathbb{Z}_q^{L \times n}$$

*



mod q =

$$\text{cm} \in \mathbb{Z}_q^L$$



$$f \in \mathbb{Z}^n$$
$$\|f\| < B$$

■ Lattice-based Commitments [Ajtai'96]

random matrix

$$A \in \mathbb{Z}_q^{L \times n}$$

*

mod q =

$$\text{cm} \in \mathbb{Z}_q^L$$

>30x faster than Pedersen

SWIFFT [LMPR'08]

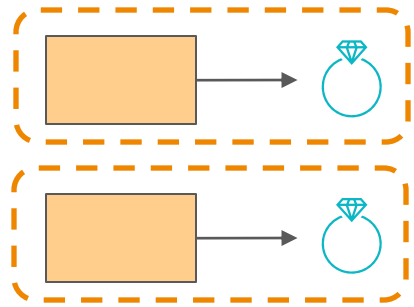
A faster variant over R_q !

$$f \in \mathbb{Z}^n$$
$$\|f\| < B$$

Q: Plug to existing folding templates?

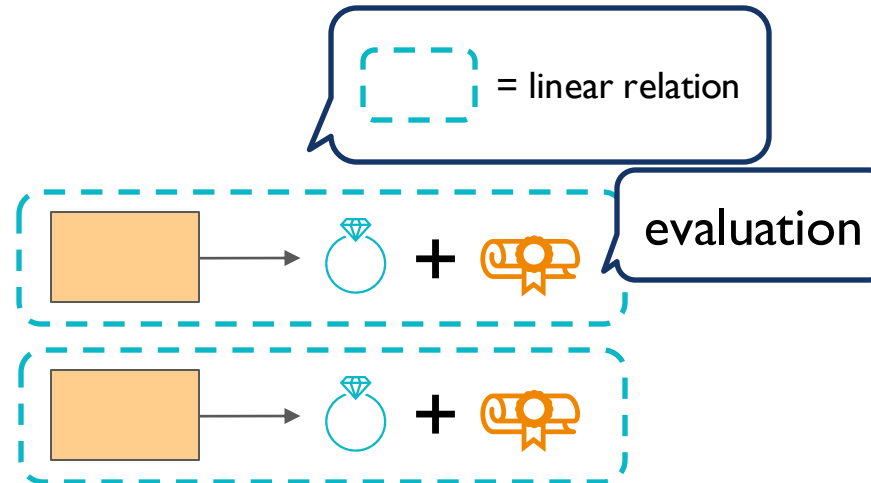
A Typical Folding Framework [KS'23]

Step 1: Commit



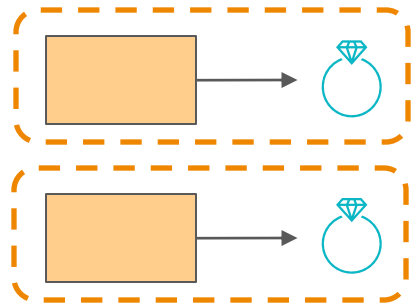
Witnesses + Comm

Step 2: Linearization



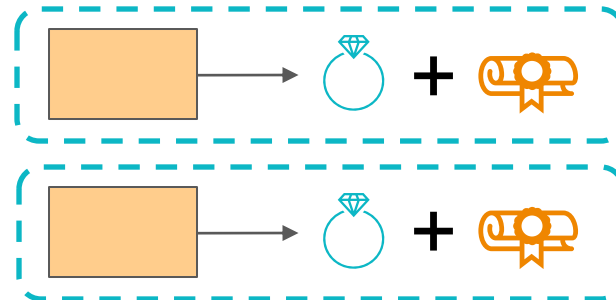
A Typical Folding Framework [KS'23]

Step 1: Commit

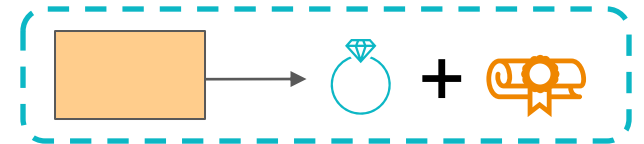


Witnesses + Comm

Step 2: Linearization

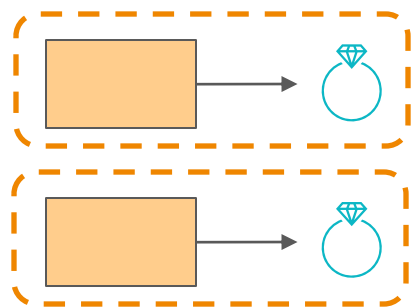


Step 3: Fold



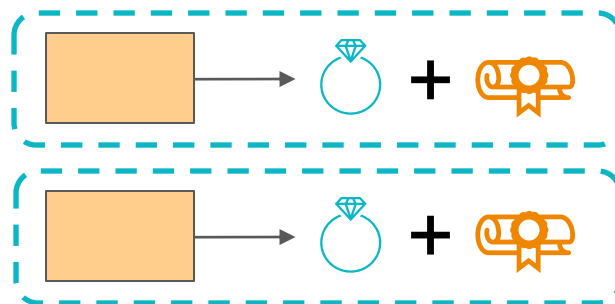
A Typical Folding Framework [KS'23]

Step 1: Commit



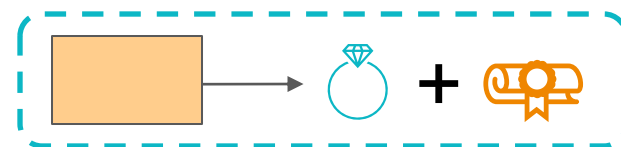
Witnesses + Comm

Step 2: Linearization



rand. lincomb

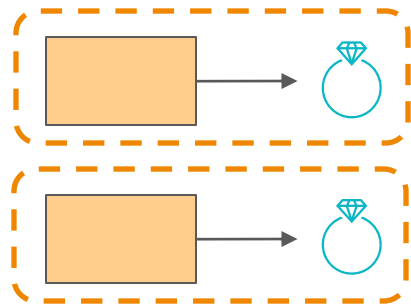
Step 3: Fold



Challenges in using lattice commitments

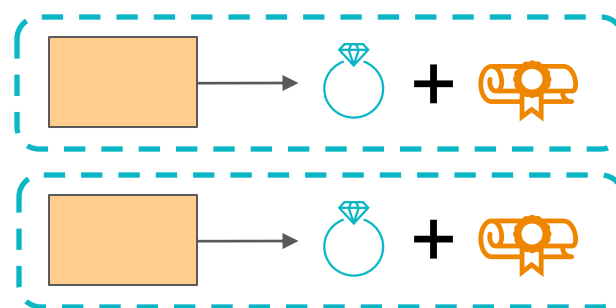
A Typical Folding Framework [KS'23]

