Folding Schemes with Selective Verification

Carla Ràfols
Pompeu Fabra University

Alexandros Zacharakis
Toposware

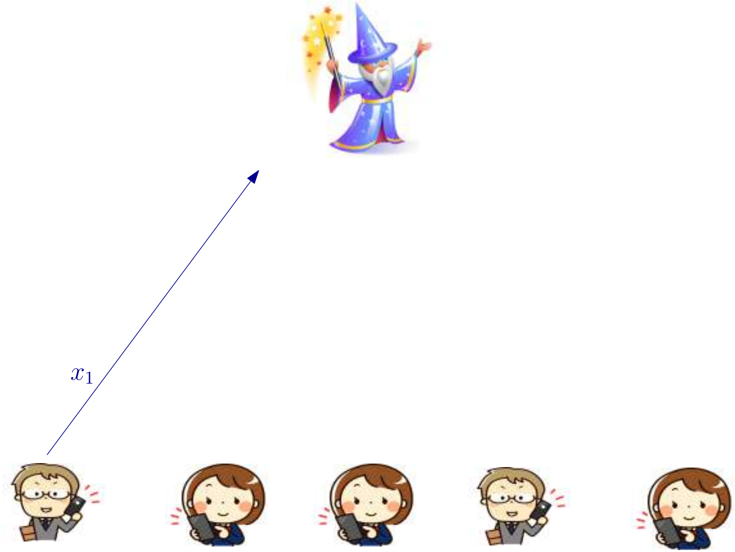
October 2, 2023

This work was partially funded by a Protocol Labs Research Grant.

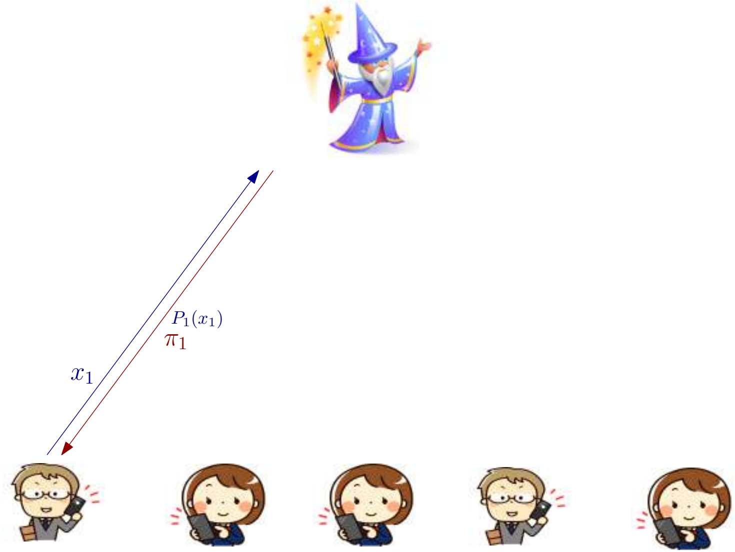
Motivation

Motivation: delegation of computation aaS

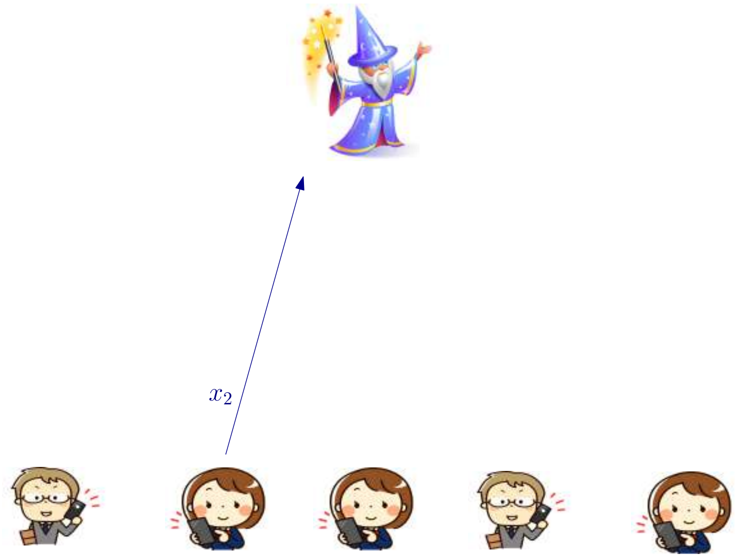
Motivation: delegation of computation aaS



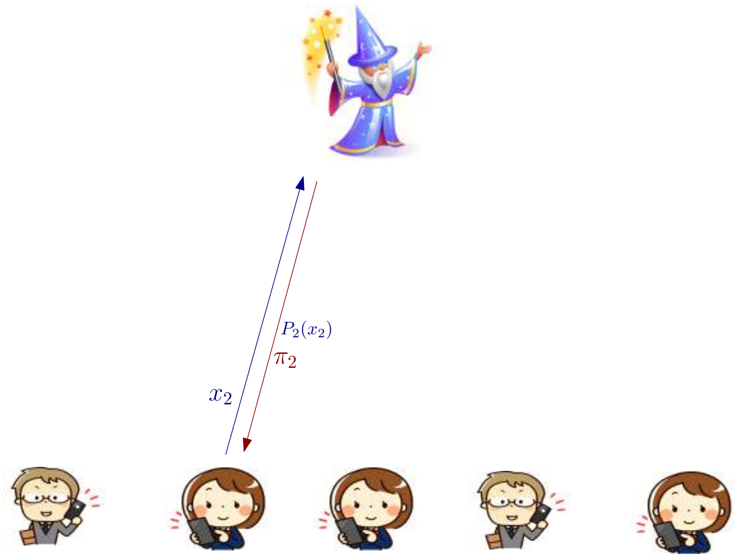
Motivation: delegation of computation aaS



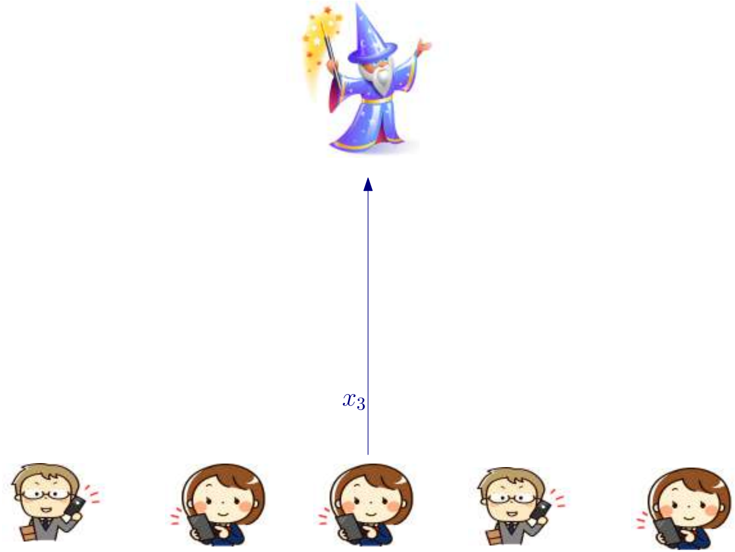
Motivation: delegation of computation aaS



