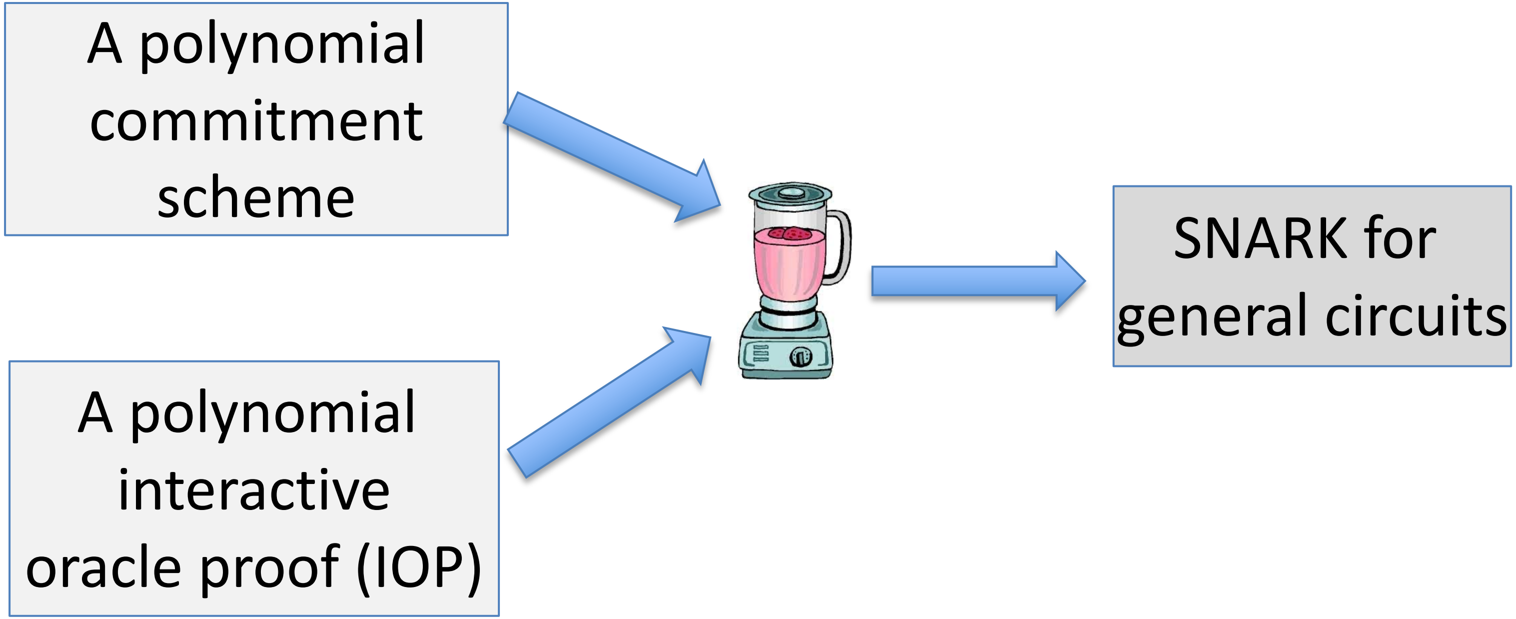
BIU Winter School on cryptography

Day 2 Lecture 4

Constructing a SNARK

Dan Boneh

Let’s build an efficient SNARK



A polynomial commitment scheme

A polynomial interactive oracle proof (IOP)

SNARK for general circuits

First, let's review poly. commitments (informally)

Prover commits to a polynomial $f(X)$ in $\mathbb{F}_p^{(\le d)}[X]$ (univariate)

- ***eval***: for public $u, v \in \mathbb{F}_p$, prover can convince the verifier that committed poly satisfies

$$\boxed{f(u) = v \quad \text{and} \quad \deg(f) \le d.}$$

verifier has (d, com_f, u, v)

$\boxed{f}$

- Eval proof size and verifier time should be $O_\lambda(\log \boldsymbol{d})$

Note: poly. commitments have many applications beyond SNARKs

Example polynomial commitments

A few examples:

- Using bilinear groups: KZG'10 (trusted setup), Dory'20, …
- Using elliptic curves: Bulletproofs (short proof, but verifier time is $O(d)$)
- Using hash functions only: based on FRI
- Using groups of unknown order: Dark'20

Proving properties of committed polynomials

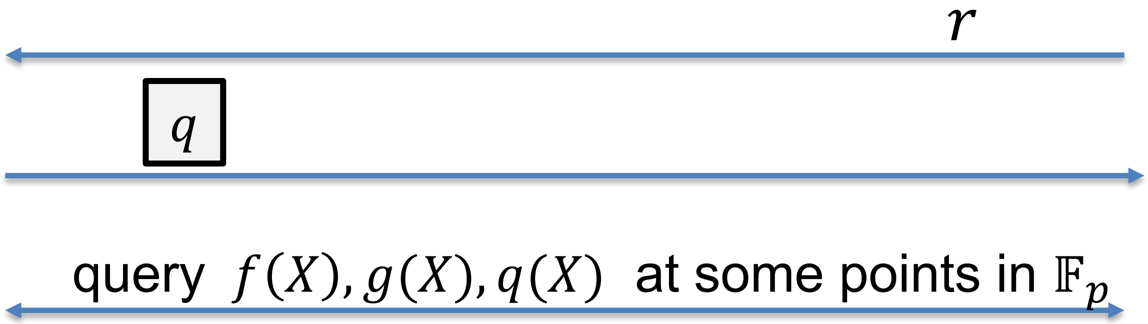
Proving properties of committed polynomials

Prover P(f, g)

Verifier V($\boxed{f}$, $\boxed{g}$)

