Lecture 13: Constant-size NIZK arguments

Zero-knowledge proofs

263-4665-00L

Lecturer: Jonathan Bootle

Announcements

- Graded homework due today, 15/12/2023 23:59 CET.

- Exam date 02/02/2023, 15:00-17:00.
- Past exam questions on Moodle. Will post summary of examinable materials later.
- Previous exams: 100 points. This year: shorter at 70 points.
- Continued office hours after New Year on 16/01, 23/01, 30/01.

- Today's exercise session: ZK implementation in Circom. It may be useful to try installation in advance.
- https://learn.microsoft.com/en-us/windows/wsl/install (windows users)
- https://docs.circom.io/getting-started/installation/

Agenda

- Non-interactive zero-knowledge (NIZK) definitions ✓

Pairing-based constructions of NIZK

- From reasonable cryptographic assumptions
  - The BGN cryptosystem ✓
  - BGN bit proofs ✓
  - BGN proofs for CSAT ✓
- **From strong cryptographic assumptions**
  - Arithmetisation of R1CS into QAP
  - Linear PCP and pairing-based compiler

$O(N)$ proof size for Boolean circuits

$O(1)$ proof size for Arithmetic circuits

“Trade off between cryptographic and non-cryptographic effort”

From R1CS to strong R1CS

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

entry-wise product

$\vec{x}$ makes the problem non-trivial. W.L.O.G first entry is 1.

Definition: *strong* R1CS instances are as above, and additionally, if $\vec{z}_i := \vec{x}||\vec{w}_i$ for $i \in [3]$, $A\vec{z}_1 \circ B\vec{z}_2 = C\vec{z}_3$ implies that $\vec{z}_1 = \vec{z}_2 = \vec{z}_3$.

Lemma: for each R1CS instance, there is a *strong* R1CS instance with exactly the same witnesses and dimensions $N_r + 2N_c, N_c$.

Proof:

$$\begin{pmatrix} & A \\ & I_{N_c} \\ 1^{N_c} & 0_{N_c \times (N_c - 1)} \end{pmatrix} \vec{z}_1 \circ \begin{pmatrix} & B \\ 1^{N_c} & 0_{N_c \times (N_c - 1)} \\ & I_{N_c} \end{pmatrix} \vec{z}_2 = \begin{pmatrix} C \\ I_{N_c} \\ I_{N_c} \end{pmatrix} \vec{z}_3$$

$$\begin{pmatrix} A\vec{z}_1 \\ 1^{N_c} \\ \vec{z}_1 \end{pmatrix} \circ \begin{pmatrix} B\vec{z}_2 \\ \vec{z}_2 \\ 1^{N_c} \end{pmatrix} = \begin{pmatrix} C\vec{z}_3 \\ \vec{z}_3 \\ \vec{z}_3 \end{pmatrix}$$

Polynomial definitions and facts

- Let $H \subseteq \mathbb{F}$ with $|H| = N$.

$L_{h,H}(\omega) = (\omega == h)$

Definition:

- The *Lagrange polynomials* on H are defined, for $\omega \in H$, by

$$L_{\omega,H}(X) \coloneqq \prod_{\omega' \in H \setminus \{\omega\}} \frac{X - \omega'}{\omega - \omega'}$$

Degree $|H| - 1$.

- The *vanishing polynomial* on H is defined as $v_H(X) \coloneqq \prod_{\omega \in H}(X - \omega)$.

Degree $|H|$.

Fact:

For $f \in \mathbb{F}[X]$, we have $f(h) = 0 \; \forall \omega \in H \Leftrightarrow v_H(X) \mid f(X)$.

R1CS as polynomial divisibility

- Choose $H = \{1, \dots, N_r\} \subseteq \mathbb{F}$ (there are better choices).
- For each $j \in [N_c]$, define $a_j(X) \coloneqq \sum_{i \in [N_r]} a_{ij} L_{i,H}(X)$.
- Define $b_j(X)$, $c_j(X)$ similarly.
- Let $\vec{z} = (z_1, \dots, z_{N_c})$ be an R1CS witness.
- Define $A_{\vec{z}}(X) \coloneqq \sum_{j \in [N_c]} z_j a_j(X)$ and $B_{\vec{z}}(X), C_{\vec{z}}(X)$ similarly.

$A = (a_{i,j})$

Note that $a_j(i) = a_{i,j}$.

Lemma: $v_H(X) \mid A_{\vec{z}}(X) \cdot B_{\vec{z}}(X) - C_{\vec{z}}(X) \iff A\vec{z} \circ B\vec{z} = C\vec{z}$.

Proof: $v_H(X) \mid A_{\vec{z}}(X) \cdot B_{\vec{z}}(X) - C_{\vec{z}}(X) \Leftrightarrow A_{\vec{z}}(X) \cdot B_{\vec{z}}(X) - C_{\vec{z}}(X)$ vanishes on H.

