STARK Arithmetization

Eli Ben-Sasson
Chief Scientist (East)

February 2019
Succinct Computational Integrity and Privacy

Goals

- Given (i) program \( P \), (ii) input \( x_{in} \), (iii) time bound \( T \)
- Bob claims \( P(x_{in}, w) = x_{out} \) after \( T \) steps, \( w \) is auxiliary (private) input
Succinct Computational Integrity and Privacy

Goals

- Given (i) program $P$, (ii) input $x_{in}$, (iii) time bound $T$
- Bob claims $P(x_{in}, w) = x_{out}$ after $T$ steps, $w$ is auxiliary (private) input
- Goals of proof system:
  - **Integrity**: Is the claim correct?
  - **Privacy**: Prevent proof from leaking $w$
  - **Succinctness**: Verify proof in time $\text{polylog}(T)$
  - **Knowledge**: Does Bob know $w$?
Succinct Computational Integrity and Privacy

Goals

- Given (i) program $P$, (ii) input $x_{in}$, (iii) time bound $T$
- Bob claims $P(x_{in}, w) = x_{out}$ after $T$ steps, $w$ is auxiliary (private) input

Goals of proof system:

- **Integrity**: Is the claim correct?
- **Privacy**: Prevent proof from leaking $w$
- **Succinctness**: Verify proof in time $\text{polylog}(T)$
- **Knowledge**: Does Bob know $w$?

Notice the problem is a special case of checking membership (of $(P, x_{in}, x_{out}, T)$) in some nondeterministic language $L$ (called the universal language, computational integrity language, . . . )
Arithmetization

- **Arithmetization**: reduction of computational problems like...
  - is $x$ a member of language $L \in NTIME(T(n))$?

...to algebraic coding problems like

- is $f : S \rightarrow \mathbb{F}$ the evaluation of a polynomial of degree $< \frac{|S|}{8}$?
Arithmetization

- **Arithmetization**: reduction of computational problems like . . .
  - is $x$ a member of language $L \in NTIME(T(n))$?
- . . . to algebraic coding problems like
  - is $f : S \rightarrow \mathbb{F}$ the evaluation of a polynomial of degree $< \frac{|S|}{8}$?

- **Brief history of arithmetization**
  - Gödel 1930's: Incompleteness theorem
  - Razborov 1980's: lower bounds on circuit size
  - Lund, Fortnow, Karloff, Nisan, late 1980's: Interactive proofs
Arithmetization

- **Arithmetization**: reduction of computational problems like . . .
  - is $x$ a member of language $L \in NTIME(T(n))$?
  
  . . . to algebraic coding problems like
  - is $f : S \rightarrow \mathbb{F}$ the evaluation of a polynomial of degree $< \frac{|S|}{8}$?

- Brief history of arithmetization
  - Gödel 1930's: Incompleteness theorem
  - Razborov 1980's: lower bounds on circuit size
  - Lund, Fortnow, Karloff, Nisan, late 1980's: Interactive proofs

- Why arithmetization?
  - polynomials are excellent error correcting codes (ECCs)
  - ECCs add redundancy and "spread information"
  - this amplifies the noticeability of errors/cheats
Arithmetization

▶ Arithmetization: reduction of computational problems like . . .
  ▶ is x a member of language L ∈ NTIME(T(n))?
  . . . to algebraic coding problems like
  ▶ is f : S → F the evaluation of a polynomial of degree < \frac{|S|}{8}?

▶ Brief history of arithmetization
  ▶ Gödel 1930’s: Incompleteness theorem
  ▶ Razborov 1980’s: lower bounds on circuit size
  ▶ Lund, Fortnow, Karloff, Nisan, late 1980’s: Interactive proofs

▶ Why arithmetization?
  ▶ polynomials are excellent error correcting codes (ECCs)
  ▶ ECCs add redundancy and “spread information”
  ▶ this amplifies the noticeability of errors/cheats

▶ Talk tl;dr: Arithmetization ↷ Succinctness & ZK
Arithmetization

- **Arithmetization**: reduction of computational problems like . . .
  - is $x$ a member of language $L \in \text{NTIME}(T(n))$?
  
  . . . to algebraic coding problems like
  - is $f : S \rightarrow \mathbb{F}$ the evaluation of a polynomial of degree $< \frac{|S|}{8}$?

- Brief history of arithmetization
  - Gödel 1930's: Incompleteness theorem
  - Razborov 1980’s: lower bounds on circuit size
  - Lund, Fortnow, Karloff, Nisan, late 1980’s: Interactive proofs

- Why arithmetization?
  - polynomials are excellent error correcting codes (ECCs)
  - ECCs add redundancy and “spread information”
  - this amplifies the noticeability of errors/cheats

- Talk tl;dr: Arithmetization $\rightsquigarrow$ Succinctness & ZK

- Work in IOP model: prover sends functions, verifier pays per query
Fact 1: If $H \subset \mathbb{F}$ multiplicative subgroup, $|H| = h$, then

$$Z_H(X) = \prod_{\alpha \in H} (X - \alpha) = X^h - 1$$ vanishes on $H$.

Evaluating $Z_H(\beta)$ requires $O(\log h)$ arithmetic operations.

Fact 2: $P(\gamma) = 0$ iff there exists $\tilde{P}(X)$ satisfying

$$\deg(\tilde{P}) = \deg(P) - 1,$$

$$(X - \gamma) \cdot \tilde{P}(X) = P(X),$$

So $P(X)$ vanishes on $H$ iff $\exists \tilde{P}(X)$ satisfying

$$\deg(\tilde{P}) = \deg(P) - h,$$

$$Z_H \cdot \tilde{P}(X) = P(X).$$

Fact 3: Two distinct polynomials of degree $d$ intersect at $\leq d$ points (e.g., two distinct lines intersect at $\leq 1$ point).