Step 1: Commit



Witnesses + Comm

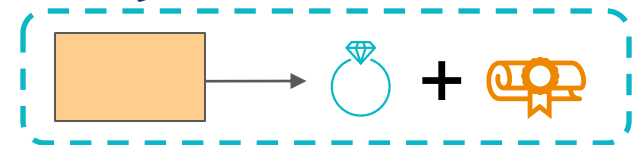
Step 2: Linearization



rand. lincomb

Step 3: Fold

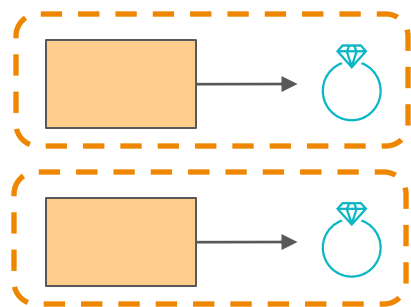
Challenge 1: Control norm blowup



Challenges in using lattice commitments

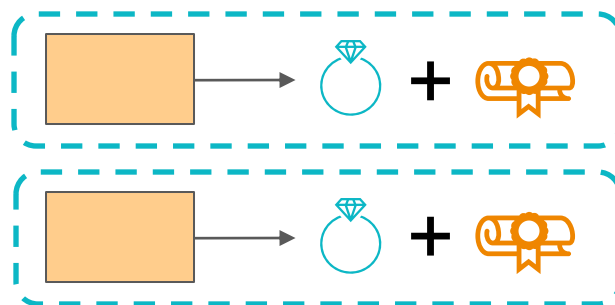
A Typical Folding Framework [KS'23]

Step 1: Commit



Witnesses + Comm

Step 2: Linearization



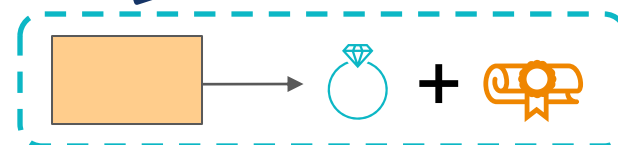
Challenge 2: Extract
low-normed witnesses



rand. lincomb

Step 3: Fold

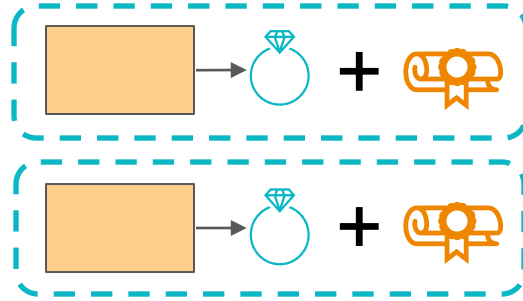
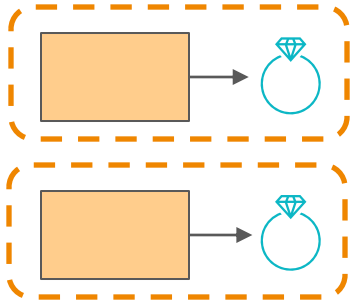
Challenge 1: Control
norm blowup



