

Lecture 12: Polynomial Interactive Oracle Proofs

CS 4803/8803 PC: Programmable Cryptography • Fall 2026

Note: my iPad is being repaired, so today's notes are typed.

1 Last time

Polynomial commitment scheme (PCS).

- $\text{Setup}(d) \rightarrow pp$
- $\text{Commit}(pp, f) \rightarrow c$
- $\text{EOpen}(pp, f, x) \rightarrow \pi$
- $\text{EVer}(pp, c, x, y, \pi) \rightarrow \{0, 1\}$

d bounds the degree of committed polynomials; π is a proof that the committed f has $f(x) = y$.

Pairings. $e: \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$

- non-degenerate, bilinear: $e(g_1^a, g_2^b) = g_T^{ab}$
- pro: 1 multiplication in the exponent – just once!
- con: makes DDH easy (sometimes) – e.g., if $\mathbb{G}_1 = \mathbb{G}_2$, then given (g^a, g^b, g^c) , check $e(g^a, g^b) \stackrel{?}{=} e(g, g^c)$.

KZG: a PCS from e Key idea: check a polynomial multiplication at a random τ , in the exponent.

$$f(x) = y \iff f(X) - y = q(X) \cdot (X - x)$$

The opening proof was a commitment to q . These ideas (committing to a quotient and checking a product at a random point) will come up again today.

2 Review and motivation

Kilian–Micali. SNARGs from PCPs + vector commitments

- Vector commitments: commit to a vector, open to indices
 - from Merkle trees
 - very practical – used in Git, TLS certificate transparency, E2EE key transparency, Ethereum, etc.
- PCPs: probabilistically checkable proofs

- proofs checked by (index) queries to a vector
- wildly impractical – never even implemented

PCPs are a crown jewel of theoretical computer science (three Gödel prizes), but no one has ever even implemented one.

Practical SNARGs: stronger commitments, weaker proofs

- Stronger commitments: PCS (last time)
 - commit to $f \in \mathbb{F}^{<d}[X]$
 - commit/open cost for P : $\mathcal{O}(d) \ll |\mathbb{F}|$
 - * for V : $\mathcal{O}(1)$ – just one pairing
 - $|\mathbb{F}|$ possible openings
- Weaker proofs: polynomial evaluation queries (now)
 - today: Fibonacci numbers – developing some tools along the way
 - next class: arithmetic circuits (which generalize boolean circuits)

A Merkle commitment to all $|\mathbb{F}|$ evaluations of f would also support $|\mathbb{F}|$ openings, but would cost $|\mathbb{F}| \approx 2^{256}$ to build. A PCS gets the same openings for $\mathcal{O}(d)$. With a commitment that strong, the proof system on top can be far simpler than a PCP.

3 PIOPs

Relation classes

Relation class Φ : a set of relations $\phi(x, w)$

- next class: arithmetic circuits
- today: Fibonacci numbers, $\Phi_{\text{Fib}} = \{\phi_1, \phi_2, \dots\}$

Definition 1 (Fibonacci relations). $\phi_i(x, w = \emptyset) \iff x = f_i$ (these relations don't need w), where

$$f_1 = 1, \quad f_2 = 1, \quad f_i = f_{i-1} + f_{i-2} \text{ for } i \geq 3.$$

i	1	2	3	4	5	6	7	8
f_i	1	1	2	3	5	8	13	21

Our protocol is field-based, so each relation needs a field. We'll choose it once we know what the protocol needs.

Goal: succinctness. There's no secret witness here: anyone can compute f_i in i steps. The point is succinctness – we want V to check $x = f_i$ much faster than computing it.

Syntax

PIOP Π for Φ : two algorithms P, V

- oracle: a polynomial $o \in \mathbb{F}^{\leq d}[X]$; query: $z \mapsto o(z)$ – d is a degree bound. This is the “polynomial” in PIOP.
- protocol: interactive PPT oracle algorithms $P(\phi, x, w), V(\phi, x)$, for $\phi \in \Phi$
 - P can: send oracles o
 - V can: query oracles o ; sample and send randomness
- $\langle P, V \rangle(\phi, x; w)$: output of $V(\phi, x)$ after interacting with $P(\phi, x, w)$

V gets ϕ as input, so this only works when ϕ has a short description, as ϕ_i does (it’s just i). For richer Φ , like circuits, V can’t even read ϕ . Next time, we’ll fix that with *preprocessing*: a Setup algorithm.