Goal: convince verifier that $f, g \in \mathbb{F}_p^{(\leq d)}[X]$ satisfy some properties

Proof systems presented as an IOP:



r

$r \xleftarrow{\$} \mathbb{F}_p$

$\boxed{q}$

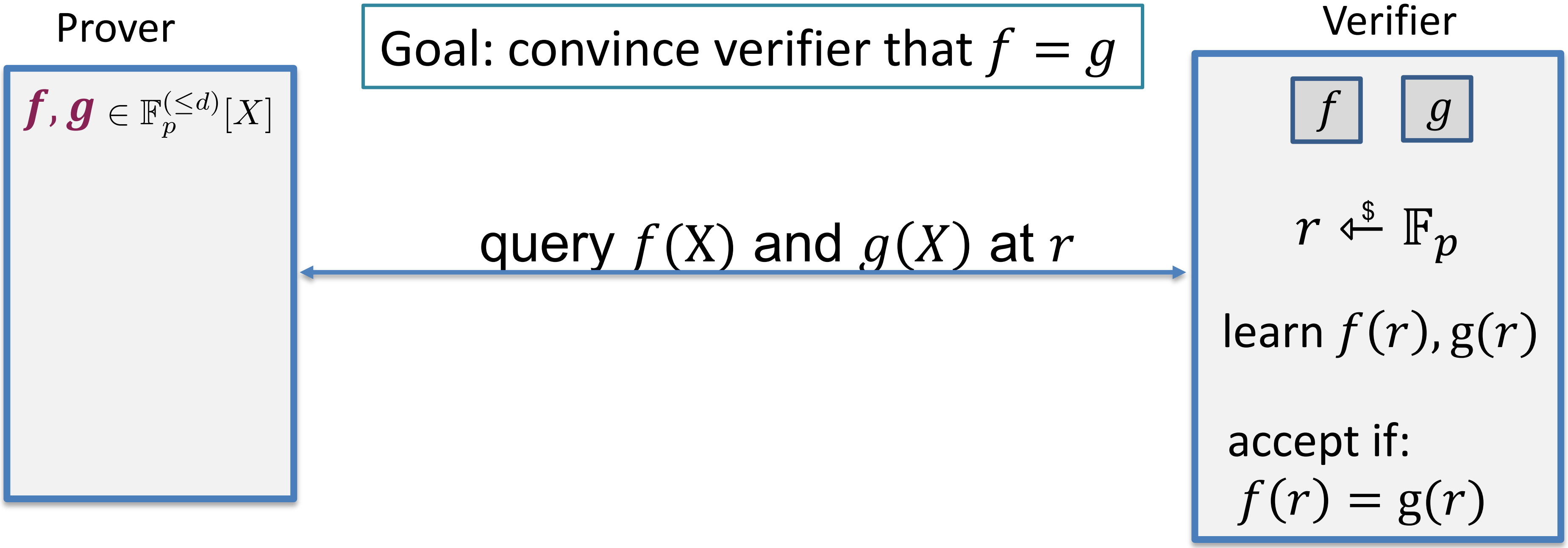
query $f(X), g(X), q(X)$ at some points in $\mathbb{F}_p$