Challenges in using lattice commitments

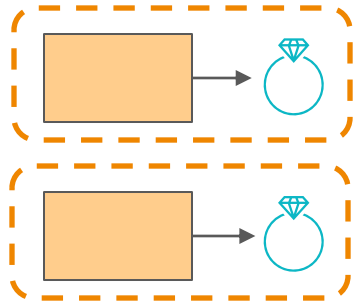
A New Lattice-Folding Framework

Step 1: Commit | Step 2: Linearize

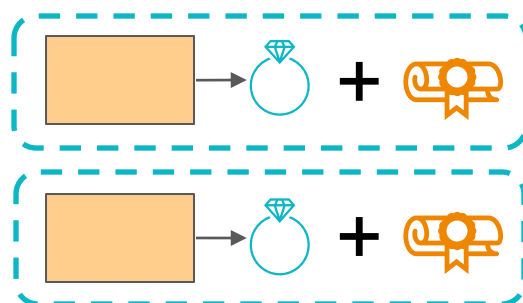


A New Lattice-Folding Framework

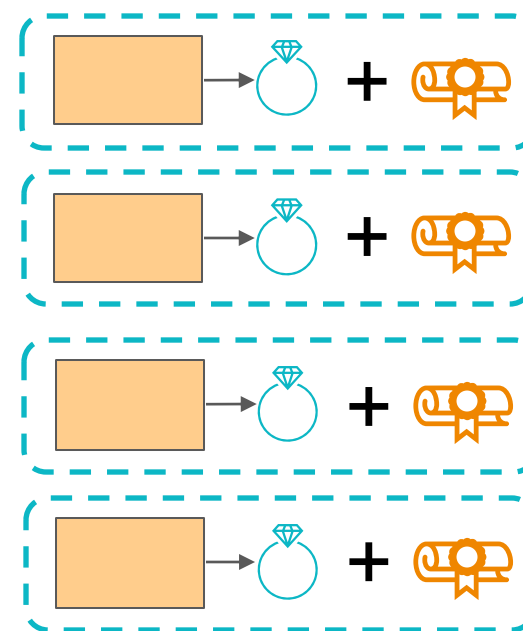
Step 1: Commit



Step 2: Linearize



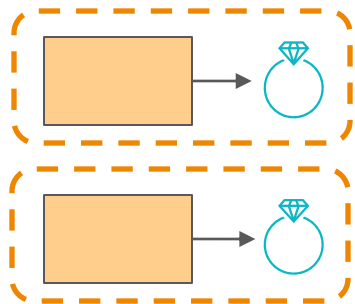
Step 3: Decomp



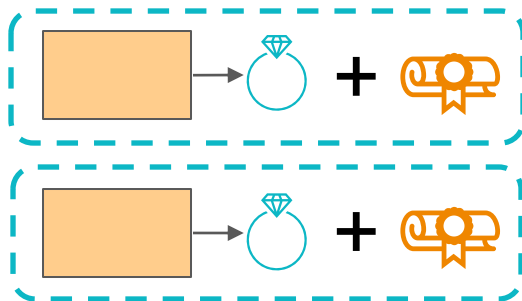
Lower norms

A New Lattice-Folding Framework

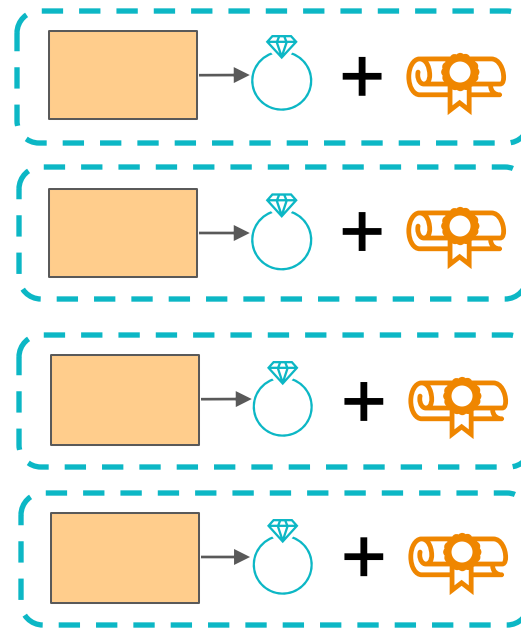
Step 1: Commit



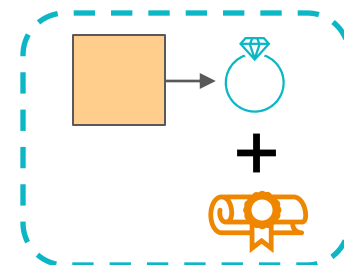
Step 2: Linearize



Step 3: Decomp



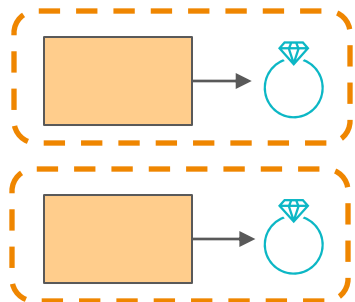
low-norm
rand. comb



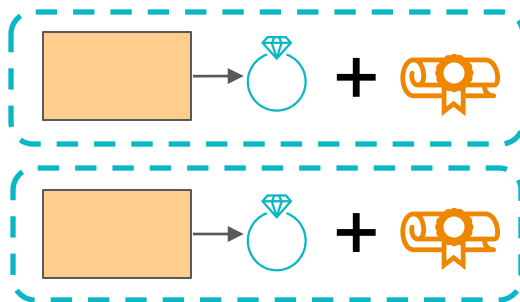
Lower norms

A New Lattice-Folding Framework

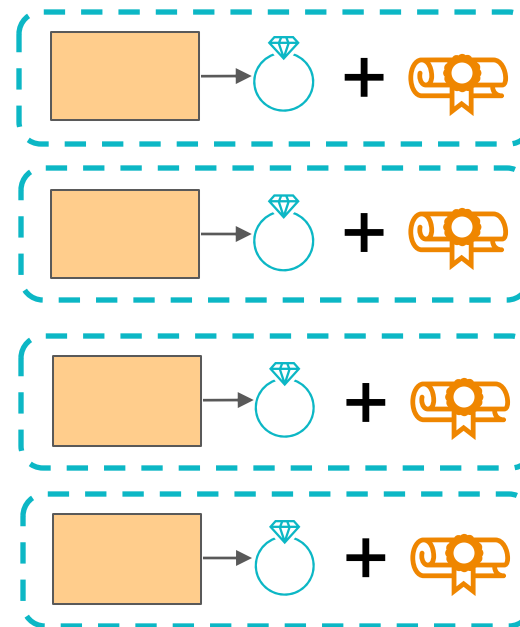
Step 1: Commit



Step 2: Linearize



Step 3: Decomp

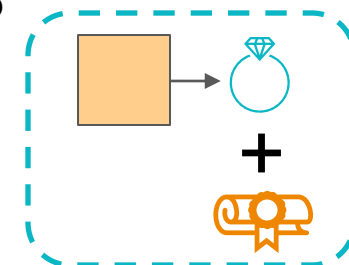


Lower norms

Step 4: Fold

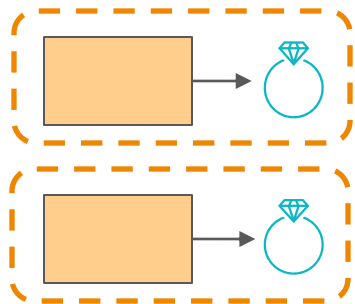
low-norm
rand. comb

No norm blowup

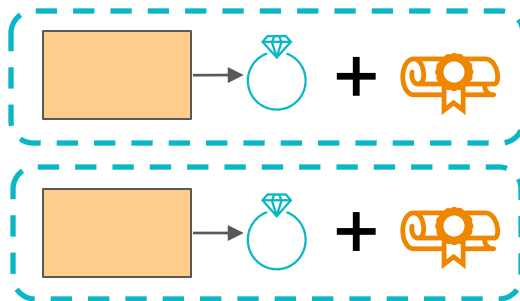


A New Lattice-Folding Framework

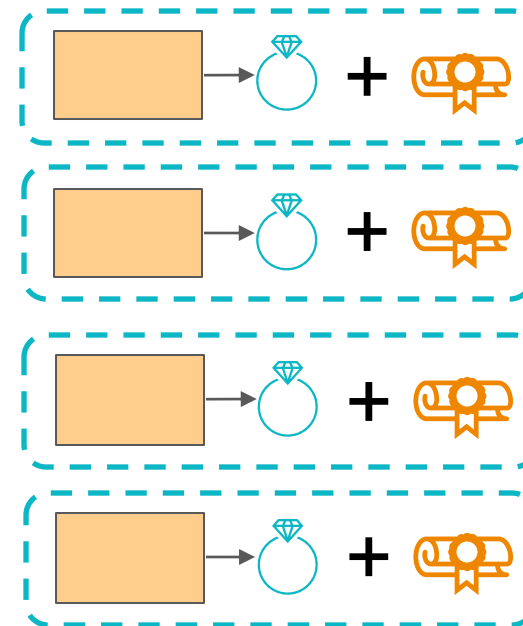
Step 1: Commit



Step 2: Linearize



Step 3: Decomp

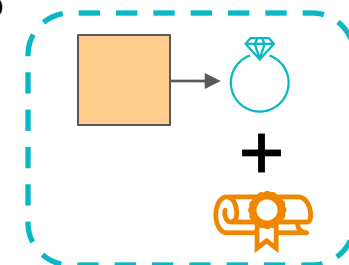


Lower norms

Step 4: Fold

low-norm
rand. comb

No norm blowup

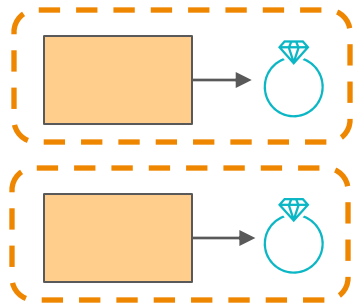


Missing Piece:

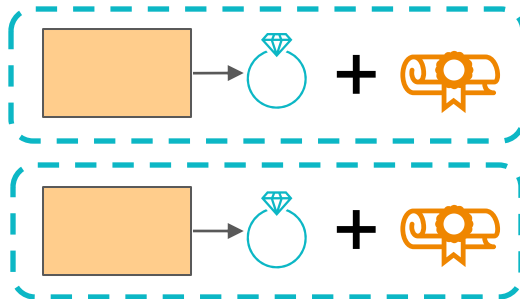
Range-proof for input witnesses

