Lecture 11: Argument compilation and NIZK

Zero-knowledge proofs

263-4665-00L

Lecturer: Jonathan Bootle

Announcements

- Exam date 02/02/2023, 15:00-17:00.
- Continued office hours after New Year on 16/01, 23/01, 30/01.

- Guests visiting today’s lecture to provide feedback

- Extra (optional, non-examinable) video about reducing verification costs for polynomial commitments will be posted early next week.

Pedersen Multicommitments Eval Summary

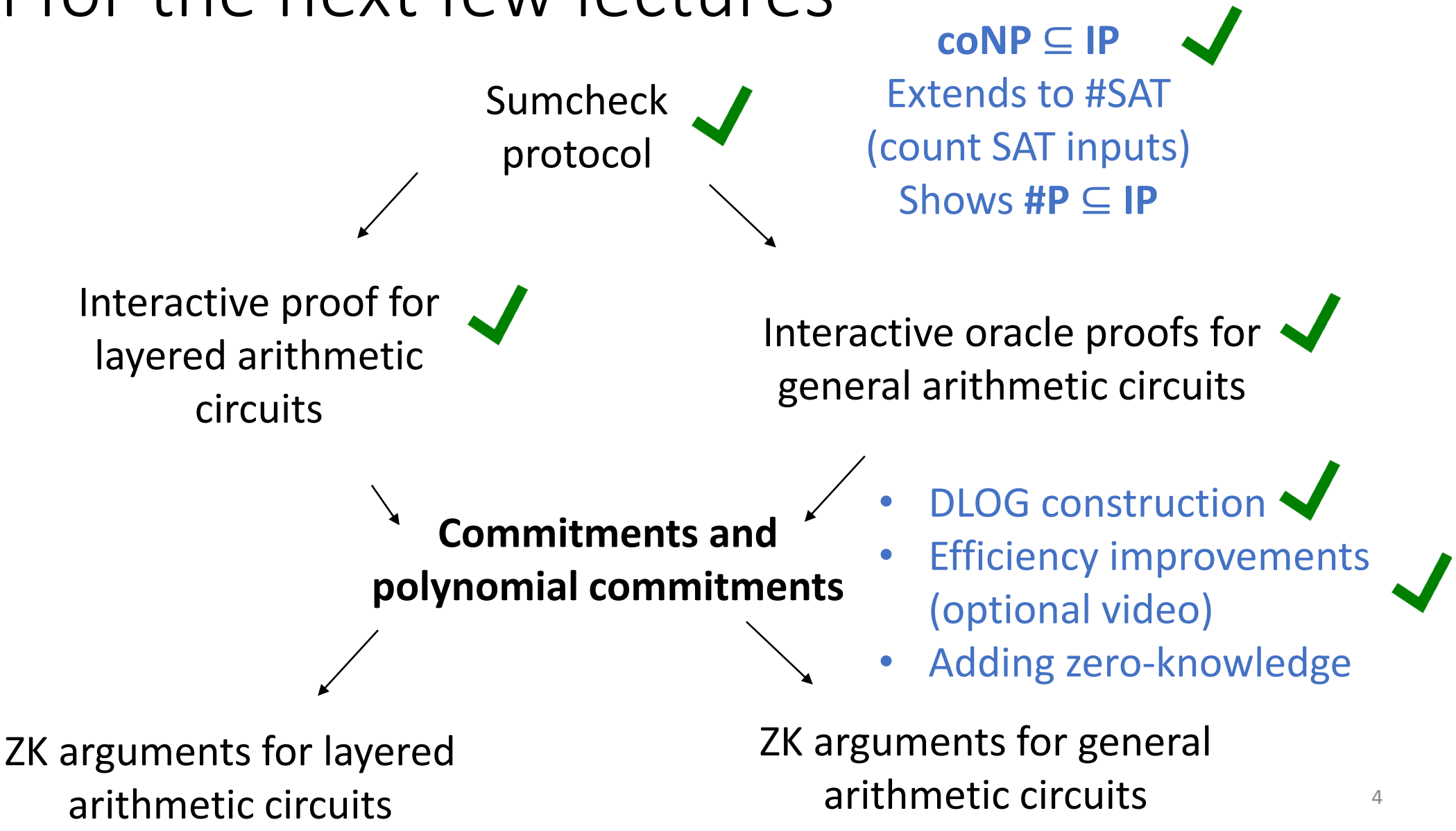
- Let $y_1, \dots, y_{\log N} \in \mathbb{Z}_p$ with $\tilde{f}(y_1, \dots, y_{\log N}) = z$.
- Let $\vec{Y} := \text{Expand}(\vec{y}) = \left(\widetilde{\text{EQ}}(\vec{y}; \vec{\imath})\right)_{\vec{\imath} \in \{0,1\}^{\log N}}$ satisfying $f(\vec{y}) = \langle \vec{f}, \vec{Y} \rangle$.

- Write $C = \langle \vec{f}, \vec{G} \rangle + r \cdot H$ and $z = \langle \vec{f}, \vec{Y} \rangle$.
- $\mathcal{R}_{PedPC}(pp) := \left\{ \left((C, z, y_1, \dots, y_\ell), \tilde{f} \right) : \begin{matrix} \tilde{f} \in \mathbb{F}^{\leq 1}[X_1, \dots, X_\ell] \\ z = \langle \vec{f}, \vec{Y} \rangle, C = \langle \vec{f}, \vec{G} \rangle \end{matrix} \right\}$

Set $r = 0$ initially
P can just send r to V.
Both remove $r \cdot H$ from C

- We constructed a protocol for Eval with prover complexity $O(N)$, proof size $O(\log N)$ and verifier complexity $O(N)$.

Plan for the next few lectures



Hiding polynomial evaluations

Evaluation relation for Pedersen multicommitments:

Given $y_1, \dots, y_{\log N}, z \in \mathbb{F}$, c, prove knowledge of openings $\{f_{\vec{i}}\}, r$ of C such that $\sum_{\vec{i} \in \{0,1\}^{log\ N}} f_{\vec{i}} \cdot \widetilde{\mathrm{EQ}}(\vec{y}; \vec{i}) = z$.

$$\mathcal{R}_{PedPC}(pp) := \left\{ \left((C, z, y_1, \dots, y_\ell), (\tilde{f}, r) \right) : \begin{array}{c} \tilde{f} \in \mathbb{F}^{\leq 1}[X_1, \dots, X_\ell] \\ \tilde{f}(y_1, \dots, y_\ell) = z, C = \langle \vec{f}, \vec{G} \rangle + r \cdot H \end{array} \right\}$$

$z, \vec{y}$ not hidden

Committed evaluation relation for Pedersen multicommitments:

Given $y_1, \dots, y_{\log N}, z \in \mathbb{F}$, c, prove knowledge of openings $\{f_{\vec{i}}\}, r$ of C such that $\sum_{\vec{i} \in \{0,1\}^{log\ N}} f_{\vec{i}} \cdot \widetilde{\mathrm{EQ}}(\vec{y}; \vec{i}) = z$.

pp for Pedersen multicommitments

pp' for plain Pedersen commitments

$$\mathcal{R}_{ComPedPC}(pp, pp') := \left\{ \left((C, C_z, y_1, \dots, y_\ell), (\tilde{f}, r, z, s) \right) : \begin{array}{c} \tilde{f} \in \mathbb{F}^{\leq 1}[X_1, \dots, X_\ell] \\ \tilde{f}(y_1, \dots, y_\ell) = z, C = \langle \vec{f}, \vec{G} \rangle + r \cdot H \\ C_z = z \cdot G + s \cdot H \end{array} \right\}$$

z now hidden

Committed evaluation protocol

Reduction completeness ⇓ Completeness

Reduction 2-soundness ⇓ $(2,4,\dots,4)$-soundness

Witness:

- vector $\vec{f} \in \mathbb{F}^N, r \in \mathbb{F}$
- $z, s \in \mathbb{F}$

Sample $\phi \leftarrow_{\$} \mathbb{F}^{\leq 1}[X_1, \dots, X_\ell]$ with coefficient vector $\vec{\phi} \in \mathbb{F}^N$.
Sample $\rho, \sigma \leftarrow_{\$} \mathbb{F}$.
Compute

$$\zeta = \phi(\vec{y})$$
$$D = \left\langle \vec{\phi}, \vec{G} \right\rangle + \rho \cdot H$$
$$D_\zeta = \zeta \cdot G + \sigma \cdot H$$