So: two distinct functions of degree $d$ evaluated at $100 \cdot d$ points are $99\%$-far in relative hamming distance.

Corollary: space of low-degree functions forms a linear error correcting code, called the Reed-Solomon (RS) code (suggested as code – 1960's).
Useful Polynomial facts

- **Fact 1**: If $H \subset \mathbb{F}$ multiplicative subgroup, $|H| = h$, then
  - The polynomial $Z_H(X) = \prod_{\alpha \in H} (X - \alpha) = X^h - 1$ vanishes on $H$
Useful Polynomial facts

- **Fact 1**: If $H \subset \mathbb{F}$ multiplicative subgroup, $|H| = h$, then
  - The polynomial $Z_H(X) = \prod_{\alpha \in H} (X - \alpha) = X^h - 1$ vanishes on $H$
  - Evaluating $Z_H(\beta)$ requires $O(\log h)$ arithmetic operations

- **Fact 2**: $P(\gamma) = 0$ iff there exists $\tilde{P}(X)$ satisfying
Useful Polynomial facts

- **Fact 1:** If $H \subset \mathbb{F}$ multiplicative subgroup, $|H| = h$, then
  - The polynomial $Z_H(X) = \prod_{\alpha \in H} (X - \alpha) = X^h - 1$ vanishes on $H$
  - Evaluating $Z_H(\beta)$ requires $O(\log h)$ arithmetic operations

- **Fact 2:** $P(\gamma) = 0$ iff there exists $\tilde{P}(X)$ satisfying
  - $\deg(\tilde{P}) = \deg(P) - 1$,
  - $(X - \gamma) \cdot \tilde{P}(X) = P(X)$

- **Fact 3:** Two distinct polynomials of degree $d$ intersect at $\leq d$ points (e.g., two distinct lines intersect at $\leq 1$ point)
  - So: two distinct functions of degree $d$ evaluated at $100 \cdot d$ points are $99\%-far$ in relative hamming distance
  - Corollary: space of low-degree functions forms a linear error correcting code, called the Reed-Solomon (RS) code (suggested as code – 1960's)
Useful Polynomial facts

- **Fact 1:** If $H \subset \mathbb{F}$ multiplicative subgroup, $|H| = h$, then
  - The polynomial $Z_H(X) = \prod_{\alpha \in H} (X - \alpha) = X^h - 1$ vanishes on $H$
  - Evaluating $Z_H(\beta)$ requires $O(\log h)$ arithmetic operations

- **Fact 2:** $P(\gamma) = 0$ iff there exists $\tilde{P}(X)$ satisfying
  - $\deg(\tilde{P}) = \deg(P) - 1$,
  - $(X - \gamma) \cdot \tilde{P}(X) = P(X)$

So $P(X)$ vanishes on $H$ iff $\exists \tilde{P}(X)$ satisfying
  - $\deg(\tilde{P}) = \deg(P) - h$,
  - $Z_H \cdot \tilde{P}(X) = P(X)$
Useful Polynomial facts

- **Fact 1**: If $H \subseteq \mathbb{F}$ multiplicative subgroup, $|H| = h$, then
  - The polynomial $Z_H(X) = \prod_{\alpha \in H} (X - \alpha) = X^h - 1$ vanishes on $H$
  - Evaluating $Z_H(\beta)$ requires $O(\log h)$ arithmetic operations

- **Fact 2**: $P(\gamma) = 0$ iff there exists $\tilde{P}(X)$ satisfying
  - $\deg(\tilde{P}) = \deg(P) - 1$,
  - $(X - \gamma) \cdot \tilde{P}(X) = P(X)$

  So $P(X)$ vanishes on $H$ iff $\exists \tilde{P}(X)$ satisfying
  - $\deg(\tilde{P}) = \deg(P) - h$,
  - $Z_H \cdot \tilde{P}(X) = P(X)$

- **Fact 3**: Two distinct polynomials of degree $d$ intersect at $\leq d$ points (e.g., two distinct lines intersect at $\leq 1$ point)
Fact 1: If $H \subset \mathbb{F}$ multiplicative subgroup, $|H| = h$, then
- The polynomial $Z_H(X) = \prod_{\alpha \in H} (X - \alpha) = X^h - 1$ vanishes on $H$.
- Evaluating $Z_H(\beta)$ requires $O(\log h)$ arithmetic operations.

Fact 2: $P(\gamma) = 0$ iff there exists $\tilde{P}(X)$ satisfying
- $\deg(\tilde{P}) = \deg(P) - 1$,
- $(X - \gamma) \cdot \tilde{P}(X) = P(X)$

So $P(X)$ vanishes on $H$ iff $\exists \tilde{P}(X)$ satisfying
- $\deg(\tilde{P}) = \deg(P) - h$,
- $Z_H \cdot \tilde{P}(X) = P(X)$

Fact 3: Two distinct polynomials of degree $d$ intersect at $\leq d$ points (e.g., two distinct lines intersect at $\leq 1$ point).
So: two distinct functions of degree $d$ evaluated at $100 \cdot d$ points are $99\%$-far in relative hamming distance.
Useful Polynomial facts

- **Fact 1**: If $H \subset \mathbb{F}$ multiplicative subgroup, $|H| = h$, then
  - The polynomial $Z_H(X) = \prod_{\alpha \in H} (X - \alpha) = X^h - 1$ vanishes on $H$
  - Evaluating $Z_H(\beta)$ requires $O(\log h)$ arithmetic operations