accept or reject

Compiled protocol: V sends x to P; P responds with $f(x)$ and eval proof π

A simple example: polynomial equality testing

Goal: convince verifier that $f = g$



Prover

$\boldsymbol{f}, \boldsymbol{g} \in \mathbb{F}_p^{(\leq d)}[X]$

query $f(X)$ and $g(X)$ at r

Verifier

f g

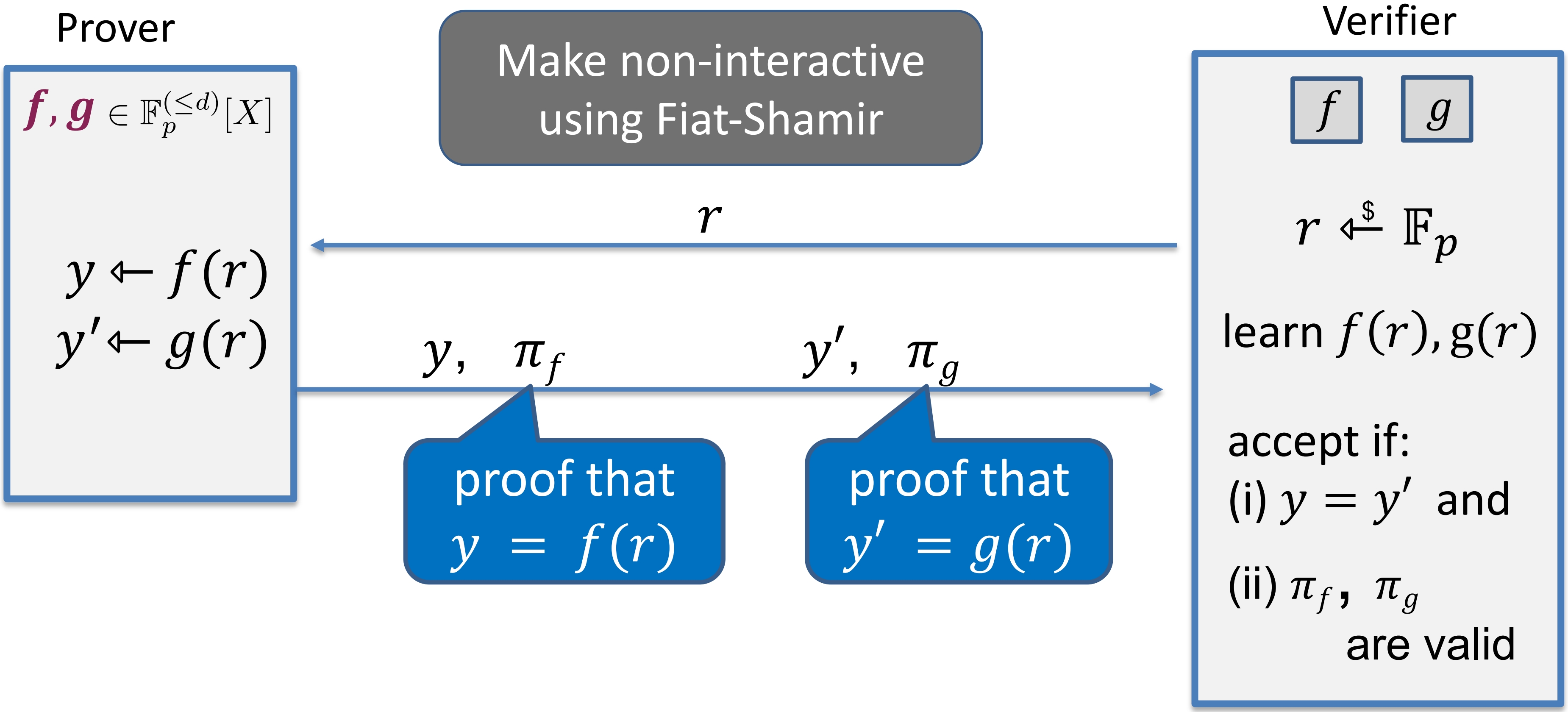
$r \xleftarrow{\$} \mathbb{F}_p$

learn $f(r), g(r)$

accept if:
$f(r) = g(r)$

Lemma: complete and sound assuming d/p is negligible

Review: the compiled proof system



Prover

$f, g \in \mathbb{F}_p^{(\leq d)}[X]$

$y \leftarrow f(r)$

$y' \leftarrow g(r)$

Make non-interactive using Fiat-Shamir

r

$y, \quad \pi_f$

$y', \quad \pi_g$

proof that $y = f(r)$

proof that $y' = g(r)$

Verifier

$\boxed{f}$ $\boxed{g}$

$r \xleftarrow{\$} \mathbb{F}_p$

learn $f(r), g(r)$

accept if:

(i) $y = y'$ and

(ii) $\pi_f, \; \pi_g$ are valid

Important proof gadgets for univariates

Let Ω be some subset of $\mathbb{F}_p$ of size k.

Let $f \in \mathbb{F}_p^{(\leq d)}[X]$ $(d \geq k)$ Verifier has $\boxed{f}$

Let us construct efficient Poly-IOPs for the following tasks:

Task 1 (**ZeroTest**): prove that f is identically zero on Ω

Task 2 (**SumCheck**): prove that $\sum_{a\in\Omega} f(a) = 0$

Task 3 (**ProdCheck**): prove that $\prod_{a\in\Omega} f(a) = 1$

The vanishing polynomial

Let Ω be some subset of $\mathbb{F}_p$ of size k.

Def: the **vanishing polynomial** of Ω is $Z_\Omega(X) \coloneqq \prod_{a\in\Omega}(X-a)$

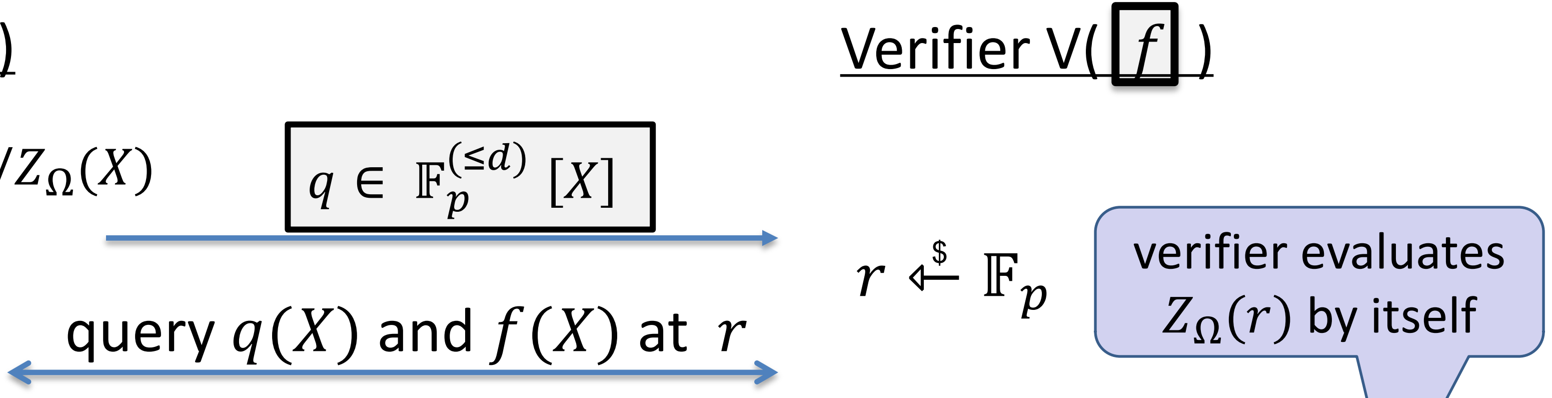
$\deg(Z_\Omega) = k$

Let $\omega \in \mathbb{F}_p$ be a primitive k-th root of unity (so that $\omega^k = 1$).

