Sumcheck Arguments and their Applications

Jonathan Bootle, Alessandro Chiesa, Katerina Sotiraki

Nikolaos Diamantis

Algorithms, Logic and Discrete Mathematics

08/06/2023

Outline

Succinct Arguments

Succinct arguments are proofs that enable checking computations exponentially faster than they can be run:

- **Completeness**
- **Soundness**
- **Succinctness:**
  - The messages are much smaller than the witness.
  - Otherwise, the prover would sent directly the witness (non ZK).

The Sumcheck Protocol

Goal: Given a polynomial $p(X_1, ..., X_l)$ over a field $\mathbb{F}$ and a value $u \in \mathbb{F}$, prove that $\sum_{\underline{\omega} \in \{0,1\}^l} p(\omega_1, ..., \omega_l) = u$.

The Sumcheck Protocol

Goal: Given a polynomial $p(X_1, ..., X_l)$ over a field $\mathbb{F}$ and a value $u \in \mathbb{F}$, prove that $\sum_{\underline{\omega} \in \{0,1\}^l} p(\omega_1, ..., \omega_l) = u$.

In the i-th of total l rounds:

- **Verifier:** Sends random challenge $r_i \leftarrow \mathbb{F}$ to prover.
- **Prover:** Computes
  $q_i(X_i) = \sum_{\underline{\omega} \in \{0,1\}^{l-i}} p(r_1, ..., r_{i-1}, X_i, \omega_{i+1}, ..., \omega_l) \in \mathbb{F}[X_i]$,
  and sends it to verifier.

The Sumcheck Protocol

Verifier: Accepts if:

- $i = 1 : \sum_{\omega_1 \in \{0,1\}} q_1(\omega_1) = u$
- $1 < i < l + 1 : \sum_{\omega_i \in \{0,1\}} q_i(\omega_i) = q_{i-1}(r_{i-1})$
- $i = l + 1 : p(r_1, ..., r_l) = q_l(r_l)$

The Sumcheck Protocol

Verifier: Accepts if:

- $i = 1 : \sum_{\omega_1 \in \{0,1\}} q_1(\omega_1) = u$
- $1 < i < l + 1 : \sum_{\omega_i \in \{0,1\}} q_i(\omega_i) = q_{i-1}(r_{i-1})$
- $i = l + 1 : p(r_1, ..., r_l) = q_l(r_l)$

- **Communication complexity:** $O(l \cdot \deg(p))$.
- **Soundness:** If $\sum_{\underline{\omega} \in \{0,1\}^l} p(\omega_1, ..., \omega_l) \neq u$, then verifier accepts with probability at most $\frac{l \cdot \deg(p)}{|\mathbb{F}|}$.

Outline

Folding Techniques

We can construct succinct arguments based on folding techniques for Pedersen commitments, in the discrete logarithm setting.

Goal: Prove knowledge of a long message, opening a given Pedersen commitment:

Folding Techniques

We can construct succinct arguments based on folding techniques for Pedersen commitments, in the discrete logarithm setting.

Goal: Prove knowledge of a long message, opening a given Pedersen commitment:

- **Prover** and **verifier** engage in a reduction that halves the message length by folding the message "around" a verifier challenge.

Folding Techniques

We can construct succinct arguments based on folding techniques for Pedersen commitments, in the discrete logarithm setting.

Goal: Prove knowledge of a long message, opening a given Pedersen commitment:

- **Prover** and **verifier** engage in a reduction that halves the message length by folding the message "around" a verifier challenge.
- Repeated until the message length is small enough to send the message directly.

What do they have in common?

Sumcheck protocol:

- Linear-time prover,
  - relative to l coefficients
- Small space,
  - the prover is a streaming algorithm.
- Strong soundness properties,
  - $\frac{l \cdot \deg(p)}{|\mathbb{F}|}$
  - it can be non-interactive without random oracles, but with some "special" hash functions.

What do they have in common?

Sumcheck protocol:

- Linear-time prover,
  - relative to l coefficients
- Small space,
  - the prover is a streaming algorithm.
- Strong soundness properties,
  - $\frac{l \cdot \deg(p)}{|\mathbb{F}|}$
  - it can be non-interactive without random oracles, but with some "special" hash functions.

Folding techniques:

- Also linear-time prover.
- The prover is a streaming algorithm,
  - they can be implemented in small space.