- **Fact 2**: $P(\gamma) = 0$ iff there exists $\tilde{P}(X)$ satisfying
  - $\deg(\tilde{P}) = \deg(P) - 1$,
  - $(X - \gamma) \cdot \tilde{P}(X) = P(X)$
  
  So $P(X)$ vanishes on $H$ iff $\exists \tilde{P}(X)$ satisfying
  - $\deg(\tilde{P}) = \deg(P) - h$,
  - $Z_H \cdot \tilde{P}(X) = P(X)$

- **Fact 3**: Two distinct polynomials of degree $d$ intersect at $\leq d$ points
  (e.g., two distinct lines intersect at $\leq 1$ point)
  
  So: two distinct functions of degree $d$ evaluated at $100 \cdot d$ points are 99%-far in relative hamming distance

- **Corollary**: space of low-degree functions forms a linear error correcting code, called the Reed-Solomon (RS) code (suggested as code – 1960’s)
Fact 1: If $H \subset F$ mult. group, $|H| = h$, then $Z_H(\beta) = \beta^h - 1$ evaluated in time $O(\log h)$

Fact 2: $P(X)$ vanishes on $H \iff \exists \tilde{P}(X), \deg(\tilde{P}) = \deg(P) - h$ and $Z_H \cdot \tilde{P}(X) = P(X)$

Fact 3: Two distinct degree $d$ functions evaluated at $100 \cdot d$ points are 99%-far
Fact 1: If $H \subset \mathbb{F}$ mult. group, $|H| = h$, then $Z_H(\beta) = \beta^h - 1$ evaluated in time $O(\log h)$

Fact 2: $P(X)$ vanishes on $H \iff \exists \tilde{P}(X), \deg(\tilde{P}) = \deg(P) - h$ and $Z_H \cdot \tilde{P}(X) = P(X)$

Fact 3: Two distinct degree $d$ functions evaluated at $100 \cdot d$ points are 99%-far

Suppose: prover uses only degree-$d$ polynomials
Computational integrity, succinctness and arithmetization

- **Fact 1:** If $H \subset \mathbb{F}$ mult. group, $|H| = h$, then $Z_H(\beta) = \beta^h - 1$ evaluated in time $O(\log h)$
- **Fact 2:** $P(X)$ vanishes on $H \iff \exists \tilde{P}(X), \deg(\tilde{P}) = \deg(P) - h$ and $Z_H \cdot \tilde{P}(X) = P(X)$
- **Fact 3:** Two distinct degree $d$ functions evaluated at $100 \cdot d$ points are $99\%$-far

**Suppose:** prover uses only degree-$d$ polynomials

**Challenge 1:** Given $f : \mathbb{F} \to \mathbb{F}, \deg(f) = d < |\mathbb{F}|/100$, devise protocol for checking succinctly and with small error if $f$ vanishes on $H$
Computational integrity, succinctness and arithmetization

- **Fact 1:** If $H \subseteq \mathbb{F}$ mult. group, $|H| = h$, then $Z_H(\beta) = \beta^h - 1$ evaluated in time $O(\log h)$

- **Fact 2:** $P(X)$ vanishes on $H \iff \exists \tilde{P}(X), \deg(\tilde{P}) = \deg(P) - h$ and $Z_H \cdot \tilde{P}(X) = P(X)$

- **Fact 3:** Two distinct degree $d$ functions evaluated at $100 \cdot d$ points are 99%-far

Suppose: prover uses only degree-$d$ polynomials

**Challenge 1:** Given $f : \mathbb{F} \to \mathbb{F}, \deg(f) = d < |\mathbb{F}|/100$, devise protocol for checking succinctly and with small error if $f$ vanishes on $H$

**The (IOP) protocol:**

- Prover sends $g : \mathbb{F} \to \mathbb{F}$ of degree $\deg(g) < d - h$
Computational integrity, succinctness and arithmetization

- **Fact 1**: If $H \subset \mathbb{F}$ mult. group, $|H| = h$, then $Z_H(\beta) = \beta^h - 1$ evaluated in time $O(\log h)$

- **Fact 2**: $P(X)$ vanishes on $H \iff \exists \tilde{P}(X), \deg(\tilde{P}) = \deg(P) - h$ and $Z_H \cdot \tilde{P}(X) = P(X)$

- **Fact 3**: Two distinct degree $d$ functions evaluated at $100 \cdot d$ points are 99%-far

**Suppose**: prover uses only degree-$d$ polynomials

**Challenge 1**: Given $f : \mathbb{F} \rightarrow \mathbb{F}, \deg(f) = d < |\mathbb{F}|/100$, devise protocol for checking succinctly and with small error if $f$ vanishes on $H$

**The (IOP) protocol:**

- Prover sends $g : \mathbb{F} \rightarrow \mathbb{F}$ of degree $\deg(g) < d - h$
- Verifier samples $\alpha \in \mathbb{F}$, accepts iff $f(\alpha) = Z_H(\alpha) \cdot g(\alpha)$
Computational integrity, succinctness and arithmetization

- **Fact 1:** If $H \subset \mathbb{F}$ mult. group, $|H| = h$, then $Z_H(\beta) = \beta^h - 1$ evaluated in time $O(\log h)$
- **Fact 2:** $P(X)$ vanishes on $H \iff \exists \tilde{P}(X), \deg(\tilde{P}) = \deg(P) - h$ and $Z_H \cdot \tilde{P}(X) = P(X)$
- **Fact 3:** Two distinct degree $d$ functions evaluated at $100 \cdot d$ points are $99\%$-far

**Suppose:** prover uses only degree-$d$ polynomials

**Challenge 1:** Given $f : \mathbb{F} \to \mathbb{F}$, $\deg(f) = d < |\mathbb{F}|/100$, devise protocol for checking **succinctly and with small error** if $f$ vanishes on $H$

