Succinct Arguments

Lecture 08:
Multilinear PIOP for R1CS

Pratyush Mishra
UPenn
Fall 2025

Summary of current PIOP for R1CS

We constructed a succinct-verifier PIOP for R1CS with the following properties:

- Prover time: $O(n \log n)$
- Verifier time: $O(\log n)$
- Number of rounds: $O(1)$

This lecture: linear prover time [Setty20]

We will construct a succinct-verifier PIOP for R1CS with the following properties:

- Prover time: $O(n)$
- Verifier time: $O(\log n)$
- Number of rounds: $O(\log n)$

Key tool:
multilinear extensions

Key tool: Multilinear extensions

Multilinear Interpolation:

Given a function $f : \{0,1\}^\ell \to \mathbb{F}$, we can **extend** f to obtain a *multilinear* polynomial $p(X_1, \ldots, X_\ell)$ such that $p(x) = f(x)$ for all $x \in \{0,1\}^\ell$.

Multilinear means the polynomial has degree at most 1 in each variable.

Multilinear Lagrange Polynomial:

For each $i \in \{0,1\}^\ell$, $\mathrm{eq}(i, X)$ is 1 at i, and 0 for all $j \in \{0,1\}^\ell, j \neq i$.

Can write $\mathrm{eq}(i, X) := \prod_{j=1}^{\ell} (i_j \cdot X_j + (1 - i_j)(1 - X_j)) \Rightarrow$ Can be evaluated in $O(\ell)$

Equiv, $\mathrm{eq}(i, X) := \prod_{j=1}^{\ell} (i_j \cdot X_j + (1 - i_j)(1 - X_j))$ is a multilinear poly over 2ℓ vars

Multilinear PIOP
For R1CS

What checks do we need?

Step 1: Correct Hadamard product

check that for each i, $z_A[i] \cdot z_B[i] = z_C[i]$

Step 2: Correct matrix-vector multiplication

check that $Mz = z_M \quad \forall M \in \{A, B, C\}$

Multilinear PIOP for Rowcheck

Prover(F, x, w)

1. Interpolate z_A, z_B, z_C to get $\hat{z}_A, \hat{z}_B, \hat{z}_C$.

$\hat{z}$

ZeroCheck PIOP for $\hat{z}_A \cdot \hat{z}_B - \hat{z}_C$

Verifier(F, x)

How to answer queries for $\hat{z}_A, \hat{z}_B, \hat{z}_C$?

How to evaluate $\hat{z}_M(r)$?

$$\hat{z}_M(r) = \sum_{i \in H} z_M[i] \cdot \mathrm{eq}(i, r)$$

$$= \sum_{i \in H} \sum_{j \in H} M[i, j] \cdot z[j] \cdot \mathrm{eq}(i, r)$$

$$= \sum_{i \in H} \sum_{j \in H} \hat{M}(i, j) \cdot \hat{z}(j) \cdot \mathrm{eq}(i, r)$$

Performing sumcheck for this will lead to verifier needing to check evaluations for $\hat{M}(\alpha, \beta), \hat{z}(\beta), \mathrm{eq}(\alpha, r)$.

How to compute/check evaluation for $\hat{M}(\alpha, \beta)$?

Recall: univariate case: encode matrix?

Polynomial Interpolation of Lists:

Given a list $A = (a_0, \ldots, a_d)$, and a set $H \subseteq \mathbb{F}$, the interpolation of A over H is

$$\hat{a}(X) := \sum_{i \in H} a_i \cdot L_H^i(X)$$

Polynomial Interpolation of Matrices?:

Given a list $M \in \mathbb{F}^{n \times n}$, and a set $H \subseteq \mathbb{F}$, the bivariate interpolation of A over H is

$$\hat{M}(X, Y) := \sum_{i \in H} \sum_{j \in H} M_{ij} \cdot L_H^i(X) \cdot L_H^j(Y)$$

Problem: computing this requires $O(|H|^2)$ work

Multilinear case?

