CS171: Cryptography

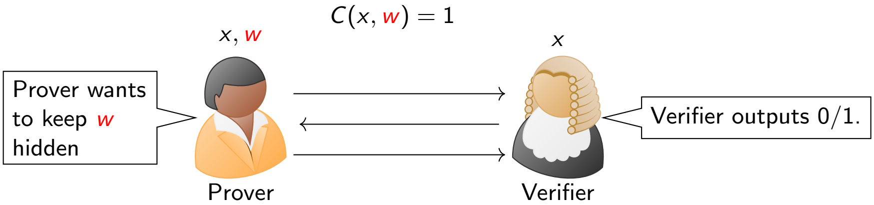
Lecture 21

Sanjam Garg

Plan for today

- Saw zero-knowledge protocol for the graph three coloring problem.
- Today: zero-knowledge protocol for graph hamiltonicity.
- Extending to arbitrary computation NP-complete.
- Succinct Arguments.

Zero-Knowledge Proof System



- Syntax: Two algorithms, $P(1^n, x, w)$ and $V(1^n, x)$.
- Completeness: Honest prover convinces an honest verifier with *overwhelming* probability.

$$\Pr[V \text{ outputs } 1 \text{ in the interaction } P(1^n, x, w) \leftrightarrow V(1^n, x)] = 1 - \mathrm{neg}(n)$$

- Soundness: A PPT cheating prover P^* cannot make a Verifier accept a false statement. For all PPT P^*, x such that $\forall w, C(x, w) = 0$then we have that

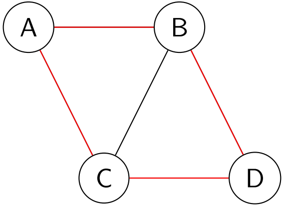
$$\Pr[V \text{ outputs } 1 \text{ in the interaction } P^*(1^n, x) \leftrightarrow V(1^n, x)] = \mathrm{neg}(n)$$

- Zero-Knowledge: The proof doesn't leak any information about the witness w. $\exists$ a PPT simulator $\mathcal{S}$ that for all PPT V^*, x, w such that $C(x, w) = 1$, we have that $\forall$ PPT D:

$$\left| \Pr[D(V^*\text{'s view in } P(1^n, x, w) \leftrightarrow V^*(1^n, x)) = 1] - \Pr[D(\mathcal{S}^{V^*}(1^n, x)) = 1] \right| \leq \mathrm{neg}(n)$$

Graph Hamiltonian Cycle Problem

- Graph $G = (V, E)$ with $V = \{1, \ldots n\}$.
- Represent as a $n \times n$ matrix M such that $M_{i,j} = 1$ if $(i, j) \in E$ and $M_{i,j} = 0$ otherwise.
- Task: Does these exist a cycle $C \subseteq E$ in G that visits each vertex exactly once?



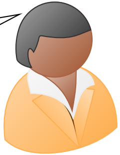
- Figuring out whether a graph has a Hamiltonian Cycle is believed to be computationally hard.

Zero-Knowledge Proof System for Graph Hamiltonicity Problem

$\exists C \subseteq E$ — a Hamiltonian Cycle in G.

Samples random permutation $\alpha : \{1, \ldots n\} \to \{1, \ldots n\}$ and randomness $r_{i,j}$ for $i, j \in \{1, \ldots n\}$. Set $\delta_{i,j} = 1$ if $(\alpha(i), \alpha(j)) \in E$ and 0 otherwise.

$G = (V, E), C$



Prover

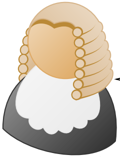
$\forall i, j, com_{i,j} = \mathrm{Com}(\delta_{i,j}; r_{i,j})$.

Challenge b

If $b = 0$ then send $\alpha, \forall i, j \ r_{i,j}$

If $b = 1$ then send $r_{i,j} \forall i, j$ such that $(\alpha(i), \alpha(j)) \in C$

$G = (V, E)$



Verifier

Sample uniform $b \leftarrow \{0, 1\}$.

If $b = 0$ then verifier checks that commitments are well-formed. If $b = 1$ then it checks that the commited/permuted graph has a Hamiltonian cycle.