A New Lattice-Folding Framework

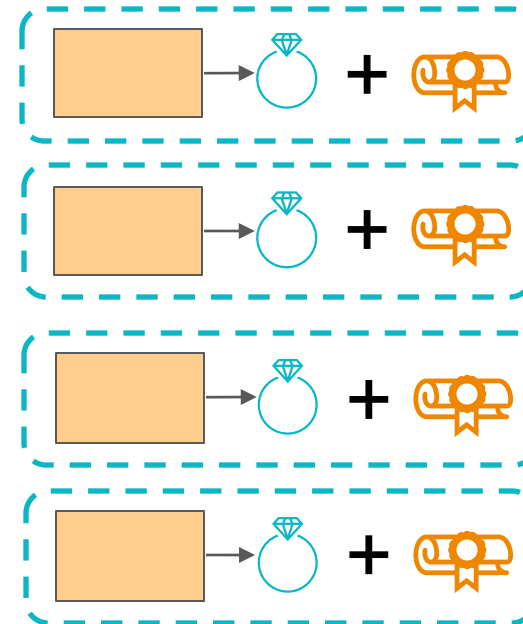
Step 1: Commit



Step 2: Linearize

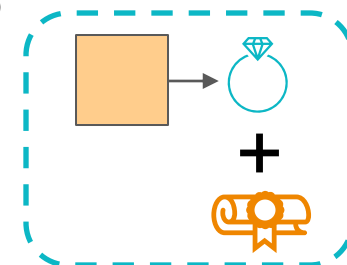


Step 3: Decomp Step 4: Fold



low-norm
rand. comb

No norm blowup



Lower norms

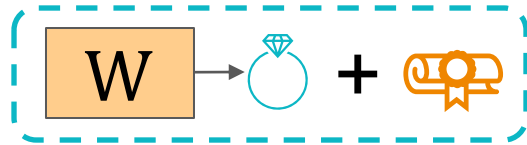
Missing Piece:

Range-proof for input witnesses

Challenge: Sublinear verification

■ Rephrasing Range Statements

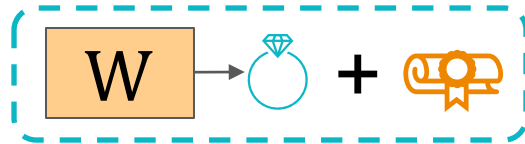
Range statement:



where each entry of W is in $[0, b)$

Rephrasing Range Statements

Range statement:

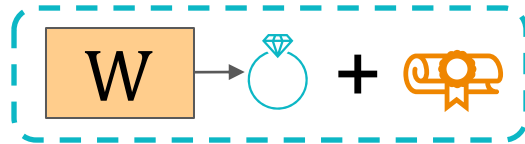


where each entry of W is in $[0, b)$

b is small

Rephrasing Range Statements

Range statement:

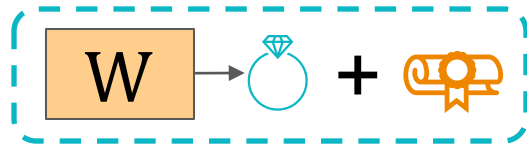


where each entry of W is in $[0, b)$

Generalize to $(-b, b)$

Rephrasing Range Statements

Range statement:



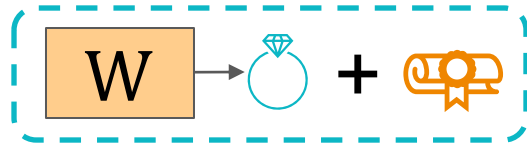
where each entry of W is in $[0, b)$

Observation:

$$W_{i,j} \in [0, b) \cap \mathbb{Z} \longleftrightarrow F(i, j) := \prod_{t \in [0, b)} (W_{i,j} - t) = 0 \bmod q$$

Rephrasing Range Statements

Range statement:



where each entry of W is in $[0, b)$

Observation:

$$W_{i,j} \in [0, b) \cap \mathbb{Z} \iff F(i,j) := \prod_{t \in [0,b)} (W_{i,j} - t) = 0 \pmod q$$

Degree- b
equation

Range Proof from Sumcheck

Observation:

$$\begin{aligned} \forall (i, j) \in \mathcal{D}: \\ F(i, j) = 0 \end{aligned}$$

[Set'20, CBBZ'23]



Sumcheck [LFKN'92]

