CS 171: Discussion Section 11 (April 15)

1 Zero-Knowledge Protocol for Graph Isomorphism

Two graphs are **isomorphic** if it is possible to permute the vertices of one graph to obtain the other graph.

Let $G = (V, E)$ be a graph with n vertices: $V = \{1, \ldots, n\} = [n]$. Let $\pi : [n] \to [n]$ be a permutation of the vertices. We can define $\pi(G)$ to be the graph that results from permuting G's vertices according to π.[1]

More formally, $\pi(G) = (V', E')$ is a graph with vertex set $V' = V$ and edge set

$$E' = \{(u, v) \in V \times V : \left(\pi^{-1}(u), \pi^{-1}(v)\right) \in E\}$$

Definition 1.1 (Isomorphic Graphs)**.** *Two graphs G_0 and G_1 are **isomorphic** (notated as $G_0 \simeq G_1$) if they have the same number of vertices n, and there exists a permutation $\pi^* : [n] \to [n]$ such that*

$$G_0 = \pi^*(G_1)$$

Question: Give a zero-knowledge proof system for the language of isomorphic graphs $\mathcal{L} = \{(G_0, G_1) : G_0 \simeq G_1\}$. Prove that the scheme satisfies completeness, soundness, and zero-knowledge.

[1] It's technically an abuse of notation to write $\pi(G)$ since π was defined to take a vertex as input, not a graph, but we'll do it anyways.

1.1 Proof System Definitions

In this problem, the prover's goal is to convince a verifier that a given pair of graphs (G_0, G_1) are isomorphic. We will use the following terminology. The **language**

$$\mathcal{L} = \{(G_0, G_1) : G_0 \simeq G_1\}$$

is the set of all pairs of graphs that are isomorphic to each other. $x := (G_0, G_1)$ is called an **instance**, and the prover's job is convince a verifier that a given instance x is in the language $\mathcal{L}$.

One simple way to prove that $G_0 \simeq G_1$ is to provide a permutation π^* such that $G_0 = \pi^*(G_1)$. Then a verifier can check whether the condition $G_0 = \pi^*(G_1)$ is satisfied.

Let's put this in more abstract terms. The **witness** $w := \pi^*$ is a proof that $x \in \mathcal{L}$. Let $R(x, w)$ be the function that verifies the witness:

$$R[(G_0, G_1), \pi^*] = \begin{cases} 1, & G_0 = \pi^*(G_1) \\ 0, & \text{otherwise} \end{cases}$$

R outputs 1 if and only if w is a valid proof that $x \in \mathcal{L}$.

Completeness and Soundness

The goal of a zero-knowledge proof system is to convince the verifier that $x \in \mathcal{L}$ without revealing any information about w to the verifier.

Syntax of the protocol: The prover takes inputs $(1^\lambda, x, w)$, and the verifier takes inputs $(1^\lambda, x)$. $\lambda \in \mathbb{N}$ is the security parameter. x is the instance that the prover will try to prove belongs to $\mathcal{L}$. In order for the proof to succeed, w should be a valid witness for x ($R(x, w) = 1$). After some interaction between the prover and verifier, the verifier outputs a bit indicating whether they accept or reject the proof that $x \in \mathcal{L}$.

This protocol should have the following three properties: completeness, soundness, and zero-knowledge. We'll define them below.

Let (P, V) be the **honest prover and verifier**, respectively, who follow the protocol as-written. Let (P^*, V^*) be a **dishonest prover and verifier**, respectively, who may deviate from the protocol.

Completeness says that a valid proof will be accepted with overwhelming probability.

Definition 1.2 (Completeness)**.** *The protocol satisfies **completeness** if when $P(1^\lambda, x, w)$ and $V(1^\lambda, x)$ interact and their inputs satisfy $R(x, w) = 1$, then the verifier will accept the proof with probability $\geq 1 - \mathsf{negl}(\lambda)$.*

Soundness says that if $x \notin \mathcal{L}$, then no adversarial prover will be able to "trick" the verifier into accepting the proof with greater than negligible probability.

Definition 1.3 (Soundness)**.** *The protocol satisfies **soundness** if for any $x \notin \mathcal{L}$ and any adversarial prover P^*, when P^* and $V(1^\lambda, x)$ interact, then the verifier will accept the proof with probability $\leq \mathsf{negl}(\lambda)$.*

Zero-Knowledge

Zero-knowledge says that an adversarial verifier cannot learn anything about w during the protocol because the information available to the verifier (their view) can be simulated without knowledge of w.

To make this definition more formal, let's establish some notation.