Polynomial Interpolation of Lists:

Given list $A = (a_0, \ldots, a_d)$, and hypercube $H = \{0,1\}^{\log d}$, interpolation of A over H:

$$\hat{a}(X) := \sum_{i \in H} a_i \cdot \mathrm{eq}(i, X)$$

Polynomial Interpolation of Matrices?:

Given matrix $M \in \mathbb{F}^{n \times n}$, and set H, the bivariate interpolation of A over H is

$$\hat{M}(X, Y) := \sum_{i \in H} \sum_{j \in H} M_{ij} \cdot \mathrm{eq}(i, X) \cdot \mathrm{eq}(j, Y)$$

Problem: evaluating this requires $O(|H|^2)$ work

Insight: The matrices are sparse!

Polynomial Interpolation of Matrices?:

Given matrix $M \in \mathbb{F}^{n \times n}$, and set H, the bivariate interpolation of A over H is

$$\hat{M}(X, Y) := \sum_{i \in H} \sum_{j \in H} M_{ij} \cdot \text{eq}(i, X) \cdot \text{eq}(j, Y)$$

Most M_{ij} are zero!

Not a polynomial!

Can rewrite as $\hat{M}(X, Y) := \sum_{k \in K} \hat{\mathsf{v}}(k) \cdot \text{eq}(\hat{\mathsf{r}}(k), X) \cdot \text{eq}(\hat{\mathsf{c}}(k), Y)$,

K is a hypercube that indexes non-zero entries

Attempt 1:

$$\hat{M}(X, Y) := \sum_{k \in K} \hat{\mathsf{v}}(k) \cdot \mathrm{eq}(\hat{\mathsf{r}}(k), X) \cdot \mathrm{eq}(\hat{\mathsf{c}}(k), Y)$$

Set $\hat{\mathsf{r}}(k)$ to be a *tuple of polynomials.* That is,

$$\hat{\mathsf{r}}(k) = (\hat{\mathsf{r}}_0(k), \hat{\mathsf{r}}_1(k), \ldots, \hat{\mathsf{r}}_{\ell-1}(k))$$

Sumcheck over ℓ-degree polynomials.
Leads to time $O(d \log d)$!

So now $\hat{M}(X, Y) := \sum_{k \in K} \hat{\mathsf{v}}(k) \cdot \mathrm{eq}(\hat{\mathsf{r}}(k), X) \cdot \mathrm{eq}(\hat{\mathsf{c}}(k), Y)$,

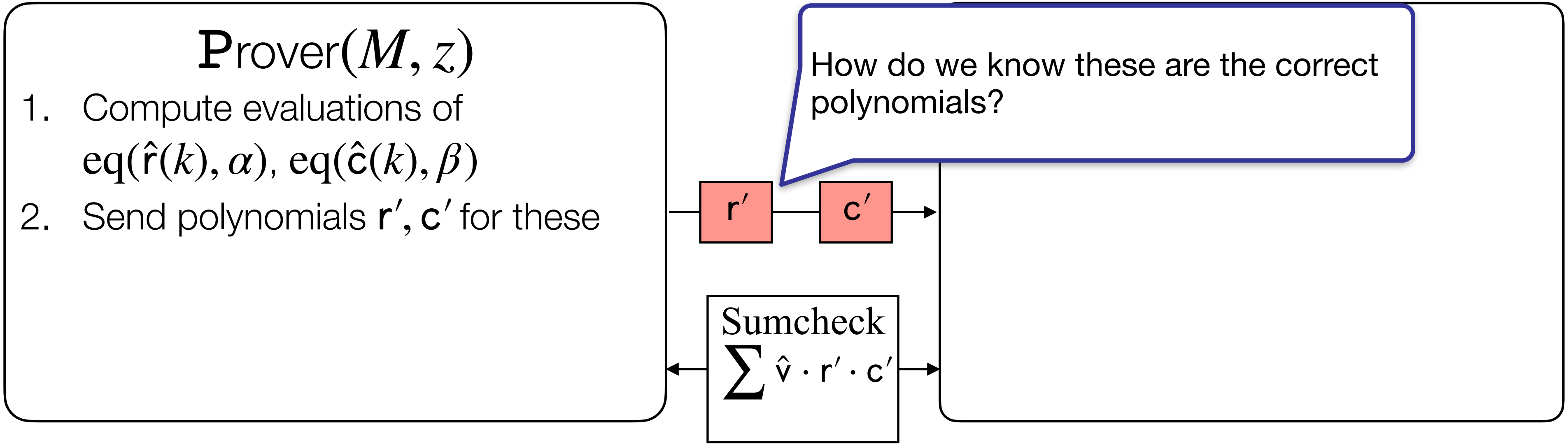
is a sum check over (composition of) polynomials!