Properties

We want the following theorem to hold:

Theorem 2 (informal). *PIOP Π for Φ + PCS $\implies$ SNARG for Φ .*

Via the following construction:

Construction.

1. PIOP $\rightarrow$ IP, using the PCS (to turn oracles and queries into ordinary messages):
 - oracle $o \rightarrow$ commitment c_o
 - query o at $z \rightarrow$
 - V sends z
 - P sends $y = o(z)$, evaluation proof π
 - V checks π against c_o – then uses y as the evaluation
2. IP $\rightarrow$ non-interactive, via Fiat–Shamir

For this construction to be a SNARG, the PIOP needs a few properties. We define these to be the properties that a PIOP must have.

Public coin: V ’s randomness is immediately sent to P or used as a polynomial query. (a syntactic requirement). NB: this ensures the IP is public-coin too, which Fiat–Shamir requires. This is a general requirement: Fiat–Shamir only works for public-coin IPs.

Efficiency:

- P : PPT – ideally quasilinear in $|\phi|$. This implies all oracles have bounded degree, too.
- V : $\mathcal{O}(|x|) + o(|\phi|)$

NB: this (and the efficiency of the PCS’s EVer) ensures succinctness.

Completeness: $\forall \phi, x, w$, if $\phi(x, w)$, then

$$\Pr[\langle P, V \rangle(\phi, x; w) = 0] \leq \text{negl.}$$

NB: this ensures completeness in the IP and SNARG.

Soundness: $\forall \phi, x$, if $x \notin \mathcal{L}_\phi$, then $\forall P^*$,

$$\Pr[\langle P^*, V \rangle(\phi, x) = 1] \leq \text{negl.}$$

NB: this ensures soundness in the IP and SNARG (given a secure PCS). Note P^* is unbounded – but even P^* can only send polynomials within the degree bound, which the PCS enforces after compilation. Computational assumptions enter only through the PCS.

4 Sub-PIOPs

Our Fibonacci PIOP will use a number of subprotocols. These subprotocols will send different oracles and then test properties about those oracles. Let's start with some of these subprotocols for testing oracle properties.

Zero test

Problem. Oracle $f \in \mathbb{F}^{\leq d}[X]$, $d \ll p$. Is f zero everywhere? That is, is $f(0) = f(1) = \dots = f(p-1) = 0$? V can clearly test this using p queries. But it can do it with just one:

Protocol. V samples $\alpha \xleftarrow{\$} \mathbb{F}$; checks $f(\alpha) = 0$.

Completeness. $f = 0 \implies f(\alpha) = 0$. (Perfect completeness.)

For soundness, we'll need the fundamental theorem of algebra:

Fact 3 (Fundamental theorem of algebra). A nonzero polynomial of degree $\leq d$ has at most d roots.

Soundness. $f \neq 0 \implies \text{at most } d \text{ roots} \implies \Pr[f(\alpha) = 0] \leq d/p$.

Sub-PIOP properties. Here the definitions are a little different than for a full PIOP:

- language: $\mathcal{L} \subseteq \mathbb{F}^{\leq d}[X]$ – a set of bounded-degree polynomial oracles. For the zero test, $\mathcal{L} = \{0\}$.
- completeness: $f \in \mathcal{L} \implies V$ accepts w.h.p.
- soundness: $f \in \mathbb{F}^{\leq d}[X] \setminus \mathcal{L} \implies V$ accepts with low probability, $\forall P^*$

In the zero test, the prover plays no role, but it will in our next protocol.

Generalization. V can zero-test any polynomial of degree $\leq d$ that it can evaluate at α via oracle queries – e.g., $f \cdot g - h$.

For example, V can test whether two oracles are equal by zero-testing their difference: query both at the same α and compare. We'll use this constantly; nobody has to send the polynomial being tested.

Subgroup zero test

What about being zero on a *subset* of $\mathbb{F}$? Say, on $H \subset \mathbb{F}$?

Let H be a *cyclic multiplicative subgroup* of $\mathbb{F}^*$, of size $|H| \ll p$.

Problem. Oracle $f \in \mathbb{F}^{\leq d}[X]$. Is $f(h) = 0$ for all $h \in H$?

The zero test is not complete: f can be zero on H but nonzero elsewhere. We need a different test. But first, what does *cyclic multiplicative subgroup* mean?