Outline

Sumcheck-Friendly Commitments

Sumcheck-Friendly Commitments

A commitment scheme is sumcheck-friendly if:

$$Com(k,m) = \sum_{\underline{\omega} \in H^l} f(p_m(\omega_1, ..., \omega_l), p_k(\omega_1, ..., \omega_l)).$$

$Com \in \mathbb{C}$ $\quad p_m \in \mathbb{M}$ $\quad p_k \in \mathbb{K}$

$$f : \mathbb{M} \times \mathbb{K} \to \mathbb{C}$$

Sumcheck-Friendly Commitments

Sumcheck-Friendly Commitments

A commitment scheme is sumcheck-friendly if:

$$Com(k,m) = \sum_{\underline{\omega} \in H^l} f(p_m(\omega_1, ..., \omega_l), p_k(\omega_1, ..., \omega_l)).$$

$Com \in \mathbb{C}$ $\quad p_m \in \mathbb{M}$ $\quad p_k \in \mathbb{K}$

$$f : \mathbb{M} \times \mathbb{K} \to \mathbb{C}$$

On Pedersen commitments:

- Commitment key: $\underline{G} \in \mathbb{G}^n$
- Commitment: $C \in \mathbb{G}$
- Opening: $\underline{\alpha} \in \mathbb{F}^n$
  - Such that: $C = \alpha_1 \cdot G_1 + \cdots + \alpha_n \cdot G_n = \langle \underline{\alpha}, \underline{G} \rangle$

Invertibility

K-Invertibility

Given polynomial $q_i(\underline{X})$ and openings $p^{(1)}(\underline{X}), ..., p^{(K)}(\underline{X})$ such that:

$$\forall j \in [K] : q_i(r_i^{(j)}) = \sum_{\underline{\omega} \in H^{l-1}} f(p(\underline{\omega}), p_k(r_1, ..., r_{i-1}, r_i^{(j)}, \underline{\omega})).$$

Invertibility

K-Invertibility

Given polynomial $q_i(\underline{X})$ and openings $p^{(1)}(\underline{X}), ..., p^{(K)}(\underline{X})$ such that:

$$\forall j \in [K] : q_i(r_i^{(j)}) = \sum_{\underline{\omega} \in H^{l-1}} f(p(\underline{\omega}), p_k(r_1, ..., r_{i-1}, r_i^{(j)}, \underline{\omega})).$$

We can find a polynomial p such that:

$$\sum_{\omega_n \in H} q_i(\omega_n) = \sum_{\underline{\omega} \in H^{l-i+1}} f(p(\underline{\omega}), p_k(r_1, ..., r_{i-1}, \underline{\omega})).$$

Sumcheck Arguments

Let a commitment scheme is sumcheck-friendly and invertible. Given a commitment key k and a commitment C, the sumcheck protocol applied to:

$$p(X_1, ..., X_l) = f(p_m(X_1, ..., X_l), p_k(X_1, ..., X_l)) \in \mathbb{C}.$$

Then, the prover has a succinct argument of knowledge for m, such that $C = Com(k, m)$,
with completeness, soundness and
communication complexity $O(l \cdot \deg(p))$.

Sumcheck Arguments for Sumcheck-Friendly Commitments

Goal: Given a key k and a commitment C, prove that $\exists m : C = \sum_{\underline{\omega} \in H^l} f(p_m(\underline{\omega}), p_k(\underline{\omega}))$.

Sumcheck Arguments for Sumcheck-Friendly Commitments

Goal: Given a key k and a commitment C, prove that $\exists m : C = \sum_{\underline{\omega} \in H^l} f(p_m(\underline{\omega}), p_k(\underline{\omega}))$.

- **Prover** and **Verifier:** Run the sumcheck protocol for: $\sum_{\underline{\omega} \in H^l} f(p_m(\underline{\omega}), p_k(\underline{\omega})) = C$.
- **Prover:** Sends $p_m(\underline{r})$
  - $\underline{r} \in \mathbb{F}^l$: the random challenges sent by verifier.
- **Verifier:** Accepts if $f(p_m(\underline{r}), p_k(\underline{r})) = q_l(r_l)$.

Outline

Sumcheck Arguments over Rings

R**(ing)-module:** Extends the notion of vector space over a field.

Bilinear Modules

A bilinear module is a triple of R-modules (M_L, M_R, M_T) over the same ring with a bilinear map $e : M_L \times M_R \to M_T$.