**The (IOP) protocol:**

- **Prover** sends $g : \mathbb{F} \to \mathbb{F}$ of degree $\deg(g) < d - h$
- **Verifier** samples $\alpha \in \mathbb{F}$, accepts iff $f(\alpha) = Z_H(\alpha) \cdot g(\alpha)$
- **Complexity:** 2 queries, $O(\log h)$ time,
Computational integrity, succinctness and arithmetization

- **Fact 1:** If $H \subset \mathbb{F}$ mult. group, $|H| = h$, then $Z_H(\beta) = \beta^h - 1$ evaluated in time $O(\log h)$
- **Fact 2:** $P(X)$ vanishes on $H \iff \exists \tilde{P}(X), \deg(\tilde{P}) = \deg(P) - h$ and $Z_H \cdot \tilde{P}(X) = P(X)$
- **Fact 3:** Two distinct degree $d$ functions evaluated at $100 \cdot d$ points are $99\%$-far

**Suppose:** prover uses only degree-$d$ polynomials

**Challenge 1:** Given $f : \mathbb{F} \to \mathbb{F}, \deg(f) = d < |\mathbb{F}|/100$, devise protocol for checking succinctly and with small error if $f$ vanishes on $H$

**The (IOP) protocol:**

- Prover sends $g : \mathbb{F} \to \mathbb{F}$ of degree $\deg(g) < d - h$
- Verifier samples $\alpha \in \mathbb{F}$, accepts iff $f(\alpha) = Z_H(\alpha) \cdot g(\alpha)$
- Complexity: 2 queries, $O(\log h)$ time,
- Soundness error $\leq 1\%$: 
Computational integrity, succinctness and arithmetization

- **Fact 1:** If $H \subset \mathbb{F}$ mult. group, $|H| = h$, then $Z_H(\beta) = \beta^h - 1$ evaluated in time $O(\log h)$
- **Fact 2:** $P(X)$ vanishes on $H \iff \exists \tilde{P}(X)$, $\deg(\tilde{P}) = \deg(P) - h$ and $Z_H \cdot \tilde{P}(X) = P(X)$
- **Fact 3:** Two distinct degree $d$ functions evaluated at $100 \cdot d$ points are $99\%$-far

**Suppose:** prover uses only degree-$d$ polynomials

**Challenge 1:** Given $f : \mathbb{F} \rightarrow \mathbb{F}$, $\deg(f) = d < \frac{|\mathbb{F}|}{100}$, devise protocol for checking succinctly and with small error if $f$ vanishes on $H$

**The (IOP) protocol:**

- **Prover** sends $g : \mathbb{F} \rightarrow \mathbb{F}$ of degree $\deg(g) < d - h$
- **Verifier** samples $\alpha \in \mathbb{F}$, accepts iff $f(\alpha) = Z_H(\alpha) \cdot g(\alpha)$
- **Complexity:** 2 queries, $O(\log h)$ time,
- **Soundness error $\leq 1\%$:**
  - Suppose $f$ does not vanish on $H$
  - then $f(X) - Z_H(X) \cdot g(X)$ non-zero polynomial
Computational integrity, succinctness and arithmetization

- **Fact 1:** If \( H \subset \mathbb{F} \) mult. group, \( |H| = h \), then \( Z_H(\beta) = \beta^h - 1 \) evaluated in time \( O(\log h) \)
- **Fact 2:** \( P(X) \) vanishes on \( H \) \( \iff \exists \tilde{P}(X), \deg(\tilde{P}) = \deg(P) - h \) and \( Z_H \cdot \tilde{P}(X) = P(X) \)
- **Fact 3:** Two distinct degree \( d \) functions evaluated at \( 100 \cdot d \) points are 99%-far

Suppose: prover uses only degree-\( d \) polynomials

**Challenge 1:** Given \( f : \mathbb{F} \rightarrow \mathbb{F}, \deg(f) = d < |\mathbb{F}|/100 \), devise protocol for checking succinctly and with small error if \( f \) vanishes on \( H \)

The (IOP) protocol:

- Prover sends \( g : \mathbb{F} \rightarrow \mathbb{F} \) of degree \( \deg(g) < d - h \)
- Verifier samples \( \alpha \in \mathbb{F} \), accepts iff \( f(\alpha) = Z_H(\alpha) \cdot g(\alpha) \)
- Complexity: 2 queries, \( O(\log h) \) time,
- Soundness error \( \leq 1\%: \)
  - Suppose \( f \) does not vanish on \( H \)
  - then \( f(X) - Z_H(X) \cdot g(X) \) non-zero polynomial
  - it has at most \( d \) roots
Computational integrity, succinctness and arithmetization

- **Fact 1:** If $H \subset \mathbb{F}$ mult. group, $|H| = h$, then $Z_H(\beta) = \beta^h - 1$ evaluated in time $O(\log h)$

- **Fact 2:** $P(X)$ vanishes on $H \iff \exists \tilde{P}(X), \deg(\tilde{P}) = \deg(P) - h$ and $Z_H \cdot \tilde{P}(X) = P(X)$

- **Fact 3:** Two distinct degree $d$ functions evaluated at $100 \cdot d$ points are 99%-far

**Suppose:** prover uses only degree-$d$ polynomials

**Challenge 1:** Given $f : \mathbb{F} \to \mathbb{F}, \deg(f) = d < |\mathbb{F}|/100$, devise protocol for checking succinctly and with small error if $f$ vanishes on $H$

The (IOP) protocol:

- Prover sends $g : \mathbb{F} \to \mathbb{F}$ of degree $\deg(g) < d - h$
- Verifier samples $\alpha \in \mathbb{F}$, accepts iff $f(\alpha) = Z_H(\alpha) \cdot g(\alpha)$
- Complexity: 2 queries, $O(\log h)$ time,
- Soundness error $\leq 1\%$:
  - Suppose $f$ does not vanish on $H$
  - then $f(X) - Z_H(X) \cdot g(X)$ non-zero polynomial
  - it has at most $d$ roots
  - So probability of error $\leq d/|\mathbb{F}| \leq 1/100$
Computational integrity, succinctness and arithmetization

- **Fact 1**: If $H \subset \mathbb{F}$ mult. group, $|H| = h$, then $Z_H(\beta) = \beta^h - 1$ evaluated in time $O(\log h)$

- **Fact 2**: $P(X)$ vanishes on $H \iff \exists \tilde{P}(X)$, $\deg(\tilde{P}) = \deg(P) - h$ and $Z_H \cdot \tilde{P}(X) = P(X)$

- **Fact 3**: Two distinct degree $d$ functions evaluated at $100 \cdot d$ points are 99%-far

Suppose: prover uses only degree-$d$ polynomials
Computational integrity, succinctness and arithmetization

- **Fact 1:** If $H \subset \mathbb{F}$ mult. group, $|H| = h$, then $Z_H(\beta) = \beta^h - 1$ evaluated in time $O(\log h)$
- **Fact 2:** $P(X)$ vanishes on $H \iff \exists \tilde{P}(X), \deg(\tilde{P}) = \deg(P) - h$ and $Z_H \cdot \tilde{P}(X) = P(X)$
- **Fact 3:** Two distinct degree $d$ functions evaluated at $100 \cdot d$ points are $99\%$-far

**Suppose:** prover uses only degree-$d$ polynomials

**Challenge 2:** Given $f : \mathbb{F} \rightarrow \mathbb{F}, \deg(f) = d < |\mathbb{F}|/100$, devise protocol for checking succinctly and with small error if $f$ is Boolean (evaluates to $\{0, 1\}$) on $H$
Computational integrity, succinctness and arithmetization

- **Fact 1**: If \( H \subset \mathbb{F} \) mult. group, \(|H| = h\), then \( Z_H(\beta) = \beta^h - 1 \) evaluated in time \( O(\log h) \)

- **Fact 2**: \( P(X) \) vanishes on \( H \iff \exists \tilde{P}(X) \) deg(\( \tilde{P} \)) = deg(\( P \)) − \( h \) and \( Z_H \cdot \tilde{P}(X) = P(X) \)

- **Fact 3**: Two distinct degree \( d \) functions evaluated at 100 \( \cdot \) \( d \) points are 99%-far

**Suppose**: prover uses only degree-\( d \) polynomials

**Challenge 2**: Given \( f : \mathbb{F} \to \mathbb{F} \), \( \deg(f) = d < |\mathbb{F}|/100 \), devise protocol for checking succinctly and with small error if \( f \) is Boolean (evaluates to \{0, 1\}) on \( H \)

**The (IOP) protocol**:

- Prover sends \( g : \mathbb{F} \to \mathbb{F} \) of degree \( \deg(g) < 2d - h \)
Computational integrity, succinctness and arithmetization

- **Fact 1:** If $H \subset \mathbb{F}$ mult. group, $|H| = h$, then $Z_H(\beta) = \beta^h - 1$ evaluated in time $O(\log h)$

- **Fact 2:** $P(X)$ vanishes on $H \iff \exists \tilde{P}(X)$, $\deg(\tilde{P}) = \deg(P) - h$ and $Z_H \cdot \tilde{P}(X) = P(X)$

- **Fact 3:** Two distinct degree $d$ functions evaluated at $100 \cdot d$ points are 99%-far

**Suppose:** prover uses only degree-$d$ polynomials

**Challenge 2:** Given $f : \mathbb{F} \to \mathbb{F}$, $\deg(f) = d < |\mathbb{F}|/100$, devise protocol for checking succinctly and with small error if $f$ is Boolean (evaluates to $\{0, 1\}$) on $H$

**The (IOP) protocol:**

- Prover sends $g : \mathbb{F} \to \mathbb{F}$ of degree $\deg(g) < 2d - h$
- Verifier samples $\alpha \in \mathbb{F}$, accepts iff $f(\alpha) \cdot (f(\alpha) - 1) = Z_H(\alpha) \cdot g(\alpha)$
Computational integrity, succinctness and arithmetization

- **Fact 1:** If $H \subset \mathbb{F}$ is a multiplicative group, $|H| = h$, then $Z_H(\beta) = \beta^h - 1$ evaluated in time $O(\log h)$

- **Fact 2:** $P(X)$ vanishes on $H \iff \exists \tilde{P}(X)$, $\deg(\tilde{P}) = \deg(P) - h$ and $Z_H \cdot \tilde{P}(X) = P(X)$

- **Fact 3:** Two distinct degree $d$ functions evaluated at $100 \cdot d$ points are 99%-far

**Suppose:** prover uses only degree-$d$ polynomials

**Challenge 2:** Given $f : \mathbb{F} \rightarrow \mathbb{F}$, $\deg(f) = d < |\mathbb{F}|/100$, devise protocol for checking succinctly and with small error if $f$ is Boolean (evaluates to $\{0, 1\}$) on $H$

The (IOP) protocol:

- **Prover** sends $g : \mathbb{F} \rightarrow \mathbb{F}$ of degree $\deg(g) < 2d - h$
- **Verifier** samples $\alpha \in \mathbb{F}$, accepts iff $f(\alpha) \cdot (f(\alpha) - 1) = Z_H(\alpha) \cdot g(\alpha)$

- **Complexity:** 2 queries, $O(\log h)$ time, error prob $\leq 2\%$
Computational integrity, succinctness and arithmetization

- **Fact 1:** If $H \subset \mathbb{F}$ mult. group, $|H| = h$, then $Z_H(\beta) = \beta^h - 1$ evaluated in time $O(\log h)$

- **Fact 2:** $P(X)$ vanishes on $H \iff \exists \tilde{P}(X), \deg(\tilde{P}) = \deg(P) - h$ and $Z_H \cdot \tilde{P}(X) = P(X)$

- **Fact 3:** Two distinct degree $d$ functions evaluated at $100 \cdot d$ points are 99%-far

**Summary:** Succinct verification of Booleanity type-checking

What about verifying correctness of general computation?