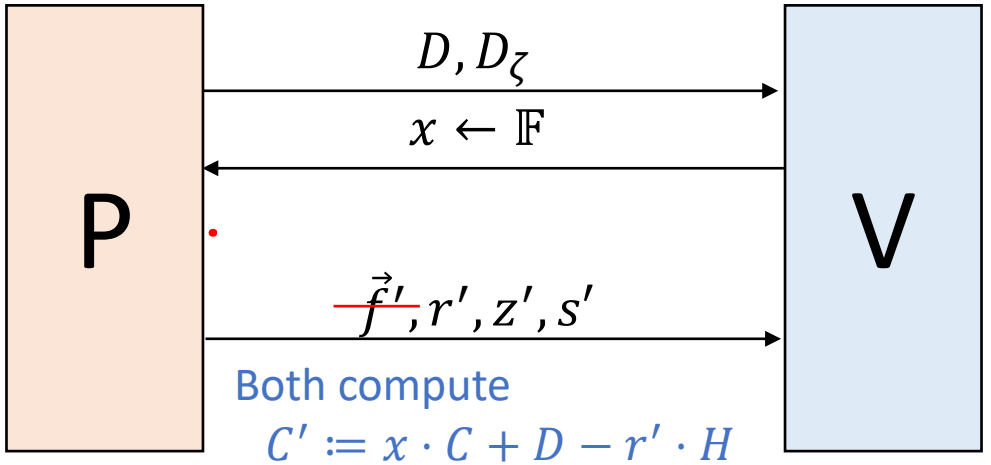
Compute $f' := xf + \phi$
$\vec{f}' := x\vec{f} + \vec{\phi}, z' := xz + \zeta$
$r' := xr + \rho, s' := xs + \sigma$

New witness:

$\vec{f}' := x\vec{f} + \vec{\phi} \in \mathbb{F}^N$

Instance:

- commitment $C \in \mathbb{G}$, key $\vec{G} \in \mathbb{G}^N$
- vector $\vec{Y} \in \mathbb{F}^N$, target $z \in \mathbb{F}$
- commitment $C_z \in \mathbb{G}$, key $G, H \in \mathbb{G}$



New instance:

- commitment $C' \in \mathbb{G}$, key $\vec{G} \in \mathbb{G}^N$
- vector $\vec{Y} \in \mathbb{F}^N$, target $z' \in \mathbb{F}$

Run previous Eval protocol

Language:

- $C = \left\langle \vec{f}, \vec{G} \right\rangle + r \cdot H \in \mathbb{G}$
- $z = \left\langle \vec{f}, \vec{Y} \right\rangle \in \mathbb{F}$
- $C_z = z \cdot G + s \cdot H$

Check
$x \cdot C + D == \left\langle \vec{f}', \vec{G} \right\rangle + r' \cdot H$
$f'(\vec{y}) == z'$
Check
$x \cdot C_z + D_\zeta == z' \cdot G + s' \cdot H$

New language:

- $C' = \left\langle \vec{f}', \vec{G} \right\rangle \in \mathbb{G}$
- $z' = \left\langle \vec{f}', \vec{Y} \right\rangle \in \mathbb{F}$

Committed evaluation protocol completeness

- $C = \left\langle \vec{f}, \vec{G} \right\rangle + r \cdot H, z = f(\vec{y}) = \left\langle \vec{f}, \vec{Y} \right\rangle, C_z = z \cdot G + s \cdot H.$ Left-multiply by x
- $D = \left\langle \vec{\phi}, \vec{G} \right\rangle + \rho \cdot H, \zeta = \phi(\vec{y}) = \left\langle \vec{\phi}, \vec{Y} \right\rangle, D_\zeta = \zeta \cdot G + \sigma \cdot H.$

Add

- $C' = \left\langle \vec{f}', \vec{G} \right\rangle + r' \cdot H, z' = f'(\vec{y}) = \left\langle \vec{f}', \vec{Y} \right\rangle, C'_z = z' \cdot G + s' \cdot H.$
- In the optimized version using Eval, we have reduced to a true instance.
- Completeness follows from the completeness of the previous Eval protocol.

SHVZK analysis

Inefficient protocol

$S(pp, pp', C, C_z, p, x)$

1. $\vec{f}' \leftarrow_{\$} \mathbb{F}^N$. $z' := f(\vec{y})$.
2. $r', s' \leftarrow_{\$} \mathbb{F}$.
3. $D = x \cdot C - \left\langle \vec{f}', \vec{G} \right\rangle - r' \cdot H$.
4. $D_\zeta = x \cdot C_z - z' \cdot G - s' \cdot H$.
5. Output $\left(D, D_\zeta, x, \vec{f}', r', z', s'\right)$.

What is the verifier's view?

- $\left(D, D_\zeta, x, \vec{f}', r', z', s'\right)$ with
- $x \cdot C + D = \left\langle \vec{f}', \vec{G} \right\rangle + r' \cdot H$, $f'(\vec{y}) = z'$ and $x \cdot C_z + D_\zeta = z' \cdot G + s' \cdot H$.
- $\rho \leftarrow_{\$} \mathbb{F}$ so $r' = xr + \rho$ is uniform in $\mathbb{F}$. Similarly for s'.
- $D = x \cdot C - \left\langle \vec{f}', \vec{G} \right\rangle - r' \cdot H$ is uniquely determined. Similarly for D_ζ.

Why is the simulator valid? (efficient, indistinguishable)

- Clearly, the simulator is efficient.
- $\vec{f}', z', r', s'$ have identical distributions to the real protocol.
- The other values are uniquely determined by the verifier checks.

SHVZK analysis

Efficient protocol

Eval protocol transcript

$S(pp, pp', C, C_z, p, x, x_1, \dots, x_\ell)$

1. $\vec{f}' \leftarrow_\$ \mathbb{F}^N$. $z' := f(\vec{y})$. $r', s' \leftarrow_\$ \mathbb{F}$.
2. $D = x \cdot C - \left\langle \vec{f}', \vec{G} \right\rangle - r' \cdot H$
3. $D_\zeta = x \cdot C_z - z' \cdot G - s' \cdot H$
4. Get transcript π by running the honest prover algorithm for Eval on the new instance.
5. Output $(D, D_\zeta, x, r', z', s', \pi)$.

What is the verifier's view?

- $\left(D, D_\zeta, x, \vec{f}', r', z', s', \pi\right)$ with
- $x \cdot C + D = \left\langle \vec{f}', \vec{G} \right\rangle + r' \cdot H$, $f'(\vec{y}) = z'$ and $x \cdot C_z + D_\zeta = z' \cdot G + s' \cdot H$.
- $\rho \leftarrow_\$ \mathbb{F}$ so $r' = xr + \rho$ is uniform in $\mathbb{F}$. Similarly for s'.
- $D = x \cdot C - \left\langle \vec{f}', \vec{G} \right\rangle - r' \cdot H$ is uniquely determined. Similarly for D_ζ.

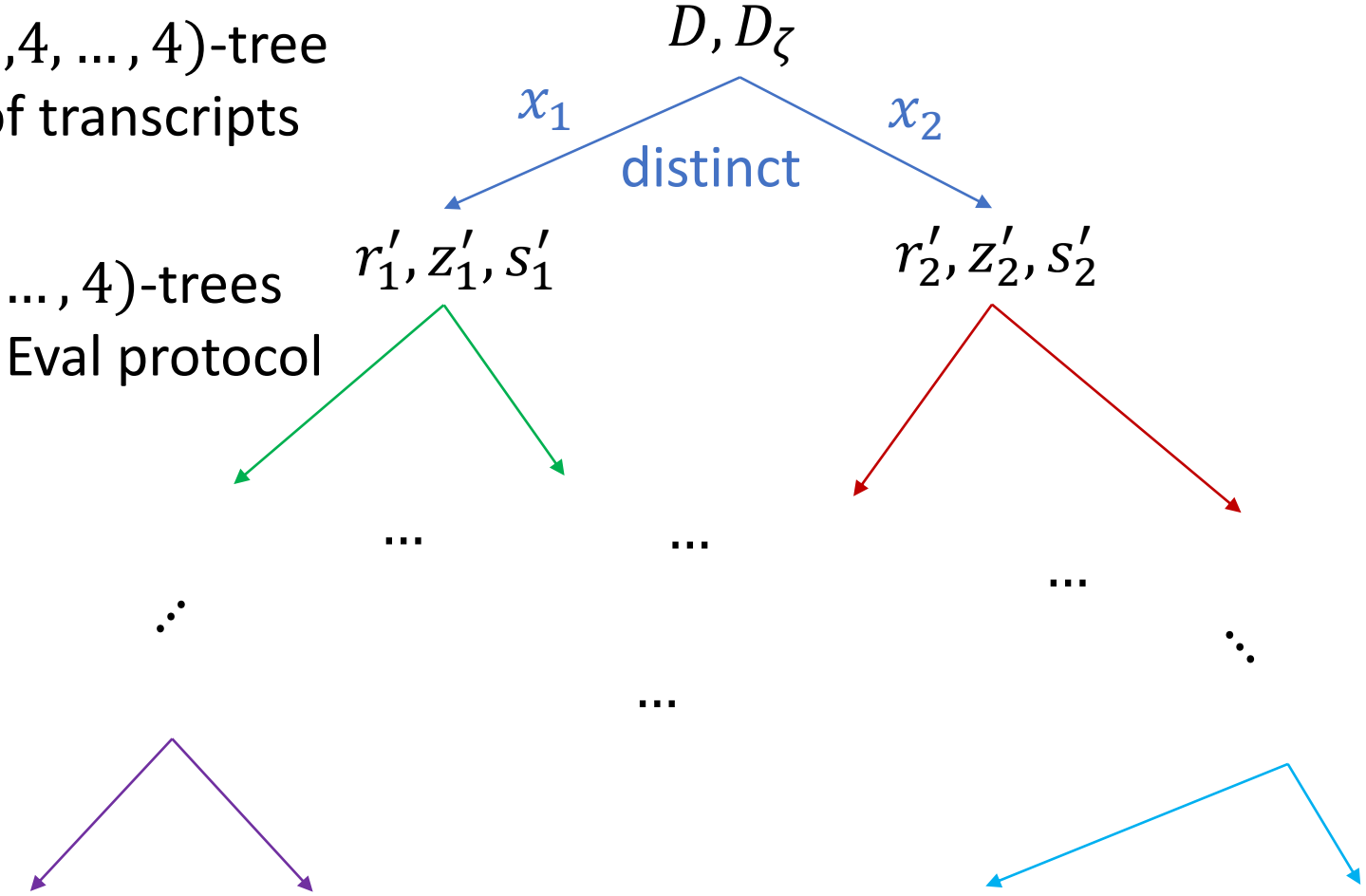
Why is the simulator valid? (efficient, indistinguishable)