Are we done?

Attempt 2:

We don't need the actual polynomial $\mathrm{eq}(\hat{\mathsf{r}}(k), \alpha)$

Instead, the polynomial that equals $\mathrm{eq}(\hat{\mathsf{r}}(k), \alpha)$ over K is good enough!



Are we done?

Checking equality of evals

$r'(X)$
$eq(0,\alpha)$
$eq(3,\alpha)$
$eq(6,\alpha)$
$eq(3,\alpha)$
$eq(2,\alpha)$
$eq(5,\alpha)$
$eq(6,\alpha)$
$eq(1,\alpha)$
$eq(7,\alpha)$
$eq(4,\alpha)$

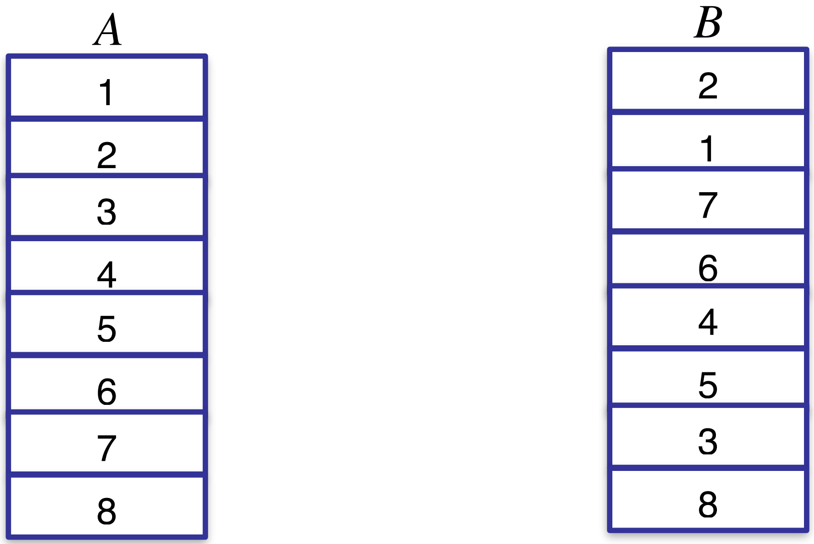
$eq_H(X, \alpha)$
$eq(0,\alpha)$
$eq(1,\alpha)$
$eq(2,\alpha)$
$eq(3,\alpha)$
$eq(4,\alpha)$
$eq(5,\alpha)$
$eq(6,\alpha)$
$eq(7,\alpha)$

Every element of the evaluation table of $r'(X)$
is an element of the evaluation table of $eq_H(X, \alpha)$!

PIOPs for multiset inclusion or *lookups*

How to check multiset inclusion?

Warmup: set equality



When are these two sets equal?
How to encode equality as a polynomial?

$$\prod_{a \in A} (X - a) = \prod_{b \in B} (X - b)$$

Polynomial fingerprint

Multiset equality?

