ZKproof 7, March 2025

LatticeFold and LatticeFold+: Lattice-Based Folding

Dan Boneh and **Binyi Chen**
Stanford University

eprint/2024/257 and eprint/2025/247

What is a SNARK?

$\boldsymbol{C}(\mathbb{x}, \mathbb{w})$: an arithmetic circuit over $\mathbb{F}_p$ for some prime p

statement $\uparrow$ $\uparrow$ witness

$(pk_C, vk_C) \leftarrow \text{Setup}(\boldsymbol{C})$ (preprocessing)

$\text{Prove}(pk_C, \mathbb{x}, \mathbb{w}) \rightarrow \pi$

$\text{Verify}(vk_C, \mathbb{x}, \pi) \rightarrow 0/1$

Succinct: $|\pi|$=polylog($|C|$) and time(Verify)=poly($\log|C|, |\mathbb{x}|$)

An example: zkPi [Laufer-Ozdemir-B, 2024]

(ACM CCS 2024)

zkPi: Proving knowledge of a LEAN proof

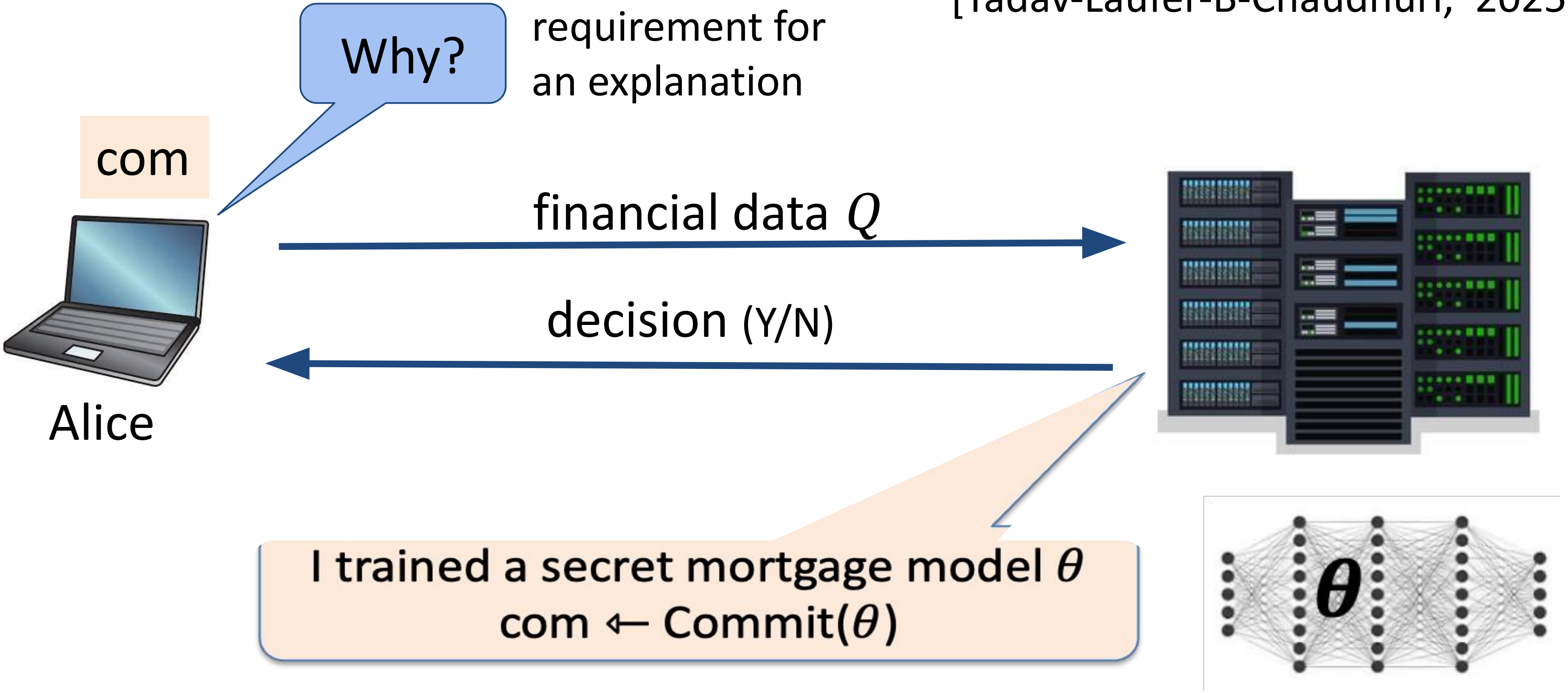
- x: a theorem written in Lean
- w: a Lean proof of the theorem x
- C: $C(x,w)=0$ iff Lean verifier accepts w as a proof for x

zkPi: a SNARK system for C that outputs succinct proofs

- π is short no matter the size of w (Fermat would have loved this)
- Currently: can generate succinct proofs for
  (43.4% of the stdlib) and (11.1% of mathlib) for Lean4

