Lecture 23

Zero Knowledge Proofs

Interactive Proofs

As a warm-up, we begin by discussing basic interactive proofs, where we aren't yet thinking about concealing information from the verifier. In an interactive proof, the goal is to prove interactively whether a string x belongs to an NP-language L, where L is

$$L = \{x \in \{0,1\}^* : \exists w \in \{0,1\}^*. R(x,w) = 1\}.$$

That is, L being an NP-language means that for any string x in L, there is some witness w for x that allows one to verify that x is indeed in L. Specifically, w needs to be polynomial-sized in x. The interaction is between a prover $P(x,w)$ who has access to the string in question and its witness w, and a verifier $V(x)$ who has access only to x.

An interactive proof system needs to satisfy two properties:

Completeness:

If a string x is in fact in L then our system should indeed verify this with probability close to 1. Formally:

$$x \in L \Longrightarrow \Pr[P(x,w) \rightleftarrows V(x) \to 1] \geq 1 - \text{ negligible.}$$

Soundness:

If a string x is not in L, then no malicious prover should be able to convince a verifier that x is actually L with reasonable probability. Formally:

$$x \notin L \Longrightarrow \forall P^*. \Pr[P^*(x) \rightleftarrows V(x) \to 1] \leq \text{negligible.}$$

Note that we don't bother to write $P^*(x,w)$ because if x is not in L, then there is not a witness to its membership.

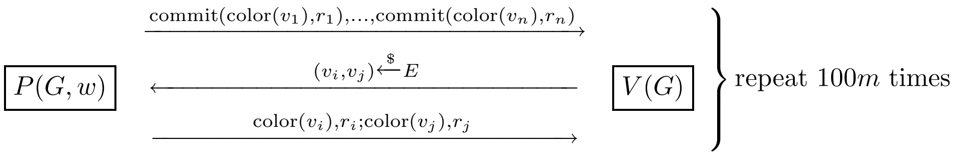
Example: 3-Coloring

As a concrete example of an interactive proof, we show how to interactively prove that a graph has a 3-coloring. Recall that a 3-coloring is a way to color each node of a graph using at most 3 different colors so that no two adjacent nodes share the same color. In our example,

$$L = \{G : G \text{ can be 3-colored}\} = \{G : \exists \text{ a 3-coloring of } G\}.$$

This is easily seen to be an NP-language, since the witness to G's 3-coloring is the coloring itself, namely we produce the color for each node. The proof is system is the following.

Let $G = (V,E)$, and let $V = \{v_1, \ldots, v_n\}$.



Each iteration of this process is called a round. After each round, V needs to check that $\text{color}(v_i) \neq \text{color}(v_j)$. If this check ever fails, then V rejects the proof. If the check succeeds on every round, then after all rounds V accepts the proof.

In other words, in this system, P uses a bit-commitment scheme to seal the color of each node in the graph, and sends these commitments to V. On each round, V chooses randomly an edge in the graph and sends this choice back to P, which then reveals the colors of the nodes that belong to that edge. V can then check whether or not those colors are the same.

Let's check if this system is complete and sound. Completeness is easy, since if w is really a 3-coloring, then no matter what edge V selects the colors of the nodes on this edge will never be the same, and the check will always succeed. Soundness is a bit more interesting. If G does not have a 3-coloring, then with any coloring, there is at least one edge somewhere in the graph where both of the vertices on this edge have the same color. The chance of discovering this edge on one round is approximately $\frac{1}{m}$, where there are m edges in the graph (the approximation comes from the fact that there is some small ϵ factor in the bit-commitment scheme). Thus the probability that we never discover this edge after $100m$ rounds is approximately $(1-\frac{1}{m})^{100m} \approx \frac{1}{e^{100}}$, which is a tiny probability. So the scheme is sound.

However, this protocol for proving the existence of a 3-coloring really stinks. After $100m$ round, it is extremely likely that we have revealed the coloring of every node in the graph. Thus with high probability we have revealed all of w, but in the process done a tremendous amount of work, when we could have simply sent w to V. However, once we introduce the idea of zero knowledge, we will see that this scheme can be easily repaired to provide a useful purpose: a zero knowledge protocol for proving that a graph has a 3-coloring.