- **Fact 4:** $\deg(f(x)) = \deg(f(ax + b))$ for all $a \neq 0, b$
Computational integrity, succinctness and arithmetization

- **Fact 1**: If $H \subset \mathbb{F}$ mult. group, $|H| = h$, then $Z_H(\beta) = \beta^h - 1$ evaluated in time $O(\log h)$

- **Fact 2**: $P(X)$ vanishes on $H \iff \exists \tilde{P}(X), \deg(\tilde{P}) = \deg(P) - h$ and $Z_H \cdot \tilde{P}(X) = P(X)$

- **Fact 3**: Two distinct degree $d$ functions evaluated at $100 \cdot d$ points are 99%-far

- **Fact 4**: for all $a \neq 0, b$ we have $\deg(f(x)) = \deg(f(ax + b))$
Computational integrity, succinctness and arithmetization

- **Fact 1:** If $H \subset F$ mult. group, $|H| = h$, then $Z_H(\beta) = \beta^h - 1$ evaluated in time $O(\log h)$
- **Fact 2:** $P(X)$ vanishes on $H \iff \exists \tilde{P}(X), \deg(\tilde{P}) = \deg(P) - h$ and $Z_H \cdot \tilde{P}(X) = P(X)$
- **Fact 3:** Two distinct degree $d$ functions evaluated at $100 \cdot d$ points are $99\%$-far
- **Fact 4:** for all $a \neq 0, b$ we have $\deg(f(x)) = \deg(f(ax + b))$

**Challenge 3:** Given $f : \mathbb{F}_p \to \mathbb{F}_p, \deg(f) = d < |\mathbb{F}|/100$, devise protocol for checking succinctly and with small error if $f$ evaluates a Fibonacci sequence on $H$ and last element equals $b \mod p$
Computational integrity, succinctness and arithmetization

- **Fact 1**: If $H \subset F$ mult. group, $|H| = h$, then $Z_H(\beta) = \beta^h - 1$ evaluated in time $O(\log h)$
- **Fact 2**: $P(X)$ vanishes on $H \iff \exists \tilde{P}(X), \deg(\tilde{P}) = \deg(P) - h$ and $Z_H \cdot \tilde{P}(X) = P(X)$
- **Fact 3**: Two distinct degree $d$ functions evaluated at $100 \cdot d$ points are $99\%$-far
- **Fact 4**: for all $a \neq 0, b$ we have $\deg(f(x)) = \deg(f(ax + b))$

**Challenge 3**: Given $f : \mathbb{F}_p \to \mathbb{F}_p, \deg(f) = d < |\mathbb{F}|/100$, devise protocol for checking succinctly and with small error if $f$ evaluates a Fibonacci sequence on $H$ and last element equals $b$ mod $p$

**The (IOP) protocol**:
- Prover sends $g, g' : \mathbb{F}_p \to \mathbb{F}_p$ of degree $\deg(g) < d - h, \deg(g') < d - 3$
Computational integrity, succinctness and arithmetization

- **Fact 1:** If $H \subset \mathbb{F}$ mult. group, $|H| = h$, then $Z_H(\beta) = \beta^h - 1$ evaluated in time $O(\log h)$
- **Fact 2:** $P(X)$ vanishes on $H \iff \exists \tilde{P}(X), \deg(\tilde{P}) = \deg(P) - h$ and $Z_H \cdot \tilde{P}(X) = P(X)$
- **Fact 3:** Two distinct degree $d$ functions evaluated at $100 \cdot d$ points are $99\%$-far
- **Fact 4:** for all $a \neq 0, b$ we have $\deg(f(x)) = \deg(f(ax + b))$

**Challenge 3:** Given $f : \mathbb{F}_p \rightarrow \mathbb{F}_p, \deg(f) = d < |\mathbb{F}|/100$, devise protocol for checking succinctly and with small error if $f$ evaluates a Fibonacci sequence on $H$ and last element equals $b \mod p$

The (IOP) protocol:

- **Prover sends** $g, g' : \mathbb{F}_p \rightarrow \mathbb{F}_p$ of degree
  $\deg(g) < d - h, \deg(g') < d - 3$
- **Let** $B(x)$ be degree 2 polynomial that satisfies $P(1) = P(\omega) = 1, P(\omega^{-1}) = b$
Computational integrity, succinctness and arithmetization

- **Fact 1:** If $H \subset \mathbb{F}$ mult. group, $|H| = h$, then $Z_H(\beta) = \beta^h - 1$ evaluated in time $O(\log h)$

- **Fact 2:** $P(X)$ vanishes on $H \iff \exists \tilde{P}(X)$, $\deg(\tilde{P}) = \deg(P) - h$ and $Z_H \cdot \tilde{P}(X) = P(X)$

- **Fact 3:** Two distinct degree $d$ functions evaluated at $100 \cdot d$ points are $99\%$-far