- When $V^*(1^\lambda, x)$ interacts with $P(1^\lambda, x, w)$, let the verifier's **view**, $\mathsf{view}(V^*; 1^\lambda, x, w)$, be a list of the verifier's inputs $(1^\lambda, x)$, any messages sent to or from the verifier during the protocol, and anything output by the verifier.
- Let the simulator Sim be an algorithm that tries to simulate the verifier's view given only $(1^\lambda, x)$. Note that Sim is not given w.

  Next, Sim is given black-box access to V^* (notated as Sim^{V^*}). This means Sim can run V^* on any inputs of its choice and rewind V^* to any step, but it cannot modify the internal workings of V^*.

  Finally, the expected value of Sim's runtime should be polynomial in the size of Sim's inputs.
- Let the distinguisher D be an algorithm that outputs a bit and tries to distinguish the verifier's real view from the one produced by the simulator.

Informally, the protocol satisfies **zero-knowledge** if whenever $R(x, w) = 1$, the distinguisher cannot distinguish the real view from the simulated view.

Here is a more-formal definition:

Definition 1.4 (Black-Box Zero-Knowledge)**.** *The protocol satisfies (black-box)* ***zero-knowledge*** *if there exists a simulator* Sim *such that for any adversarial* V^* *and any inputs* $(1^\lambda, x, w)$ *that satisfy* $R(x, w) = 1$ *and any distinguisher* D*:*

$$\left| \Pr\left[D\big(\mathsf{view}(V^*; 1^\lambda, x, w)\big) \to 1\right] - \Pr\left[D\big(\mathsf{Sim}^{V^*}(1^\lambda, x)\big) \to 1\right] \right| \leq \mathsf{negl}(\lambda)$$

Finally, **honest-verifier zero-knowledge** is a weaker form of security in which zero-knowledge only holds when the verifier follows the protocol honestly.

Definition 1.5 (Black-Box Honest-Verifier Zero-Knowledge)**.** *The protocol satisfies (black-box)* ***honest-verifier zero-knowledge*** *if there exists a simulator* Sim *such that for the honest verifier* V *and any inputs* $(1^\lambda, x, w)$ *that satisfy* $R(x, w) = 1$ *and any distinguisher* D*:*

$$\left| \Pr\left[D\big(\mathsf{view}(V; 1^\lambda, x, w)\big) \to 1\right] - \Pr\left[D\big(\mathsf{Sim}^{V}(1^\lambda, x)\big) \to 1\right] \right| \leq \mathsf{negl}(\lambda)$$

2 Polynomial Commitments

Question: Prove that the KZG commitment scheme is not hiding.

2.1 The KZG Commitment Scheme

1. $\mathsf{Gen}(1^n)$:
   (a) Let d be polynomial in n.
   (b) Set up a bilinear map by sampling
   $$\mathsf{pp} = (\mathbb{G}, \mathbb{G}_T, q, g, e) \leftarrow \mathcal{G}(1^n)$$
   (c) Sample $\tau \leftarrow \mathbb{Z}_q^*$.
   (d) Finally, output
   $$\mathsf{params} = \left(\mathsf{pp}, g^{\tau}, g^{(\tau^2)}, \ldots, g^{(\tau^d)}\right)$$
2. $\mathsf{Commit}(\mathsf{params}, f)$:
   (a) Let f be a polynomial $\in \mathbb{Z}_q[X]$ of degree $\leq d$:
   $$f(X) = \sum_{i=0}^{d} \alpha_i \cdot X^i$$
   where every $\alpha_i \in \mathbb{Z}_q$.
   (b) Compute and output the commitment:
   $$\begin{aligned}\mathsf{com}_f &= \prod_{i=0}^{d} \left(g^{(\tau^i)}\right)^{\alpha_i} \\ &= g^{f(\tau)}\end{aligned}$$
   (c) Open:
      i. Let $z \in \mathbb{Z}_q$ be an input on which to open the commitment, and let $s = f(z)$. Now the sender will prove that $s = f(z)$.
      ii. The sender computes the polynomial:
      $$t(X) := \frac{f(X) - s}{X - z}$$
      and a commitment $\mathsf{com}_t = \mathsf{Commit}(\mathsf{params}, t)$. Then they send (z, s, T) to the receiver.
      iii. The receiver accepts the opening if and only if:
      $$e(\mathsf{com}_f \cdot g^{-s}, g) = e(\mathsf{com}_t, g^{\tau} \cdot g^{-z}) \tag{2.1}$$
   Note that equation 2.1 is satisfied if and only if:
   $$\begin{aligned}e(g^{f(\tau)-s}, g) &= e(g^{t(\tau)}, g^{\tau - z}) \\ f(\tau) - s &= t(\tau) \cdot (\tau - z)\end{aligned}$$