- if $\Omega = \{1, \omega, \omega^2, \ldots, \omega^{k\text{-}1}\} \subseteq \mathbb{F}_p$ then $Z_\Omega(X) = X^k - 1$

$\Rightarrow$ for $r \in \mathbb{F}_p$, evaluating $Z_\Omega(r)$ takes $2\log_2 k$ field operations

(1) ZeroTest on Ω ($\Omega = \{1, \omega, \omega^2, \ldots, \omega^{k\text{-}1}\}$)



Prover P(f)

Verifier V($\boxed{f}$)

$q(X) \leftarrow f(X)/Z_\Omega(X)$

$\boxed{q \in \mathbb{F}_p^{(\leq d)}[X]}$

$r \xleftarrow{\$} \mathbb{F}_p$

verifier evaluates $Z_\Omega(r)$ by itself

query $q(X)$ and $f(X)$ at r

learn $q(r), f(r)$

accept if $f(r) \overset{?}{=} q(r) \cdot Z_\Omega(r)$

(implies that $f(X) = q(X) \cdot Z_\Omega(X)$ w.h.p)

Lemma: f is zero on Ω if and only if $f(X)$ is divisible by $Z_\Omega(X)$

Thm: this protocol is complete and sound, assuming d/p is negligible.

(1) ZeroTest on Ω $(\Omega = \{ 1, \omega, \omega^2, \ldots, \omega^{k\text{-}1} \})$

Prover P(f)

Verifier V($\boxed{f}$)

$q(X) \leftarrow f(X)/Z_\Omega(X)$

$\boxed{q \in \mathbb{F}_p^{(\le d)}[X]}$ $\longrightarrow$

$r \xleftarrow{\$} \mathbb{F}_p$

verifier evaluates $Z_\Omega(r)$ by itself

query $q(X)$ and $f(X)$ at r $\longleftrightarrow$

learn $q(r)$, $f(r)$

Lemma: f is zero on Ω if and only if $f(X)$ is divisible by $Z_\Omega(X)$

accept if $f(r) \overset{?}{=} q(r) \cdot Z_\Omega(r)$

(implies that $f(X) = q(X) \cdot Z_\Omega(X)$ w.h.p)

Verifier time: O(log k) and two poly queries (but can be batched)

Prover time: dominated by time to compute $q(X)$ and commit to $q(X)$

(4) Another useful gadget: permutation check

Let f, g polynomials in $\mathbb{F}_p^{(\leq d)}[X]$. Verifier has $\boxed{f}$, $\boxed{g}$.

Prover wants to prove that $(\, f(1), f(\omega), f(\omega^2), \dots, f(\omega^{k-1})\,) \in \mathbb{F}_p^k$

is a permutation of $(\, g(1), g(\omega), g(\omega^2), \dots, g(\omega^{k-1})\,) \in \mathbb{F}_p^k$

⇒ Proves that $g(\Omega)$ is the same as $f(\Omega)$, just permuted

(4) Another useful gadget: permutation check

Prover P(f, g) Verifier V($\boxed{f}$, $\boxed{g}$)

Let $\hat{f}(X) = \prod_{a\in\Omega}(X - f(a))$ and $\hat{g}(X) = \prod_{a\in\Omega}(X - g(a))$

Then: $\hat{f}(X) = \hat{g}(X) \iff g(\Omega)$ is a permutation of $f(\Omega)$

$$\xleftarrow{\qquad\qquad r \qquad\qquad} \quad r \overset{\$}{\leftarrow} \mathbb{F}_p$$

prove that $\hat{f}(r) = \hat{g}(r)$

prod-check: $\frac{\hat{f}(r)}{\hat{g}(r)} = \prod_{a\in\Omega}\left(\frac{r - f(a)}{r - g(a)}\right) = 1$

$$\xleftrightarrow{\qquad\qquad\qquad\qquad}$$

implies $\hat{f}(X) = \hat{g}(X)$ w.h.p

accept or reject

[Lipton's trick, 1989]

(5) final gadget: prescribed permutation check

$W: \Omega \rightarrow \Omega$ is a **permutation of** Ω if $\forall i \in [k]$: $W(\omega^i) = \omega^j$ a bijection

example $(k = 3)$: $W(\omega^0) = \omega^2$, $W(\omega^1) = \omega^0$, $W(\omega^2) = \omega^1$

Let f, g polynomials in $\mathbb{F}_p^{(\leq d)}[X]$. Verifier has $\boxed{f}$, $\boxed{g}$, $\boxed{W}$.

<u>Goal</u>: prover wants to prove that $f(y) = g(W(y))$ for all $y \in \Omega$

$\Rightarrow$ Proves that $g(\Omega)$ is the same as $f(\Omega)$, permuted by the prescribed W

Prescribed permutation check

How? Use a zero-test to prove $\boxed{f(y) - g\big(W(y)\big) = 0 \quad \text{on } \Omega}$

The problem: the polynomial $f(y) - g\big(W(y)\big)$ has degree k^2

$\Rightarrow$ prover would need to manipulate polynomials of degree k^2

$\Rightarrow$ quadratic time prover !! (goal: linear time prover)

Can reduce this to a prod-check on a poly of degree $2k$ (not k^2)

