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*:*

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

Solution

1. Here is a zero-knowledge protocol for graph isomorphism:

   (a) Inputs: The prover and verifier both take as input a security parameter (1^n) and a pair of graphs $x = (G_0, G_1)$ on n vertices. The prover also takes a private input $w = \pi^*$, which is a permutation satisfying: $G_0 = \pi^*(G_1)$.

   (b) Repeat the following procedure λ times:

      i. The prover samples a random permutation $\pi_R \leftarrow S_n$ and sends the graph $G_R = \pi_R(G_0)$ to the verifier.

      ii. The verifier samples $b \leftarrow \{0, 1\}$ and sends it to the prover.

      iii. If $b = 0$, the prover sets $\pi_P = \pi_R$, and if $b = 1$, the prover sets $\pi_P = \pi_R \circ \pi^*$. Then they send π_P to the verifier.

      iv. The verifier checks that:

      $$G_R = \pi_P(G_b)$$

      If the check fails, then the protocol ends, and the verifier outputs 0 (reject).

   (c) If all rounds of the protocol succeeded without rejection, then the verifier outputs 1 (accept).

2. **Claim 1.6** (Completeness)**.** *If $G_0 = \pi^*(G_1)$, and both the prover and verifier follow the protocol honestly, then the verifier will certainly accept.*

   Proof.

   (a) If $b = 0$, then $\pi_P = \pi_R$, and

   $$\pi_P(G_b) = \pi_R(G_0) = G_R$$

   (b) If $b = 1$, then $\pi_P = \pi_R \circ \pi^*$, and

   $$\pi_P(G_b) = \pi_R \circ \pi^*(G_1) = \pi_R(G_0) = G_R$$

   (c) The verifier's check will pass in both cases, so the verifier will accept the proof.

   □

3. **Claim 1.7** (Soundness)**.** *If $G_0 \not\simeq G_1$, then for any adversarial prover P^* interacting with the honest verifier V, the verifier will accept the proof with probability $\leq 2^{-\lambda}$.*

   Proof.

(a) On each round, the graph G_R cannot be isomorphic to both G_0 and G_1 because otherwise, G_0 would be isomorphic to G_1. This follows from the transitive property of isomorphic graphs: If $G_R \simeq G_0$ and $G_R \simeq G_1$, then $G_0 \simeq G_1$.
This means that with probability $\geq \frac{1}{2}$, the verifier picks a b such that $G_R \not\simeq G_b$.

(b) If $G_R \not\simeq G_b$, then there is no permutation π_P that the prover can send for which $G_R = \pi_P(G_b)$, so the verifier will reject the proof.

(c) We've shown that on each round of the protocol, the probability that the verifier accepts is $\leq \frac{1}{2}$. Since b is sampled independently on each round, the probability that the verifier's check passes on all of the λ rounds is $\leq 2^{-\lambda}$. This means that the probability that the verifier accepts the proof is $\leq 2^{-\lambda}$.

□

4.

Claim 1.8 (Zero-Knowledge)**.** *The protocol above satisfies zero-knowledge.*

Proof.

(a) Intuition: Hypothetically, if the prover was told b before they had to output G_R or π_P, then it's easy for them to find a (G_R, π_P) that the verifier will accept. They just sample π_P randomly, and then choose $G_R = \pi_P(G_b)$. This can be done without any knowledge of π^*.
The zero-knowledge simulator will do something similar. They will guess b before they have to compute (G_R, π_P). It their guess is wrong, they can just rewind the protocol and try again.

(b) To prove zero-knowledge, we must construct a simulator Sim that can simulate the verifier's view without knowing π^*.

$\mathsf{Sim}^{V^*}(1^\lambda, G_0, G_1)$:

i. For each round of the protocol:
   A. Sim samples $b' \leftarrow \{0,1\}$, samples a random permutation π_P, and computes $G_R = \pi_P(G_b)$. Then they run V^* on input G_R.
   B. Next, V^* outputs a bit b. If $b' = b$, then Sim sends π_P to V^*. If $b' \neq b$, then Sim rewinds V^*, and restarts the simulation at the beginning of the round.

ii. Finally, Sim outputs the verifier's inputs $(1^n, G_0, G_1)$ and any messages sent to or from V^* in the simulated protocol.

(c) On each round, $\Pr[b' = b] = \frac{1}{2}$ because b' is sampled uniformly at random. Therefore, Sim requires 2 attempts on average to simulate a round correctly.

(d) On any given round, if $b' = b$, then the transcript of the simulated protocol has the same distribution as in the real protocol.
In the real protocol, given (G_0, G_1, b, π^*), π_P is a uniformly random permutation, and G_R is the unique graph satisfying $G_R = \pi_P(G_b)$ (see lemma 1.9). This is the same distribution as in the simulated protocol.