Completeness: Since G has a Hamiltonian Cycle C, Verifier should accept in both cases.

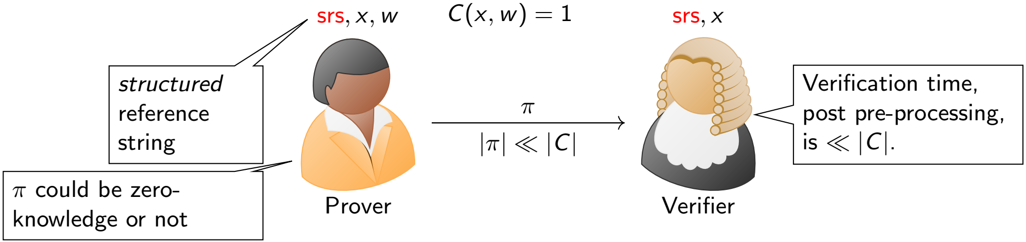
Soundness: Let $com_{i,j} = \mathrm{Com}(\delta_{i,j}; r_{i,j})$. Since the graph is not Hamiltonian either committed graph is not a permutation of G or the committed graph doesn't have a cycle. Verifier rejects with probability at least $\frac{1}{2}$. Amplification by repetition.

Zero-Knoweldge: Cropping Argument. Like Graph 3 Coloring.

Extending to any computation

- Give C, x we can construct a graph $G = (V, E)$.
- Such that: $\exists$ a hamiltonian cycle in G if and only if $\exists w$ such that $C(x, w) = 1$.
- Very useful!

Succinct Non-Interactive Argument System (SNARG)



- Completeness: An honest prover should be able to convince an honest verifier with *overwhelming* probability.
- Soundness: A PPT cheating prover cannot generate an accepting proof for a false statement.
- Zero-Knowledge: The proof doesn't leak any information about the witness w.
  - Not all applications need zero knowledge, e.g. zk-rollups.

Polynomial equality check

- Alice has a string $A = (a_0, \dots a_{n-1})$ and Bob has $B = (b_0, \dots b_{n-1})$ where each $a_i, b_i \in \{0, 1\}$.
- They want to check if $A \stackrel{?}{=} B$ with minimal communication.
- Let q be large prime.
- Alice computes polynomial $a(x) = \sum_i a_i \cdot x^i \mod q$ at a random point $r \in \{0, \dots q-1\}$ and sends $y = a(r)$ to Bob.
- bob computes polynomial $b(x) = \sum_i b_i \cdot x^i \mod q$ at point r and checks that $y = b(r)$. If yes, then Bob assertains that $A = B$ and no otherwise.
- If $a(x) \neq b(x)$ then

$$\Pr_r[A(r) = B(r)] \leq \frac{n-1}{q}$$

Verifying Matrix Multiplication

- Given two input matrices $A, B \in \mathbb{F}^{n \times n}$ we want to compute $A \cdot B$.
- Let's say $\mathbb{F} = \{0, \ldots p-1\}$ and addition, multiplication and division are modulo p.
- Fastest know algorithm takes time $n^{2.37}$.
- Can a prover P who knows the answer C convince a verifier V that the answer is correct in less time?
- Yes, here is the protocol.
- Both P and V get A, B, C and P wants to convince V that $C = A \cdot B$.
- Verifier picks random $r \in \mathbb{F}$.
- Let $x = (r, r^2, \ldots r^n)$.
- V checks if $C \cdot x \stackrel{?}{=} A \cdot B \cdot x$.
- Takes time $O(n^2)$.
- If $A \cdot B = C$ then V accepts with probability 1.
- If $A \cdot B \neq C$ then V accepts with probability $\leq n/|\mathbb{F}|$.