- Clearly, the simulator is efficient.
- $\vec{f}', z', r', s'$ have identical distributions to the real protocol.
- The other values are uniquely determined by the verifier checks.
- The Eval prover algorithm is run on the same input distribution in both simulated and real protocol executions.

(2,4,..,4)-soundness from reduction 2-soundness

$(2,4,\ldots,4)$-tree of transcripts

$(4,\ldots,4)$-trees for Eval protocol x2



Witnesses for new instance

$$\langle \vec{f}'_1, \vec{G} \rangle + r'_1 \cdot H = x_1 \cdot C + D$$
$$\langle \vec{f}'_2, \vec{G} \rangle + r'_2 \cdot H = x_2 \cdot C + D$$

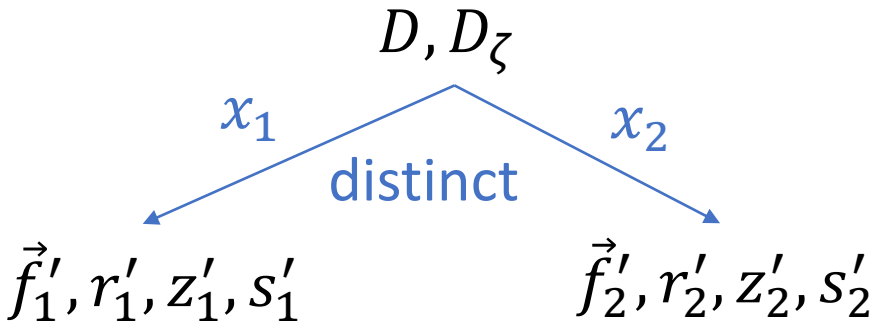
$$C'_1 := x_1 \cdot C + D - r'_1 \cdot H$$
$$C'_2 := x_2 \cdot C + D - r'_2 \cdot H$$

Eval protocol extractor produces $\vec{f}'_1$, $\vec{f}'_2$ such that:

$$C'_1 = \langle \vec{f}'_1, \vec{G} \rangle \qquad C'_2 = \langle \vec{f}'_2, \vec{G} \rangle$$
$$z'_2 = \langle \vec{f}'_1, \vec{Y} \rangle \qquad z'_2 = \langle \vec{f}'_2, \vec{Y} \rangle$$

2-soundness analysis of reduction

2-tree of transcripts



Satisfying

$$\langle \vec{f}_1', \vec{G} \rangle + r_1' \cdot H = x_1 \cdot C + D$$
$$\langle \vec{f}_2', \vec{G} \rangle + r_2' \cdot H = x_2 \cdot C + D$$
$$z_1' = \langle \vec{f}_1', \vec{Y} \rangle$$
$$z_2' = \langle \vec{f}_2', \vec{Y} \rangle$$
$$x_1 \cdot C_z + D_\zeta = z_1' \cdot G + s_1' \cdot H$$
$$x_2 \cdot C_z + D_\zeta = z_2' \cdot G + s_2' \cdot H$$

$$\begin{pmatrix} 1 & x_1 \\ 1 & x_2 \end{pmatrix} \begin{pmatrix} D \\ C \end{pmatrix} = \begin{pmatrix} \vec{f}_1' \\ \vec{f}_2' \end{pmatrix} \cdot \vec{G} + \begin{pmatrix} r_1' \\ r_2' \end{pmatrix} \cdot H$$

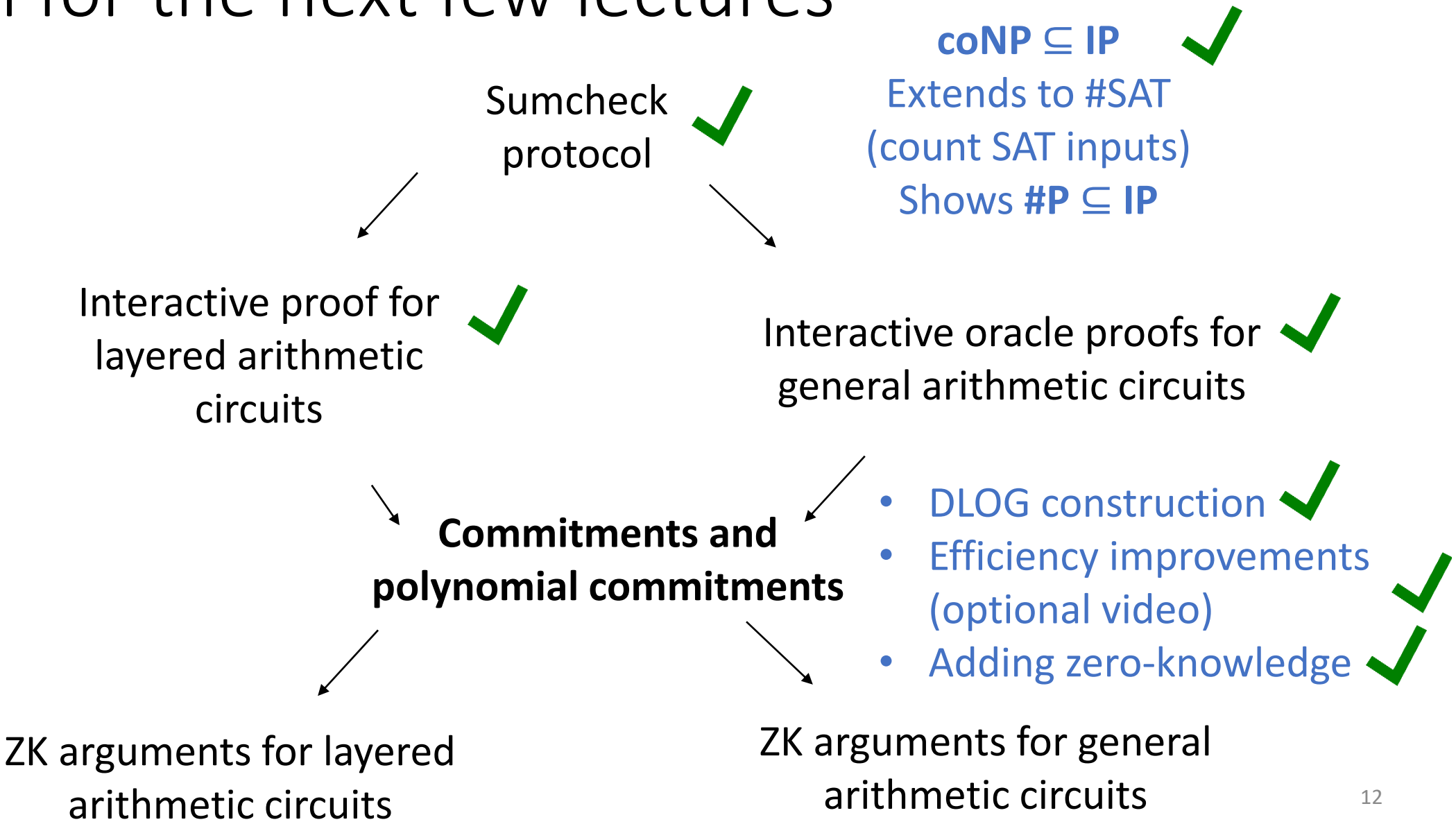
$$\begin{pmatrix} 1 & x_1 \\ 1 & x_2 \end{pmatrix} \begin{pmatrix} D_\zeta \\ C_z \end{pmatrix} = \begin{pmatrix} z_1' \\ z_2' \end{pmatrix} \cdot G + \begin{pmatrix} s_1' \\ s_2' \end{pmatrix} \cdot H = \begin{pmatrix} \vec{f}_1' \\ \vec{f}_2' \end{pmatrix} \cdot (\vec{Y} \cdot G) + \begin{pmatrix} s_1' \\ s_2' \end{pmatrix} \cdot H$$