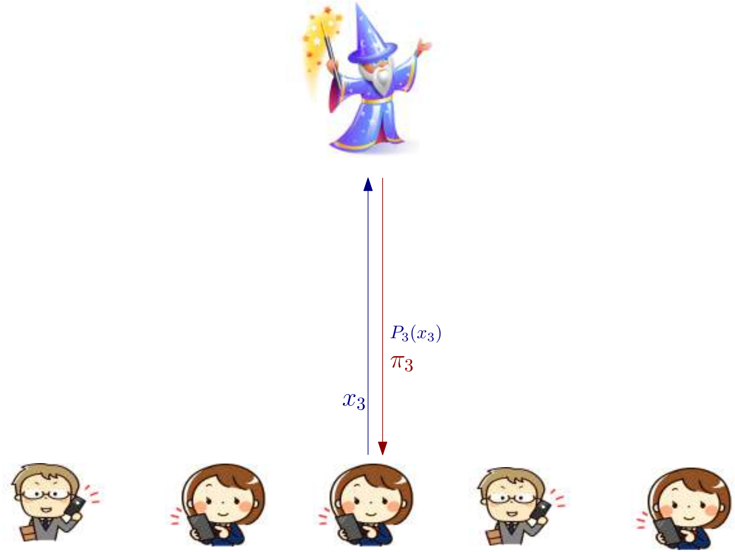
Motivation: delegation of computation aaS



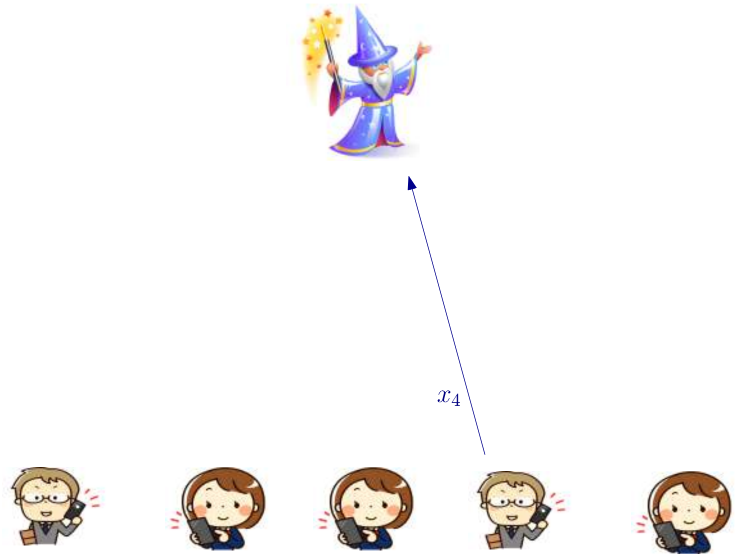
Motivation: delegation of computation aaS



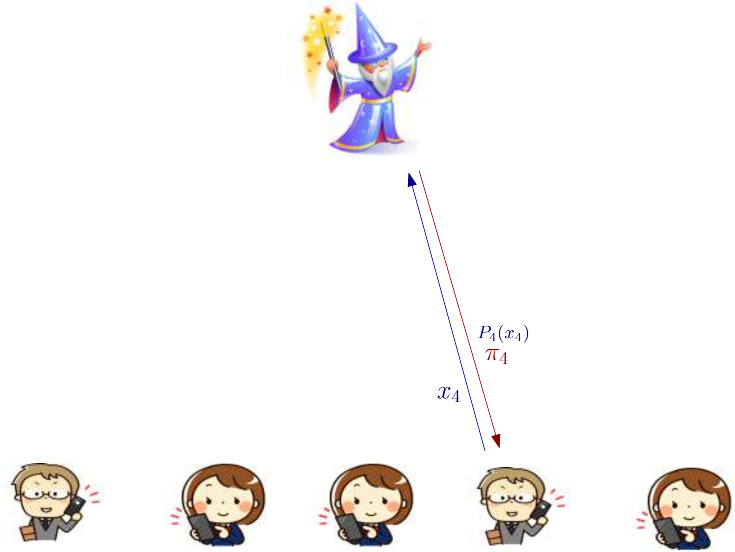
Motivation: delegation of computation aaS



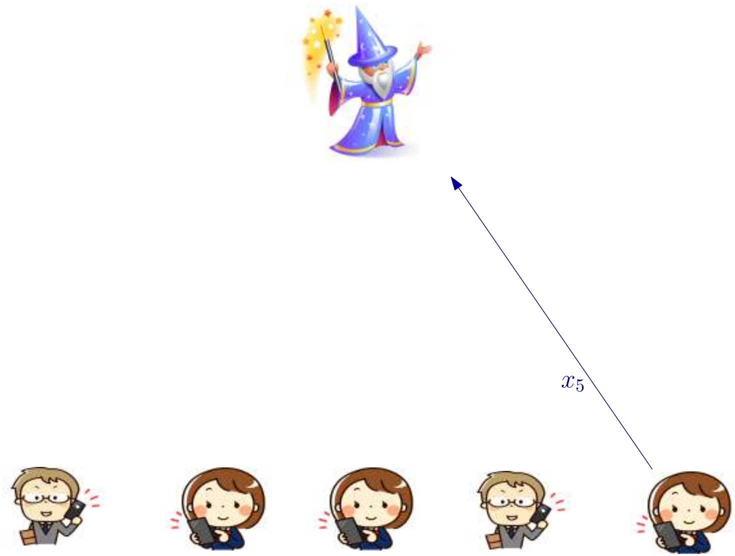
Motivation: delegation of computation aaS