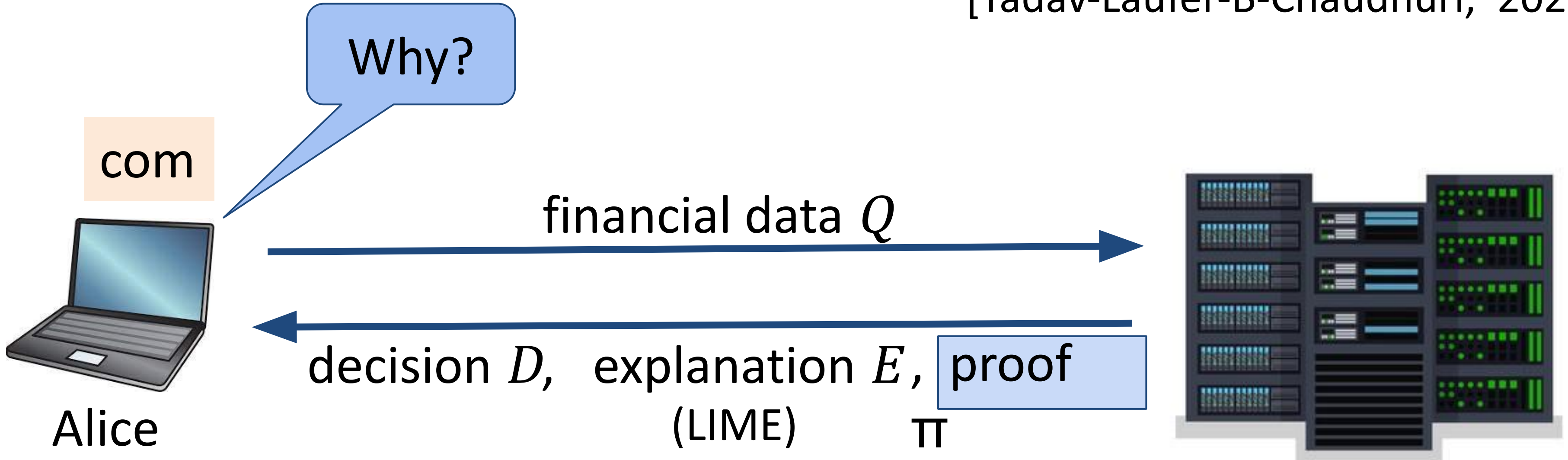
ExpProof: proving AI model explanation in ZK

[Yadav-Laufer-B-Chaudhuri, 2025]



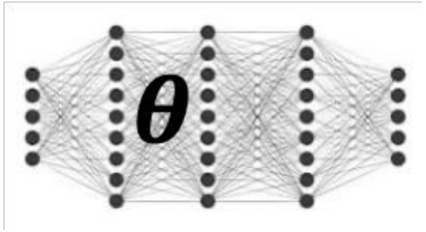
ExpProof: proving AI model explanation in ZK

[Yadav-Laufer-B-Chaudhuri, 2025]