For each $i \in H = [N_r]$,

$$A_{\vec{z}}(i) B_{\vec{z}}(i) - C_{\vec{z}}(i) = \left(\sum_{j \in [N_c]} z_j a_j(i)\right)\left(\sum_{j \in [N_c]} z_j b_j(i)\right) - \left(\sum_{j \in [N_c]} z_j c_j(i)\right)$$

$$= \left(\sum_{j \in [N_c]} z_j a_{ij}\right)\left(\sum_{j \in [N_c]} z_j b_{ij}\right) - \left(\sum_{j \in [N_c]} z_j c_{ij}\right) = (A\vec{z})_i (B\vec{z})_i = (C\vec{z})_i.$$

The Quadratic Arithmetic Program (QAP) problem

Definition:

degree $\leq h-1$ N is the size. $h := |H|$ is the degree.

- QAP instance $\mathbb{x} = \left(\mathbb{F}, \{a_j(X), b_j(X), c_j(X)\}_{j\in[N]}, \vec{x}, H\right)$, with $\vec{x} \in \mathbb{F}^k, H \subseteq \mathbb{F}$.
- QAP witness $\vec{w} \in \mathbb{F}^{N-k}$, such that if $\vec{z} := \vec{x}||\vec{w}$, $\exists Q(X) \in \mathbb{F}[X]$ such that

$A_{\vec{z}}(X)B_{\vec{z}}(X) = C_{\vec{z}}(X) + Q(X)v_H(X)$.

- A QAP instance is *strong* if

$$\left(\exists \vec{z}_1, \vec{z}_2, \vec{z}_3 \in \mathbb{F}^N, \exists Q(X) \in \mathbb{F}[X] : A_{\vec{z}_1}(X)B_{\vec{z}_2}(X) = C_{\vec{z}_3}(X) + Q(X)v_H(X)\right)$$

$\vec{z}_i := \vec{x}||\vec{w}_i$ $\Rightarrow \vec{z}_1 = \vec{z}_2 = \vec{z}_3$ $\vec{w}_1 = \vec{w}_2 = \vec{w}_3$

We can transform CSAT → R1CS → Strong R1CS → Strong QAP

Agenda

- Non-interactive zero-knowledge (NIZK) definitions ✓

Pairing-based constructions of NIZK

- From reasonable cryptographic assumptions
  - The BGN cryptosystem ✓
  - BGN bit proofs ✓
  - BGN proofs for CSAT ✓
- **From strong cryptographic assumptions**
  - Arithmetisation of R1CS into QAP ✓
  - **Linear PCP** and pairing-based compiler

$O(N)$ proof size for Boolean circuits

$O(1)$ proof size for Arithmetic circuits

Succinct Non-Interactive Arguments via Linear Interactive Proofs

Nir Bitansky[*]
Tel Aviv University

Alessandro Chiesa
MIT

Yuval Ishai[†]
Technion

Rafail Ostrovsky[‡]
UCLA

Omer Paneth[§]
Boston University

point-query PCPs

linear-query PCPs

A linear-query PCP for QAP

$P(\mathbb{x}, \vec{w})$

Sample $r_A, r_B \leftarrow_\$ \mathbb{F}$. Compute $\vec{z} := \vec{x}||\vec{w}$.

$$Q'(X) := \frac{(A_{\vec{z}}(X)+r_A \cdot v_H(X))(B_{\vec{z}}(X)+r_B \cdot v_H(X)) - C_{\vec{z}}(X)}{v_H(X)}$$

$\vec{Q}' := \text{Coeffs}(Q'(X))$ degree $\leq h$

pad to $\mathbb{F}^{h+1}$

$\vec{w}||r_A||r_B||\vec{w}||\vec{Q}'$

Length $O(N+h)$

$V(\mathbb{x})$

Sample $s \leftarrow_\$ \mathbb{F} \setminus |H|$.

Compute $a_{\vec{x}} := \sum_{j \leq k} x_j a_j(s)$

$b_{\vec{x}} := \sum_{j \leq k} x_j b_j(s)$

$c_{\vec{x}} := \sum_{j \leq k} x_j c_j(s)$

Query to get

$a_{\vec{w}, r_A} := \sum_{j > k} z_j a_j(s) + r_A \cdot v_H(s)$

$b_{\vec{w}, r_B} := \sum_{j > k} z_j b_j(s) + r_B \cdot v_H(s)$

$c_{\vec{w}, \vec{Q}'} := \sum_{j > k} z_j c_j(s) + \sum_{j \in [0,\ldots,|H|]} Q'_j s^j v_H(s)$

Accept iff

$(a_{\vec{x}} + a_{\vec{w}, r_A})(b_{\vec{x}} + b_{\vec{w}, r_A}) == \left(c_{\vec{x}} + c_{\vec{w}, \vec{Q}'}\right)$.

Allowing $v_H(X)$ multiples does not affect QAP satisfiability.

Note: $s \in \mathbb{F} \setminus |H|$ so $v_H(s) \neq 0$ so $a_{\vec{w}, r_A}, b_{\vec{w}, r_B}$ are uniformly random in $\mathbb{F}$.