Cyclic subgroups. For $\omega \in \mathbb{F}^* = \mathbb{F} \setminus \{0\}$:

$$\langle \omega \rangle \triangleq \{1 = \omega^0, \omega, \omega^2, \omega^3, \dots\}$$

- finite – since $\mathbb{F}$ is
- subgroup: $\omega^i \cdot \omega^j = \omega^{i+j}$ – closed under multiplication, the group operation
- H is cyclic if $H = \langle \omega \rangle$ for some ω – we say ω generates H
- $\omega^{|H|} = 1$, and $H = \{\omega^0, \omega^1, \dots, \omega^{|H|-1}\}$

Why $\omega^{|H|} = 1$: the powers of ω must eventually repeat, say $\omega^i = \omega^j$ with $i < j$; then $\omega^{j-i} = 1$. Let n be the smallest positive power with $\omega^n = 1$. Then the powers cycle with period n : $1, \omega, \dots, \omega^{n-1}$ are distinct, and then they repeat. So $|H| = n$. (This also gives inverses: $\omega^{-1} = \omega^{n-1}$.)

$\langle \omega \rangle$ may or may not be all of $\mathbb{F}^*$: many elements ω do not generate all of $\mathbb{F}^*$, but some do.

Definition 4 (Vanishing polynomial). $v_H(X) \triangleq X^{|H|} - 1 \in \mathbb{F}[X]$.

Why “vanishing”? It vanishes (is zero) on exactly H :

Fact 5. $\text{roots}(v_H) = H$.

Proof.

- $\text{roots}(v_H) \supseteq H$: $v_H(\omega^i) = (\omega^i)^{|H|} - 1 = (\omega^{|H|})^i - 1 = 1^i - 1 = 0$.
- $\text{roots}(v_H) \subseteq H$: $|\text{roots}(v_H)| \leq \deg v_H = |H|$. It has at most $|H|$ roots, and H already accounts for all of them. □

Intuition: how can we show that f is zero on H ? We show that it is divisible by v_H .

Lemma 6. f is zero on $H \iff v_H \mid f$.

Proof. $(\Leftarrow) f = q \cdot v_H \implies f(h) = q(h) \cdot 0 = 0$.

$(\Rightarrow) f = q \cdot v_H + r, \deg r < |H|$ (divide with remainder). For $h \in H$: $r(h) = f(h) - q(h) \cdot v_H(h) = 0$. So r has $\geq |H|$ roots $\implies r = 0$. Its degree is less than $|H|$. $\square$

Protocol.

1. P sends oracle $q = f / v_H \in \mathbb{F}^{\leq d-|H|}[X]$ – polynomial division, without remainder; $\tilde{O}(d)$ time
2. V samples $\alpha \xleftarrow{\$} \mathbb{F}$
3. V checks $q(\alpha) \cdot v_H(\alpha) = f(\alpha)$
 - queries $q(\alpha), f(\alpha)$
 - computes $v_H(\alpha) = \alpha^{|H|} - 1$ directly

This is just a zero test on $q \cdot v_H - f$.

Completeness. Fix f zero on H .

- $v_H \mid f$ (Lemma 6)
- so $q \cdot v_H = f$
- so $q(\alpha) \cdot v_H(\alpha) = f(\alpha)$: V accepts

Soundness. Fix f not zero on H , any P^* ; it sends $q^* \in \mathbb{F}^{\leq d-|H|}[X]$.

- $v_H \nmid f$ (Lemma 6)
- so $q^* \cdot v_H \neq f$
- so $q^* \cdot v_H - f$ is nonzero, of degree $\leq d$
- so $\Pr[V \text{ accepts}] = \Pr[\alpha \text{ is a root}] \leq d/p$

Here's where the degree bound on q^* matters: without it, $q^* \cdot v_H - f$ could have degree far above d , and many roots.

Efficiency. V does two queries and $\mathcal{O}(\log |H|)$ multiplications to evaluate v_H .

v_H is sparse, so V can evaluate it with $\mathcal{O}(\log |H|)$ multiplications (repeated squaring). That's why we use a multiplicative subgroup: for an arbitrary set S , the vanishing polynomial $\prod_{s \in S} (X - s)$ costs $|S|$ to evaluate.

5 A PIOP for Fibonacci

Goal. Convince V that $x = f_i$, with V doing $\mathcal{O}(\log i)$ field operations.

Naively, V would do $\mathcal{O}(i)$ work to compute f_i itself.

The plan: P writes down the whole computation $f_1, \dots, f_i$ as a polynomial, and V spot-checks it with the tools above.

