COM S 6830 – Cryptography October 22nd, 2009

Lecture 16: Zero Knowledge

Instructor: Rafael Pass *Scribe: Matthew Paff*

1 Views

Let $\mathcal{A}$ and $\mathcal{B}$ be two interactive Turing machines. In the context of interactive proofs, $\mathcal{A}$ and $\mathcal{B}$ will be given a common input (denoted x), and possibly auxiliary inputs (denoted z_1 and z_2). During execution, $\mathcal{A}$ and $\mathcal{B}$ have access to a random bit generator, and the messages sent by the other machine. The execution is of $\mathcal{A}$ and $\mathcal{B}$ is denoted $\mathcal{A}(x, z_1) \leftrightarrow \mathcal{B}(x, z_2)$. $\mathcal{A}$'s view of the execution, denoted $\mathsf{view}_{\mathcal{A}}[\mathcal{A}(x, z_1) \leftrightarrow \mathcal{B}(x, z_2)]$, is the tuple (x, z_1, r_1, M_1), where r_1 contains the results of $\mathcal{A}$'s random bit generator, and M_1 is the set of messages sent from $\mathcal{B}$ to $\mathcal{A}$.

2 Zero Knowledge Definitions:

Definition 1 $\langle \mathcal{P}, \mathcal{V} \rangle$ *is* honest verifier zero knowledge *for the language* $L \in \mathsf{NP}$ *if there exists an expected PPT* $\mathcal{S}$ *st* $\forall\ x \in L,\ w \in R_L(x)$*, and all* $z \in \{0,1\}^*$*:*

$$\{\mathsf{view}_{\mathcal{V}}\left[\mathcal{P}(x, w) \leftrightarrow \mathcal{V}(x, z)\right]\} \approx \{\mathcal{S}(x, z)\}$$

z above contains any auxiliary information the verifier has about x. What this definition basically says that no matter what information V has about x, $\mathcal{V}$ will not learn any new information from the execution since it could have just simulated its view of the execution (using $\mathcal{S}$) on its own.

There is a big problem with the above definition however: what if $\mathcal{V}$ doesn't do what it is supposed to? $\mathcal{V}$ could be malicious, and the above definition says nothing about whether the verifier could gain knowledge if it ran a different machine. Therefore, we have a second, stronger definiton of zero knowledge:

Definition 2 $\langle \mathcal{P}, \mathcal{V} \rangle$ *is* perfect zero knowledge *for the language* $L \in \mathsf{NP}$ *if for all PPT* $\mathcal{V}^*$*, there exists an expected PPT* $\mathcal{S}$ *st* $\forall\ x \in L,\ w \in R_L(x)$*, and all* $z \in \{0,1\}^*$*:*

$$\{\mathsf{view}_{\mathcal{V}^*}\left[\mathcal{P}(x, w) \leftrightarrow \mathcal{V}^*(x, z)\right]\} \approx \{\mathcal{S}(x, z)\}$$

In other words, no matter what the verifier does, it cannot gain any knowledge from an execution.

We now consider a stronger definition. Above, for any $\mathcal{V}^*$, there has to be a simulator $\mathcal{S}$ that simulates $\mathcal{V}^*$'s view of the execution. Instead, let us require that there is a simulator

that can simulate $\mathcal{V}^*$'s view for any $\mathcal{V}^*$. An obvious requirement is that $\mathcal{S}$ must be given oracle access to $\mathcal{V}^*$, because otherwise, how would it possibly know how to simulate $\mathcal{V}^*$'s view of the action? Here is the definition:

Definition 3 *$\langle \mathcal{P}, \mathcal{V} \rangle$ is* perfect black box zero knowledge *for the language $L \in \mathsf{NP}$ if there exists an expected PPT $\mathcal{S}$ st $\forall$ PPT $\mathcal{V}$, $x \in L$, $r \in \{0,1\}^*$, $w \in R_L(x)$, and all $z \in \{0,1\}^*$:*

$$\{\mathsf{view}_{\mathcal{V}^*}[\mathcal{P}(x,w) \leftrightarrow \mathcal{V}_r^*(x,z)]\} \approx \left\{\mathcal{S}^{\mathcal{V}_r^*(x,z)}(x,z), r\right\}$$

where r represents the randomness for $\mathcal{V}^*$ during the current execution.

For non-interactive proofs, this definition is equivalent to the definition of perfect zero knowledge. However, for interactive proofs, this definition is strictly stronger. It should be clear why this definition is at least as strong as perfect zero knowledge. In the perfect zero knowledge definition, $\mathcal{S}$ depended on $\mathcal{V}^*$, so it can easily simulate $\mathcal{V}^*$, so giving $\mathcal{S}$ oracle access to $\mathcal{V}_r^*(x,z)$ is okay. As for why this definition is strictly stronger, that will possibly appear as a homework problem...