Random

$$\sum_{(i,j) \in \mathcal{D}} F(i, j) \cdot eq((i, j), (r, r')) = 0$$

Range Proof from Sumcheck

Observation:

$$\forall (i, j) \in \mathcal{D}: \\ F(i, j) = 0$$



$$\sum_{(i,j) \in \mathcal{D}} F(i, j) \cdot eq((i, j), (r, r')) = 0$$

Sumcheck reduction:

$$\sum_{(i,j) \in \mathcal{D}} F(i, j) \cdot eq((i, j), (r, r')) = 0$$

Range Proof from Sumcheck

Observation:

$$\forall (i, j) \in \mathcal{D}: \\ F(i, j) = 0$$

[Set'20, CBBZ'23]



$$\sum_{(i, j) \in \mathcal{D}} F(i, j) \cdot eq((i, j), (r, r')) = 0$$

Sumcheck reduction:

$$\sum_{(i, j) \in \mathcal{D}} F(i, j) \cdot eq((i, j), (r, r')) = 0$$

[LFKN'92]



Linear evaluation

$$\tilde{W}(\vec{\gamma}) = v$$

Range Proof from Sumcheck

Observation:

$$\forall (i, j) \in \mathcal{D}: \\ F(i, j) = 0$$

[Set'20, CBBZ'23]



$$\sum_{(i, j) \in \mathcal{D}} F(i, j) \cdot eq((i, j), (r, r')) = 0$$

Sumcheck reduction:

$$\sum_{(i, j) \in \mathcal{D}} F(i, j) \cdot eq((i, j), (r, r')) = 0$$

[LFKN'92]



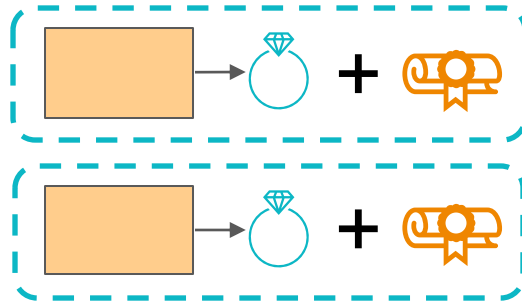
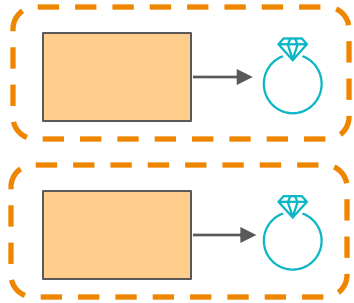
Linear evaluation