Completeness analysis

If $\mathbb{x} \in \mathcal{L}_{QAP}$ then $\exists \vec{w} \in \mathbb{F}^k$ such that setting $\vec{z} = \vec{x}||\vec{w}$,

- $\exists Q(X):\ A_{\vec{z}}(X)B_{\vec{z}}(X) = C_{\vec{z}}(X) + Q(X)v_H(X)$.
- $\exists Q'(X): \big(A_{\vec{z}}(X) + r_A \cdot v_H(X)\big)\big(B_{\vec{z}}(X) + r_B \cdot v_H(X)\big) = C_{\vec{z}}(X) + Q'(X)v_H(X)$.
- $Q'(X) \coloneqq Q(X) + r_A \cdot B_{\vec{z}}(X) + r_B \cdot A_{\vec{z}}(X) + r_A r_B \cdot v_H(X)$

- $A_{\vec{z}}(s) + r_A \cdot v_H(s) = \sum_{j\in[N]} z_j a_j(s) + r_A \cdot v_H(s) = a_{\vec{x}} + a_{\vec{w},r_A}$. Similarly for $B_{\vec{z}}(s)$.
- $C_{\vec{z}}(s) + \sum_{j\in[0,\ldots,|H|]} Q'_j s^j = c_{\vec{x}} + c_{\vec{w},\vec{Q}'}$.
- Hence $\left(a_{\vec{x}} + a_{\vec{w},r_A}\right)\left(b_{\vec{x}} + b_{\vec{w},r_A}\right) = \left(c_{\vec{x}} + c_{\vec{w},\vec{Q}'}\right)$ and V accepts.

Soundness analysis

If $\mathbb{x} \notin \mathcal{L}_{QAP}$ then $\forall \vec{w} \in \mathbb{F}^k$, setting $\vec{z} = \vec{x}||\vec{w}$,

- $\forall Q(X):\ A_{\vec{z}}(X)B_{\vec{z}}(X) \neq C_{\vec{z}}(X) + Q(X)v_H(X)$.
- $\forall Q'(X), r_A, r_B : \left(A_{\vec{z}}(X) + r_A \cdot v_H(X)\right)\left(B_{\vec{z}}(X) + r_B \cdot v_H(X)\right) \neq C_{\vec{z}}(X) + Q'(X)v_H(X)$.
- $\left(A_{\vec{z}}(s) + r_A \cdot v_H(s)\right)\left(B_{\vec{z}}(s) + r_B \cdot v_H(s)\right) \neq C_{\vec{z}}(s) + Q'(s)v_H(s)$ except w.p. $\leq \frac{2h}{|\mathbb{F}|-h}$
- $A_{\vec{z}}(s) + r_A \cdot v_H(s) = a_{\vec{x}} + a_{\vec{w},r_A}$.
- $B_{\vec{z}}(s) + r_B \cdot v_H(s) = b_{\vec{x}} + b_{\vec{w},r_B}$.
- $C_{\vec{z}}(s) + \sum_{j\in[0,..,|H|]} Q'_j s^j = c_{\vec{x}} + c_{\vec{w},\vec{Q}'}$.
- Hence $\left(a_{\vec{x}} + a_{\vec{w},r_A}\right)\left(b_{\vec{x}} + b_{\vec{w},r_A}\right) \neq \left(c_{\vec{x}} + c_{\vec{w},\vec{Q}'}\right)$ and V rejects.

Adding multiples of $v_H(X)$ does not affect divisibility.

Apply S.Z. Lemma with degree $2|H|$ and $s \leftarrow_{\$} \mathbb{F} \setminus |H|$

Prover complexity analysis

- P computes $A_{\vec{z}}(X), B_{\vec{z}}(X), C_{\vec{z}}(X)$ from $\vec{z}, \{a_j(X), b_j(X), c_j(X)\}, v_H(X)$.
- Each $\{a_j(X), b_j(X), c_j(X)\}$ has $O(h)$ coefficients.
- $O(Nh)$ to compute $A_{\vec{z}}(X), B_{\vec{z}}(X), C_{\vec{z}}(X)$.
- $O(h^2)$ to compute $Q'(X) \coloneqq \frac{\left(A_{\vec{z}}(X)+r_A \cdot v_H(X)\right)\left(B_{\vec{z}}(X)+r_B \cdot v_H(X)\right)-C_{\vec{z}}(X)}{v_H(X)}$ using long division.
- When H is specially chosen, we can reduce $O(h^2)$ to $O(h \log h)$ using the Fast Fourier Transform.