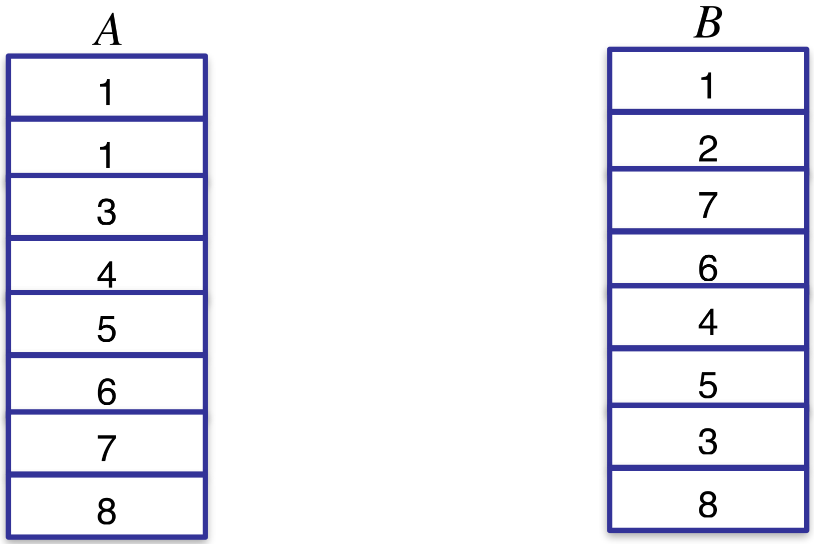
A	B
1	1
1	1
3	7
4	6
5	4
6	5
7	3
8	8

When are these two *multi*sets equal?

How to encode equality as a polynomial?

$$\prod_{a \in A} (X - a) = \prod_{b \in B} (X - b)$$

Multiset *inclusion*?



When is multiset A included in B?

How to encode equality as a polynomial?

$\prod_{a \in A} (X - a) = \prod_{b \in B} (X - b)$ doesn't work!

Multiset *inclusion*?

$$\prod_{a \in A} (X - a) = (X - 1)^2 \cdot (X - 3) \cdots (X - 8)$$

$$\prod_{b \in B} (X - b) = (X - 1) \cdot (X - 2) \cdot (X - 3) \cdots (X - 8)$$

They have common roots (up to multiplicity)!

In particular, A is included in B if and only if the roots of A's polynomial are a subset of those of B's polynomial!

Multiset *inclusion*?

$$\prod_{a \in A} (X - a) = (X - 1)^2 \cdot (X - 3) \cdots (X - 8)$$

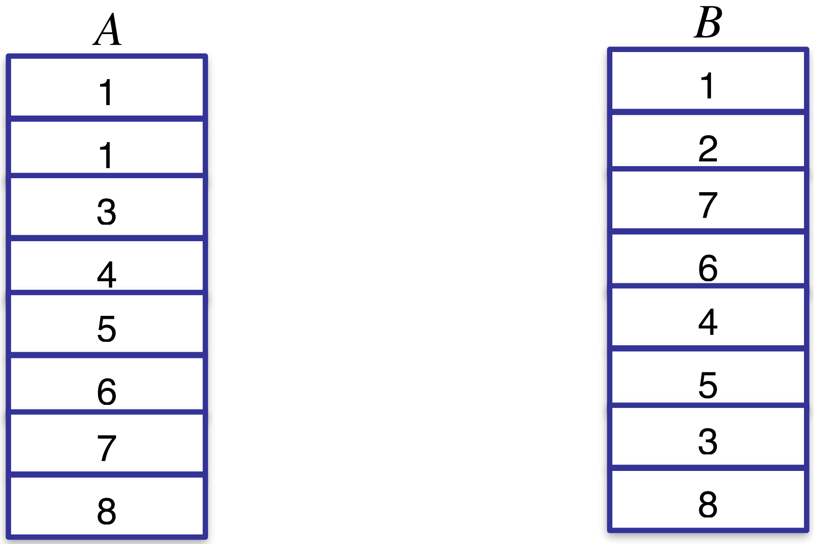
$$\prod_{b \in B} (X - b) = (X - 1) \cdot (X - 2) \cdot (X - 3) \cdots (X - 8)$$

Need to handle two things:

1. Elements in B not in A
2. Repeated elements in A

To handle this, we will introduce a *multiplicity* function m such that $m(b)$:= number of times $b \in B$ appears in A

Multiset *inclusion*?



When is multiset A included in B?