Summary of proof gadgets

polynomial equality testing

zero test on Ω

product check, sum check

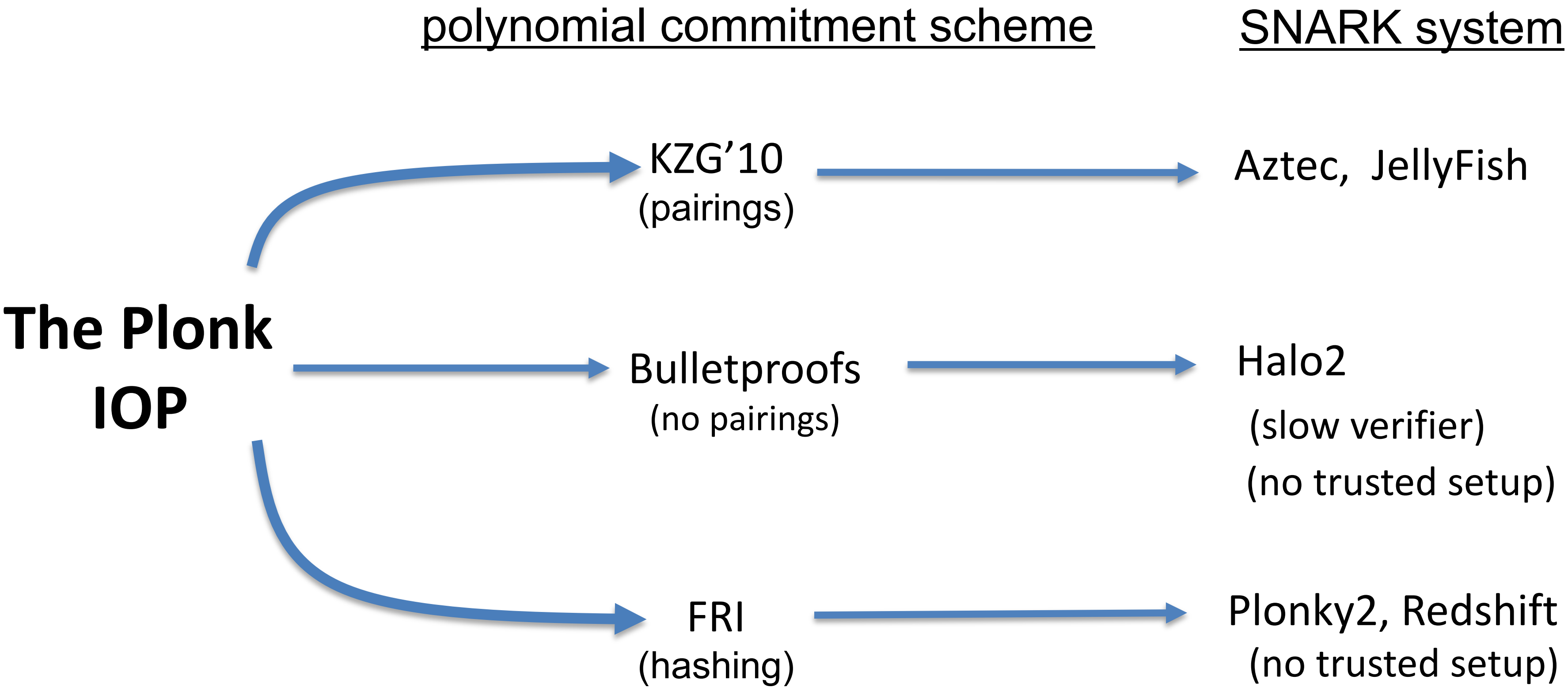
permutation check

prescribed permutation check

The PLONK IOP
for general circuits

eprint/2019/953

PLONK: widely used in practice



polynomial commitment scheme

SNARK system

The Plonk IOP

KZG'10
(pairings)

Aztec, JellyFish

Bulletproofs
(no pairings)

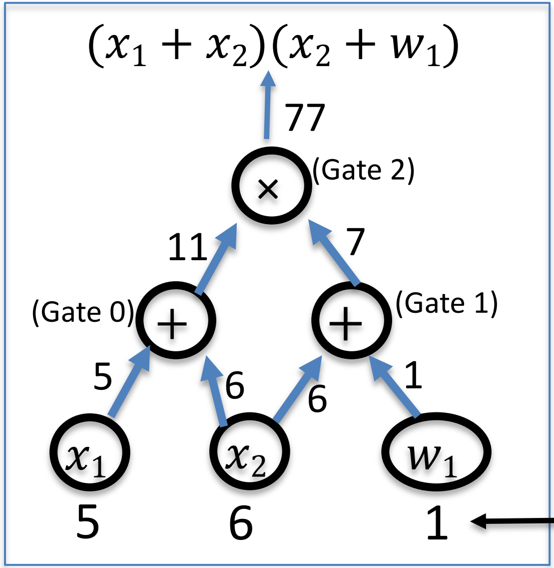
Halo2
(slow verifier)
(no trusted setup)

FRI
(hashing)

Plonky2, Redshift
(no trusted setup)

PLONK: a poly-IOP for a general circuit $C(x, w)$

Step 1: compile circuit to a computation trace (gate fan-in = 2)



example input

The computation trace (arithmetization):

inputs:	5,	6,	1
Gate 0:	5 ,	6 ,	11
Gate 1:	6 ,	1 ,	7
Gate 2:	11,	7,	77

left inputs

right inputs

outputs

Encoding the trace as a polynomial

$|C| :=$ total # of gates in C , $|I| := |I_x| + |I_w| =$ # inputs to C

let $d := 3\,|C| + |I|$ (in example, $d = 12$) and $\Omega := \{\, 1, \omega, \omega^2, \ldots, \omega^{d-1} \,\}$

The plan:

prover interpolates a poly. $T \in \mathbb{F}_p^{(\leq d)}[X]$

that encodes the entire trace.