Agenda

- Non-interactive zero-knowledge (NIZK) definitions ✓

Pairing-based constructions of NIZK

- From reasonable cryptographic assumptions
  - The BGN cryptosystem ✓
  - BGN bit proofs ✓
  - BGN proofs for CSAT ✓
- **From strong cryptographic assumptions**
  - Arithmetisation of R1CS into QAP ✓
  - Linear PCP and **pairing-based compiler** ✓

$O(N)$ proof size for Boolean circuits

$O(1)$ proof size for Arithmetic circuits

Analysis here is sketchy because

- The real proof contains many subtleties
- Protocol doesn't match our definitions

Prime-order asymmetric pairings

Definition:

An *asymmetric bilinear group* is a triple of 3 groups of prime order p and a *bilinear map* $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ satisfying

$$\forall a, b \in \mathbb{Z}_p, \forall G, \in \mathbb{G}_1, \forall H, \in \mathbb{G}_2,$$
$$e(a \cdot G, b \cdot H) = ab \cdot e(G, H)$$

Pairing maps 'multiply DLOGs'

which is non-degenerate i.e.

If $\mathbb{G}_1 = \langle G \rangle, \mathbb{G}_2 = \langle H \rangle$, then $\mathbb{G}_T = \langle e(G, H) \rangle$

Clash for H

Assumptions

For some generation algorithm
$(e, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, G, H, p) \leftarrow_\$ \mathrm{Gen}(1^\lambda)$,
$p \approx 2^\lambda$, $\mathbb{G}_1 = \langle G \rangle$, $\mathbb{G}_2 = \langle H \rangle$.

Does not hold for all choices of polynomials $\{v_i(X)\}$!

Definition:

Let $\lambda \in \mathbb{N}$ and $N, h = \mathrm{poly}(\lambda)$. The *Knowledge of Exponent assumption* (KEA) over $\mathbb{G}_1$ for polynomials $\{v_j(X)\}_{j \in [N]}$ holds if for all efficient A,

Degree $\leq h$

there exists an efficient extractor X_A such that

$$\Pr\left[\begin{matrix} C, \hat{C} \in \mathbb{G}_1 \\ \hat{C} = \alpha \cdot C \\ C \neq \left(\sum_{i=1}^N z_i v_i(s)\right) \cdot G \end{matrix} : \begin{matrix} (e, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, G, H, p) \leftarrow_\$ \mathrm{Gen}(1^\lambda) \\ \alpha, s \leftarrow_\$ \mathbb{Z}_p^*, \sigma := \left(\{v_j(s) \cdot G\}, \{\alpha v_j(s) \cdot G\}\right) \\ \left(C, \hat{C} || z_1, \dots, z_N\right) \leftarrow_\$ (A || X_A)(\sigma, Z) \end{matrix}\right] \approx 0.$$

Auxiliary information Z

Non-falsifiable assumption

- DLOG can be 'falsified' by providing a DLOG breaker
- To break KEA, you have to provide A and prove that no X_A exists

Necessary for $O(1)$ proof size [GW'10]

Previous knowledge soundness definition

Definition:

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

Quantifiers on extractor and adversary are in the opposite order!

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

Idea for linear PCP to argument compiler

$P(\mathbb{x}, \vec{w})$

Sample $r_A, r_B \leftarrow_\$ \mathbb{F}$. Compute $\vec{z} \coloneqq \vec{x}||\vec{w}$.

$Q'(X) \coloneqq \frac{(A_{\vec{z}}(X)+r_A \cdot v_H(X))(B_{\vec{z}}(X)+r_B \cdot v_H(X))-C_{\vec{z}}(X)}{v_H(X)}$

$\vec{Q}' \coloneqq \mathrm{Coeffs}(Q'(X))$

$\vec{w}||r_A||r_B||\vec{w}||\vec{Q}'$ $\longrightarrow$

$V(\mathbb{x})$

Sample $s \leftarrow_\$ \mathbb{F} \setminus |H|$.

Compute $a_{\vec{x}} \coloneqq \sum_{j \le k} x_j a_j(s)$

$b_{\vec{x}} \coloneqq \sum_{j \le k} x_j b_j(s)$

$c_{\vec{x}} \coloneqq \sum_{j \le k} x_j c_j(s)$

Query to get

$a_{\vec{w}, r_A} \coloneqq \sum_{j>k} z_j a_j(s) + r_A \cdot v_H(s)$

$b_{\vec{w}, r_B} \coloneqq \sum_{j>k} z_j b_j(s) + r_B \cdot v_H(s)$

$c_{\vec{w}, \vec{Q}'} \coloneqq \sum_{j>k} z_j c_j(s) +$
$\sum_{j \in [0,..,|H|]} Q'_j s^j v_H(s)$

Accept iff

$(a_{\vec{x}} + a_{\vec{w}, r_A})(b_{\vec{x}} + b_{\vec{w}, r_A})$
$== \left(c_{\vec{x}} + c_{\vec{w}, \vec{Q}'}\right).$

- In a real argument, we need P to answer the queries but stick to a linear strategy.
- We can't allow V to choose and send s because that requires interaction.
- We will use KEA to guarantee 'linear' answers to each query.
- We will use pairings to replace the verifier check.

The $a_{\vec{w},r_A}$ query

$K(\mathbb{x})$: Sample $(e, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, G, H, p), \alpha, \beta, \gamma, s \leftarrow_\$ \mathbb{Z}_p^*$.