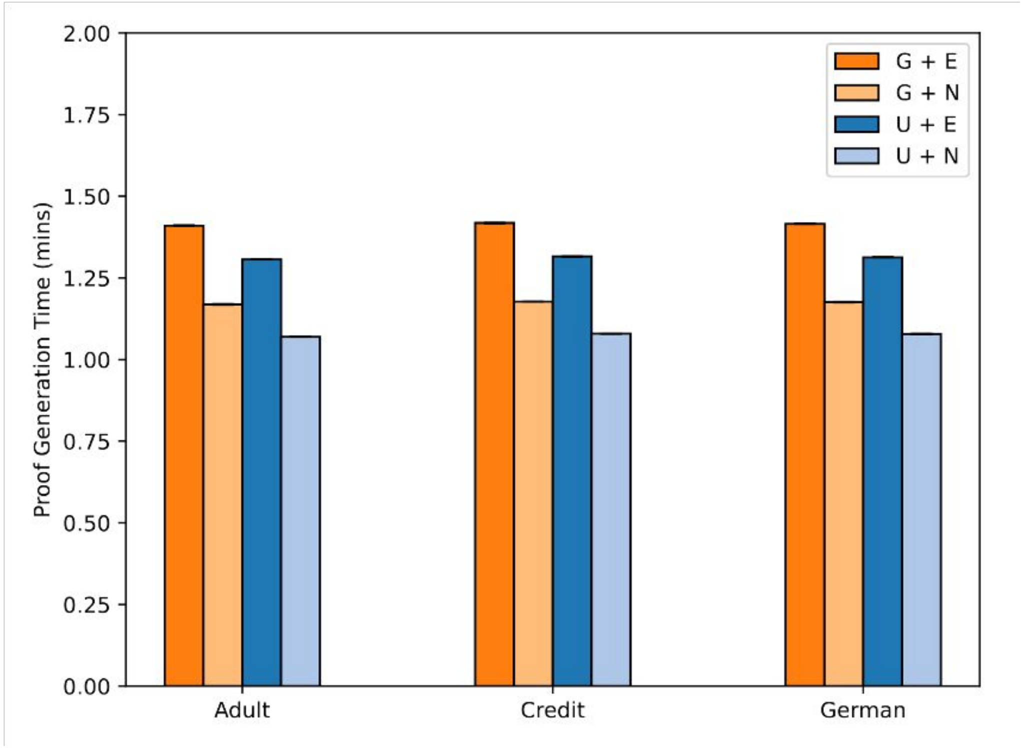
Proof π:

$$\mathrm{commit}(\Theta) = \mathrm{com}, \quad f_\Theta(Q) = D, \quad \boxed{\mathrm{LIME}(\Theta, Q) = E}$$



ExpProof: proving AI model explanation in ZK

[Yadav-Laufer-B-Chaudhuri, 2025]



income predictor

loan application risk

Likelihood to default on credicard payment

ExpProof time for three models

… an extension of EZKL

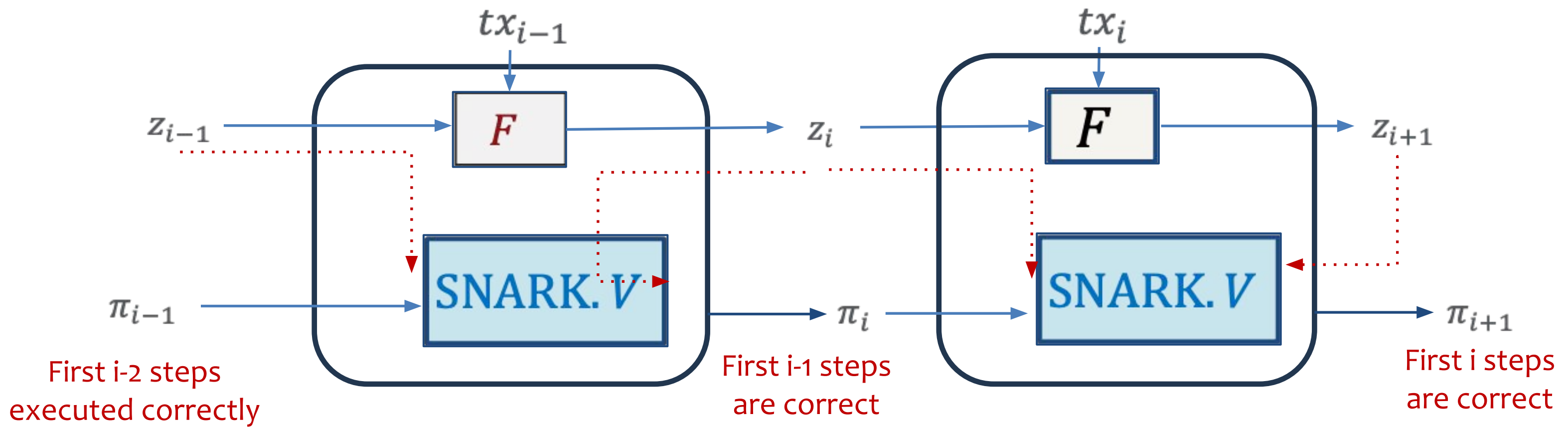
Pre- vs. Post- Quantum SNARKs

Three SNARK families

	pairing-based	**hash-based**	**lattice-based**
Proof size:	< 1KB	50-100KB	30-100KB
Verifier time:	< 10ms	< 10ms	slower (e.g., Greyhound’24)
Folding:	fast (e.g., Hypernova)	doable (Arc)	??? This work: Yes!