- **Fact 4:** for all $a \neq 0, b$ we have $\deg(f(x)) = \deg(f(ax + b))$

**Challenge 3:** Given $f : \mathbb{F}_p \to \mathbb{F}_p$, $\deg(f) = d < |\mathbb{F}|/100$, devise protocol for checking succinctly and with small error if $f$ evaluates a Fibonacci sequence on $H$ and last element equals $b$ mod $p$

The (IOP) protocol:

- Prover sends $g, g' : \mathbb{F}_p \to \mathbb{F}_p$ of degree $\deg(g) < d - h, \deg(g') < d - 3$

- Let $B(x)$ be degree 2 polynomial that satisfies $P(1) = P(\omega) = 1, P(\omega^{-1}) = b$

- Let $D(X)$ be the degree-3 polynomial that vanishes on $1, \omega, \omega^{-1}$
Fact 1: If $H \subseteq F$ mult. group, $|H| = h$, then $Z_H(\beta) = \beta^h - 1$ evaluated in time $O(\log h)$

Fact 2: $P(X)$ vanishes on $H \iff \exists \tilde{P}(X), \deg(\tilde{P}) = \deg(P) - h$ and $Z_H \cdot \tilde{P}(X) = P(X)$

Fact 3: Two distinct degree $d$ functions evaluated at $100 \cdot d$ points are $99\%$-far

Fact 4: for all $a \neq 0, b$ we have $\deg(f(x)) = \deg(f(ax + b))$

Challenge 3: Given $f : \mathbb{F}_p \to \mathbb{F}_p, \deg(f) = d < |\mathbb{F}|/100$, devise protocol for checking succinctly and with small error if $f$ evaluates a Fibonacci sequence on $H$ and last element equals $b \mod p$

The (IOP) protocol:

- Prover sends $g, g' : \mathbb{F}_p \to \mathbb{F}_p$ of degree $\deg(g) < d - h, \deg(g') < d - 3$
- Let $B(x)$ be degree 2 polynomial that satisfies $P(1) = P(\omega) = 1, P(\omega^{-1}) = b$
- Let $D(X)$ be the degree-3 polynomial that vanishes on $1, \omega, \omega^{-1}$
- Verifier samples $\alpha \in \mathbb{F} \setminus \{1, \omega\}$, accepts iff
  - $f(\alpha) - f(\alpha/\omega) - f(\alpha/\omega^2) = Z_H(\alpha) \cdot g(\alpha)/((\alpha - 1)(\alpha - \omega))$
  - $f(\alpha) - B(\alpha) = g'(\alpha) \cdot D(\alpha)$
Computational integrity, succinctness and arithmetization

- **Fact 1:** If $H \subset \mathbb{F}$ mult. group, $|H| = h$, then $Z_H(\beta) = \beta^h - 1$ evaluated in time $O(\log h)$
- **Fact 2:** $P(X)$ vanishes on $H \iff \exists \tilde{P}(X), \deg(\tilde{P}) = \deg(P) - h$ and $Z_H \cdot \tilde{P}(X) = P(X)$
- **Fact 3:** Two distinct degree $d$ functions evaluated at $100 \cdot d$ points are $99\%$-far
- **Fact 4:** for all $a \neq 0, b$ we have $\deg(f(x)) = \deg(f(ax + b))$

**Challenge 3:** Given $f: \mathbb{F}_p \rightarrow \mathbb{F}_p, \deg(f) = d < |\mathbb{F}|/100$, devise protocol for checking succinctly and with small error if $f$ evaluates a Fibonacci sequence on $H$ and last element equals $b \mod p$

**The (IOP) protocol:**
- **Prover sends** $g, g' : \mathbb{F}_p \rightarrow \mathbb{F}_p$ of degree $\deg(g) < d - h, \deg(g') < d - 3$
- **Let $B(x)$** be degree 2 polynomial that satisfies $P(1) = P(\omega) = 1, P(\omega^{-1}) = b$
- **Let $D(X)$** be the degree-3 polynomial that vanishes on $1, \omega, \omega^{-1}$
- **Verifier samples** $\alpha \in \mathbb{F} \setminus \{1, \omega\}$, accepts iff
  - $f(\alpha) - f(\alpha/\omega) - f(\alpha/\omega^2) = Z_H(\alpha) \cdot g(\alpha)/((\alpha - 1)(\alpha - \omega))$
  - $f(\alpha) - B(\alpha) = g'(\alpha) \cdot D(\alpha)$
- **Complexity:** 5 queries, $O(\log h)$ time, error prob $\leq 1\%$
Computational integrity, succinctness and arithmetization

- **Fact 1:** If $H \subset \mathbb{F}$ mult. group, $|H| = h$, then $Z_H(\beta) = \beta^h - 1$ evaluated in time $O(\log h)$
- **Fact 2:** $P(X)$ vanishes on $H \iff \exists \tilde{P}(X), \deg(\tilde{P}) = \deg(P) - h$ and $Z_H \cdot \tilde{P}(X) = P(X)$
- **Fact 3:** Two distinct degree $d$ functions evaluated at $100 \cdot d$ points are $99\%$-far
- **Fact 4:** for all $a \neq 0, b$ we have $\deg(f(x)) = \deg(f(ax + b))$

General theme:

- Write transition function as polynomial constraints
- Write boundary constraints as polynomial constraints
Fact 1: If $H \subset \mathbb{F}$ mult. group, $|H| = h$, then $Z_H(\beta) = \beta^h - 1$ evaluated in time $O(\log h)$

Fact 2: $P(X)$ vanishes on $H \iff \exists \tilde{P}(X), \deg(\tilde{P}) = \deg(P) - h$ and $Z_H \cdot \tilde{P}(X) = P(X)$

Fact 3: Two distinct degree $d$ functions evaluated at $100 \cdot d$ points are 99%-far