Output $\sigma_A \coloneqq \begin{pmatrix} H, \alpha \cdot H, \{a_j(s) \cdot G\}_{j>k}, v_H(s) \cdot G, \\ \{\alpha a_j(s) \cdot G\}_{j>k}, \alpha v_H(s) \cdot G \end{pmatrix}$.

$P(\sigma, \mathbb{x}, \mathbb{w})$: Sample $r_A \leftarrow_\$ \mathbb{Z}_p$. Compute $\vec{z} \coloneqq \vec{x} || \vec{w}$.

- Compute $A \coloneqq \left(\sum_{j>k} z_j a_j(s) + r_A \cdot v_H(s)\right) \cdot G \in \mathbb{G}_1$.
- Compute $\hat{A} \coloneqq \left(\sum_{j>k} z_j a_j(s) + r_A \cdot v_H(s)\right) \cdot \alpha G \in \mathbb{G}_1$.
- Output $\pi \coloneqq \left(A, \hat{A}\right) \in \mathbb{G}_1^2$.

$V(\sigma, \mathbb{x}, \pi)$: Output 1 if and only if $e(A, \alpha \cdot H) == e(\hat{A}, H)$.

Completeness sketch:

Since $\hat{A} = \alpha \cdot A$, $e(A, \alpha \cdot H) = e(\alpha \cdot A, H) = e(\hat{A}, H)$.

Hence, the verifier will accept.

Knowledge soundness sketch:

- Suppose $P^*(\sigma, \mathbb{x}) = \left(A, \hat{A}\right)$, satisfying $e(A, \alpha \cdot H) = e(\hat{A}, H)$.
- Write $A = a \cdot G$, $\hat{A} = \hat{a} \cdot G$.
- We have $e(A, \alpha \cdot H) = a\alpha \cdot e(G, H)$, so $e\left(\hat{A}, H\right) = \hat{a} \cdot e(G, H)$.
- Since $e(G, H)$ is a generator, $a\alpha = \hat{a}$, so $\hat{A} = \alpha \cdot A$.
- By the KEA assumption, ∃ efficient X_A producing $z_{k+1}, \ldots, z_N, r_A$ satisfying $A = \left(\sum_{j>k} z_j a_j(s) + r_A v_H(s)\right) \cdot G$.

CRS generator for all three queries

$K(\mathbb{x})$: Sample $(e, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, G, H, p), \alpha, \beta, \gamma, s \leftarrow_\$ \mathbb{Z}_p^*$.

Output $\sigma \coloneqq \begin{pmatrix} \sigma_A \\ \sigma_B \\ \sigma_C \end{pmatrix}$

$$= \begin{pmatrix} H, \alpha \cdot H, \{a_j(s) \cdot G\}_{j>k}, v_H(s) \cdot G, \{\alpha a_j(s) \cdot G\}_{j>k}, \alpha v_H(s) \cdot G, \\ G, \beta \cdot G, \{b_j(s) \cdot H\}_{j>k}, v_H(s) \cdot H, \{\beta b_j(s) \cdot H\}_{j>k}, \beta v_H(s) \cdot H, \\ H, \gamma \cdot H, \{c_j(s) \cdot G\}_{j>k}, \{s^j v_H(s) \cdot G\}_{j=0}^h, \{\gamma c_j(s) \cdot G\}_{j>k}, \{\gamma s^j v_H(s) \cdot G\}_{j=0}^h \end{pmatrix}.$$

- σ_B reverses $\mathbb{G}_1, \mathbb{G}_2$ because we will want the $b_{\vec{w}, r_B}$ in $\mathbb{G}_2$ later.
- σ_C uses a different multiplier from σ_A because if they both used α, $e(A, \alpha \cdot H) = e(\hat{A}, H)$ would imply that A was made from the $a_j(s)$ and $c_j(s)$, not just the $a_j(s)$.

All three queries

$K(\mathbb{x})$: Sample $(e, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, G, H, p), \alpha, \beta, \gamma, s \leftarrow_\$ \mathbb{Z}_p^*$.
Output $\sigma \coloneqq (\sigma_A, \sigma_B, \sigma_C)$.

$P(\sigma, \mathbb{x}, \mathbb{w})$: Sample $r_A, r_B \leftarrow_\$ \mathbb{Z}_p$. Compute $\vec{z} \coloneqq \vec{x}||\vec{w}$ and

- $A \coloneqq \left(\sum_{j>k} z_j a_j(s) + r_A \cdot v_H(s)\right) \cdot G \in \mathbb{G}_1$.
- $\hat{A} \coloneqq \left(\sum_{j>k} z_j a_j(s) + r_A \cdot v_H(s)\right) \cdot \alpha G \in \mathbb{G}_1$.
- $B \coloneqq \left(\sum_{j>k} z_j b_j(s) + r_B \cdot v_H(s)\right) \cdot H \in \mathbb{G}_2$.
- $\hat{B} \coloneqq \left(\sum_{j>k} z_j b_j(s) + r_B \cdot v_H(s)\right) \cdot \beta H \in \mathbb{G}_2$.
- $C \coloneqq \left(\sum_{j>k} z_j c_j(s) + \sum_{j\in[0,..,h]} Q_j' s^j v_H(s)\right) \cdot G \in \mathbb{G}_1$.
- $\hat{C} \coloneqq \left(\sum_{j>k} z_j c_j(s) + \sum_{j\in[0,..,h]} Q_j' s^j v_H(s)\right) \cdot \gamma G \in \mathbb{G}_1$.
- Output $\pi \coloneqq (A, \hat{A}, B, \hat{B}, C, \hat{C}) \in \mathbb{G}_1^2 \times \mathbb{G}_2^2 \times \mathbb{G}_1^2$.