Piecemeal SNARKs: IVC/PCD [Valiant08, BCCT12]

Break a large circuit into small steps, prove each step on its own



Pros:

- Small memory footprint
- Build proof as statements stream in
- Parallelizable using a tree

The problem:

- Expensive SNARK.V circuit
- Expensive proof generation per step

Need a better way to compose steps

Folding Schemes [BCLMS20,KST21]

Recall: linearly homomorphic commitments:

Commit: long vector $w \in \mathbb{F}^n$ $\longrightarrow$ short c_w

Homomorphism: $w_1 + w_2$ $\longrightarrow$ $c_{w_1+w_2} = c_{w_1} \cdot c_{w_2}$

(note: Arc builds folding without relying on a linear homomorphism)

Folding Schemes [BCLMS20,KST21]

Folding: Compress multiple NP statements into one

$$R_{\text{com}} := \{\ (u = (x, c_w);\ w) :\ \ (x, w) \in R_{NP}\ \ \wedge\ \ c_w = \text{Comm}(w)\ \ \}$$

Much faster than SNARK.P!

Much faster than SNARK.V!

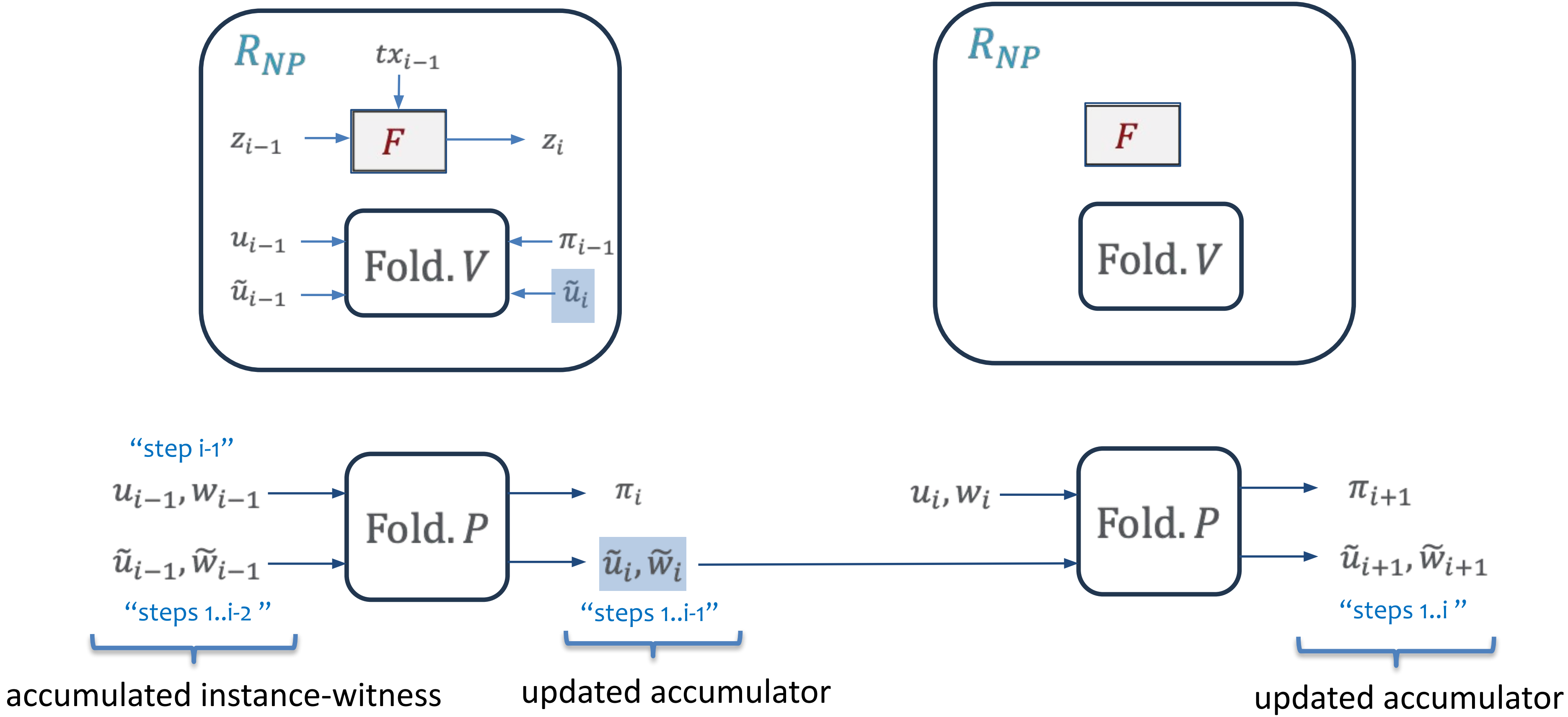
Complete: if $\text{Fold}.P$ ran correctly, then $\text{Fold}.V$ accepts

