COM S 6810 Theory of Computing March 26, 2009

Lecture 18: Zero-Knowledge Proofs

Instructor: Rafael Pass *Scribe: Igor Gorodezky*

1 The formal definition

We intuitively defined an interactive proof to be zero-knowledge if after reading the proof the verifier cannot (efficiently) perform computational tasks that she could not (efficiently) perform before. Just as soundness is a desirable property in an interactive proof system with a possibly malicious prover, zero-knowledge is a property that could be plausibly desired by a prover who suspects a malicious verifier.

Let us state the formal definition of a zero-knowledge proof system, due to Goldwasser, Micali and Rackoff. Given an interactive proof system $\langle P, V \rangle$ for a language $\mathcal{L}$ and an input x, we define the *view of V on x* to be all messages sent from P to V as well as all random bits used by V during the execution of the protocol on x. We will write the view on x as

$$\mathrm{View}_V[P(x) \leftrightarrow V(x)].$$

Also, we define an *expected PPT* Turing machine to be a TM whose expected running time is polynomial in the input length.

Definition 1 (Zero-knowledge proof system) *An interactive proof system $\langle P, V \rangle$ for a language $\mathcal{L}$ is* zero-knowledge *if for any PPT verifier V^* there exists an expected PPT simulator S such that*

$$\forall\, x \in \mathcal{L}, z \in \{0,1\}^*, \quad \mathrm{View}_{V^*}[P(x) \leftrightarrow V^*(x,z)] = S(x,z)$$

As usual, P has unlimited computation power (in practice, P must be a randomized TM). Intuitively, the definition states that an interactive proof system $\langle P, V \rangle$ is zero-knowledge if for any verifier V^* there exists an efficient simulator S that can essentially produce a transcript of the conversation that would have taken place between P and V^* on any given input. Therefore, having a conversation with P cannot teach V^* to compute something she could not have computed before.[1]

The string z in the definition plays the role of "prior knowledge"; V^* cannot use any prior knowledge string z to mine information out of its conversation with P because we demand that if S is also given this prior knowledge then it can reproduce the conversation between V^* and P just as before.

[1]Modulo the fact that S is slightly more powerful than V^* since it is allowed to run for *expected* polynomial time.

Observe that both $S(x, z)$ and View_{V^*} are distributions over random strings. Therefore the equality between these quantities in the definition means that these distributions are identical. This requirement is formally called *perfect* zero-knowledge. This can be relaxed by requiring that instead of being identical, the two distributions cannot be told apart by any efficient computation; this defines what is called *computational* zero-knowledge. In this lecture, zero-knowledge proofs will be perfect unless qualified otherwise.

We can also relax the assumption that a simulator S exists for any V^* and instead require only that there exists some V^*, an honest verifier, for which there exists an S with the desired property. This yields what is called *honest verifier* zero-knowledge.

2 Graph isomorphism

As a first example let us give a zero-knowledge proof system for graph isomorphism (GI).

Given two graphs G_0 and G_1, P wants to convince V that he knows a permutation π such that $\pi(G_0) = G_1$. P could simply send π to V, but that is hardly zero-knowledge; we want to convince V that π is an isomorphism without revealing anything about it. The protocol is as follows.

- $P \to V$: P randomly chooses a permutation σ and a bit $b \in \{0, 1\}$, computes $H = \sigma(G_b)$, and sends H to V.
- $V \to P$: V chooses a bit $b' \in \{0, 1\}$ and sends it to P.
- $P \to V$: P sends the permutation τ to V, where

$$\tau = \begin{cases} \sigma & \text{if } b = b' \\ \sigma\pi^{-1} & \text{if } b = 0, b' = 1 \\ \sigma\pi & \text{if } b = 1, b' = 0 \end{cases}$$

- V accepts if and only if $H = \tau(G_{b'})$.

This protocol has perfect completeness because if π really is an isomorphism then it will always be the case that $\tau(G_{b'}) = \sigma(G_b)$. Moreover, assume for the moment that V is an honest verifier that chooses b' randomly. In this case the protocol is also sound, for if π is not an isomorphism then $\tau(G_{b'}) = \sigma(G_b)$ only if $b = b'$, and this happens with probability 1/2 by our choice of verifier.

We also claim that this is an honest verifier zero-knowledge protocol for the verifier V that chooses b' randomly. To see why, note first that

$$\text{View}_V[P(G_0, G_1) \leftrightarrow V(G_0, G_1)] = (b', H, \sigma).$$

The simulator S for V operates by randomly choosing b' and σ and then outputting $(b', \sigma(G_{b'}), \sigma)$. S satisfies the definition because the distribution of $S(G_0, G_1)$ is precisely the distribution of View_V whenever $G_0 \cong G_1$.

What if we allow malicious verifiers? Is the protocol perfect zero-knowledge as well as honest verifier zero-knowledge?