$$\tilde{W}(\vec{\gamma}) = v$$

Folding-friendly

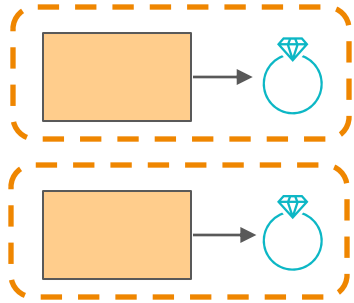
Final Scheme

Step 1: Commit | Step 2: Linearize

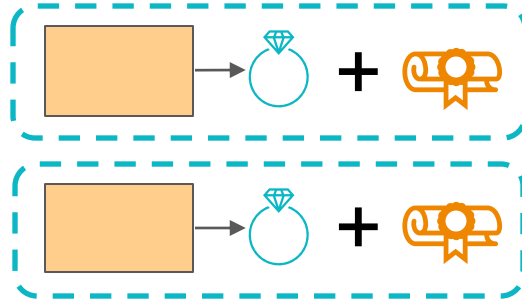


Final Scheme

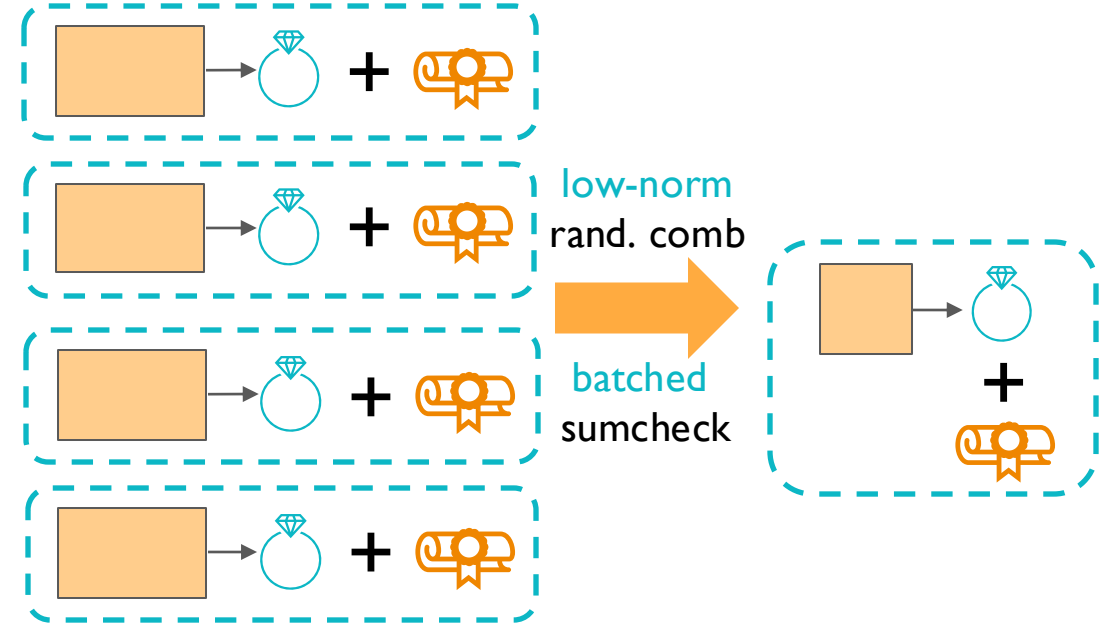
Step 1: Commit



Step 2: Linearize

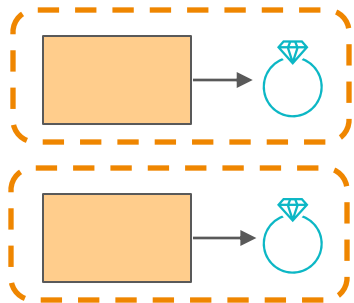


(Combined) Step 3:
Decomp + Range-chk + Fold

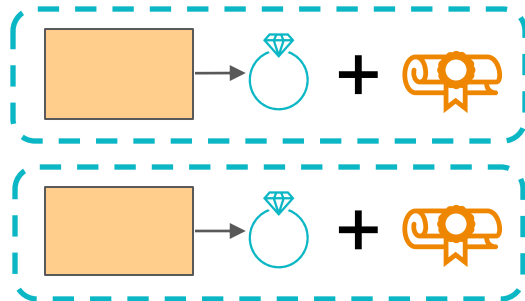


Final Scheme

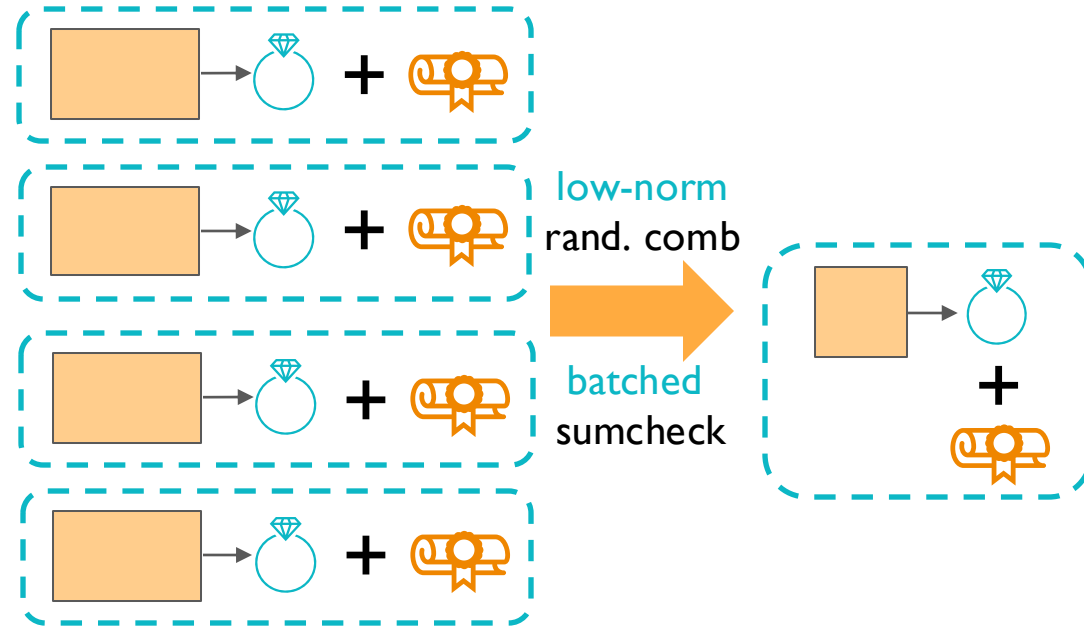
Step 1: Commit



Step 2: Linearize



(Combined) Step 3:
Decomp + Range-chk + Fold

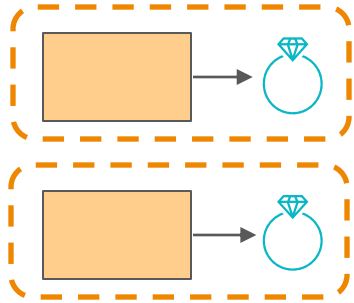


Folding Verifier:

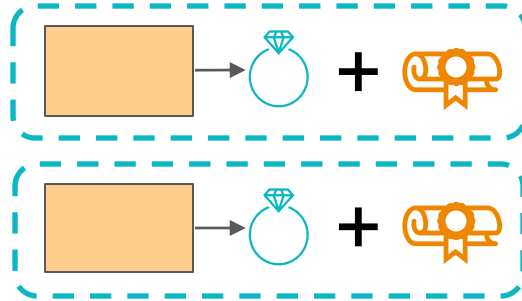
- Batched Sumcheck: $O(\log n)$ Rq ops
- Rand. comb: $O(\lambda)$ field ops

Final Scheme

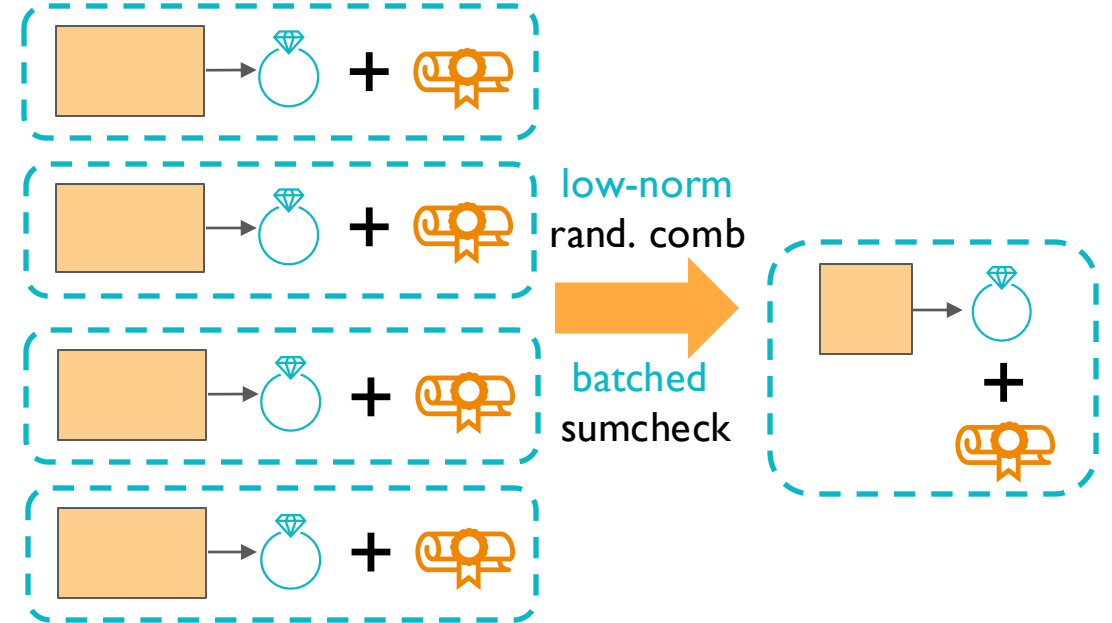
Step 1: Commit



Step 2: Linearize



(Combined) Step 3:
Decomp + Range-chk + Fold



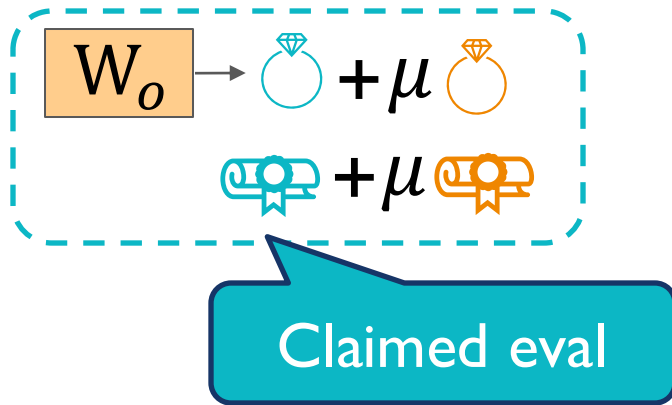
Folding Verifier:

- Batched Sumcheck: $O(\log n)$ Rq ops
- Rand. comb: $O(\lambda)$ field ops

[BC'25, NS'25]: Replace ring w/ field ops

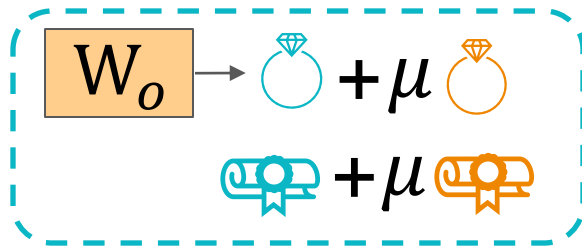
Knowledge Extraction

1st folded instance-witness pair:



Knowledge Extraction

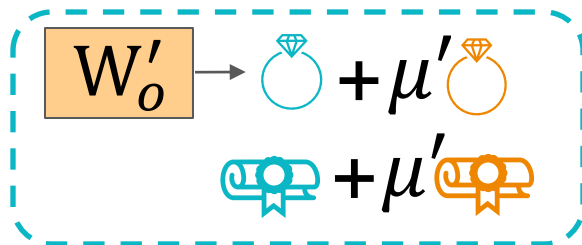
1st folded instance-witness pair:



Rewind*



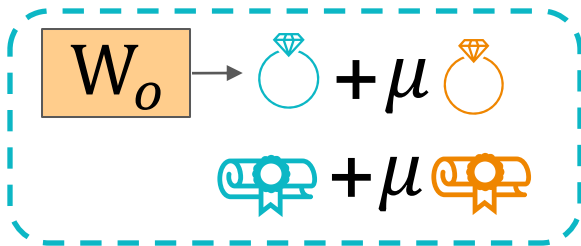
2nd folded instance-witness pair:



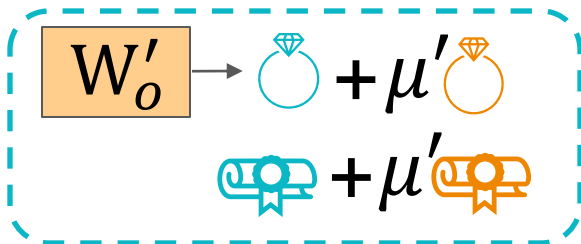
*: We actually rewind more than two times.

Knowledge Extraction

1st folded instance-witness pair:



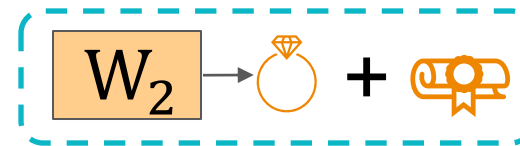
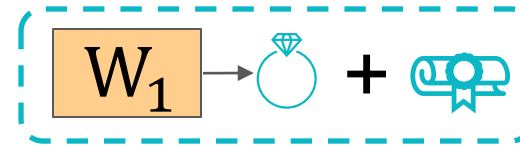
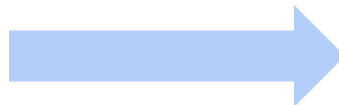
2nd folded instance-witness pair:



Rewind*



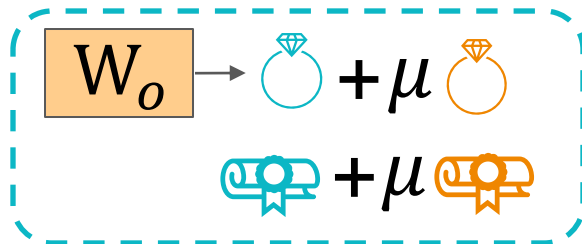
Interpolate



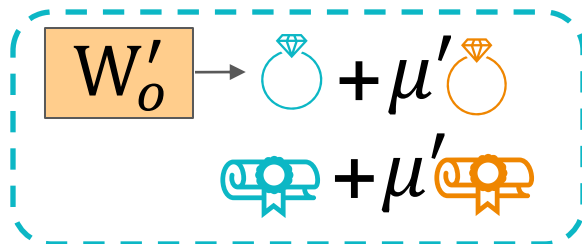
*: We actually rewind more than two times.

Knowledge Extraction

1st folded instance-witness pair:



2nd folded instance-witness pair:



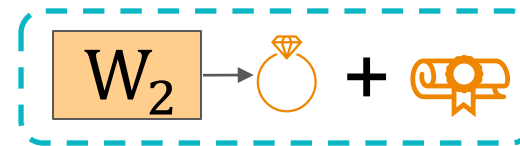
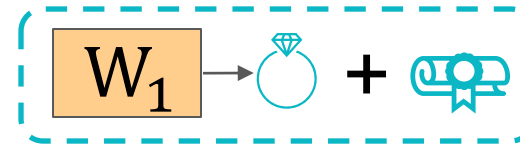
Rewind*



Interpolate



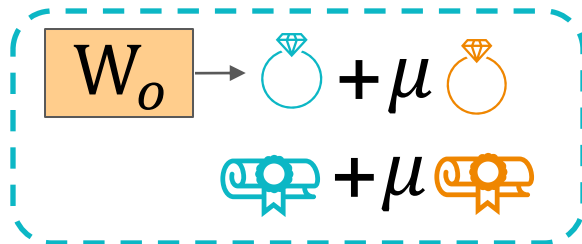
Relaxed binding



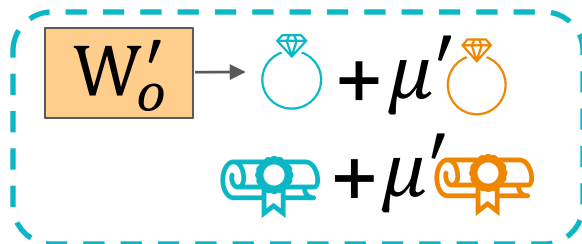
*: We actually rewind more than two times.