Sumcheck Arguments over Rings

R**(ing)-module:** Extends the notion of vector space over a field.

Bilinear Modules

A bilinear module is a triple of R-modules (M_L, M_R, M_T) over the same ring with a bilinear map $e : M_L \times M_R \to M_T$.

On Pedersen commitments:

- **Message:** small $\underline{\alpha} \in M_L$
- **Key:** $\underline{G} \in M_R$
- **Commitment:** $C \in M_T$
- **Assumption:** Bilinear relation assumption
  - analog of discrete logarithm

Sumcheck Arguments over Rings

R**(ing)-module:** Extends the notion of vector space over a field.

Bilinear Modules

A bilinear module is a triple of R-modules (M_L, M_R, M_T) over the same ring with a bilinear map $e : M_L \times M_R \to M_T$.

On Pedersen commitments:

- **Message:** small $\underline{\alpha} \in M_L$
- **Key:** $\underline{G} \in M_R$
- **Commitment:** $C \in M_T$
- **Assumption:** Bilinear relation assumption
  - analog of discrete logarithm
- $C = \alpha_1 \cdot G_1 + \cdots + \alpha_n \cdot G_n = \langle \underline{\alpha}, \underline{G} \rangle$
  - Hard to find small $\underline{\alpha}$ such that $\langle \underline{\alpha}, \underline{G} \rangle = 0$ (ring-SIS).

Sumcheck Arguments for Sumcheck-Friendly Commitments over Rings

Goal: Given a key k and a commitment C, prove that $\exists m, ||m|| \leq B : C = \sum_{\underline{\omega} \in H^l} f(p_m(\underline{\omega}), p_k(\underline{\omega}))$.

- **Prover** and **verifier** run the sumcheck protocol for: $\sum_{\underline{\omega} \in H^l} f(p_m(\underline{\omega}), p_k(\underline{\omega})) = C$.

Sumcheck Arguments for Sumcheck-Friendly Commitments over Rings

Goal: Given a key k and a commitment C, prove that $\exists m, ||m|| \leq B : C = \sum_{\underline{\omega} \in H^l} f(p_m(\underline{\omega}), p_k(\underline{\omega}))$.

- **Prover** and **verifier** run the sumcheck protocol for: $\sum_{\underline{\omega} \in H^l} f(p_m(\underline{\omega}), p_k(\underline{\omega})) = C$.
- **Prover:** Sends $p_m(\underline{r})$
  - $\underline{r} \in \mathcal{C}^l$: the random challenges sent by verifier.
    - $\mathcal{C} \subseteq R$
- **Verifier:** Accepts if:
  - $f(p_m(\underline{r}), p_k(\underline{r})) = q_l(r_l)$
  - $||p_m(\underline{r})|| \leq B$

Succinct Arguments for R1CS

Rank-1 Constraint Satisfiability

Given a ring R and matrices $A, B, C \in R^{n \times n}$, does there exist $z \in R^n$ satisfying $Az \circ Bz = Cz$?

Succinct Arguments for R1CS

Rank-1 Constraint Satisfiability

Given a ring R and matrices $A, B, C \in R^{n \times n}$, does there exist $z \in R^n$ satisfying $Az \circ Bz = Cz$?

Theorem

Let (M_L, M_R, M_T, e) be a "secure" bilinear module where M_L is a ring. Let $I \subseteq M_L$ be a suitable ideal. There is a zero-knowledge succinct argument of knowledge for R1CS with:

- **R1CS Ring:** M_L/I
- **Prover and Verifier time:** $O(n)$,
  - relative to the number of operations.
- **Proof size:** $O(\log n)$,
  - relative to the size of M_T.

Lattice-Based Succinct Arguments for R1CS

Corollary

Let $R_p := \mathbb{Z}_p[X]/\langle X^d + 1\rangle$, $R_q := \mathbb{Z}_q[X]/\langle X^d + 1\rangle$ for d a power of 2, and p, q primes with $p \ll q$. If the SIS problem is hard over R_q, then there is a zero-knowledge succinct argument of knowledge for R1CS with:

- **R1CS Ring:** R_p
- **Prover and Verifier time:** $O(n)$,
  - relative to the number of operations.
- **Proof size:** $O(\log n)$
  - relative to the size of R_q.

Thank you!