Zero Knowledge Interactive Proof (ZKIP)

We want a proof protocol in which the verifier doesn't gain any "knowledge" during the interaction, other than the validity of the theorem being proved. But we need some kind of formalization of what "knowledge" is. In essence we decide that a zero knowledge interaction is one in which the verifier's interaction with the prover, who knows the witness w, can be simulated by an interaction with some algorithm S which does not know w. That is, S can make a fake transcript of the interaction that has the same distribution of responses as a real interaction. There are two formalizations.

Honest-verifier ZK:

Here we assume that the verifier is not malicious and plays by the rules. A protocol is *honest-verifier ZK* if

$$\exists S. \forall x \in L. \text{View}_V(P(x,w) \rightleftarrows V(x)) \sim S(x).$$

Here the view of V, View_V, is everything that V sees during the course of an interaction, namely the transcript of inputs to V, any messages it sees, and any coin tosses that it makes.

ZK:

Here we make no assumptions about the verifier. It may try to cheat by choosing the values it sends out in some calculated way, rather than choosing them at random. A protocol is *ZK* if

$$\forall V^*. \exists S^*. \forall x \in L. \text{View}_{V^*}(P(x,w) \rightleftarrows V^*(x)) \sim S^*(x).$$

Note here the placement of the quantifiers. This definition says that the simulator S^* can depend on the verifier V^*, not that there is some universal simulator. However, it often happens as a common case that there is a kind of universal simulator that needs only oracle access to V^* in order to properly simulate. That is, there is some $S^* = S^{V^*}$.

Modification of the Previous Example

We now go and fix up the example of the 3-coloring proof in order to make it zero knowledge. Define the protocol as follows:

Let $G = (V, E)$, and let $V = \{v_1, \ldots, v_n\}$.
$\pi \xleftarrow{\$} \text{Perm}(\{\text{red}, \text{green}, \text{blue}\})$

$$\boxed{P(G,w)} \quad \begin{array}{c} \xrightarrow{\text{commit}(\pi(\text{color}(v_1)),r_1),\ldots,\text{commit}(\pi(\text{color}(v_n)),r_n)} \\ \xleftarrow{\quad(v_i,v_j)\xleftarrow{\$}E\quad} \\ \xrightarrow{\pi(\text{color}(v_i)),r_i;\pi(\text{color}(v_j)),r_j} \end{array} \quad \boxed{V(G)} \left.\vphantom{\begin{array}{c}a\\a\\a\end{array}}\right\} \text{repeat } 100m \text{ times}$$

After each round, V checks whether $\pi(\text{color}(v_i)) \neq \pi(\text{color}(v_j))$. If the check ever fails, V rejects the proof, but if the check succeeds on every round then V accepts the proof.

The only difference between this protocol and the last is that instead of commiting to the actual coloring of the graph, on each round P selects a random permutation of the 3 colors involved and commits to the permuted coloring. Thus the actual colors that P commits to will vary from round to round, but if P indeed has a valid 3-coloring, the commitment on each round will indeed be to a valid 3-coloring.

We once again check the completeness and soundness of this protocol. It is complete for the same reason as before. If P has a 3-coloring, then no matter which edge V asks for, the adjacent vertices on that edge will have a different coloring, no matter what the permuted coloring actually is. It is also sound, for the same reasons and with the same probabilities. Merely permuting the colors doesn't change anything about the fact that if there isn't a valid 3-coloring, then there is at least one edge whose adjacent vertices have the same color, and the probability of discovering that edge is the same.

ZK

We now show that this protocol is in fact ZK, since V's view of this can be simulated, knowing only that the graph is 3-colorable. The principal involved in creating the simulation is that there is some probability on each round of guessing correctly which edge V will request. Once we make this guess, say that V will request the edge (v_i, v_j), we can prepare in advance a coloring that ensures that v_i and v_j are colored differently. If V actually does request this edge, then we're able to simulate what the real prover P would have responded. If V requests a different edge, we can't guarantee that we've sent it a coloring that will pass the check. But this is no matter, we simply don't record this round in the transcript, and go back to the top, make another guess, and try again, hoping that V will pick our edge this time. In other words, we're making an edited transcript that only records our successes and never our mistakes, even though we mostly will make mistakes. Formally, the protocol is:

$(v_i, v_j) \xleftarrow{\$} E$.
pick c such that $c(v_i) \neq c(v_j)$.