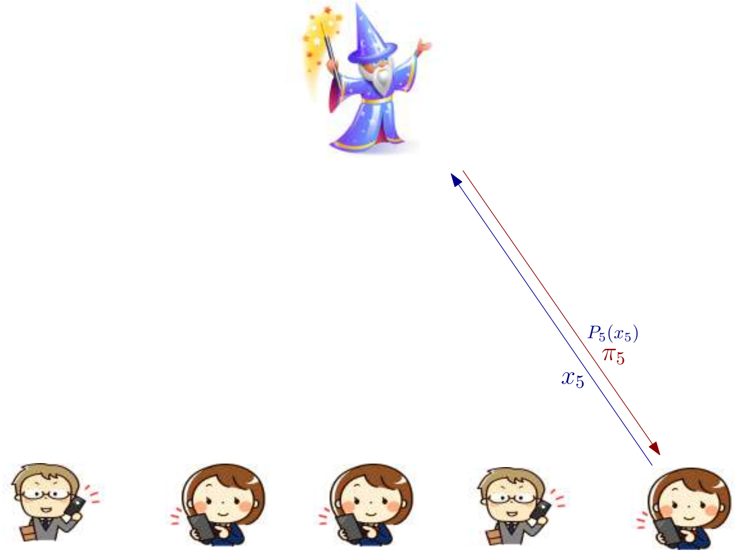
Motivation: delegation of computation aaS



Motivation: delegation of computation aaS



Motivation: delegation of computation aaS



Motivation: delegation of computation aaS

Much work for the prover...

Motivation: delegation of computation aaS

Much work for the prover...

- Compute $P_1(x_1), \ldots, P_5(x_5)$
- Many proofs...

Motivation: delegation of computation aaS

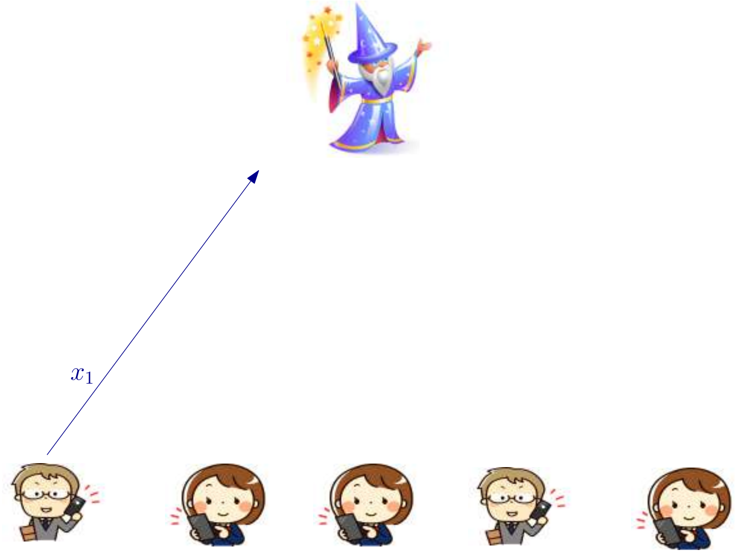
Much work for the prover...

- Compute $P_1(x_1), \ldots, P_5(x_5)$
- Many proofs...

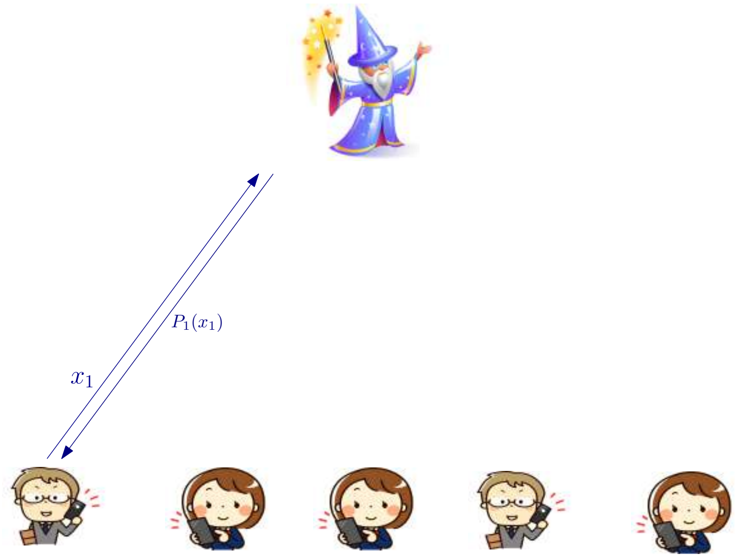
Idea: Prove everything at once!

Motivation: delegation of computation aaS

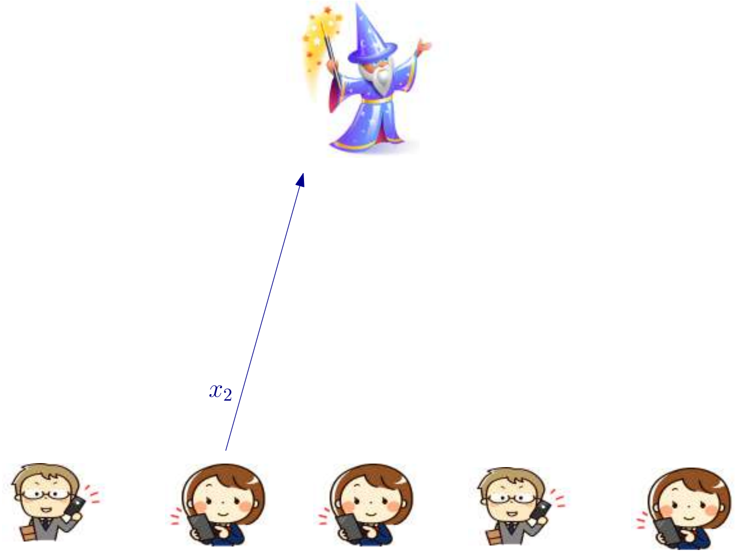
Motivation: delegation of computation aaS



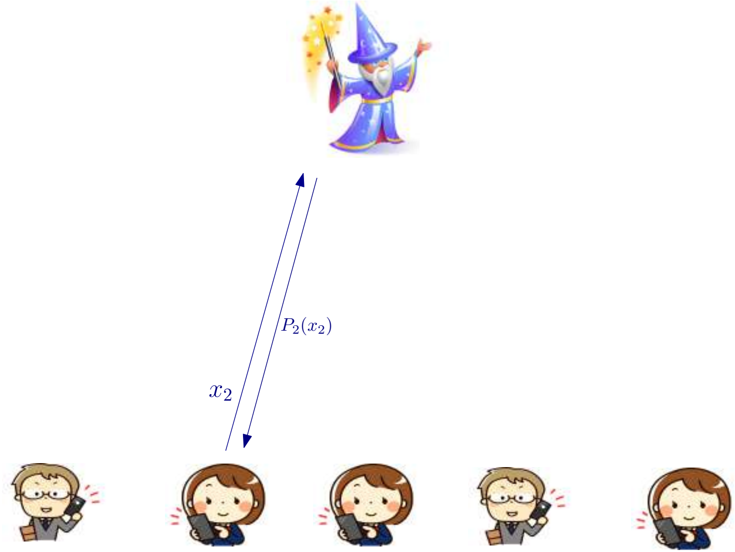
Motivation: delegation of computation aaS