Reduction of knowledge [KP'22]: $(\tilde{u}, \tilde{w}) \in R_{\text{com}}$ and $\text{Fold}.V$ accepts $u_1, u_2, \tilde{u}, \pi$

(informal)

then prover "knows" w_1, w_2 s.t. $(u_1, w_1), (u_2, w_2) \in R_{\text{com}}$

A piecemeal SNARK from folding (IVC/PCD)



Instantiating Folding: which hom. commit?

Option 1: Pedersen p, q: ≈256-bit primes

$$w := (w_1, w_2 \dots, w_n) \in \mathbb{F}_p^n \longrightarrow c_w := g_1^{w_1} g_2^{w_2} \cdots g_n^{w_n} \in \mathbb{G} \subseteq \mathbb{F}_q \times \mathbb{F}_q$$

Pros: clean proof of knowledge soundness (from 3-special soundness)

Cons:

- Fold.P: expensive group exponentiations over **large** fields (256-bit)
- Fold.V: uses $\mathbb{G}$-ops and $\mathbb{F}_p$ field ops over
  - need to support both $\mathbb{F}_p, \mathbb{F}_q$ ⇒ field emulation (e.g. $\mathbb{F}_p$-ops over $\mathbb{F}_q$)
- Not post-quantum (not binding)

Can we fold with a lattice hom. commit?

We show: Yes! $\Rightarrow$ **LatticeFold** … but lots of complications

eprint.iacr.org/2024/257

Many benefits to lattice folding:

- **Fold.P**: arithmetic over **small** fields (64-bit)
  - Ajtai hash is fast: 30x faster than Pedersen [Nethermind]
  - Accelerate prover using the same hardware for accelerating FHE
- **Fold.V**: uses arithmetic in **only one field** $\Rightarrow$ no field emulation
- Post-quantum

Review: Ajtai Binding Commitment

Note: A is generated from a seed

Params: $q =$ 64-bit prime, $\beta = 2^{16}$, $n \gg \lambda$

vector $w \in [-\beta, \beta]^n \xrightarrow{\boxed{A \leftarrow \mathbb{Z}_q^{\lambda \times n}}} c_w = A \cdot w \bmod q \in \mathbb{Z}_q^{\lambda}$

note: this is a binding commitment only for vectors w in $[-\beta, \beta]^n$!!

<u>Trivially linearly homomorphic:</u>

$$c_{w_1} + c_{w_2} = (Aw_1 + Aw_2) \bmod q = A(w_1 + w_2) \bmod q = c_{w_1 + w_2}$$

… but committing complexity is $O(\lambda n)$ $\mathbb{Z}_q$-ops

Review: Ring/Module Ajtai [LM07,PR07]

Params: $R_q := \mathbb{Z}_q[X]/(X^d+1)$

vector $w \in [-\beta,\beta]^n \xrightarrow{\boxed{A \leftarrow \mathbb{Z}_q^{\lambda\times n}}} c_w = A\cdot w \bmod q \in \mathbb{Z}_q^{\lambda}$

$\mathbb{Z}_q^d \to R_q$

$\tilde{w} \in R_q^{n/d} \xrightarrow{\tilde{A} \leftarrow R_q^{\lambda/d \times n/d}} \tilde{c}_w = \tilde{A}\cdot\tilde{w} \in R_q^{\lambda/d}$

only binding when coefficients of $\tilde{w}$ *are* in $\{-\beta,\ldots,\beta\}$

Now, committing complexity is only $O(\lambda n/d^2)$ R_q-ops (fast via FFT)

Challenge #1 with Folding with Ajtai

Folding with Ajtai:

c_{w_1}, w_1

c_{w_2}, w_2

$\xrightarrow{\text{random } \gamma \in \mathbb{F}_p}$

$$\boxed{\begin{array}{l} \tilde{c} := c_{w_1} + \gamma c_{w_2} \\ \widetilde{w} := w_1 + \gamma w_2 \end{array}}$$