Check the roots of a polynomial

- P wants to prove that a given poylonomial $f(x)$ evlautes to 0 on inputs $H = \{0, 1, \ldots n-1\}$.
- Note that $\prod_{i \in H}(x - i) \mid f(x)$.
- Or, $f(x) = g(x) \cdot Z_H(x)$, where $Z_H(x) = \prod_{i \in H}(x - i)$.
- P commits to $f(x)$ and $g(x)$.
- V samples a random challenge r and sends to P.
- P opens $f(r)$ and $g(r)$.
- V checks that $f(r) = g(r) \cdot Z_H(r)$.
- What is V's running time? Grows with $|H|$. Can we make it smaller?

Choice of H

- Using $H = \{0, 1 ... n-1\}$ is inefficient.
- Instead we use $H = \{\omega, ... \omega^n\}$ the n^{th} (where $n = 2^k$) roots of unity $\omega^n = 1$ and $\omega^{n/2} \neq 1$. The exponent space needs to be such that 2^k divides $p-1$, which is the case for BLS12-381 for $k = 32$.
- **How do we find these roots of unity?**
- By Fermat's Littel Theorem for all $\alpha \in \mathbb{Z}_p$ we have $\alpha^{p-1} = 1$.
- For a random α, set $\omega = \alpha^{\frac{p-1}{n}}$ is one of the n^{th} roots of unity in $\mathbb{F}$. Check if $\omega^n = 1$ and $\omega^{n/2} \neq 1$. If not true, then repeat. Have to do it only once.

What do we gain?

- $Z_H(x) = (x-\omega)(x-\omega^2)...(x-1) = (x^n - 1)$
- $L_i(x) = L_{\omega^i}(x) = \frac{\prod_{j \neq i}(x-\omega^j)}{\prod_{j \neq i}(\omega^i - \omega^j)} = \frac{\omega^i}{n} \cdot \frac{x^n - 1}{x - \omega^i}$.

KZG Polynomial Commitment/Pairing Curve BLS12-381

- Gives groups $G_1 = \langle g_1 \rangle, G_2 = \langle g_2 \rangle$ and G_T (of the same prime order p) along with a bilinear pairing operation e.
- For every $\alpha, \beta \in \mathbb{Z}_p^*$, we have that $e(g_1^\alpha, g_2^\beta) = e(g_1, g_2)^{\alpha\beta}$.
- **Setup:** srs generation that supports committing to degree $d-1$ polynomials:
  - Sample $\tau \leftarrow \mathbb{Z}_p^*$.
  - srs $= (h_0 = g_1, h_1 = g_1^\tau, g_1^{\tau^2}, \ldots h_d = g_1^{\tau^{d-1}}, g_2, h' = g_2^\tau)$
- **Commitment:** Given srs and a polynomial $f(x) = c_0 + c_1 x + \ldots c_{d-1}x^{d-1}$ of degree $d-1$, we can compute $\mathrm{Com}(f)$ as:

$$F = \mathrm{Com}(f) = g_1^{f(\tau)} = \prod_{i=0}^{d-1} h_i^{c_i}$$

- **Opening:** Show that $f(z) = s$. In this case, $g(x) = f(x) - s$ is such that $g(z) = 0$. Or, $x - z$ divides $f(x) - s$.
- Sender computes $T(x) = \frac{f(x)-f(z)}{x-z}$ and sends $W = \mathrm{Com}(T)$.
- **Receiver Accepts if:** $e\left(\frac{F}{g_1^s}, g_2\right) = e\left(W, \frac{h'}{g_2^z}\right)$.

Permutation Check: How to Prove — Warmup!

Permutation Check: How to check that $\sigma(\zeta_1 ... \zeta_n) = (\zeta_1 ... \zeta_n)$.

How to test?

- Check two multisets $(\zeta_1, \zeta_2 ... \zeta_n)$ and $(\zeta'_1, \zeta'_2 ... \zeta'_n)$ are the same. How about a check:

$$\prod_i \zeta_i \stackrel{?}{=} \prod_i \zeta'_i$$

- How about this instead over polynomials?

$$\prod_i (\zeta_i + x) \stackrel{?}{=} \prod_i (\zeta'_i + x)$$

- How about a specific permutation σ?

$$\prod_{i=1}^{n} (\zeta_i + iy + x) \stackrel{?}{=} \prod_{i=1}^{n} (\zeta_i + \sigma(i)y + x)$$