How to encode equality as a polynomial?

$$\prod_{a \in A} (X - a) = \prod_{b \in B} (X - b)^{m(b)}$$

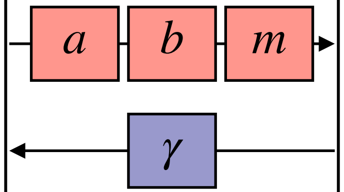
PIOP for polynomial fingerprinting

Attempt 1:

Prover

1. Send polynomials a, b whose evaluations are elements of A, B, and interpolation of m
2. Need to prove somehow that

$$\prod_{h \in H} (\gamma - a(h)) = \prod_{h \in H} (\gamma - b(h))^{m(h)}$$



Verifier

1. Sample $\gamma \leftarrow \mathbb{F}$

How to do product check?

Number of approaches today:

1. Direct construction [GW19]

2. Construct specialized circuit [Setty20]

3. *Logarithmic derivatives* [Habock22]

Logarithmic derivative

The logarithmic derivative of a polynomial $p(X)$ is $\frac{p'(X)}{p(X)}$

Important properties:

1. log-derivative of product is sum of log-derivatives:

$$\frac{(p_1(X) \cdot p_2(X))'}{p_1(X) \cdot p_2(X)} = \frac{p_1'(X) \cdot p_2(X) + p_1(X) \cdot p_2'(X)}{p_1(X) \cdot p_2(X)} = \frac{p_1'(X)}{p_1(X)} + \frac{p_2'(X)}{p_2(X)}.$$

2. Log-derivative of $\prod_a (X - a) = \sum \frac{1}{X - a}$

PIOP for Multiset inclusion!

Prover

1. Send polynomials a, b whose evaluations are elements of A, B, and interpolation of m
2. Rational sumcheck to prove that

$$\prod_{h \in H} (\gamma - a(h)) = \prod_{h \in H} (\gamma - b(h))^{m(h)}$$

Prover → Verifier: a, b, m

Verifier → Prover: γ

Rational sumcheck:

$$\sum_{h \in H} \frac{1}{\gamma - a(h)} = \sum_{h \in H} \frac{m(h)}{\gamma - b(h)}$$

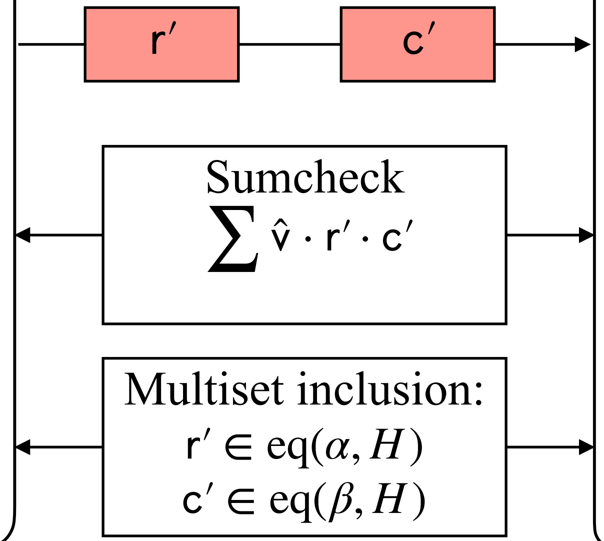
Verifier

1. Sample $\gamma \leftarrow \mathbb{F}$

Back to PIOP for lincheck:

$\mathbf{P}\text{rover}(M, z)$

1. Compute evaluations of $\text{eq}(\hat{\mathsf{r}}(k), \alpha)$, $\text{eq}(\hat{\mathsf{c}}(k), \beta)$
2. Send polynomials r', c' for these



$\mathbf{V}\text{erifier}^{\hat{\mathsf{r}}, \hat{\mathsf{c}}, \hat{\mathsf{v}}}()$

Other apps of multiset inclusion

Lookups

For many computations, expressing them as circuits over $\mathbb{F}$ is wasteful.

Eg: 8-bit XOR is cheap on a CPU, but requires 8 constraints in R1CS.

Instead,
during preprocessing, build table T containing all input-output triples for 8-bit XOR
during proving, instead of constraining witnesses with R1CS, constrain with multiset-inclusion in T!