Choosing the field

Fact 7. $\mathbb{F}_p^*$ is cyclic, of order $p - 1$. So $\mathbb{F}_p$ has a cyclic subgroup of size n iff $n \mid p - 1$.

Let $n = i$. ϕ_i uses $\mathbb{F}_i = \mathbb{Z}_{p_i}$ for a prime p_i with

- $p_i > f_i$ – so that x fits in $\mathbb{F}_i$
- $p_i > 2^\lambda$ – for soundness
- $n \mid p_i - 1$ – so that H exists

Fix $H = \langle \omega \rangle \subseteq \mathbb{F}_i^*$, $|H| = n$.

Such primes do exist: there are infinitely many primes $\equiv 1 \pmod{n}$ for any n (Dirichlet).

We'll still write p and $\mathbb{F} = \mathbb{Z}_p$, for brevity.

In practice, n need not equal i : any $n \geq i$ works, with padding. P keeps running the recurrence out to f_n , and (Final) moves to $T(\omega^{i-1}) = x$. So one takes n to be the smallest power of two $\geq i$, and p with $2^k \mid p - 1$ for large k . The power of two lets P use FFTs for interpolation and evaluation, which improves efficiency.

We'll ignore that here, and just assume $n = i$ divides $p - 1$.

First step: P sends a trace as an oracle

First, P computes all the intermediate Fibonacci numbers $f_1, \dots, f_n$ – the whole computation.

Then, P interpolates $T \in \mathbb{F}^{<n}[X]$: $T(\omega^j) = f_{j+1}$ for all $j < n$.

The $+1$ is because H starts at ω^0 and Fibonacci at f_1 .

j	0	1	2	3	$\dots$	$n-1$
point	$\omega^0 = 1$	ω	ω^2	ω^3	$\dots$	ω^{n-1}
$T(\cdot)$	1	1	2	3	$\dots$	$f_n = f_i$

There's a unique polynomial of degree $< n$ through n points (Lagrange interpolation), and it can be computed in $\tilde{O}(n)$ time.

Then, P sends T as an oracle to V . – This is the first message in the PIOP.

Second step: V checks the trace

It suffices for V to check three things: (Base), (Rec), and (Final).

Claim 8. $x = f_i \iff \exists T \in \mathbb{F}^{<n}[X]$ such that:

$$\text{(Base)} \quad T(1) = T(\omega) = 1$$

$$\text{(Rec)} \quad T(\omega^j) = T(\omega^{j-1}) + T(\omega^{j-2}) \text{ for all } j \in \{2, \dots, n-1\}$$

$$\text{(Final)} \quad T(\omega^{n-1}) = x$$

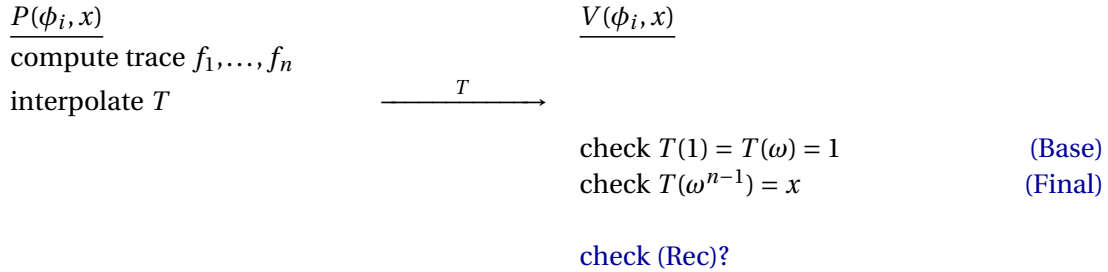
Proof.

- $(\Rightarrow)$ The trace polynomial.
- $(\Leftarrow)$
 - (Base), (Rec) $\implies T(\omega^j) = f_{j+1}$ for all $j < n$ (induction on j)
 - (Final) $\implies x = T(\omega^{n-1}) = f_n = f_i$

□

This is where we use $p > f_i$: (Rec) holds in $\mathbb{F}$, so the induction only shows $T(\omega^j) = f_{j+1} \bmod p$. Since $f_i < p$, that's f_i itself.

Checking (Base), (Final). V queries T at $1, \omega, \omega^{n-1}$. Three queries.