Let’s see how …

inputs:	5,	6,	1
Gate 0:	5 ,	6 ,	11
Gate 1:	6 ,	1 ,	7
Gate 2:	11,	7,	77

Encoding the trace as a polynomial

The plan: Prover interpolates $T \in \mathbb{F}_p^{(\leq d)}[\mathrm{X}]$ such that

(1) ***T* encodes all inputs**: $\mathrm{T}(\omega^{-j}) = \text{input } \#j$ for $j = 1, \ldots, |I|$

(2) ***T* encodes all wires**: $\forall\, l = 0, \ldots, |C| - 1$:

- $\mathrm{T}(\omega^{3l})$: left input to gate $\#l$
- $\mathrm{T}(\omega^{3l+1})$: right input to gate $\#l$
- $\mathrm{T}(\omega^{3l+2})$: output of gate $\#l$

inputs:	5,	6,	1
Gate 0:	5 ,	6 ,	11
Gate 1:	6 ,	1 ,	7
Gate 2:	11,	7,	77

Encoding the trace as a polynomial

In our example, Prover interpolates $T(X)$ such that:

inputs: $\mathrm{T}(\omega^{-1}) = 5,\quad \mathrm{T}(\omega^{-2}) = 6,\quad \mathrm{T}(\omega^{-3}) = 1,$
gate 0: $\mathrm{T}(\omega^{0}) = 5,\quad \mathrm{T}(\omega^{1}) = 6,\quad \mathrm{T}(\omega^{2}) = 11,$
gate 1: $\mathrm{T}(\omega^{3}) = 6,\quad \mathrm{T}(\omega^{4}) = 1,\quad \mathrm{T}(\omega^{5}) = 7,$
gate 2: $\mathrm{T}(\omega^{6}) = 11,\quad \mathrm{T}(\omega^{7}) = 7,\quad \mathrm{T}(\omega^{8}) = 77$

degree(T) = 11

inputs:	5,	6,	1
Gate 0:	5 ,	6 ,	11
Gate 1:	6 ,	1 ,	7
Gate 2:	11,	7,	77

Prover can use NTT to compute the coefficients of T in time $O(d \log d)$

Step 2: proving validity of T

Prover P($S_p, \boldsymbol{x}, \mathbf{w}$)

Verifier V($S_v, \boldsymbol{x}$)

build T(X) ∈ $\mathbb{F}_p^{(\leq d)}$[X]

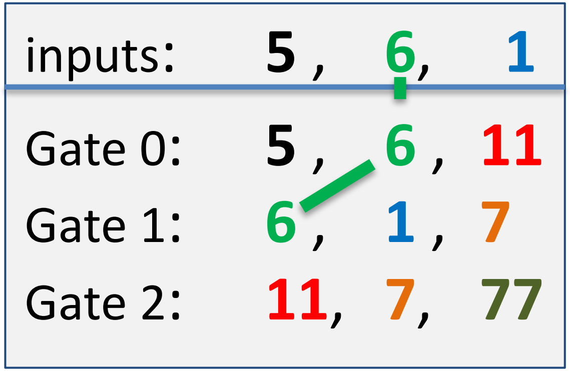
$$\xrightarrow{\boxed{T}}$$

Prover needs to prove that T is a correct computation trace:

1. (1) T encodes the correct inputs,
2. (2) every gate is evaluated correctly,
3. (3) the wiring is implemented correctly,
4. (4) the output of last gate is 0

Proving (4) is easy: prove $T(\omega^{3|C|-1}) = 0$

(wiring constraints)



inputs:	5	6	1
Gate 0:	5	6	11
Gate 1:	6	1	7
Gate 2:	11	7	77

Proving (1): T encodes the correct inputs

Both prover and verifier interpolate a polynomial $v(X) \in \mathbb{F}_p^{(\leq|I_x|)}[X]$ that encodes the x-inputs to the circuit:

for $j = 1, \ldots, |I_x|$: $v(\omega^{-j})$ = input #j

In our example: $v(\omega^{-1}) = 5$, $v(\omega^{-2}) = 6$. (v is linear)

constructing $v(X)$ takes time proportional to the size of input x

⇒ verifier has time do this

Proving (1): T encodes the correct inputs

Both prover and verifier interpolate a polynomial $v(X) \in \mathbb{F}_p^{(\leq|I_x|)}[X]$ that encodes the x-inputs to the circuit:

$$\text{for } j = 1, \dots, |I_x|: \qquad v(\omega^{-j}) = \text{input \#j}$$

Let $\Omega_{\text{inp}} := \{\, \omega^{-1}, \omega^{-2}, \dots, \omega^{-|I_x|} \,\} \subseteq \Omega$ (points encoding the input)

Prover proves (1) by using a ZeroTest on Ω_{inp} to prove that

$$\boxed{T(y) - v(y) = 0 \qquad \forall\, y \in \Omega_{\text{inp}}}$$

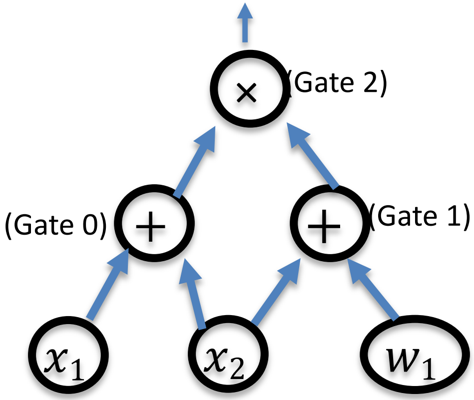
Proving (2): every gate is evaluated correctly

Idea: encode gate types using a *selector* polynomial S(X)

define S(X) $\in \mathbb{F}_p^{(\le d)}[X]$ such that $\forall\, l = 0, \dots, |C| - 1$:

S(ω^{3l}) = 1 if gate #l is an addition gate

S(ω^{3l}) = 0 if gate #l is a multiplication gate



inputs:	5 ,	6,	1	$S(X)$	
Gate 0 (ω^0):	5 ,	6 ,	11	1	(+)
Gate 1 (ω^3):	6 ,	1 ,	7	1	(+)
Gate 2 (ω^6):	11,	7,	77	0	(×)

Proving (2): every gate is evaluated correctly

Idea: encode gate types using a *selector* polynomial S(X)

define $S(X) \in \mathbb{F}_p^{(\leq d)}[X]$ such that $\forall\, l = 0, \ldots, |C| - 1$:

$S(\omega^{3l}) = 1$ if gate $\#l$ is an addition gate

$S(\omega^{3l}) = 0$ if gate $\#l$ is a multiplication gate

Then $\forall\, y \in \Omega_{\text{gates}} := \{\, 1, \omega^3, \omega^6, \omega^9, \ldots, \omega^{3(|C|-1)} \,\}$:

$$S(y)\cdot[\underbrace{T(y)}_{\text{left input}} + \underbrace{T(\omega y)}_{\text{right input}}] \;+\; (1 - S(y))\cdot\underbrace{T(y)}_{\text{left input}}\cdot\underbrace{T(\omega y)}_{\text{right input}} \;=\; \underbrace{T(\omega^2 y)}_{\text{output}}$$

Proving (2): every gate is evaluated correctly

$$\text{Setup}(C) \;\rightarrow\; pp{:=}\text{S} \quad \text{and} \quad vp{:=}\left(\; \boxed{\text{S}} \;\right)$$

Prover P($pp, \boldsymbol{x}, \mathbf{w}$) Verifier V($vp, \boldsymbol{x}$)

build T(X) ∈ $\mathbb{F}_p^{(\leq d)}[\text{X}]$ $\xrightarrow{\quad\boxed{T}\quad}$

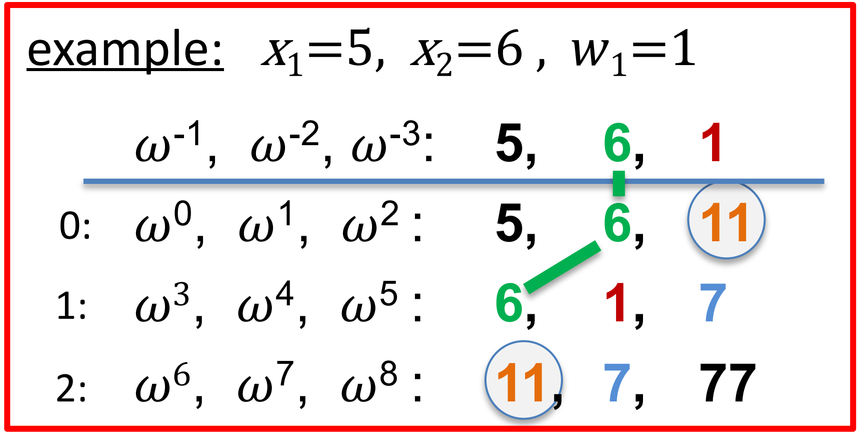
Prover uses ZeroTest to prove that for all $\forall\, y \in \Omega_{gates}$:

$$\boxed{\text{S}(\text{y})\cdot[\text{T}(\text{y}) + \text{T}(\omega\text{y})] \;+\; (1 - \text{S}(\text{y}))\cdot\text{T}(\text{y})\cdot\text{T}(\omega\text{y}) \;-\; \text{T}(\omega^2\text{y}) \;=\; 0}$$

Proving (3): the wire is correct

Step 4: encode the wires of C:

$$\left\{\begin{array}{l} \mathrm{T}(\omega^{-2}) = \mathrm{T}(\omega^{1}) = \mathrm{T}(\omega^{3}) \\ \mathrm{T}(\omega^{-1}) = \mathrm{T}(\omega^{0}) \\ \mathrm{T}(\omega^{2}) = \mathrm{T}(\omega^{6}) \\ \mathrm{T}(\omega^{-3}) = \mathrm{T}(\omega^{4}) \end{array}\right.$$



<u>example:</u> $x_1=5$, $x_2=6$, $w_1=1$

	$\omega^{-1}, \omega^{-2}, \omega^{-3}$:	5,	6,	1
0:	$\omega^{0}, \omega^{1}, \omega^{2}$:	5,	6,	11
1:	$\omega^{3}, \omega^{4}, \omega^{5}$:	6,	1,	7
2:	$\omega^{6}, \omega^{7}, \omega^{8}$:	11,	7,	77

Define a polynomial W: $\Omega \rightarrow \Omega$ that implements a rotation:

$$\mathrm{W}(\omega^{-2}, \omega^{1}, \omega^{3}) = (\omega^{1}, \omega^{3}, \omega^{-2}) \;, \quad \mathrm{W}(\omega^{-1}, \omega^{0}) = (\omega^{0}, \omega^{-1}) \;, \;\ldots$$

<u>**Lemma**</u>: $\forall\, y \in \Omega$: $\mathrm{T}(y) = \mathrm{T}(\mathrm{W}(y))$ ⇒ wire constraints are satisfied

Proving (3): the wiring is correct

Step 4: encode the wires of C:

$$\begin{cases} T(\omega^{-2}) = T(\omega^{1}) = T(\omega^{3}) \\ T(\omega^{-1}) = T(\omega^{0}) \\ T(\omega^{2}) = T(\omega^{6}) \end{cases}$$

example: $x_1=5,\ x_2=6,\ w_1=1$

	$\omega^{-1},\ \omega^{-2},\ \omega^{-3}$:	5,	6,	1
0:	$\omega^{0},\ \omega^{1},\ \omega^{2}$:	5,	6,	11
1:	$\omega^{3},\ \omega^{4},\ \omega^{5}$:	6,	1,	7
				77

Proved using a prescribed permutation check

Define a polynom $\rightarrow \Omega$ that implements a rotation:

$W(\omega^{-2}, \omega^{1}, \omega^{3})$ $\omega^{3}, \omega^{-2})$, $W(\omega^{-1}, \omega^{0}) = (\omega^{0}, \omega^{-1})$, …

Lemma: $\forall\, y \in \Omega$: $T(y) = T(W(y)) \Rightarrow$ wire constraints are satisfied

The complete Plonk Poly-IOP (and SNARK)