$V(\sigma, \mathbb{x}, \pi)$: Output 1 if and only if $e(A, \alpha \cdot H) == e(\hat{A}, H)$,
$e(\beta \cdot G, B) == e(G, \widehat{H})$ and $e(C, \gamma \cdot H) == e(\hat{C}, H)$.

Completeness sketch:
Since $\hat{A} = \alpha \cdot A$, $e(A, \alpha \cdot H) = e(\hat{A}, H)$.
Since $\hat{B} = \beta \cdot B$, $e(\beta \cdot G, B) = e(G, \widehat{H})$.
Since $\hat{C} = \gamma \cdot C$, $e(C, \gamma \cdot H) = e(\hat{C}, H)$.
Hence V accepts.

Knowledge soundness sketch:
Suppose $P^*(\sigma, \mathbb{x}) = \pi$, satisfying all V's checks.
By various KEA assumptions, ∃ efficient X_{P^*} producing

- $z_{k+1}, \dots, z_N, r_A$ satisfying $A = (\sum_{j>k} z_j a_j(s) + r_A v_H(s)) \cdot G$,
- $z'_{k+1}, \dots, z'_N, r_B$ satisfying $B = (\sum_{j>k} z'_j b_j(s) + r_B v_H(s)) \cdot H$,
- $z''_{k+1}, \dots, z''_N, Q'_0, \dots, Q'_h$ satisfying $C = (\sum_{j>k} z''_j c_j(s) + \sum_{j\in[0,..,h]} Q'_j s^j v_H(s)) \cdot G$.

CRS generator when adding the final check

$K(\mathbb{x})$: Sample $(e, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, G, H, p), \alpha, \beta, \gamma, s \leftarrow_\$ \mathbb{Z}_p^*$.

Output $\sigma := \begin{pmatrix} \sigma'_A \\ \sigma'_B \\ \sigma'_C \end{pmatrix}$

CRS generation is heavily instance-dependent

Universal circuits: capture any N-gate circuit in $O(N \log N)$ gates using 'control bits'

$$= \begin{pmatrix} H, \alpha \cdot H, \{a_j(s) \cdot G\}_{j \in [N]}, v_H(s) \cdot G, \{\alpha a_j(s) \cdot G\}_{j>k}, \alpha v_H(s) \cdot G, \\ G, \beta \cdot G, \{b_j(s) \cdot H\}_{j \in [N]}, v_H(s) \cdot H, \{\beta b_j(s) \cdot H\}_{j>k}, \beta v_H(s) \cdot H, \\ H, \gamma \cdot H, \{c_j(s) \cdot G\}_{j \in [N]}, \{s^j v_H(s) \cdot G\}_{j=0}^h, \{\gamma c_j(s) \cdot G\}_{j>k}, \{\gamma s^j v_H(s) \cdot G\}_{j=0}^h \end{pmatrix}.$$

- V needs to adjust A, B, C to incorporate $\vec{x}$.
- Since σ_A uses multiplier α, $e(A, \alpha \cdot H) = e(\hat{A}, H)$ implies that A was only made from $\{a_j(s)\}_{j>k}$, not $\{a_j(s)\}_{j \in [N]}$.
- This means a malicious prover cannot change $\vec{x}$.

Adding the final check

$K(\mathbb{x})$: Sample $(e, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, G, H, p), \alpha, \beta, \gamma, s \leftarrow_\$ \mathbb{Z}_p^*$.
Output $\sigma := (\sigma_A', \sigma_B', \sigma_C')$.

$P(\sigma, \mathbb{x}, \mathbb{w})$: Sample $r_A, r_B \leftarrow_\$ \mathbb{Z}_p$. Compute $\vec{z} := \vec{x}||\vec{w}$ and output $\pi := \left(A, \hat{A}, B, \hat{B}, C, \hat{C}\right) \in \mathbb{G}_1^2 \times \mathbb{G}_2^2 \times \mathbb{G}_1^2$ as before.

$V(\sigma, \mathbb{x}, \pi)$:

Compute $A_x := \left(\sum_{j \le k} x_j a_j(s)\right) \cdot G \in \mathbb{G}_1$.

Compute $B_x := \left(\sum_{j \le k} x_j b_j(s)\right) \cdot H \in \mathbb{G}_2$.