Inverting the linear system, we get $\vec{f}, \vec{\phi}, r, s, \rho, \sigma$ satisfying

$$\begin{pmatrix} D \\ C \end{pmatrix} = \begin{pmatrix} \vec{\phi} \\ \vec{f} \end{pmatrix} \cdot \vec{G} + \begin{pmatrix} \rho \\ r \end{pmatrix} \cdot H,$$

$$\begin{pmatrix} D_\zeta \\ C_z \end{pmatrix} = \begin{pmatrix} \vec{\phi} \\ \vec{f} \end{pmatrix} \cdot (\vec{Y} \cdot G) + \begin{pmatrix} \sigma \\ s \end{pmatrix} \cdot H = \begin{pmatrix} \langle \vec{\phi}, \vec{Y} \rangle \\ \langle \vec{f}, \vec{Y} \rangle \end{pmatrix} \cdot G + \begin{pmatrix} \sigma \\ s \end{pmatrix} \cdot H$$
$$= f(\vec{y})$$

Plan for the next few lectures



How to make protocols succinct and ZK

Succinctness:

- Replace each large/oracle message with a polynomial commitment.
- Whenever V would have made a polynomial evaluation query to perform a check, P sends the evaluation, and they run the Eval protocol together.

Zero-knowledge:

- Also commit to each small message with a plain Pedersen commitment.
- Whenever V would have made a polynomial evaluation query to perform a check, P sends a commitment to the evaluation, and they run the hidden Eval protocol together.
- P and V run Σ-protocols on committed values to check that each verification equation would have been satisfied.

Compiling GKR to a ZK argument

- GKR language: $\left\{\left(\{add_i, mult_i\}_{i=0}^{D-1}, \vec{x}, \vec{y}\right) : C(\vec{x}) = \vec{y}\right\}$. **P** language

- Compiled GKR relation:

Easily generalizes to different lengths

$$\left\{\left(\left(\{add_i, mult_i\}_{i=0}^{D-1}, C_{\vec{x}}, C_{\vec{y}}\right), (\vec{x}, r, \vec{y}, s)\right) : \begin{array}{c} C_{\vec{x}} = \langle \vec{x}, \vec{G} \rangle + r \cdot H \\ C_{\vec{y}} = \langle \vec{y}, \vec{G} \rangle + s \cdot H \\ C(\vec{x}) = \vec{y} \end{array}\right\}.$$

Instance: (polynomial) commitments

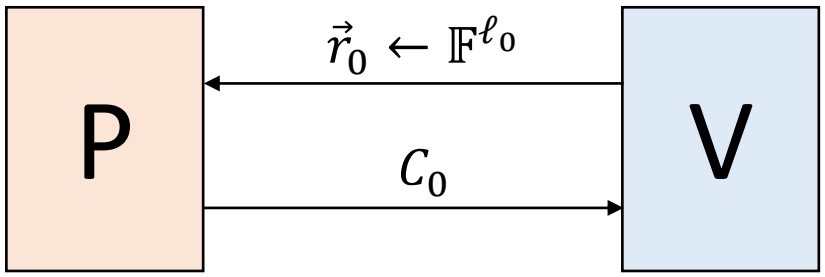
Witness

NP relation

Example: initial reduction

w_0 computed from $\vec{x}$

Initial claim: $C_{\vec{y}}$ contains $\vec{y}$ satisfying $\widetilde{w}_0 \equiv \tilde{y}$



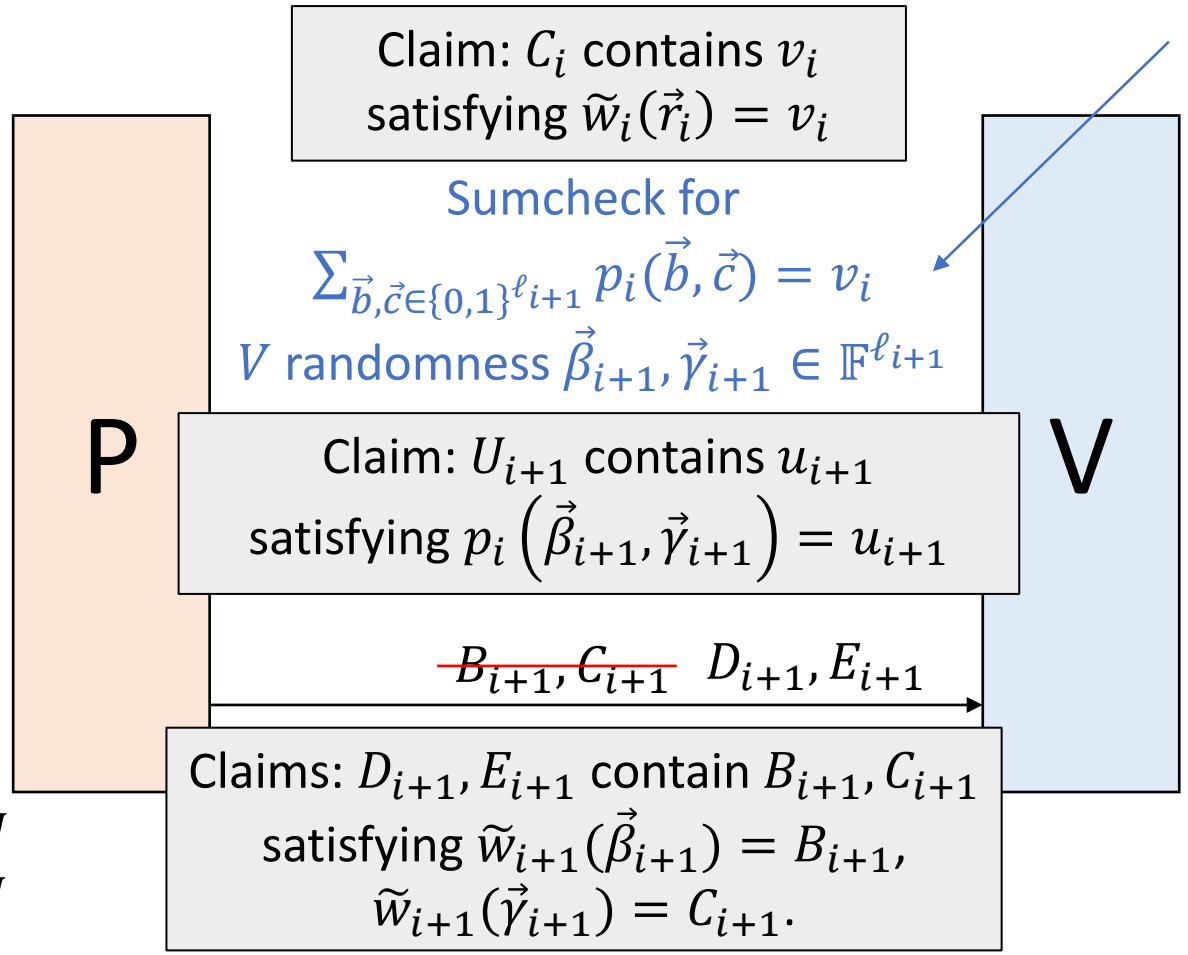
Compute $v_0 := \tilde{y}(\vec{r}_0)$.
Sample $s \leftarrow_{\$} \mathbb{F}$.
Compute $C_0 = v_0 \cdot G + s \cdot H$.

~~Both compute $v_0 := \tilde{y}(\vec{r}_0)$~~

Committed eval protocol proves: $C_{\vec{y}}, C_0$ contain $\vec{y}, v_0$ satisfying $\tilde{y}(\vec{r}_0) = v_0$

Claim: C_0 contains v_0 satisfying $\widetilde{w}_0(\vec{r}_0) = v_0$

Example: sumcheck level i to level i+1 reduction



- P sends *commitments* to sumcheck messages.
- V uses Σ-protocols to perform sumcheck checks but on committed values.
- V computes commitment U_{i+1} to final target evaluation using homomorphism.

Sample $r_i, s_i \leftarrow \mathbb{F}$.