3 Graph Isomorphism Problem

The graph isomorphism interactive proof presented in the last class is honest verifier zero knowledge. The simulator in that case is:

$$\begin{aligned}
&c \leftarrow \{1,2\} \\
&\pi \leftarrow S_n \\
&H \leftarrow \pi(G_c) \\
&\sigma \leftarrow \pi^{-1} \\
&\text{Output } \left[(H, \pi^{-1}), c\right]
\end{aligned}$$

Note that c above is the same c that $\mathcal{V}$ picks. Since $\mathcal{V}$ is honest, $\mathcal{S}$ knows what c it is going to pick.

However, this simulator does not work in the perfect zero knowledge case because $\mathcal{V}^*$ can lie about which c it is going to pick. So to show that this graph isomorphism protocol is perfect zero knowledge, we must construct a different simulator. We will actually prove that it is perfect black box zero knowledge. Consider the following simulator:

$$\begin{aligned}
&c \leftarrow \{1,2\} \\
&\pi \leftarrow S_n \\
&H \leftarrow \pi(G_c) \\
&c' \leftarrow \mathcal{V}_r^*(x,z,H) \\
&\text{if } c = c' \text{ then output } (H, \pi^{-1}) \text{ else repeat}
\end{aligned}$$

We need to show that the expected runtime of $\mathcal{S}$ is polynomial, and that the output distribution is correct (ie, indistinguishable from the view of $\mathcal{V}^*$). Both of these follow from the following lemma:

Lemma 1 $\Pr[c = c'] = 1/2$

Proof. If $G_1 \approx G_2$, then $\{\pi \leftarrow S_n : \pi(G_1)\} \approx \{\pi \leftarrow S_n : \pi(G_2)\}$. Therefore, although H is computed using c, the distribution of H is the same regardless of c, so H is actually independent from c since π is randomly chosen. Hence, c' is independent from c, from which the desired result follows. ■

From that lemma, we easily get that the expected runtime of $\mathcal{S}$ is polynomial since the expected number of iterations is 2.

For the second claim (that the output distribution is correct), the simulator generates the correct output distribution, but then half of the elements are randomly rejected. The professor gave the analogy that it's like you write a correct distribution on the board, and then ask a blind man to randomly cross out half of them. Since he couldn't possibly have crossed them out based on their characteristics, the output distribution is still the same.

4 Zero Knowledge Proofs for NP Problems

We want to show that there exists a zero knowledge for all problems in NP. It is sufficient to find a zero knowledge proof for any NP-complete problem. We will consider the 3-coloring problem.

First, consider the following protocol for the 3-coloring problem set in the real world (as opposed to between computers):

> Let Alice be the prover, and Bob be the verifier. Alice, given a coloring C of the graph G, chooses a random permutation of the colors $\pi \in S_3$, paints the graph according to the new permutation, and covers all the nodes with plastic cups. Bob then walks in the room and is allowed to pick any edge (u, v). Alice then lifts the plastic cups covering u and v, and Bob checks that the colors are not the same. This process is repeated m^2 times (where m is the number of edges in the graph).

To see that this satisfies completeness, it is obvious that if Alice is given a correct 3-coloring C, no matter what edge Bob picks, the corresponding nodes will always be painted with different colors. For soundness, since this is repeated m^2 times, the probability Alice is lying is $\left(1 - \frac{1}{m}\right)^{m^2} < \left(\frac{1}{e}\right)^m$.

Constructing the simulators is very similar to for the graph isomorphism problem. In the honest verifier case, the simulator knows which edge Bob is going to pick, so the simulator just needs to paint the graph such that those two nodes are different. In the perfect black box case, the simulator guesses which edge Bob is going to pick, and picks a coloring with those nodes different. It then simulates what Bob will output with the current coloring, and recolors if necessary. The probability of guessing wrong is $(1 - \frac{1}{m})$, so the expected number of recolorings is m. Therefore, this is a perfect black box zero knowledge proof of the 3-coloring problem.

The only question that remains is if you can run this on machines. The only thing that is not trivial to simulate on a computer is the plastic cups. Basically, the plastic cups ensure that the prover cannot change the colors after the verifier picks an edge, and the verifier cannot see the colors until the prover lifts the plastic cups. There are known schemes for this that won't be discussed in this class.

5 Sudoko

We ended lecture by discussing a zero knowledge proof for Sudoko. The protocol is to place 3 cards face down on each empty square, with the number for that square on each card, and three cards face up on each already filled square, with the corresponding numbers on those cards. Then for each column, take one card from each square, shuffle them, then allow the verifier to check that the cards contain the numbers 1 through 9. Then repeat for all rows and all 3-by-3 squares.

This protocol works if the prover is honest, but not otherwise because there is no quarentee that the 3 cards on each square have the same number.