(e) Since the distribution of the verifier's view is identical in the real protocol and the simulated protocol, then for any distinguisher D:

$$\left| \Pr\left[D\big(\mathsf{view}(V^*; 1^\lambda, G_0, G_1, \pi^*)\big) \to 1\right] - \Pr\left[D\big(\mathsf{Sim}^{V^*}(1^\lambda, G_0, G_1)\big) \to 1\right] \right| = 0$$

Therefore, the protocol satisfies zero-knowledge.

□

5.

Lemma 1.9. *In the real protocol, given (G_0, G_1, b, π^*), π_P is a uniformly random permutation, and G_R is the unique graph satisfying: $G_R = \pi_P(G_b)$.*

Proof. We will show that π_R is a uniformly random permutation. First, if $b = 0$, then $\pi_P = \pi_R$, so π_P is uniformly random as well. If $b = 1$, the $\pi_P = \pi_R \circ \pi^*$. This π_P is uniformly random as well because it is a uniformly random permutation (π_R) composed with a fixed permutation (π^*).

Next, G_R satisfies $\pi_P(G_b)$, so G_R is completely determined by (G_0, G_1, b, π_P). □

□

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:

$$e(g^{f(\tau)-s}, g) = e(g^{t(\tau)}, g^{\tau-z})$$
$$f(\tau) - s = t(\tau) \cdot (\tau - z)$$

Solution

(a) Key Ideas: The function $\mathsf{Commit}(\mathsf{params}, f)$ is deterministic, and two different polynomials will produce different commitments, with overwhelming probability. Given two different polynomials f_0, f_1, an adversarial receiver can compute $\mathsf{Commit}(\mathsf{params}, f_0)$ and $\mathsf{Commit}(\mathsf{params}, f_1)$ on their own and then check which of the two values the sender produces as their commitment.

(b)

Claim 2.1. *For any two distinct polynomials $f_0, f_1 \in \mathbb{Z}_q[X]$ of degree $\leq d$:*

$$\Pr_{\tau \leftarrow \mathbb{Z}_q^*}[f_0(\tau) \neq f_1(\tau)] \geq 1 - \mathsf{negl}(n)$$

Proof. Since f_0 and f_1 have degree $\leq d$ and $f_0 \neq f_1$, then $f_0(X) - f_1(X)$ is a non-zero polynomial of degree $\leq d$. Therefore, $f_0(X) - f_1(X)$ has at most d roots. Next, $f_0(\tau) = f_1(\tau)$ if and only if τ is a root of $f_0(X) - f_1(X)$. Since τ is sampled uniformly from $\mathbb{Z}_q^*$,

$$\Pr[f_0(\tau) = f_1(\tau)] \leq \frac{d}{q-1} = \mathsf{negl}(n)$$

where we used the fact that $d = \mathsf{poly}(n)$, and $\frac{1}{q} = \mathsf{negl}(n)$.
Therefore,

$$\Pr_{\tau \leftarrow \mathbb{Z}_q^*}[f_0(\tau) \neq f_1(\tau)] \geq 1 - \mathsf{negl}(n)$$

□

(c) Here is how to break the hiding property of the KZG commitment:

i. The adversary selects two distinct polynomials $f_0, f_1 \in \mathbb{Z}_q[X]$ of degree $\leq d$, and asks the sender to commit to one of them.
ii. The adversary computes $\mathsf{com}_0 = \mathsf{Commit}(\mathsf{params}, f_0)$ and $\mathsf{com}_1 = \mathsf{Commit}(\mathsf{params}, f_1)$. These are deterministic computations.
iii. The sender commits to one of the polynomials by computing $\mathsf{com}^* = \mathsf{Commit}(\mathsf{params}, f_b)$ for some $b \in \{0, 1\}$. They send com^* to the adversary.
iv. The adversary checks whether $\mathsf{com}^* = \mathsf{com}_0$. If so, they output 0. If not, they output 1.

(d) If $f_0(\tau) \neq f_1(\tau)$ (which occurs with overwhelming probability over the choice of τ), then the adversary correctly guesses which polynomial was committed to. This is because

$$\mathsf{com}_0 = g^{f_0(\tau)}, \quad \mathsf{com}_1 = g^{f_1(\tau)}, \quad \mathsf{com}^* = g^{f_b(\tau)}$$

Then $\mathsf{com}_0 = \mathsf{com}^*$ if and only if $f_0(\tau) = f_b(\tau)$, which occurs if and only if $b = 0$.

(e) Therefore, this adversary breaks hiding.

□