$B_{i+1} := \widetilde{w}_{i+1}(\vec{\beta}_{i+1})$
$C_{i+1} := \widetilde{w}_{i+1}(\vec{\gamma}_{i+1})$

$D_{i+1} = B_{i+1} \cdot G + r_i \cdot H$
$E_{i+1} = C_{i+1} \cdot G + s_i \cdot H$

Both P, V run a Σ-protocol to check that $D_{i+1}, E_{i+1}, U_{i+1}$ contain $B_{i+1}, C_{i+1}, u_{i+1}$ satisfying

$$\widetilde{\mathrm{add}}_i\left(\vec{r}_i, \vec{\beta}_{i+1}, \vec{\gamma}_{i+1}\right)[B_{i+1} + C_{i+1}] + \widetilde{\mathrm{mult}}_i\left(\vec{r}_i, \vec{\beta}_{i+1}, \vec{\gamma}_{i+1}\right)[B_{i+1} \cdot C_{i+1}] == u_{i+1}$$

Compiling R1CS IOP to a ZK argument

$$\mathcal{R}_{R1CS} = \left\{ \left((\mathbb{F}, A, B, C, \vec{x}) , \vec{w} \right) : \begin{array}{c} A, B, C \in \mathbb{F}^{N_r \times N_c}, \vec{x} \in \mathbb{F}^k \\ \vec{w} \in \mathbb{F}^{N_c - k}, \vec{z} := \vec{x} || \vec{w} \\ A\vec{z} \circ B\vec{z} = C\vec{z} \end{array} \right\}.$$

NP relation

- Compiled R1CS relation:

$$\left\{ \left((\mathbb{F}, A, B, C, \vec{x}, C_{\vec{w}}) , (\vec{w}, r) \right) : \begin{array}{c} A, B, C \in \mathbb{F}^{N \times N}, \vec{x} \in \mathbb{F}^k \\ C_{\vec{w}} = \langle \vec{w}, \vec{G} \rangle + r \cdot H \\ \vec{w} \in \mathbb{F}^{N-k}, \vec{z} := \vec{x} || \vec{w} \\ A\vec{z} \circ B\vec{z} = C\vec{z} \end{array} \right\}.$$

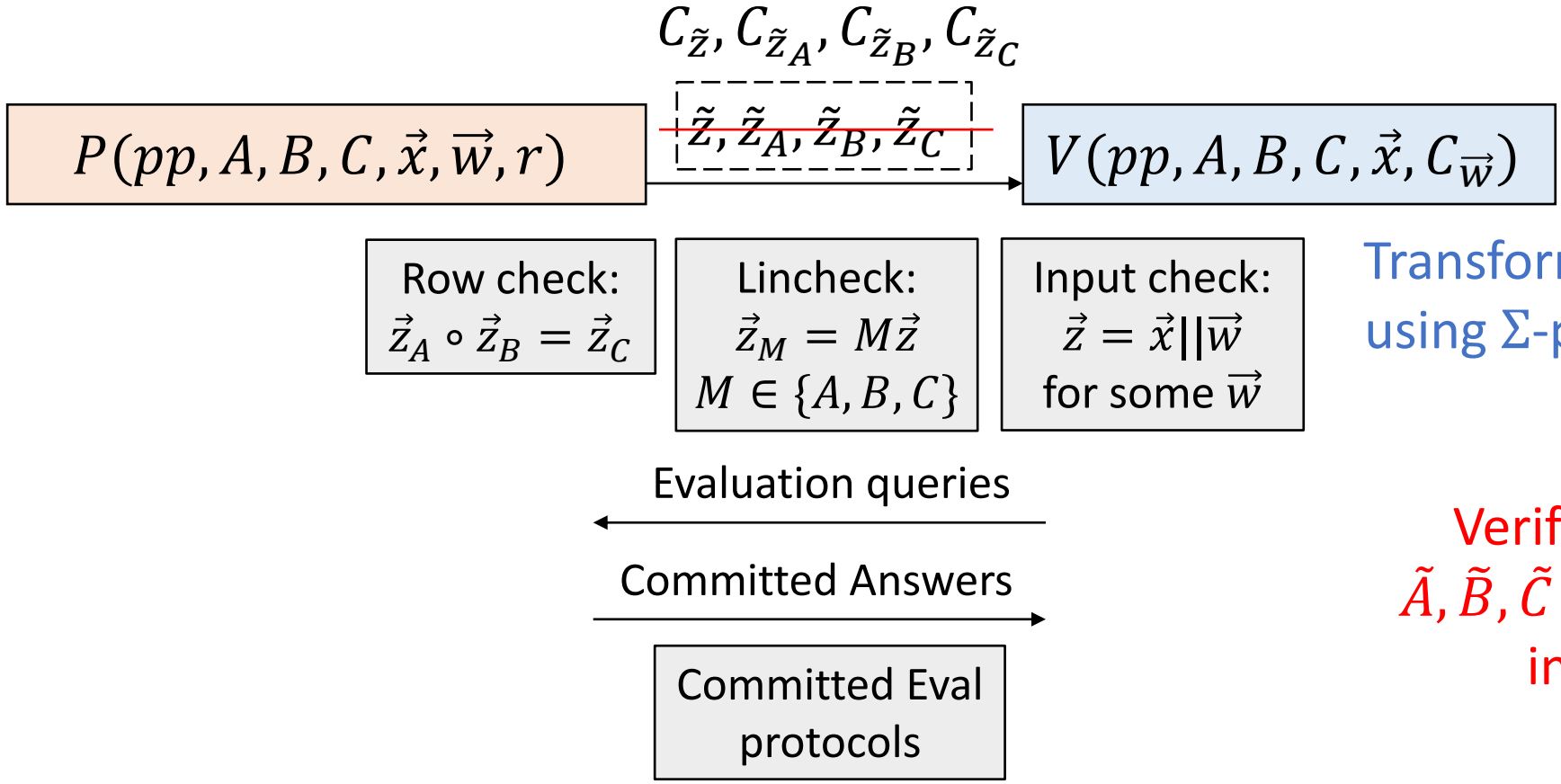
Instance

Witness

Alternatively, send $C_{\vec{w}}$ in first message

NP relation

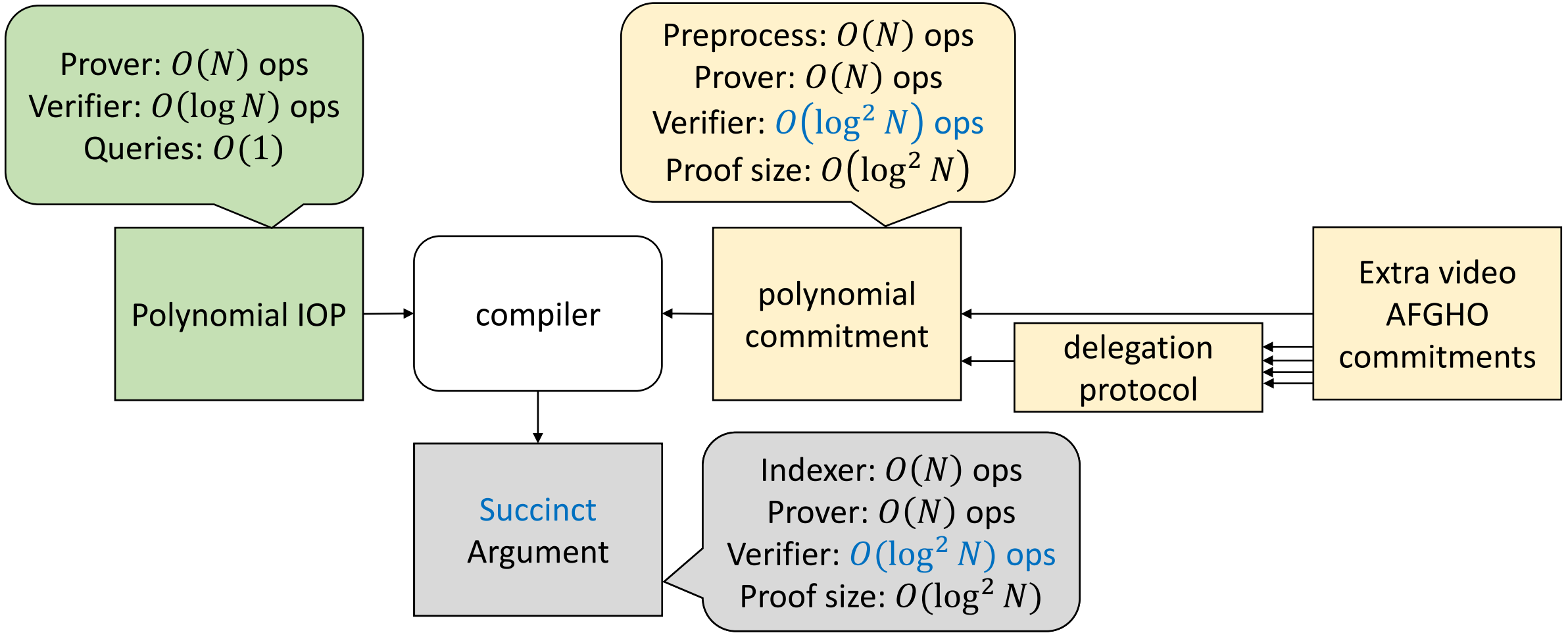
ZK argument for R1CS



Transform verifier checks using Σ-protocols as with GKR

Verifier evaluates $\tilde{A}, \tilde{B}, \tilde{C}$ for themselves in Lincheck

Summary of CSAT argument result



Security sketch

Completeness:

- Inherited from the underlying IP/IOP, hidden Eval and Σ-protocols

SHVZK:

- Sample uniformly random commitments C and C_z for each polynomial f and evaluation z.
- Sample uniformly random commitments for each small message.
- Simulate the Σ-protocols and committed Eval protocols.