Compute $C_x := \left(\sum_{j \le k} z_j a_j(s)\right) \cdot G \in \mathbb{G}_1$.

Output 1 if and only if
$e(A, \alpha \cdot H) == e(\hat{A}, H)$,
$e(\beta \cdot G, B) == e(G, \widehat{H})$,
$e(C, \gamma \cdot H) == e(\hat{C}, H)$, and
$e(A_x + A, B_x + B) == e(C_x + C, H)$.

Can be optimized to 3 group elements, and strong QAP property removed.

Completeness:
As before, plus

$A_x = a_{\vec{x}} \cdot G, \qquad A = a_{\vec{w}, r_A} \cdot G,$
$B_x = b_{\vec{x}} \cdot H, \qquad B = b_{\vec{w}, r_B} \cdot H,$
$C_x = c_{\vec{x}} \cdot G, \qquad C = a_{\vec{w}, \vec{Q}'} \cdot G.$

Final check implies $\left(a_{\vec{x}} + a_{\vec{w}, r_A}\right)\left(b_{\vec{x}} + b_{\vec{w}, r_A}\right) = \left(c_{\vec{x}} + c_{\vec{w}, \vec{Q}'}\right)$.
Hence V accepts by PCP completeness.

Communication complexity: $4\mathbb{G}_1 + 2\mathbb{G}_2$.

Verifier complexity: $O(k)$ $\mathbb{G}_1, \mathbb{G}_2$-ops and 1 pairing.

Prover complexity:

- $O\left(Nh + h^2\right)$ $\mathbb{Z}_p$-ops from the PCP.
- $O(N + h)$ $\mathbb{G}_1, \mathbb{G}_2$–ops to compute $A, \hat{A}, B, \hat{B}, C, \hat{C}$ from σ.
- Can be optimized a lot.

Zero-knowledge analysis

What is the verifier's view?

$\pi := (A, \hat{A}, B, \hat{B}, C, \hat{C}) \in \mathbb{G}_1^2 \times \mathbb{G}_2^2 \times \mathbb{G}_1^2$.

- $e(A, \alpha \cdot H) = e(\hat{A}, H)$,
- $e(\beta \cdot G, B) = e(G, \hat{H})$,
- $e(C, \gamma \cdot H) = e(\hat{C}, H)$, and
- $e(A_x + A, B_x + B) = e(C_x + C, H)$
- A, B are uniformly random.
- C is uniquely determined by the final check.
- We have seen that e.g. $\hat{A}$ must satisfy $\hat{A} = \alpha \cdot A$ so $\hat{A}, \hat{B}, \hat{C}$ are uniquely determined from A, B, C.

Why is the simulator valid?

- The distributions of A, B are uniform.
- The other values are uniquely determined by V's checks, which are all satisfied are satisfied.

Use knowledge of DLOGs
to satisfy all checks.

$S_1(\mathbb{x}) \to (\sigma, \tau := s, \alpha, \beta, \gamma)$

$S_2(\sigma, \mathbb{x}, \tau)$:

- Sample $r_A, r_B \leftarrow_\$ \mathbb{Z}_p$.
- Compute $A := r_A \cdot G$ and $B := r_B \cdot H$.
- Compute $a_{\vec{x}} := \sum_{j \le k} x_j a_j(s)$
- Compute $b_{\vec{x}} := \sum_{j \le k} x_j b_j(s)$
- Compute $c_{\vec{x}} := \sum_{j \le k} x_j c_j(s)$
- Compute $r_C := (a_{\vec{x}} + r_A)(b_{\vec{x}} + r_b) - c_{\vec{x}}$.
- Compute $C := r_C \cdot G$.
- $\hat{A} := \alpha \cdot A$, $\hat{B} := \beta \cdot B$, $\hat{C} := \gamma \cdot C$.
- Output $\pi := (A, \hat{A}, B, \hat{B}, C, \hat{C})$.

s, α, β, γ are 'toxic waste'
Must be forgotten after CRS generation
or can be used to forge proofs

Knowledge soundness sketch I

- As before, using KEA, we have

Defines r_A, and $\vec{z} = \vec{x}||\vec{w}$

- $z_{k+1}, \ldots, z_N, r_A$ satisfying $A = (\sum_{j>k} z_j a_j(s) + r_A v_H(s)) \cdot G$,

Defines r_B, and $\vec{z}' = \vec{x}||\vec{w}'$

- $z'_{k+1}, \ldots, z'_N, r_B$ satisfying $B = (\sum_{j>k} z'_j b_j(s) + r_B v_H(s)) \cdot H$,

- $z''_{k+1}, \ldots, z''_N, Q'_0, \ldots, Q'_h$ satisfying

Defines $\vec{Q}'$, and $\vec{z}'' = \vec{x}||\vec{w}''$

$C = (\sum_{j>k} z''_j c_j(s) + \sum_{j\in[0,..,h]} Q'_j s^j v_H(s)) \cdot G$.