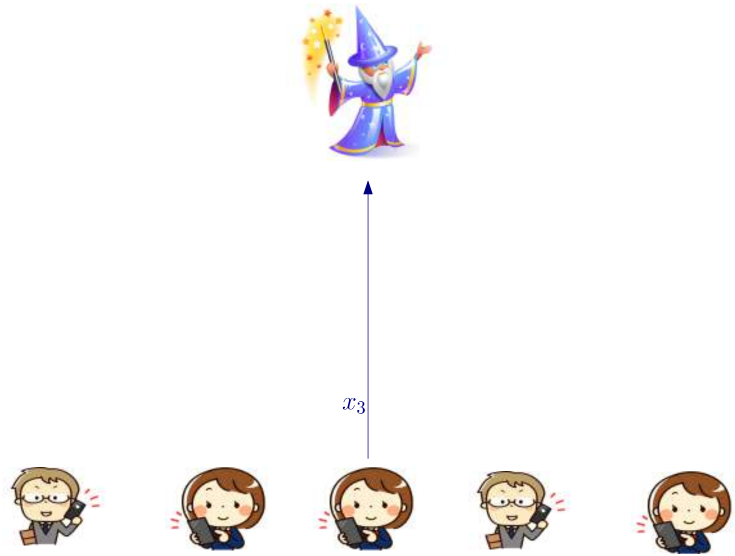
Motivation: delegation of computation aaS



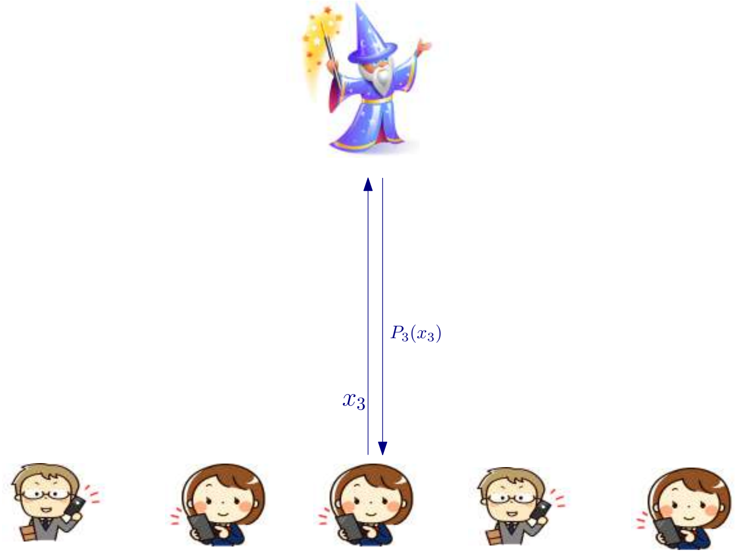
Motivation: delegation of computation aaS



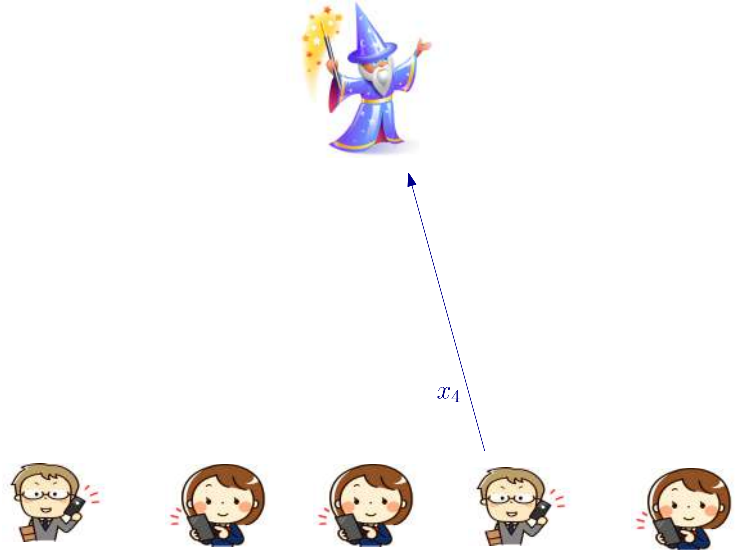
Motivation: delegation of computation aaS



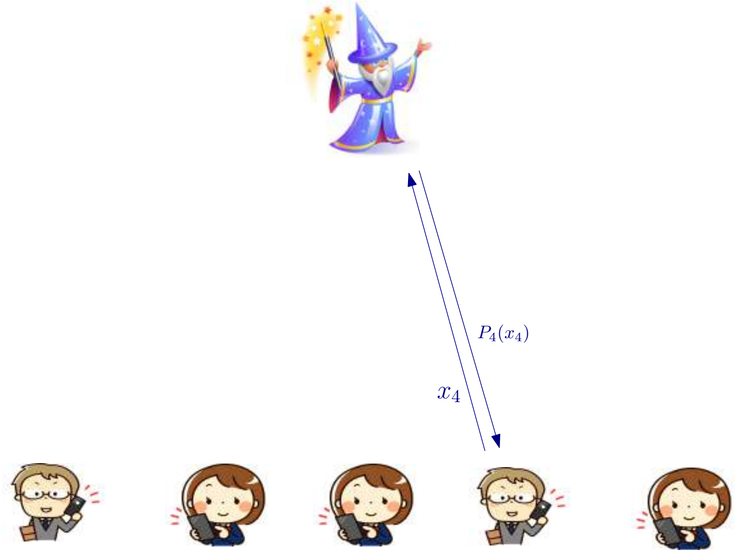
Motivation: delegation of computation aaS