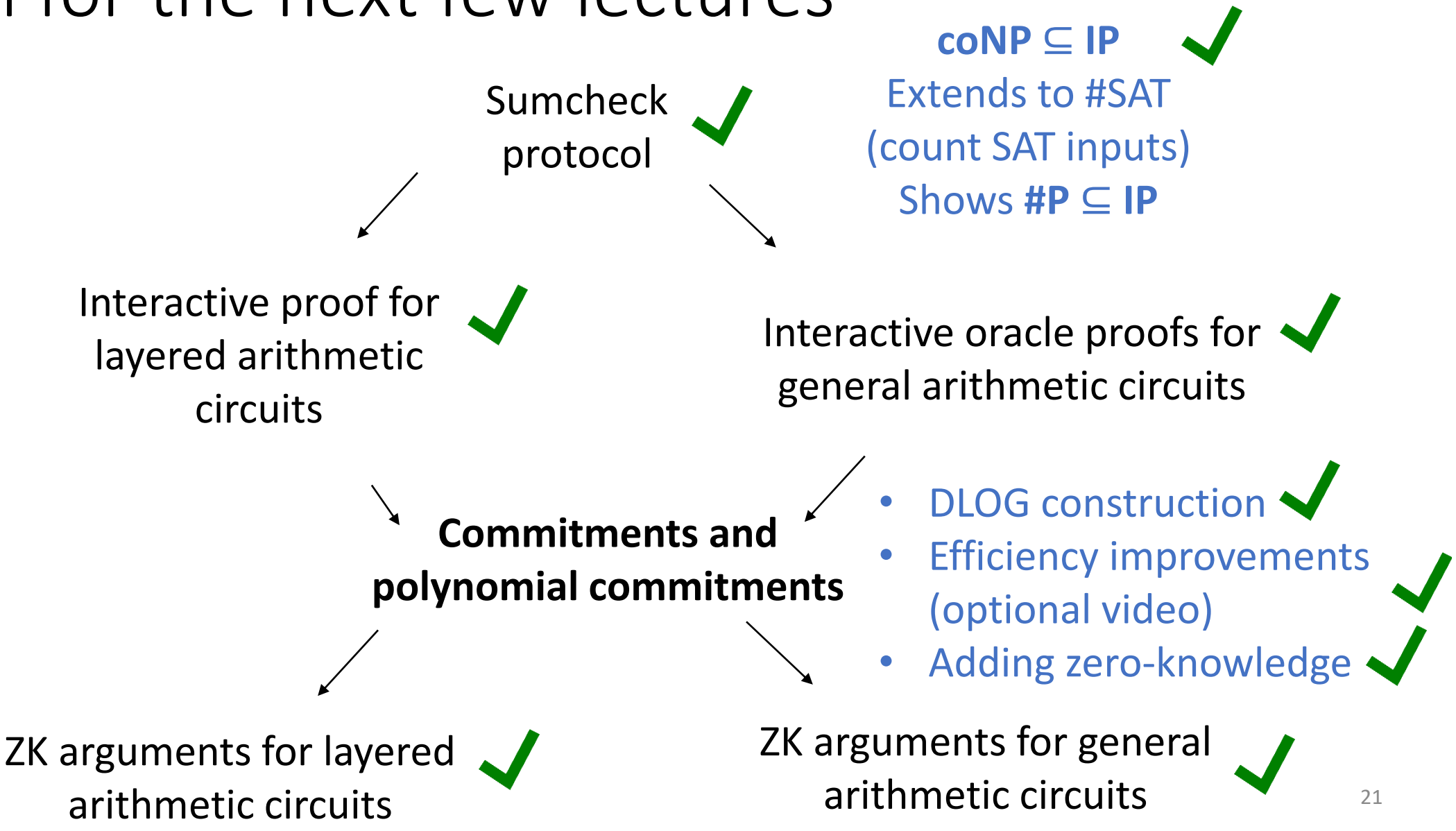
Soundness:

ZK argument adversary → (Hidden eval, Σ-protocol extractors) → IP/IOP argument adversary

Knowledge error bounds in terms of

- IOP/IP soundness error
- Comitted Eval knowledge error
- Σ-protocol knowledge error

Plan for the next few lectures



Course Outline (13 lectures)

1. **Introduction and definitions** ~2 lectures

2. **Sigma protocols** ~3 lectures

3. **ZK arguments with short proofs** ~4 lectures

4. **Non-interactive zero-knowledge** ~3 lectures

5. **Bonus material?** ~1 lecture

Non-interactive zero-knowledge

- Non-interactive zero-knowledge (NIZK) definitions

Pairing-based constructions of NIZK

- From reasonable cryptographic assumptions

$O(N)$ proof size for Boolean circuits

- From strong cryptographic assumptions

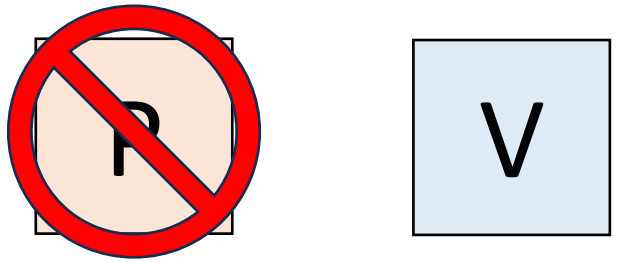
$O(1)$ proof size for Arithmetic circuits

Non-interactive proofs

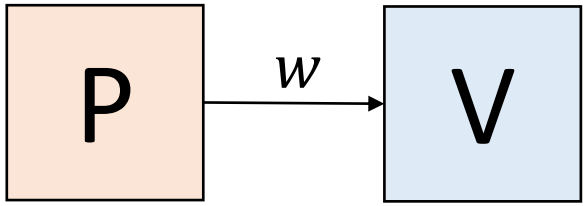
- A non-interactive proof consists of a single message from P to V.

Examples:

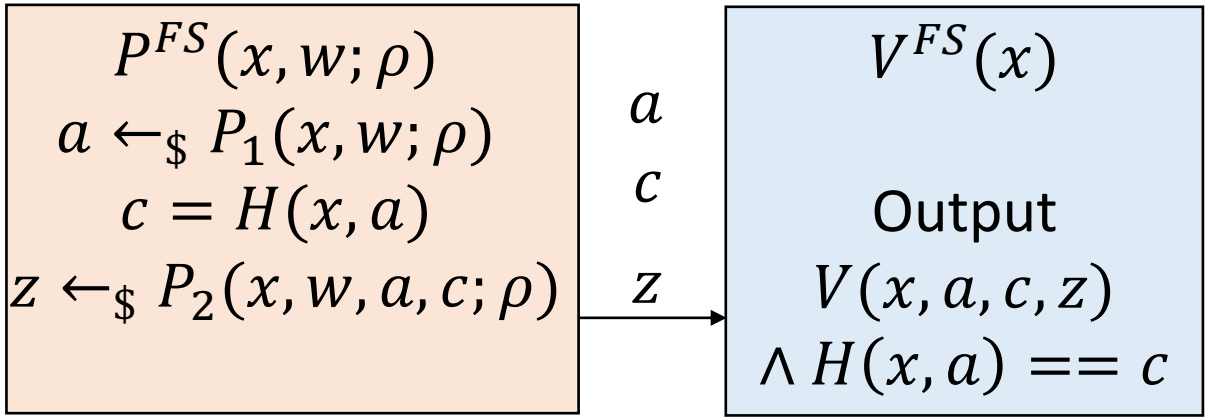
$\mathcal{L} \in \mathbf{P} \longleftrightarrow$ $x \in \mathcal{L}$ is obvious and needs no proof!