Lemma 1 *The GI protocol above is perfect zero-knowledge.*

Proof. We must verify that a simulator S exists for an arbitrary verifier V^*. Given a verifier V^* let $h^*(G_0, G_1, z, H)$ be the bit chosen by $V^*(G_0, G_1, z)$ at the second step of the protocol, after having received H.

Define $S(G_0, G_1, z)$ to operate as follows:

- Randomly choose a permutation σ and $b \in \{0, 1\}$.
- Set $H = \sigma(G_b)$ and $b' = h^*(G_0, G_1, z, H)$.
- If $b' = b$ output (b', H, σ), otherwise restart with a new σ, b.

In order to show that S is a valid simulator it suffices to prove that if $G_0 \cong G_1$ then (a) S runs in expected polynomial time and (b) the distribution of its output equals the distribution of View_{V^*}.

To prove the first claim, observe first that if $G_0 \cong G_1$ then, since σ is random, h^* cannot depend on b. Therefore b' and b are chosen independently, hence are equal with probability 1/2. Therefore S must run only twice in expectation before halting.

The observation that $\text{Prob}[b = b'] = 1/2$ also proves that the distribution of S equals the distribution of View_{V^*}, because it implies that whether or not S halts on a particular choice of $\{b, \sigma\}$ is independent of $\{b, \sigma\}$ and therefore of H. It follows that the distribution of the output (b', H, σ) is exactly the distribution of View_{V^*}. ■

3 Computational zero knowledge

Recall that $\langle P, V \rangle$ is a computational zero-knowledge protocol if it satisfies Definition 1 with the condition

$$\text{View}_{V^*}[P(x) \leftrightarrow V^*(x, z)] = S(x, z)$$

weakened to the requirement that the two distributions be indistinguishable by efficient computations (we'll keep this discussion casual and therefore omit any formal definition of efficiency).

We write CZK for the class of languages that have computational zero-knowledge proofs. The following theorem is due to Goldreich, Micali and Wigderson.

Theorem 2 *If one-way functions exist then* $\mathsf{NP} \subseteq \mathsf{CZK}$.

Let us sketch the proof. The existence of one-way functions allows *bit commitment*: it is possible for P to "seal" a bit b and communicate the sealed bit to V in a way that only reveals the bit to V with P's permission and also guarantees that P could not have changed b between the time it was sealed and the time it was revealed.

To prove the theorem it suffices to give a computational zero-knowledge protocol for 3-coloring (3COL), which is NP-complete. Given a graph G with $|E|$ edges, P wants to convince V that $c : V(G) \to \{R, G, B\}$ is a valid 3-coloring. The protocol is as follows

- P permutes the three colors and seals the label on each vertex using bit commitment.
- V picks an edge (u, v) and send u, v to P.
- P reveals $c(u)$ and $c(v)$.
- V accepts if and only if $c(u) \neq c(v)$.

We observe that this protocol has perfect completeness against an arbitrary prover: if c is a valid 3-coloring then $c(u) \neq c(v)$ for any edge (u, v). This protocol is also sound because if c is not a valid 3-coloring then $c(u) = c(v)$ for some edge (u, v), meaning any prover fails to convince V with probability at least $1/|E|$.

All very good, but is this a zero-knowledge protocol? It is, and the simulator $S(G)$ utilizes the same restarting trick we saw in the GI protocol. $S(G)$ operates as follows:

- Pick a random edge (u, v) and define a coloring c by
$$c(w) = \begin{cases} R & \text{if } w = u \\ G & \text{if } w = v \\ B & \text{otherwise} \end{cases}$$
- Permute the three colors and seal the label on each vertex using bit commitment (just like P).
- Query V for an edge (u', v').
- If $(u', v') = (u, v)$, reveal $c(u')$, $c(v')$ and output $(u, v, c(u), c(v))$; otherwise restart.

It is easy to see that $S(G)$ makes only $|E|$ restarts in expectation hence has constant expected running time. It is also the case, though this is quite difficult to prove, that the distribution of its output is indistinguishable from the distribution of View_V, which implies that our protocol for 3COL is computational zero-knowledge.

We end with the following generalization of Theorem 2 due to Ben-Or, Goldreich, Goldwasser, Håstad, Kilian, Micali and Rogaway.

Theorem 3 *If one-way functions exist then* $\mathsf{IP} = \mathsf{CZK}$.

Clearly, $\mathsf{CZK} \subseteq \mathsf{IP}$ by definition. The opposite inclusion is proved by showing that any public-coin protocol has an equivalent computational zero-knowledge protocol. It is possible to construct such a protocol by, very roughly speaking, first having the prover use bit commitment to seal the transcript of an accepting interaction in the original public-coin protocol. Then, the claim that the sealed bits encode an accepting transcript is an NP statement, hence by Theorem 2 can be verified using a computational zero-knowledge protocol.