Setup(C) → $pp := (S, W)$ and $vp := (\boxed{S}$ and $\boxed{W})$ (untrusted)

Prover P($pp, \boldsymbol{x}, \mathbf{w}$) | Verifier V($vp, \boldsymbol{x}$)

build T(X) ∈ $\mathbb{F}_p^{(\leq d)}[X]$ $\xrightarrow{\boxed{T}}$ build $v(X) \in \mathbb{F}_p^{(\leq|I_x|)}[X]$

Prover proves:

gates: (1) $S(y)\cdot[T(y) + T(\omega y)] + (1 - S(y))\cdot T(y)\cdot T(\omega y) - T(\omega^2 y) = 0$; $\forall\ y \in \Omega_{\text{gates}}$

inputs: (2) $T(y) - v(y) = 0$ $\forall\ y \in \Omega_{\text{inp}}$

wires: (3) $T(y) - T(W(y)) = 0$ (using prescribed perm. check) $\forall\ y \in \Omega$

output: (4) $T(\omega^{3|C|-1}) = 0$ (output of last gate = 0)

The complete Plonk Poly-IOP (and SNARK)

Setup(C) → $pp := (S, W)$ and $vp := (\boxed{S}$ and $\boxed{W})$ (untrusted)

Prover P($pp, \boldsymbol{x}, \mathbf{w}$)

Verifier V($vp, \boldsymbol{x}$)

build T(X) ∈ $\mathbb{F}_p^{(\le d)}[X]$ $\xrightarrow{\boxed{T}}$ build $v(X) \in \mathbb{F}_p^{(\le |I_x|)}[X]$

Thm: The Plonk Poly-IOP is complete and knowledge sound, assuming $7|C|/p$ is negligible

(eprint/2019/953)

Many extensions …

- Plonk proof: a short proof (O(1) commitments), fast verifier
- The SNARK can be made into a zk-SNARK

Main challenge: reduce prover time

- **Hyperplonk**: replace Ω with $\{0,1\}^t$ (where $t = \log_2|\Omega|$)
  - The polynomial T is now a multilinear polynomial in t variables
  - ZeroTest is replaced by a multilinear SumCheck (linear time)

A generalization: plonkish arithmetization

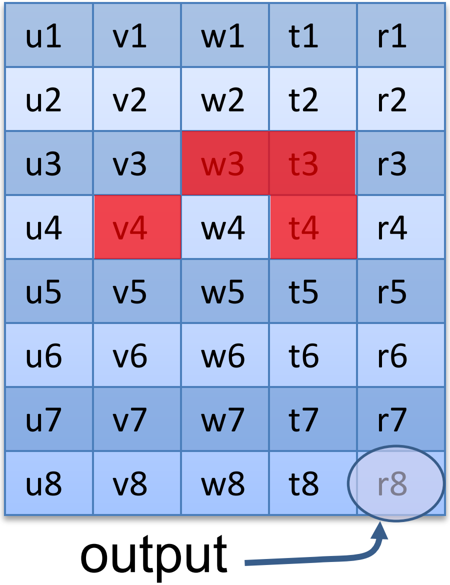
Plonk for circuits with gates other than + and × on rows:

Plonkish computation trace: (also used in AIR)

An example custom gate:

$$\boxed{\forall\, y \in \Omega: \quad v(y\omega) + w(y) \cdot t(y) - t(y\omega) = 0}$$

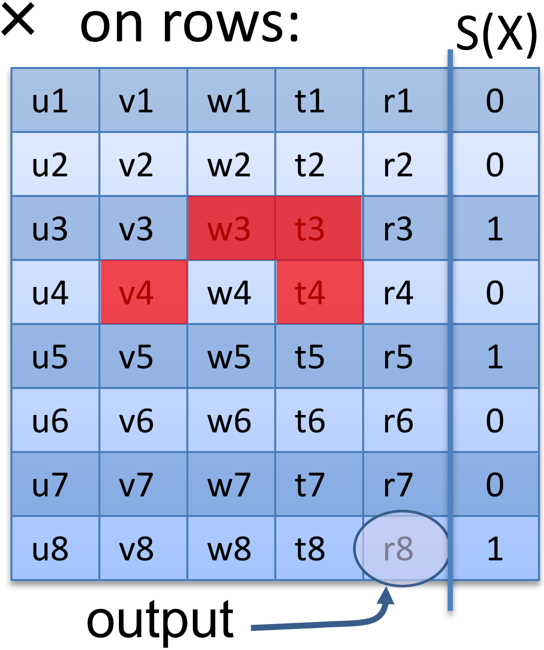
All such gate checks are included in the gate check



u1	v1	w1	t1	r1
u2	v2	w2	t2	r2
u3	v3	w3	t3	r3
u4	v4	w4	t4	r4
u5	v5	w5	t5	r5
u6	v6	w6	t6	r6
u7	v7	w7	t7	r7
u8	v8	w8	t8	r8

A generalization: plonkish arithmetization

Plonk for circuits with gates other than + and × on rows:



					S(X)
u1	v1	w1	t1	r1	0
u2	v2	w2	t2	r2	0
u3	v3	w3	t3	r3	1
u4	v4	w4	t4	r4	0
u5	v5	w5	t5	r5	1
u6	v6	w6	t6	r6	0
u7	v7	w7	t7	r7	0
u8	v8	w8	t8	r8	1

Plonkish computation trace: (also used in AIR)

An example custom gate:

$$\forall\, y \in \Omega:\ \boxed{\mathrm{S(X)}} \cdot [v(y\omega) + w(y)\cdot t(y) - t(y\omega)] = 0$$

Selector poly $S(X)$ can choose when to apply gate

Plookup: ensure some values are in a pre-defined list ... tomorrow

THE END