$\mathcal{L} \in \mathbf{NP} \longleftrightarrow$ $x \in \mathcal{L}$ has a proof which is easy to check



Σ-protocol for $\mathcal{R}$ + Fiat-Shamir Heuristic



The complexity class **BPP**

Definition:

$\mathcal{L} \in$ **BPP** if $\exists$ efficient decision algorithm M (polynomial time in $|x|$) satisfying:

- $\forall x \in \mathcal{L}, \Pr_r[M(x,r) = 1] \geq {}^3/_4$ "True statements usually accepted"
- $\forall x \notin \mathcal{L}, \Pr_r[M(x,r) = 1] \leq {}^1/_2$ "False statements usually rejected"

Any constants with a gap define the same complexity class

Why doesn't this apply to interactive proofs or the examples on the previous slide?

Impossibility of interesting NIZK?

Theorem:

- Let (P, V) be a NIZK for $\mathcal{L}$. Then $\mathcal{L} \in$ **BPP**.

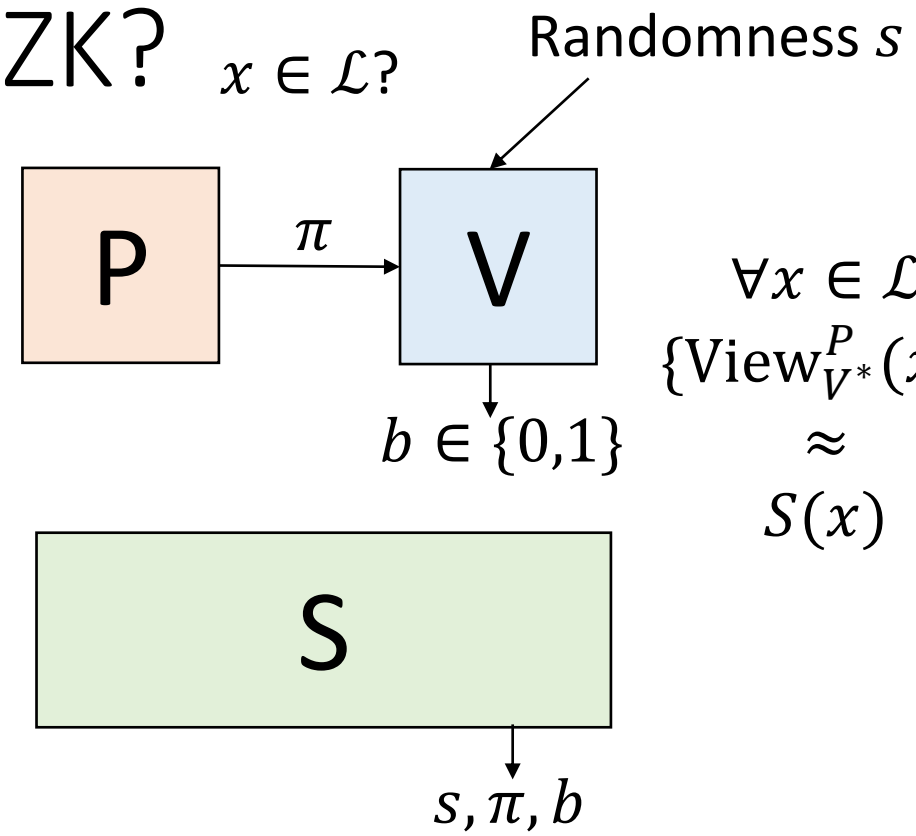
Proof: **BPP** decider M for $\mathcal{L}$:

- $x \in \mathcal{L} \Rightarrow S$ output indistinguishable from P output (by ZK of (P, V)).

$\Rightarrow M$ outputs 1 with probability $\geq \frac{3}{4}$

(by completeness of (P, V)).

- $x \notin \mathcal{L} \Rightarrow M$ outputs 1 with probability $\leq \frac{1}{2}$ (by soundness of (P, V)).
- M is efficient because S and V are.

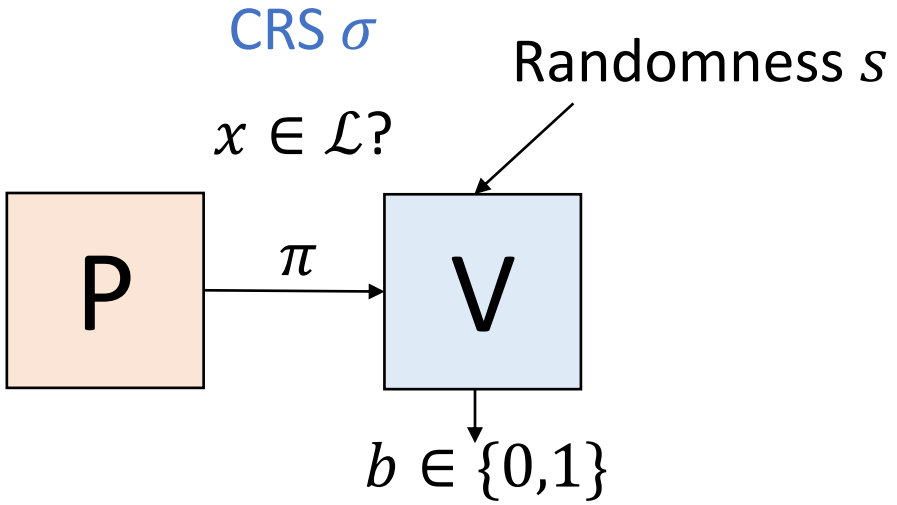


$M(x)$

1. $s, \pi, b \leftarrow S(x)$.
2. Sample s'.
3. Output $V(x, \pi; s') \in \{0,1\}$.

Syntax for non-interactive zero-knowledge

- Include a common reference string (CRS) containing ingredients used in the proof.



Definition:

A *non-interactive proof system* for an NP relation $\mathcal{R}$ consists of three efficient algorithms (K, P, V) which are

- the CRS generator $K(1^\lambda) \to \sigma$,
- the prover $P(\sigma, x, w) \to \pi$,
- the verifier $V(\sigma, x, \pi) \to b$.

Suppressing random inputs

K may take $|x|$ or even x as input

Ideally σ is uniformly random but may be structured

Security of non-interactive proofs

Easy to modify to get computational and statistical security notions

- **Completeness:** $\forall (x, w) \in \mathcal{R}$,

$$\Pr[b = 1 | \, \sigma \leftarrow K(1^\lambda), \pi \leftarrow P(\sigma, x, w), b \leftarrow V(\sigma, x, \pi)] = 1$$

Adaptive

- **Soundness:** $\forall P^*$,

$$\Pr[x \notin \mathcal{L}_\mathcal{R}, b = 1 | \, \sigma \leftarrow K(1^\lambda), (x, \pi) \leftarrow P^*(\sigma), b \leftarrow V(\sigma, x, \pi)] \approx 0$$

Adaptive

- **Zero-knowledge:** $\exists$ efficient simulators (S_1, S_2) such that $\forall A$ producing $(x, w) \in \mathcal{R}$,

Simulated σ indistinguishable from normal σ

$$\{(\sigma, \pi) : \sigma \leftarrow K(1^\lambda), (x, w) \leftarrow A(\sigma), \pi \leftarrow P(\sigma, x, w)\}$$
$$\approx \{(\sigma, \pi) : (\sigma, \tau) \leftarrow S_1(1^\lambda), (x, w) \leftarrow A(\sigma), \pi \leftarrow S_2(\sigma, x, \tau)\}$$

Simulation trapdoor τ (replaces oracle access to V^*)

In non-adaptive definitions, x is not chosen based on σ

These are *single-theorem* definitions. No security guarantees reusing σ for many x.

Knowledge soundness

Definition:

(K, P, V) is a *proof of knowledge* for a relation $\mathcal{R}$ if ∃ efficient extractors E_1, E_2 such that for all P^*,

Extractor's σ indistinguishable from normal σ

Extraction trapdoor ξ
(replaces oracle access to P^*)

- $\{\sigma : (\sigma, \xi) \leftarrow E_1(1^\lambda)\} \approx \{\sigma : \sigma \leftarrow K(1^\lambda)\}$, and

- $$\Pr\left[\begin{array}{c} V(\sigma, x, \pi) = 0 \\ \vee\ (x, w) \in \mathcal{R} \end{array} : \begin{array}{c} (\sigma, \xi) \leftarrow E_1(1^\lambda), (x, \pi) \leftarrow P^*(\sigma) \\ w \leftarrow E_2(\sigma, \xi, x, pi) \end{array}\right] \approx 1$$

$V(\sigma, x, \pi) \Rightarrow (x, w) \in \mathcal{R}$

How can we trust the CRS?

- Simulation trapdoors let us produce proofs without knowing witnesses (breaking soundness)
- Extraction trapdoors let us extract witnesses from proofs (breaking ZK)
- Trapdoor σ are indistinguishable from normal σ.