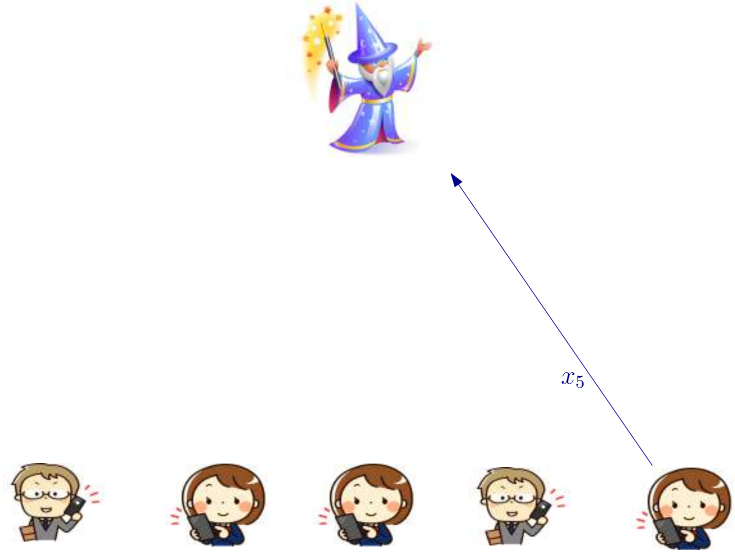
Motivation: delegation of computation aaS



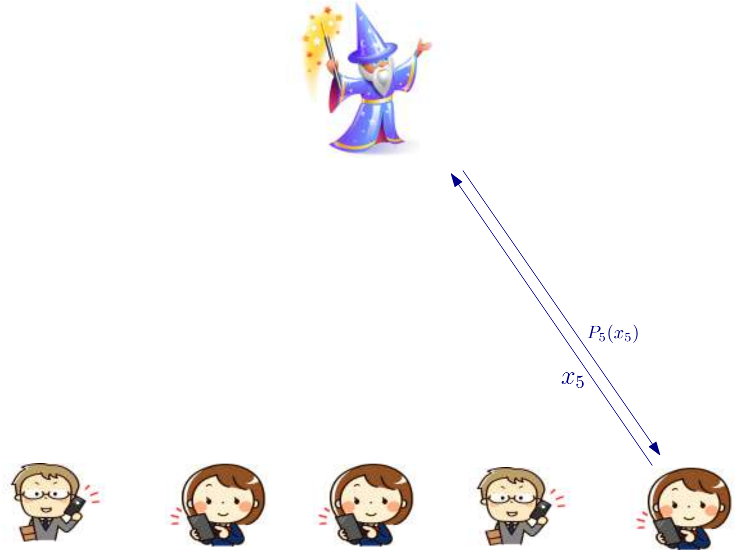
Motivation: delegation of computation aaS



Motivation: delegation of computation aaS



Motivation: delegation of computation aaS



Motivation: delegation of computation aaS



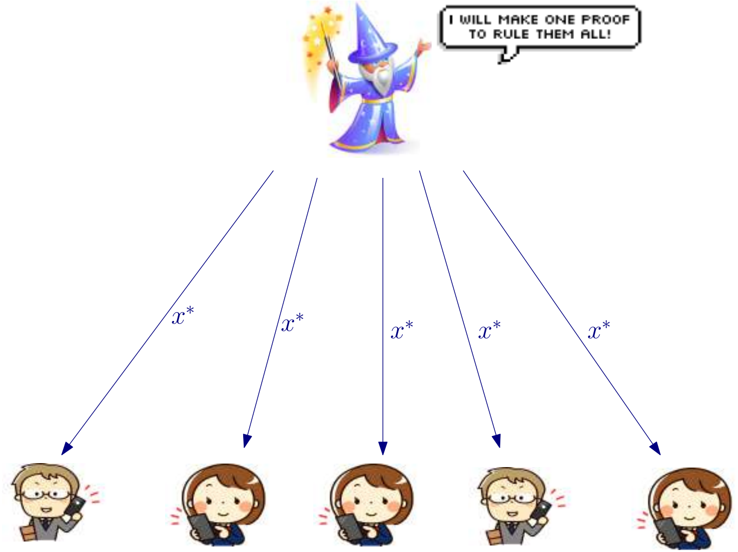


Motivation: delegation of computation aaS

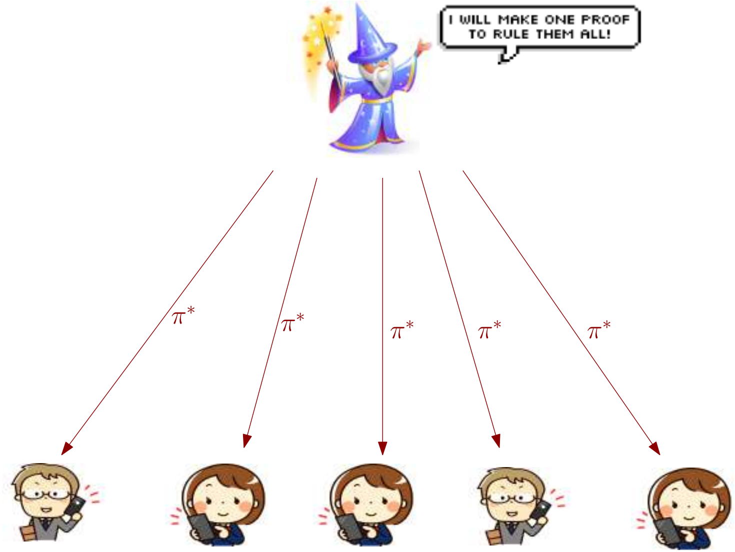




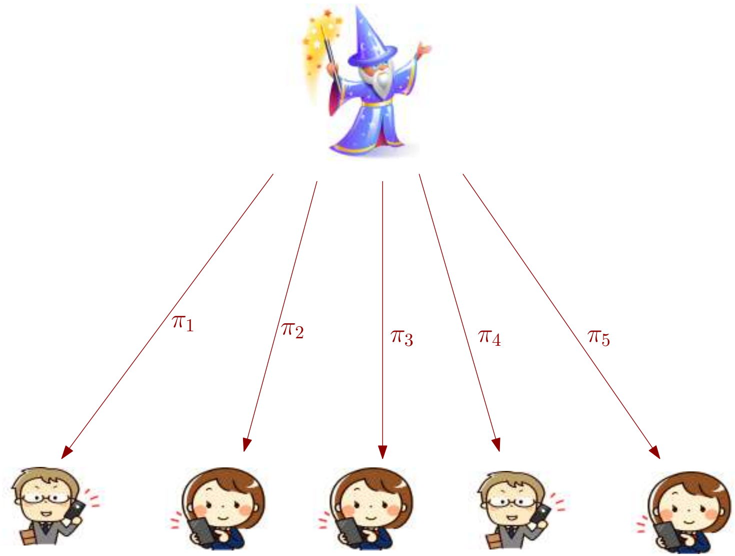
Motivation: delegation of computation aaS



Motivation: delegation of computation aaS



Motivation: delegation of computation aaS



Motivation: delegation of computation aaS

Requirements:

1. "Heavy" part done once
2. Cheap individual proofs

Motivation: delegation of computation aaS

Requirements:

1. "Heavy" part done once
2. Cheap individual proofs

✔ More statements $\Rightarrow$ cheaper prover

✔ All verifiers check the same proof π^*

Folding schemes

NP language $\mathcal{L}$ with corresponding relation $\mathcal{R}$.