$\notin [-\beta, \beta]^n$ anymore

Challenge: folded witness must stay in the ***bounded*** msg space

… otherwise folding is not knowledge sound

Solution: norm reduction before folding

Decomposition: $a \in (-\beta, \beta) \xrightarrow[(\mathrm{b} = \beta^{1/k})]{\text{split}} a_1, \ldots, a_k \in (-b, b)$

$a = a_1 + b \cdot a_2 + \cdots + b^{k-1} \cdot a_k$

Folding: (e.g., $k = 2$ and $\mathrm{b} = \beta^{1/2}$)

$c_{w_1}, w_1 \xrightarrow{\text{split}} \left[c_{w_1}^1, w_1^1 \in (-\boldsymbol{b}, \boldsymbol{b})^n ;\; c_{w_1}^2, w_1^2 \right]$

$c_{w_2}, w_2 \xrightarrow{\text{split}} \left[c_{w_2}^1, w_2^1 ;\; c_{w_2}^2, w_2^2 \right]$

$\xrightarrow[\text{4-way folding with random low norm } \vec{\gamma} \text{ (in a sampling set)}]{\gamma_1, \gamma_2, \gamma_3, \gamma_4 \in R_q}$

$\tilde{c} = \mathrm{combine}([\gamma_i], [c_{w_1}^1 \ldots c_{w_2}^2])$

$\tilde{w} = \mathrm{combine}([\gamma_i], [w_1^1 \ldots w_2^2])$

ensures $\tilde{w} \in [-\beta, \beta]^n$

Challenge #2: extracting low-norm witnesses

Folding verifier:

Malicious P^*

$$c_{w_1}, \; c_{w_2} \xrightarrow{\text{random } \gamma} c_\gamma = c_{w_1} + \gamma c_{w_2}$$

$w_\gamma^* \in [-\beta, \beta]^n$ s.t. $(c_\gamma, w_\gamma^*) \in R_{\text{com}}$

Knowledge soundness:

Given P^*, extract low-norm openings $w_1, w_2 \in [-\beta, \beta]^n$ for c_{w_1}, c_{w_2}

The problem:

Rewind P^* to get $(\gamma_1, w_{\gamma_1}^*), (\gamma_2, w_{\gamma_2}^*)$ and find w_1, w_2 by dividing by $(\gamma_1 - \gamma_2)^{-1}$

… but then $w_1, w_2 \in R_q^{n/d}$ will have too large norms

Challenge #2: extracting low-norm witnesses

Solution (for knowledge soundness):

Fold. P must prove that the input witnesses $w_1, w_2 \in R_q^{n/d}$ are low-norm: their coefficients are in $(-\beta, \beta)$

… otherwise, commitments c_{w_1}, c_{w_2} need not be binding

How to do efficiently?

LatticeFold / LatticeFold+ range proofs

LatticeFold: range proof on w_1, w_2 using bit decomposition

$\Rightarrow$ single sum-check protocol for both range proof and CCS

LatticeFold+: a far more efficient range proof on w_1, w_2 .

(also, norm reduction <u>after</u> folding)

LatticeFold+ Range Proof

Let $f = (f_1, \ldots, f_n) \in \mathbb{Z}_q^n$ (n scalars)

Need to prove $f_i \in (-d/2, d/2)$ for all $i \in [n]$

(1) Prover Ajtai-commits to $[f] := (X^{f_1}, \ldots, X^{f_n}) \in R_q^n$ (fast!)

(2) Prove that $[f]$ is constructed correctly and $f_i \in (-d/2, d/2)$

$\Rightarrow$ both proofs can be done algebraically using sumcheck.

No bit decomposition!

LatticeFold+ Double Commitments

Problem: when f is a vector of n ring elems. prover sends d commitments:

$$c := (c_0, \ldots, c_{d-1}) \in R_q^{\lambda}$$

(high norm)

To shrink proof: send a commitment to c

- First, decompose c to reduce its norm, then Ajtai-commit to result
- We call this a **double commitment**

New problem: a double commitment is not linearly homomorphic!

Solved by a sumcheck-based commitment transformation

LatticeFold: Summary

LatticeFold, LatticeFold+, Neo: lattice-based folding schemes

- Small field; efficient verifier circuit; post-quantum
- Support for high-deg. constraint systems (CCS)
- Transparent setup

For piecemeal SNARKs: post-quantum LatticeFold+ is likely better, than pre-quantum Pedersen folding

THE END

eprint/2024/257
LatticeFold

eprint/2025/247
LatticeFold+

eprint/2025/294
Neo