Mitigate risks using

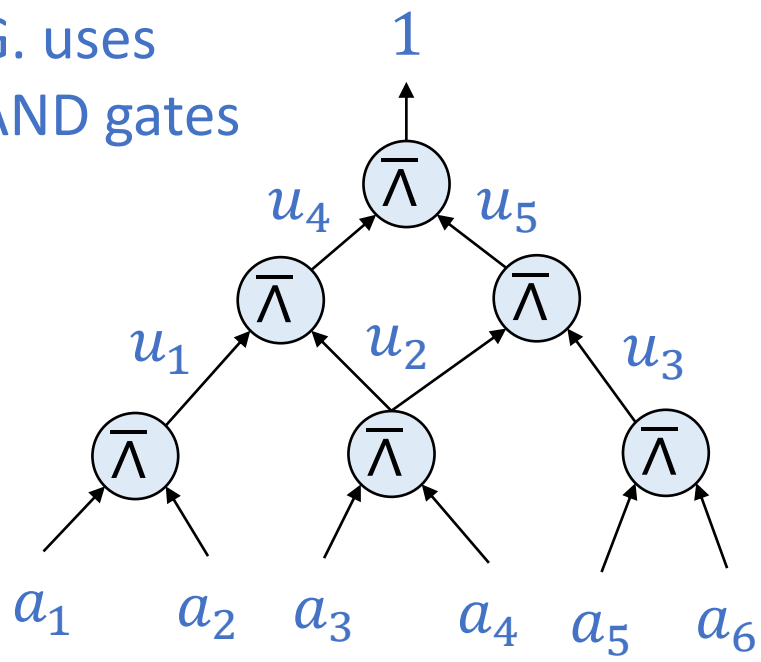
- “Subversion resistant” NIZK constructions
- “Updatable CRS” NIZK constructions
- “Verifiable CRS” NIZK constructions
- MPC protocols to generate σ

How can we trust the CRS?

[2] This curious property makes our result potentially applicable. For instance, all libraries in the country possess identical copies of the random tables prepared by the Rand Corporation. Thus, we may think of ourselves as being already in the scenario needed for noninteractive zero-knowledge

Boolean circuit NIZK idea

W.L.O.G. uses only NAND gates



Instance: circuit over $\mathbb{Z}_2$ with output.

Witness: input wire values giving correct output.

a	b	c	$\overline{a \wedge b} == c$	$a + b + 2c - 2$
0	0	0	0	-2
0	0	1	1	0
0	1	0	0	-1
0	1	1	1	1
1	0	0	0	-1
1	0	1	1	1
1	1	0	1	0
1	1	1	0	2

$$\overline{a \wedge b} = c \Leftrightarrow a + b + 2c - 2 \in \{0,1\}$$

Proof idea:

- Commit to each wire value.
- Prove each wire value $\in \{0,1\}$.
- Prove $a + b + 2c - 2 \in \{0,1\}$ for wires around each gate.

Need a commitment scheme with NI bit proofs

Composite-order symmetric pairings

Definition:

A *symmetric bilinear group* is a triple of two groups of order $n = pq$ (where p, q are distinct primes) and a *bilinear map* $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ satisfying

$$\forall a, b \in \mathbb{Z}_p, \forall G_1, G_2 \in \mathbb{G},$$
$$e(a \cdot G_1, b \cdot G_2) = ab \cdot e(G_1, G_2)$$

Pairing maps
'multiply DLOGs'

which is non-degenerate i.e.

$$\text{If } \mathbb{G} = \langle G_1 \rangle = \langle G_2 \rangle, \text{ then } \mathbb{G}_T = \langle e(G_1, G_2) \rangle$$

Changes from last time:

- Order n instead of p
- "Type 1" setting with $\mathbb{G}_1 \cong \mathbb{G}_2 \cong \mathbb{G}$

Facts

Claim: (symmetry)

$\forall G_1, G_2 \in \mathbb{G}, e(G_1, G_2) = e(G_2, G_1)$

Proof:

Let $\mathbb{G} = \langle G \rangle$. Write $G_1 = a \cdot G$ and $G_2 = b \cdot G$. Then

$e(G_1, G_2) = ab \cdot e(G, G) = ba \cdot e(G, G) = e(G_2, G_1)$ by bilinearity.

Both claims together imply homomorphism in the right input

Claim:

$\forall G_1, G_2, H \in \mathbb{G}, e(G_1 + G_2, H) = e(G_1, H) + e(G_2, H).$

Proof: exercise

The Boneh-Goh-Nissim Cryptosystem

Large enough that n is difficult to factor e.g. $O(\lambda^3)$ bits

Setup: on input $\lambda \in \mathbb{N}$, sample distinct primes p, q and composite order symmetric bilinear group $e, \mathbb{G}, \mathbb{G}_T$ of order $n = pq$, and $G, H \in \mathbb{G}$.

Sample $B \in \mathbb{N}$. Output $pp := (e, \mathbb{G}, \mathbb{G}_T, G, H, n, B)$.

$B \leq \text{poly}(\lambda)$

Exact sampling method to be discussed

Commit: given $m \in \{0, \dots, B-1\}, pp$, sample $r \leftarrow \mathbb{Z}_n$.

Compute $C = m \cdot G + r \cdot H$. Output (C, r).

Verify: check $m \in \{0, \dots, B-1\}$ and $C == m \cdot G + r \cdot H$.

Homomorphic (looks like Pedersen)

For us, a commitment scheme (but can be used as an encryption scheme)

Dual-mode parameter generation

- $\mathbb{G}$ has order $n = pq$.
- $\text{Setup}_{binding}: G \leftarrow \mathbb{G},\ s \leftarrow \mathbb{Z}_n^*,\ H = ps \cdot G.$

random generator of order q subgroup

a generator

- $\text{Setup}_{hiding}: G \leftarrow \mathbb{G},\ s \leftarrow \mathbb{Z}_n^*,\ H = s \cdot G.$

random generator of whole group $\mathbb{G}$

Definition:

The subgroup hiding assumption holds if

$$\{\text{Setup}_{binding}(1^\lambda)\} \approx_c \{\text{Setup}_{hiding}(1^\lambda)\}$$

The BGN cryptosystem is hiding

Proof:

- Using Setup_{hiding}: $G \leftarrow_{\$} \mathbb{G},\ s \leftarrow_{\$} \mathbb{Z}_n^*,\ H = s \cdot G$.

random generator of whole group $\mathbb{G}$

- For $r \leftarrow_{\$} \mathbb{Z}_n$, $r \cdot H$ is uniformly random in $\mathbb{G}$.
- Hence $C = m \cdot G + r \cdot H$ is uniformly random in $\mathbb{G}$.
- Therefore Setup_{hiding} gives *perfect* hiding.

With Setup_{hiding}, BGN is *equivocable* with equivocation key s.

$C = m \cdot G + r \cdot H = m' \cdot G + r' \cdot H$ where $a = \frac{m - m'}{s} mod\ n$.

- The output of $\text{Setup}_{binding}$ is computationally indistinguishable from Setup_{hiding} under the subgroup hiding assumption.
- Therefore $\text{Setup}_{binding}$ still gives *computational* hiding.

The BGN cryptosystem is perfectly binding

Proof:

- Using $\text{Setup}_{binding}$: $G \leftarrow_{\$} \mathbb{G},\ s \leftarrow_{\$} \mathbb{Z}_n^*,\ H = ps \cdot G$.

random generator of order q subgroup

- Suppose $C = m \cdot G + r \cdot H = m' \cdot G + r' \cdot H$ for distinct $m, m' \in \{0, \dots, B-1\}$.
- Then $e(C, q \cdot G) = e(m \cdot G + r \cdot H, q \cdot G) = e(m \cdot G + rps \cdot G, q \cdot G)$
$= qm \cdot e(G,G) + rspq \cdot e(G,G) = qm \cdot e(G,G)$
- Similarly, $e(C, q \cdot G) = qm' \cdot e(G,G)$. Hence $q(m - m') \cdot e(G,G) = 0$.
- By non-degeneracy, $e(G,G)$ has order n so $n \mid q(m - m')$.
- Hence $q(m - m') = kn$, so $(m - m') = kp$.
- $m \equiv m' mod\ p$ but $B \ll p$ so $m = m'$.

With $\text{Setup}_{binding}$, BGN is *extractable* with extraction key s.

Compute $e(C, q \cdot G)$, check whether it is equal to $qm \cdot e(G,G)$ for each $m \in \{0, \dots, B-1\}$.

- The output of Setup_{hiding} is computationally indistinguishable from $\text{Setup}_{binding}$ under subgroup hiding, so Setup_{hiding} gives *computational* binding.