Knowledge soundness sketch II

- $A_x := \left(\sum_{j\le k} x_j a_j(s)\right)\cdot G,$ $\qquad A = \left(\sum_{j>k} z_j a_j(s) + r_A v_H(s)\right)\cdot G,$
- $B_x := \left(\sum_{j\le k} x_j b_j(s)\right)\cdot H,$ $\qquad B = \left(\sum_{j>k} z_j' b_j(s) + r_B v_H(s)\right)\cdot H,$
- $C_x := \left(\sum_{j\le k} z_j a_j(s)\right)\cdot G,$ $\qquad C = \left(\sum_{j>k} z_j'' c_j(s) + \sum_{j\in[0,..,h]} Q_j' s^j v_H(s)\right)\cdot G.$
- $e(A_x + A, B_x + B) = e(C_x + C, H).$
- Taking DLOGs w.r.t. $e(G,H)$, we have $\left(a_{\vec{x}} + a_{\vec{w},r_A}\right)\left(b_{\vec{x}} + b_{\vec{w}',r_A}\right) = \left(c_{\vec{x}} + c_{\vec{w}'',\vec{Q}'}\right).$
- Suppose $\left(A_{\vec{z}}(X) + r_A\cdot v_H(X)\right)\left(B_{\vec{z}'}(X) + r_B\cdot v_H(X)\right) \neq C_{\vec{z}''}(X) + Q'(X)v_H(X).$
- $\left(A_{\vec{z}}(s) + r_A\cdot v_H(s)\right)\left(B_{\vec{z}'}(s) + r_B\cdot v_H(s)\right) \neq C_{\vec{z}''}(s) + Q'(s)v_H(s)$ except w.p. $\le \frac{2h}{p-h}$.
- This means $\left(a_{\vec{x}} + a_{\vec{w},r_A}\right)\left(b_{\vec{x}} + b_{\vec{w}',r_A}\right) \neq \left(c_{\vec{x}} + c_{\vec{w}'',\vec{Q}'}\right)$, so V would not accept.

Not rigorous; assumes π produced by P^* is independent of s.
The real security proof needs additional (CDH style) assumptions.

Knowledge soundness sketch III

- So $\left(A_{\vec{z}}(X) + r_A \cdot v_H(X)\right)\left(B_{\vec{z}'}(X) + r_B \cdot v_H(X)\right) = C_{\vec{z}''}(X) + Q'(X)v_H(X)$.
- $A_{\vec{z}}(X)B_{\vec{z}'}(X) = C_{\vec{z}''}(X) + Q(X)v_H(X)$.
- $Q(X) \coloneqq Q'^{(X)} - r_A \cdot B_{\vec{z}}(X) - r_B \cdot A_{\vec{z}}(X) - r_A r_B \cdot v_H(X)$

- By the strong QAP property, $\vec{z} = \vec{z}' = \vec{z}''$.

- Hence $A_{\vec{z}}(X)B_{\vec{z}}(X) = C_{\vec{z}}(X) + Q(X)v_H(X)$, and we have extracted a QAP witness.

Agenda

- Non-interactive zero-knowledge (NIZK) definitions ✓

Pairing-based constructions of NIZK

- From reasonable cryptographic assumptions
  - The BGN cryptosystem ✓
  - BGN bit proofs ✓
  - BGN proofs for CSAT ✓
- **From strong cryptographic assumptions**
  - Arithmetisation of R1CS into QAP ✓
  - Linear PCP and pairing-based compiler ✓ ✓

$O(N)$ proof size for Boolean circuits

$O(1)$ proof size for Arithmetic circuits

What we saw in this course:

Explosion of activity!

Sumcheck
[LFKN’92]

GKR protocol
[GKR’08]

IOPs
[BCS’16]

PolyCommit, logarithmic verification from pairings
[Lee’21]

Interactive Proofs
Zero-knowledge
[GMW’88]

Sigma protocols
[Cramer’96]

MPC in the head
[IKOS’07]

PolyCommit, logarithmic proofs from DLOG
[BCCGP’16]

PolyIOP for CSAT
[Setty’20]

NIZK
[BFS’88]

BGN-based NIZKs
[GOS’06]

NIZKs from KEA
[GGPR’13]

3-element NIZKs
[Groth’16]

These papers don’t correspond exactly to course material due to subsequent mixing and simplification of ideas.

Sometimes the originals took a different view.
Sometimes later papers (not the originals) were easier to present here.

Other important and relevant papers: too many to mention!

Other topics:

- Advanced security properties
- Malleability
- Recursive proof composition
- Proofs from point-query IOPs and codes
- Low-memory proofs
- Lattices and quantum-safe ZK
- Quantum ZK

If you want more zero-knowledge…

- Libraries: https://github.com/arkworks-rs , https://docs.circom.io/
- Standardization effort: https://zkproof.org/
- Podcast: https://zeroknowledge.fm/
- Events: https://www.zksummit.com/
- More: https://github.com/ventali/awesome-zk

- Or ask me for random trivia/references/open problems/MSc thesis topics!

End of course