Folding schemes

NP language $\mathcal{L}$ with corresponding relation $\mathcal{R}$.

- $\mathsf{Fold}(x_1, w_1, x_2, w_2) \rightarrow x, w, \pi_{\mathsf{Fold}}$
- $\mathsf{FoldVrfy}(x_1, x_2, x, \pi_{\mathsf{Fold}}) \rightarrow 0/1$

Folding schemes

NP language $\mathcal{L}$ with corresponding relation $\mathcal{R}$.

- $\mathsf{Fold}(x_1, w_1, x_2, w_2) \rightarrow x, w, \pi_{\mathsf{Fold}}$
- $\mathsf{FoldVrfy}(x_1, x_2, x, \pi_{\mathsf{Fold}}) \rightarrow 0/1$

Properties:

- **Completeness**: $(x_1, w_1), (x_2, w_2) \in \mathcal{R} \Rightarrow (x, w) \in \mathcal{R}$ & π_{Fold} verifies
- **Knowledge soundness**: valid π_{Fold} & $w \Rightarrow w_1, w_2$.

Folding schemes

NP language $\mathcal{L}$ with corresponding relation $\mathcal{R}$.

- $\mathsf{Fold}(x_1, w_1, x_2, w_2) \to x, w, \pi_{\mathsf{Fold}}$
- $\mathsf{FoldVrfy}(x_1, x_2, x, \pi_{\mathsf{Fold}}) \to 0/1$

Properties:

- **Completeness**: $(x_1, w_1), (x_2, w_2) \in \mathcal{R} \Rightarrow (x, w) \in \mathcal{R}$ & π_{Fold} verifies
- **Knowledge soundness**: valid π_{Fold} & $w \Rightarrow w_1, w_2$.

Extends to m statements/witness pairs

Folding schemes with *selective verification*

NP language $\mathcal{L}$ with corresponding relation $\mathcal{R}$.

Folding schemes with *selective verification*

NP language $\mathcal{L}$ with corresponding relation $\mathcal{R}$.

- $\mathsf{Fold}, \mathsf{FoldVrfy}$
- $\mathsf{SelProve}(x_1, \ldots, x_m, x, \pi_{\mathsf{Fold}}) \to \pi_1, \ldots, \pi_m$

  π_i asserts that x_i was included in aggregation
- $\mathsf{SelVerify}(x, i, x_i, \pi_i) \to 0/1$

Folding schemes with *selective verification*

NP language $\mathcal{L}$ with corresponding relation $\mathcal{R}$.

- Fold, FoldVrfy
- $\mathsf{SelProve}(x_1, \ldots, x_m, x, \pi_{\mathsf{Fold}}) \to \pi_1, \ldots, \pi_m$

  π_i asserts that x_i was included in aggregation
- $\mathsf{SelVerify}(x, i, x_i, \pi_i) \to 0/1$

Additional properties:

1. **Selective completeness**: honest proof π_i verifies
2. **Selective knowledge soundness**: valid π_i & $w \Rightarrow w_i$
3. **Efficiency**: π_i sublinear in m

Folding schemes with *selective verification*

NP language $\mathcal{L}$ with corresponding relation $\mathcal{R}$.

- $\mathsf{Fold}, \mathsf{FoldVrfy}$
- $\mathsf{SelProve}(x_1, \ldots, x_m, x, \pi_{\mathsf{Fold}}) \rightarrow \pi_1, \ldots, \pi_m$

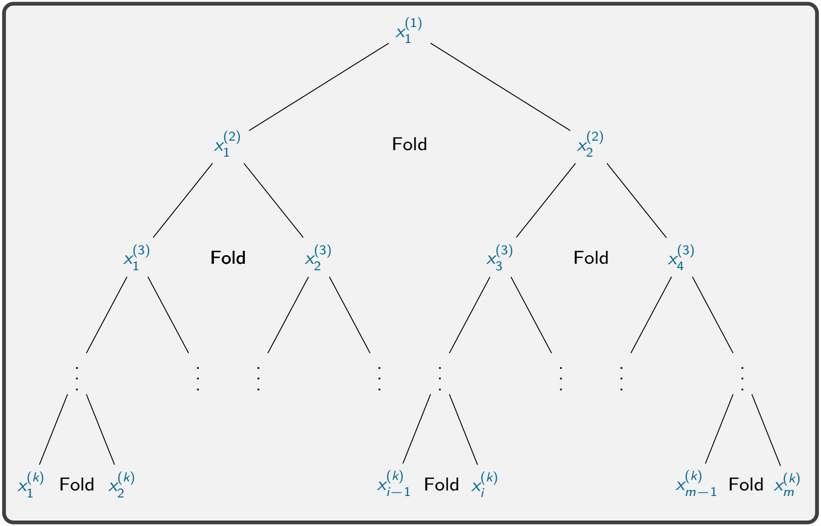
  π_i asserts that x_i was included in aggregation
- $\mathsf{SelVerify}(x, i, x_i, \pi_i) \rightarrow 0/1$

Additional properties:

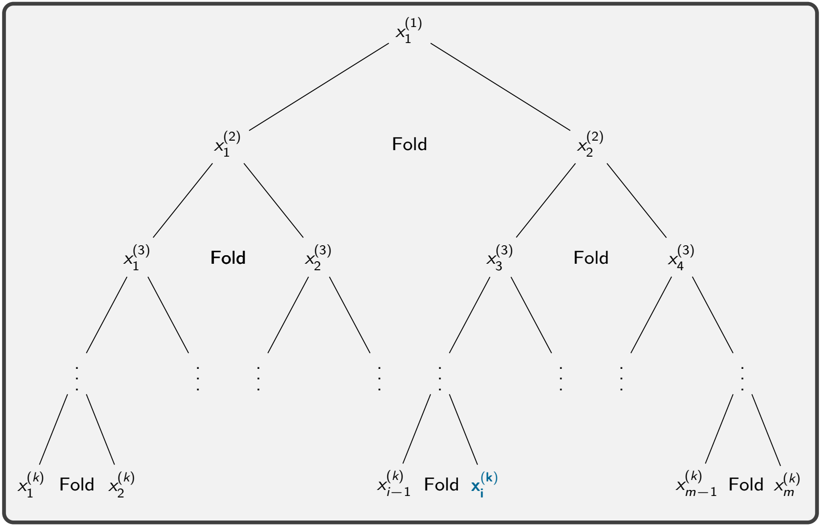
1. **Selective completeness**: honest proof π_i verifies
2. **Selective knowledge soundness**: valid π_i & $w \Rightarrow w_i$
3. **Efficiency**: π_i sublinear in m

Folding scheme $\Rightarrow$ folding scheme with selective verification

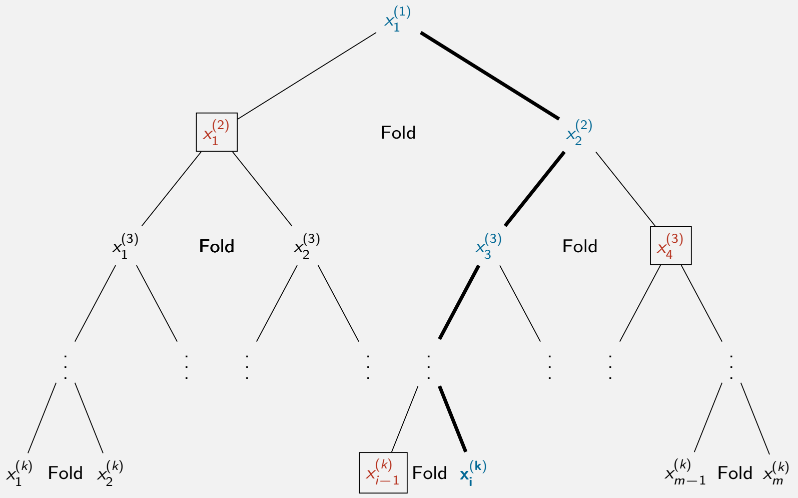
Statement aggregation tree



Statement aggregation tree



Statement aggregation tree



Give as *proof* the sibling statements & 2-folding proofs

Properties

- Prover: $\mathcal{O}(m)$ aggregations
- Verifier: $\mathcal{O}(\log m)$ verifications

Final statement

- Prove (e.g. NIZK)
- Aggregate

Notation

Implicit notation for groups

- Let $\mathbb{G}$ be a group and $\mathcal{P}$ a fixed generator.
- $[x]$ is the element $x\mathcal{P}$.

Implicit notation for groups

- Let $\mathbb{G}$ be a group and $\mathcal{P}$ a fixed generator.
- $[x]$ is the element $x\mathcal{P}$.

Example:

$$[1], [a], [b], [ab] \in \mathsf{DDH}$$

$$\equiv$$

$$\mathcal{P}, a\mathcal{P}, b\mathcal{P}, ab\mathcal{P} \in \mathsf{DDH}$$

Implicit notation for groups

- Let $\mathbb{G}$ be a group and $\mathcal{P}$ a fixed generator.
- $[x]$ is the element $x\mathcal{P}$.

Example:

$$[1], [a], [b], [ab] \in \mathrm{DDH}$$

$$\equiv$$

$$\mathcal{P}, a\mathcal{P}, b\mathcal{P}, ab\mathcal{P} \in \mathrm{DDH}$$

With this notation:

$$[\mathbf{r}] = ([r_1], \ldots, [r_n]), \qquad \mathbf{x} = (x_1, \ldots, x_n)\,,$$

$$[\mathbf{r}]^\top \mathbf{x} = \sum [r_i] x_i \quad (= x_1 r_1 \mathcal{P} + \cdots + x_n r_n \mathcal{P})$$

Algebraic commitments

Generalization of Pedersen commitments

- keygen(1^λ):

  sample $\mathbf{r} \in \mathbb{F}^n$ from some hard distribution

  output $[\mathbf{r}]$

- com($[\mathbf{r}], \mathbf{x}$):

  output $[c] = [\mathbf{r}]^\top \mathbf{x}$

- verify($[\mathbf{r}], [c], \mathbf{x}$):

  $[c] \stackrel{?}{=} [\mathbf{r}]^\top \mathbf{x}$

Folding VC through IP

Aggregation of vector commitment openings

Let's fold VC openings!

- Statement: $[c_i]$ opens to $x_{i_1}, \ldots, x_{i_k}$ at positions $i_1, \ldots, i_k$
- Witnees: opening $\mathbf{x}$

Aggregation of vector commitment openings

Let's fold VC openings!

- Statement: $[c_i]$ opens to $x_{i_1}, \ldots, x_{i_k}$ at positions $i_1, \ldots, i_k$
- Witnees: opening $\mathbf{x}$

1. reduce to IP: $([c], [d], z) : \exists \mathbf{a}, \mathbf{b}$ s.t $z = \mathbf{a}^\top \mathbf{b}$

Aggregation of vector commitment openings

Let's fold VC openings!

- Statement: $[c_i]$ opens to $x_{i_1}, \ldots, x_{i_k}$ at positions $i_1, \ldots, i_k$
- Witnees: opening $\mathbf{x}$

1. reduce to IP: $([c], [d], z) : \exists \mathbf{a}, \mathbf{b}$ s.t $z = \mathbf{a}^\top \mathbf{b}$

x_1	x_2	x_3	x_4	x_5

Aggregation of vector commitment openings

Let's fold VC openings!

- Statement: $[c_i]$ opens to $x_{i_1}, \ldots, x_{i_k}$ at positions $i_1, \ldots, i_k$
- Witnees: opening $\mathbf{x}$

1. reduce to IP: $([c], [d], z) : \exists \mathbf{a}, \mathbf{b}$ s.t $z = \mathbf{a}^\top \mathbf{b}$

x_1	x_2	x_3	x_4	x_5

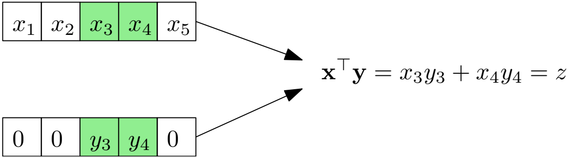
0	0	y_3	y_4	0

Aggregation of vector commitment openings

Let's fold VC openings!