$$\boxed{S(G)} \quad \begin{array}{c} \xrightarrow{\text{commit}(c_1,r_1),\ldots,\text{commit}(c_n,r_n)} \\ \xleftarrow{\quad(v'_i,v'_j)\xleftarrow{\$}E\quad} \\ \xrightarrow{\text{if } i'=i, j'=j, \text{ send } c_i,r_i,c_j,v_j} \\ \text{otherwise, rewind} \end{array} \quad \boxed{V(G)} \left.\vphantom{\begin{array}{c}a\\a\\a\\a\end{array}}\right\} \text{repeat } 100m \text{ times}$$

The probability that S guesses the correct edge on each round is $\frac{1}{m}$. So we need to run this process about m times per round to get a correct guess. But this only means that the overall running time of the verifier is about $100m^2$. Against verifiers who choose (v_i, v_j) at random, we now have a simulator, so this protocol is honest-verifier ZK.

This scheme works against cheating verifiers as well. The only difference is that a cheating verifier doesn't choose its (v_i, v_j) randomly. But it still has no way of learning that the values $\text{commit}(c_1, r_1), \ldots, \text{commit}(c_n, r_n)$ are not a valid 3-coloring, if the commitment scheme being used has the hiding property. Thus this scheme is fully ZK.

Another Example: Graph Isomorphisms

Our language is the set of pairs of graphs that are isomorphic to each other:

$$L = \{(G, H) : \exists \pi . H = \pi G\}.$$

We'll establish that the following proof protocol is ZK.

$\sigma \xleftarrow{\$} \text{Perm}(V)$

$$\left.\begin{array}{ccc} & \xrightarrow{\qquad G' = \sigma G \qquad} & \\ \boxed{P(G,H,w)} & \xleftarrow{\qquad c \xleftarrow{\$} \{0,1\} \qquad} & \boxed{V(G,H)} \\ & \xrightarrow{\gamma = \begin{cases} \sigma & \text{if } c = 0 \\ \sigma \circ \pi^{-1} & \text{if } c = 1 \end{cases}} & \end{array}\right\} \text{repeat 100 times}$$

After each round, V checks whether

$$G' = \begin{cases} \gamma G & \text{if } c = 0 \\ \gamma H & \text{if } c = 1 \end{cases}.$$

Completeness of this protocol is clear, because if G and H are really isomorphic, then G' will be isomorphic to both G and H and the check will always be successful. Soundness follows from the fact that if G and H are not isomorphic, then there is only a $\frac{1}{2}$ chance that the check will be successful. That is, the check will only work if $c = 0$, so that V is checking that $G' = \gamma G = \sigma G$, which it clearly is by definition. So after 100 rounds of this, the probability the check never will have failed will be $\frac{1}{2^{100}}$ which of course is negligible.

ZK and Honest-Verifier ZK:

The scheme is both honest-verifier ZK and ZK, as the following simulation demonstrates.

$c' \xleftarrow{\$} \{0,1\}$
$\gamma \xleftarrow{\$} \text{Perm}(V)$

$$\left.\begin{array}{ccc} & \xrightarrow{G' = \begin{cases} \gamma G & \text{if } c = 0 \\ \gamma H & \text{if } c = 1 \end{cases}} & \\ \boxed{S(G,H)} & \xleftarrow{\qquad c \qquad} & \boxed{V^*(G,H)} \\ & \xrightarrow{\text{if } c \neq c',\ \text{rewind,else } \sigma} & \end{array}\right\} \text{repeat 100 times}$$

Here, the simulator has a probability of $\frac{1}{2}$ of guessing correctly the bit that V^* will choose, and so we need to run the above steps about twice per round to get a successful guess. The scheme works no matter how V^* chooses its bit c, whether it is cheating or not, so the simulation shows that the protocol is both honest-verifier ZK and ZK.

We conclude with a useful and elegant theorem.

Theorem. *Every NP-language has a ZKIP.*

Proof. Let L be an NP-language. Then since 3-coloring is an NP-complete problem, L is Karp-reducible to 3-coloring. In other words, for every $x \in L$, we can construct a graph G that is 3-colorable if and only if $x \in L$. Now simply prove that that graph G is 3-colorable, for instance using the scheme just described. Thus we have a ZKIP for L. $\square$