Checking (Rec). (Rec) is *almost* a subgroup zero test:

$$\begin{aligned} \text{(Rec)} &\iff T(\omega^j) - T(\omega^{j-1}) - T(\omega^{j-2}) = 0 && \text{for all } j \in \{2, \dots, n-1\} \\ &\iff T(\omega^j) - T(\omega^{-1}\omega^j) - T(\omega^{-2}\omega^j) = 0 && \text{for all } j \in \{2, \dots, n-1\} \\ &\iff T(h) - T(\omega^{-1}h) - T(\omega^{-2}h) = 0 && \text{for all } h \in H \setminus \{1, \omega\} \end{aligned}$$

So, we need a way to check that a polynomial is zero on a subgroup, except for two points. We'll do that by multiplying by polynomials that vanish on those two points.

$$C(X) \triangleq \underbrace{(T(X) - T(\omega^{-1}X) - T(\omega^{-2}X))}_{\text{zero on } H \setminus \{1, \omega\} \iff \text{(Rec)}} \cdot \underbrace{(X-1)}_{\text{zero at } 1} \cdot \underbrace{(X-\omega)}_{\text{zero at } \omega}$$

Claim 9. $(\text{Rec}) \iff C \text{ is zero on } H.$

So, P and V run the subgroup zero test on C . P does **not** send C as an oracle: it can be evaluated from T with three queries. So, when V would query C at α , it queries T at α , $\omega^{-1}\alpha$, and $\omega^{-2}\alpha$ instead, and computes $C(\alpha)$ from those three values.

To query $C(\alpha)$, V queries T at α , $\omega^{-1}\alpha$, and $\omega^{-2}\alpha$.

The full protocol

$\underline{P(\phi_i, x)}$		$\underline{V(\phi_i, x)}$
compute trace $f_1, \dots, f_n$		
interpolate T		
compute C , $q = C / v_H$	$\xrightarrow{T, q}$	
		check $T(1) = T(\omega) = 1$ (Base)
		check $T(\omega^{n-1}) = x$ (Final)
		$\alpha \xleftarrow{\$} \mathbb{F}$
		$C(\alpha) \leftarrow (T(\alpha) - T(\omega^{-1}\alpha) - T(\omega^{-2}\alpha))$
		$\quad \cdot (\alpha - 1)(\alpha - \omega)$
		check $q(\alpha) \cdot v_H(\alpha) = C(\alpha)$ (Rec)

Analysis

Efficiency. V makes 6 queries to T and 1 to q , and does $\mathcal{O}(\log n)$ field operations. P does $\tilde{O}(n)$ work to compute the trace, interpolate T , compute C , and divide by v_H .

Public coin. V 's only randomness: α . It sends α to P .

Completeness.

- Per Claim 8 ($\Rightarrow$), T satisfies (Base), (Rec), (Final)
- So, (Base) and (Final) pass.
- Then, per Claim 9, C is zero on H , so $q \cdot v_H = C$.
- So $q \cdot v_H = C$, and (Rec) passes.

(Perfect completeness.)

Soundness. Fix $x \neq f_i$ and any P^* ; it sends $T^* \in \mathbb{F}^{<n}[X]$, $q^* \in \mathbb{F}^{\leq 1}[X]$. Let C^* be built from T^* .

- Since $x \neq f_i$, per Claim 8 ($\Leftarrow$), T^* violates (Base), (Rec), or (Final).
- If it violates (Base) or (Final), then that check fails, and V rejects.

- Otherwise, it violates (Rec). Then, per Claim 9, C^* is not zero on H .
- So, per the subgroup zero test's soundness (with $d = n + 1$), (Rec) passes w.p. $\leq (n + 1)/p$.

So $\Pr[V \text{ accepts}] \leq (n + 1)/p \leq \text{negl}(\lambda)$.

– here, we use $p > 2^\lambda$ for soundness.

Compiling to a SNARG.

- P sends commitments c_T, c_q
- Fiat–Shamir: $\alpha = \text{Hash}(\phi_i, x, c_T, c_q)$
- P sends 7 evaluations + 7 evaluation proofs

With KZG, that's 2 commitments, 7 field elements, and 7 evaluation proofs – a constant-size proof, no matter how large i is. V runs 7 EVer, plus $\mathcal{O}(\log i)$ field operations.

6 Recap

- $\text{PIOP} + \text{PCS} \implies \text{SNARG}$
- today: PIOP for Fibonacci numbers
- tools: zero test; subgroup zero test (send the quotient by ν_H); shifting by ω (for induction on a trace); direct queries (for boundary conditions)
- next time: PIOP for arithmetic circuits