Fact 4: for all $a \neq 0, b$ we have $\deg(f(x)) = \deg(f(ax + b))$

General theme:

- Write transition function as polynomial constraints
- Write boundary constraints as polynomial constraints
- Check that applied to $f$, all constraints vanish on $H$
Computational integrity, succinctness and arithmetization

- **Fact 1:** If $H \subset \mathbb{F}$ mult. group, $|H| = h$, then $Z_H(\beta) = \beta^h - 1$ evaluated in time $O(\log h)$
- **Fact 2:** $P(X)$ vanishes on $H \iff \exists \tilde{P}(X), \deg(\tilde{P}) = \deg(P) - h$ and $Z_H \cdot \tilde{P}(X) = P(X)$
- **Fact 3:** Two distinct degree $d$ functions evaluated at $100 \cdot d$ points are 99%-far
- **Fact 4:** for all $a \neq 0, b$ we have $\deg(f(x)) = \deg(f(ax + b))$

General theme:
- Write transition function as polynomial constraints
- Write boundary constraints as polynomial constraints
- Check that applied to $f$, all constraints vanish on $H$
- **Question:** What about ZK? $f\mid_H$ reveals the computation!
Computational integrity, succinctness and arithmetization

- **Fact 1**: If $H \subset \mathbb{F}$ mult. group, $|H| = h$, then $Z_H(\beta) = \beta^h - 1$ evaluated in time $O(\log h)$

- **Fact 2**: $P(X)$ vanishes on $H \iff \exists \tilde{P}(X), \deg(\tilde{P}) = \deg(P) - h$ and $Z_H \cdot \tilde{P}(X) = P(X)$

- **Fact 3**: Two distinct degree $d$ functions evaluated at $100 \cdot d$ points are 99%-far

- **Fact 4**: for all $a \neq 0, b$ we have $\deg(f(x)) = \deg(f(ax + b))$

General theme:

- Write transition function as polynomial constraints
- Write boundary constraints as polynomial constraints
- Check that applied to $f$, all constraints vanish on $H$

**Question**: What about ZK? $f|_H$ reveals the computation!

- never sample from $H$,
- if test uses $q$ queries, slacken degree, $\deg(f) = d + q$,
- prover samples $f$ to agree with correct execution trace on $H$ and be random otherwise.
- this gives ZK!
We saw

- arithmetization solves succinct checking of computational integrity

Solution 1 requires trusted setup, leads to zkSNARKs (and many other constructions)

Solution 2 is transparent, leads to zkSTARKs (and many other constructions)

want to learn more? workshop@starkware.co
want to realize in practice? jobs@starkware.co
Our lecture on Arithmetization concluded with the following summary:

We saw:
- Arithmetization solves succinct checking of computational integrity.
- Adding randomness and increasing degree gives ZK.

To address the challenge of preventing Bob from presenting functions that are not of the required degree, two solutions were discussed:

1. [IKO07]: Use additively homomorphic encryption (and more) to limit Bob to using only low-degree polynomials.
2. [PCPs 1990s]: Have Bob Commit-then-reveal entries of $f, g$ and add a special "proximity-to-low-degree-testing" protocol (next lecture).

Solution 1 requires trusted setup, leading to zkSNARKs (and many other constructions).
Solution 2 is transparent, leading to zkSTARKs (and many other constructions).

If you want to learn more, contact workshop@starkware.co. If you want to realize these concepts in practice, contact jobs@starkware.co.
We saw

- arithmetization solves succinct checking of computational integrity
- adding randomness and increasing degree gives ZK

We didn’t see

- How can Bob be prevented from presenting $f, g$ that are not of needed degree?

Solution 1 requires trusted setup, leads to zkSNARKs (and many other constructions)

Solution 2 is transparent, leads to zkSTARKs (and many other constructions)
We saw

- arithmetization solves succinct checking of computational integrity
  - adding randomness and increasing degree gives ZK

We didn’t see

- How can Bob be prevented from presenting $f, g$ that are not of needed degree?
- Two kinds of solutions
  1. [IKO07]: Use additively homomorphic encryption (and more) to limit Bob to using only low-degree polynomials
  2. [PCPs 1990s]: Have Bob Commit-then-reveal entries of $f, g$ and add special “proximity-to-low-degree-testing” protocol (next lecture)
Arithmetization — Summary

We saw

- arithmetization solves succinct checking of computational integrity
- adding randomness and increasing degree gives ZK

We didn’t see

- How can Bob be prevented from presenting $f, g$ that are not of needed degree?
- Two kinds of solutions
  1. [IKO07]: Use additively homomorphic encryption (and more) to limit Bob to using only low-degree polynomials
  2. [PCPs 1990s]: Have Bob Commit-then-reveal entries of $f, g$ and add special “proximity-to-low-degree-testing” protocol (next lecture)

- Solution 1 requires trusted setup, leads to zkSNARKs (and many other constructions)
- Solution 2 is transparent, leads to zkSTARKs (and many other constructions)
Arithmetization — Summary

We saw

- arithmetization solves succinct checking of computational integrity
- adding randomness and increasing degree gives ZK

We didn’t see

- How can Bob be prevented from presenting $f, g$ that are not of needed degree?

Two kinds of solutions

1. [IKO07]: Use additively homomorphic encryption (and more) to limit Bob to using only low-degree polynomials
2. [PCPs 1990s]: Have Bob Commit-then-reveal entries of $f, g$ and add special “proximity-to-low-degree-testing” protocol (next lecture)

Solution 1 requires trusted setup, leads to zkSNARKs (and many other constructions)

Solution 2 is transparent, leads to zkSTARKs (and many other constructions)

want to learn more? workshop@starkware.co
want to realize in practice? jobs@starkware.co