Knowledge Extraction

1st folded instance-witness pair:



2nd folded instance-witness pair:



Rewind*

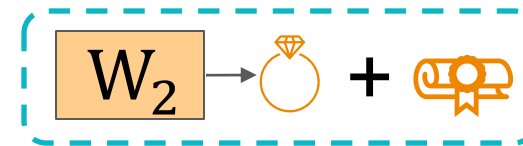
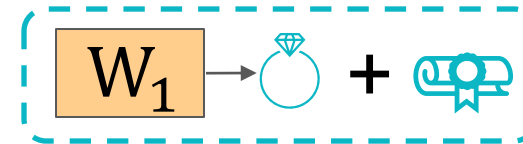


Interpolate



Relaxed binding

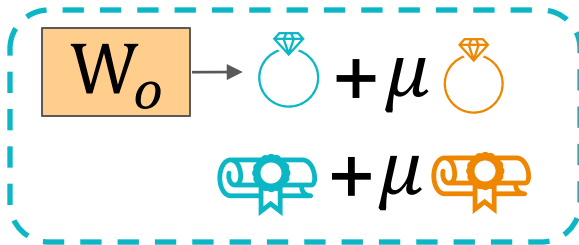
Satisfies eval claim



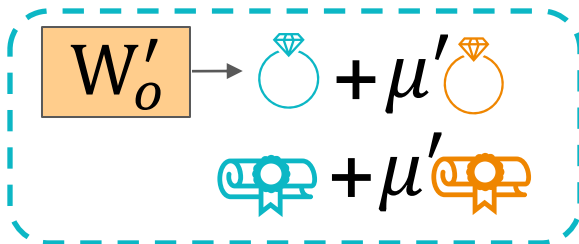
*: We actually rewind more than two times.

Knowledge Extraction

1st folded instance-witness pair:



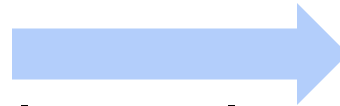
2nd folded instance-witness pair:



Rewind*

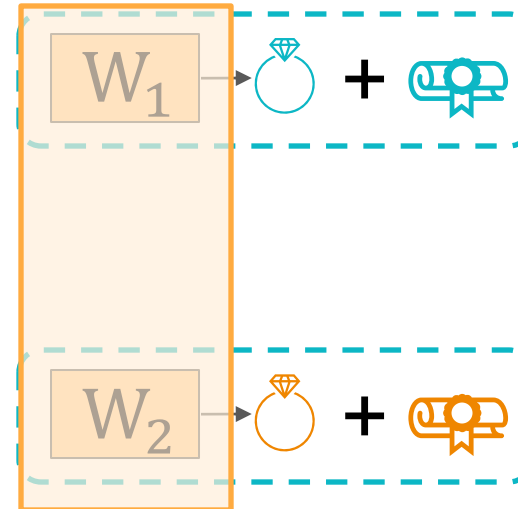


Interpolate



Relaxed binding

Satisfies eval claim

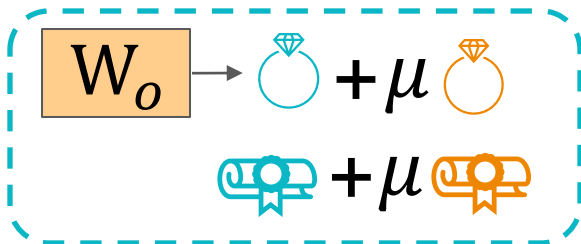


Low-norm by
Sumcheck soundness

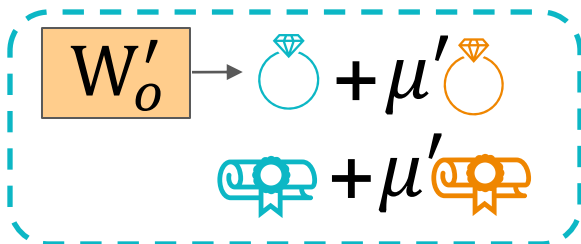
*: We actually rewind more than two times.

Knowledge Extraction

1st folded instance-witness pair:



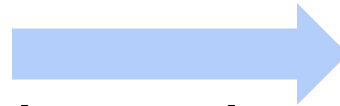
2nd folded instance-witness pair:



Rewind*

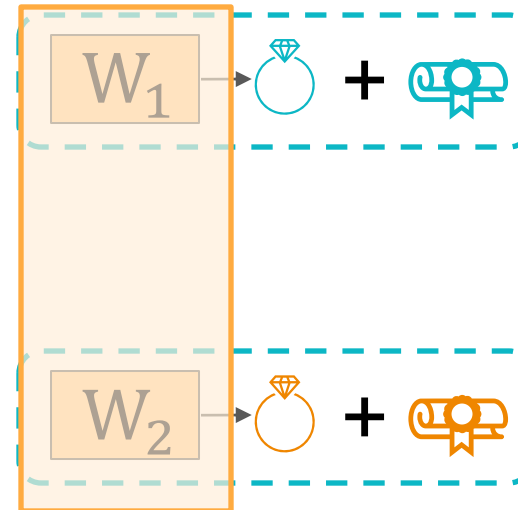


Interpolate



Relaxed binding

Satisfies eval claim



Low-norm by
Sumcheck soundness

More subtleties on supporting witness vectors over R_q

*: We actually rewind more than two times.

Summary & Future Work

Summary & Extensions

- **First** plausibly PQ-secure folding scheme from Module-SIS
- Extensions: Small modulus & high-degree constraint system

Summary & Future Work

Summary & Extensions

- **First** plausibly PQ-secure folding scheme from Module-SIS
- Extensions: Small modulus & high-degree constraint system

Follow-up

- [BC'25, C'25]: Efficient range-proofs w/o decomposition
- [NS'25]: Pay-per-bit commitment + better support to small fields
- [FKNP'24]: Security from SIS

Summary & Future Work

Summary & Extensions

- **First** plausibly PQ-secure folding scheme from Module-SIS
- Extensions: Small modulus & high-degree constraint system

Follow-up

- [BC'25, C'25]: Efficient range-proofs w/o decomposition
- [NS'25]: Pay-per-bit commitment + better support to small fields
- [FKNP'24]: Security from SIS

Future work

QROM Security Analysis

THANK YOU

Eprint 2024/257