- Statement: $[c_i]$ opens to $x_{i_1}, \ldots, x_{i_k}$ at positions $i_1, \ldots, i_k$
- Witnees: opening $\mathbf{x}$

1. reduce to IP: $([c], [d], z) : \exists \mathbf{a}, \mathbf{b}$ s.t $z = \mathbf{a}^\top \mathbf{b}$

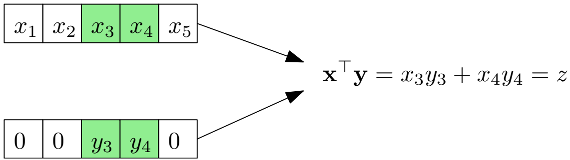


Aggregation of vector commitment openings

Let's fold VC openings!

- Statement: $[c_i]$ opens to $x_{i_1}, \ldots, x_{i_k}$ at positions $i_1, \ldots, i_k$
- Witnees: opening $\mathbf{x}$

1. reduce to IP: $([c], [d], z) : \exists \mathbf{a}, \mathbf{b}$ s.t $z = \mathbf{a}^\top \mathbf{b}$



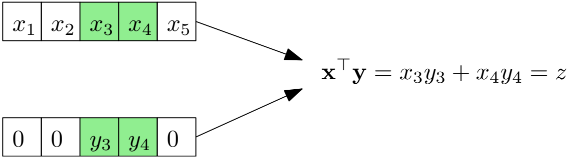
2. (Simple) folding scheme for IP

Aggregation of vector commitment openings

Let's fold VC openings!

- Statement: $[c_i]$ opens to $x_{i_1}, \ldots, x_{i_k}$ at positions $i_1, \ldots, i_k$
- Witnees: opening $\mathbf{x}$

1. reduce to IP: $([c], [d], z) : \exists \mathbf{a}, \mathbf{b}$ s.t $z = \mathbf{a}^\top \mathbf{b}$



2. (Simple) folding scheme for IP
3. Use bootstrapping

Folding scheme for IP

Claim:

- Statement: $([c_1], [d_1], z_1)$, $([c_2], [d_2], z_2) \in \mathrm{IP}$,
- Witness $(\mathbf{a}_1, \mathbf{b}_1), (\mathbf{a}_2, \mathbf{b}_2)$.

Folding scheme for IP

Claim:

- Statement: $([c_1], [d_1], z_1)$, $([c_2], [d_2], z_2) \in \mathrm{IP}$,
- Witness $(\mathbf{a}_1, \mathbf{b}_1), (\mathbf{a}_2, \mathbf{b}_2)$.

$\mathcal{P}$: $\mathbf{a}_1, \mathbf{b}_1, \mathbf{a}_2, \mathbf{b}_2$		$\mathcal{V}$: $[c_1], [d_1], z_1, [c_2], [d_2], z_2$
$z_{1,2} = \mathbf{a}_1{}^\top \mathbf{b}_2$		
$z_{2,1} = \mathbf{a}_2{}^\top \mathbf{b}_1$	$\xrightarrow{z_{1,2},\, z_{2,1}}$	
	$\xleftarrow{x}$	$x \leftarrow \mathbb{F}$
$\mathbf{a} = \mathbf{a}_1 + x\mathbf{a}_2$		$[c] = [c_1] + x[c_2]$
$\mathbf{b} = \mathbf{b}_1 + x^2\mathbf{b}_2$		$[d] = [d_1] + x^2[d_2]$
		$z = z_1 + x \cdot z_{2,1} + x^2 \cdot z_{1,2} + x^3 z_2$

Folding scheme for IP

Claim:

- Statement: $([c_1],[d_1],z_1),\ ([c_2],[d_2],z_2) \in \mathrm{IP}$,
- Witness $(\mathbf{a}_1,\mathbf{b}_1),(\mathbf{a}_2,\mathbf{b}_2)$.

$\mathcal{P}:\ \mathbf{a}_1,\mathbf{b}_1,\mathbf{a}_2,\mathbf{b}_2$		$\mathcal{V}:\ [c_1],[d_1],z_1,[c_2],[d_2],z_2$
$z_{1,2}=\mathbf{a}_1{}^\top\mathbf{b}_2$		
$z_{2,1}=\mathbf{a}_2{}^\top\mathbf{b}_1$	$\xrightarrow{z_{1,2},\,z_{2,1}}$	
	$\xleftarrow{x}$	$x\leftarrow\mathbb{F}$
$\mathbf{a}=\mathbf{a}_1+x\mathbf{a}_2$		$[c]=[c_1]+x[c_2]$
$\mathbf{b}=\mathbf{b}_1+x^2\mathbf{b}_2$		$[d]=[d_1]+x^2[d_2]$
		$z=z_1+x\cdot z_{2,1}+x^2\cdot z_{1,2}+x^3z_2$

✔ Extemely fast: $|\mathit{witness}|$ computations in $\mathbb{F}$!

VC opening

Proving m openings:

- One single NIZK for IP
- $\mathcal{O}(m)$ hash function[1] computations (FS)
- $\mathcal{O}(m)$ inner-products in $\mathbb{F}$ (comparable to reading the statement)

[1] No recursion circuit involved!

VC opening

Proving m openings:

- One single NIZK for IP
- $\mathcal{O}(m)$ hash function[1] computations (FS)
- $\mathcal{O}(m)$ inner-products in $\mathbb{F}$ (comparable to reading the statement)

Verification:

- $O(\log m)$ group operations
- $O(\log m)$ hash computations

[1] No recursion circuit involved!

Applications

Folding schemes for:

- Inner product relations
- Polynomial commitment opening
- Relaxed R1CS [NOVA]

Applications

Folding schemes for:

- Inner product relations
- Polynomial commitment opening
- Relaxed R1CS [NOVA]

Folding much cheaper than NIZK proof!

Applications

Folding schemes for:

- Inner product relations
- Polynomial commitment opening
- Relaxed R1CS [NOVA]

Folding much cheaper than NIZK proof!

Use cases:

- Aggregation of polynomial holographic proofs based SNARKs
- NOVA's style aggregation
- ...

Future work

- Applications? (Public verifiability vs aaS...)

Future work

- Applications? (Public verifiability vs aaS...)
- Privacy?

Future work

- Applications? (Public verifiability vs aaS...)
- Privacy?
- Statement Vector Commitments?

Future work

- Applications? (Public verifiability vs aaS...)
- Privacy?
- Statement Vector Commitments?
- Other relations *PLONK/AIR style NOVA*?

Thank You!