6.876/18.426: Advanced Cryptography February 26, 2003

Lecture 6: Power and Efficiency of ZK

Scribed by: abhi shelat

1 Introduction

Previously, we proved that 3-coloring is in ZK. This proof (as all proofs in these notes that use encryption of any messages) is contingent on the existence of one way functions. Assumptions aside, what this means is that membership in any NP language can be proven in zero knowledge! In this lecture, we want to illustrate a few interesting points about this benign yet powerful property.

Specifically, we hope to cover the following:

1. The power of this result
2. The efficiency of ZK

2 Power of NP $\subseteq$ ZK

We present one application today which emphatically demonstrates the power of NP $\subseteq$ ZK. In particular, we show that IP $\subseteq$ ZK. Essentially, any theorem provable in any way (ie, IP) can be proven in zero knowledge, which is quite an amazing view of the world (again, this assumes the existence of one-way functions). We shall base this result on an interesting complexity theory result due to Goldwasser and Sipser that states that IP = AM [GS86]. Recall that IP is the set of languages with proofs systems that are complete and sound. The class AM (Arthur-Merlin games) are proof systems with (our standard notion of) completeness and soundness and the additional restriction that all of the verifier's random coins are public. Sometimes this model is referred to as the public coins model.

To elaborate, an AM interaction follows a general pattern. Merlin, (the all powerful prover) communicates messages to Arthur (the verifier). Arthur is only allowed to send random bits back to Merlin, and so on. At the end, Arthur gets to decide whether to accept or reject based on the transcript. One of the first languages that we studied, ISO, is in fact in AM. In our zero-knowledge protocol for ISO, the prover first sends a graph $C = \pi(G)$ which is a permutation of G to the verifier, the verifier flips a coin and sends it to the prover. Depending on the coin, the prover either sends back the permutation π or the permutation τ such that $\tau(H) = C$. On the other hand, our protocol for NISO is in IP since the verifier starts the game by sending a permutation of either G or H to the prover. More importantly, it is important that the prover not see the verifier's random coins for the protocol to be sound.

Intuition would tell us that having hidden coins is useful and so AM should be a smaller set of languages than IP. Although the Goldwasser-Sipser results runs against this, keep in

mind that the number of rounds might grow by a polynomial factor. Using this powerful complexity theoretic result, we can prove the following:

Theorem 1 IP = ZK.

Proof We begin this proof by noting that ZK $\subseteq$ IP by definition. In order to show that IP $\subseteq$ ZK, first consider any language $L \in$ IP. Since AM = IP [GS86], there exists an Arthur-Merlin protocol $(\mathsf{P}_M, \mathsf{V}_A)$ for the language L. By the simplicity of Arthur-Merlin games, this protocol will look something like:

$$
\begin{array}{lcl}
\mathsf{P}_M & \xrightarrow{m_1} & \mathsf{V}_A \\
\mathsf{P}_M & \xleftarrow{r_1} & \mathsf{V}_A \\
\mathsf{P}_M & \xrightarrow{m_2} & \mathsf{V}_A \\
\mathsf{P}_M & \xleftarrow{r_2} & \mathsf{V}_A \\
 & \cdots & \\
\mathsf{P}_M & \xrightarrow{m_j} & \mathsf{V}_A \\
\mathsf{P}_M & \xleftarrow{\text{y or n}} & \mathsf{V}_A
\end{array}
$$

Let us construct a new protocol $(\mathsf{P}_{M'}, \mathsf{V}_{A'})$ in the following way.

1. The new $\mathsf{P}_{M'}$ first generates a probabilistic encryption scheme (G, E, D), and sends E to the verifier.

2. Let m_i be the message that P_M sends in round i after having seen $(m_1, r_1, ..., m_{i-1}, r_{i-1})$ during the earlier rounds of the protocol. The new prover, $\mathsf{P}_{M'}$, computes m_i based on what the original P_M would send to V_A but then sends $E(m_i, \rho_i)$ where ρ_i are the random coins used by the encryption function.

3. Since the $\mathsf{V}_{A'}$ can only send back random coins, the new verifier does exactly the same as V_A. For example, the interaction might appear something like this:

$$
\begin{array}{lcl}
\mathsf{P}_{M'} & \xrightarrow{E, E(m_1, \rho_1)} & \mathsf{V}_{A'} \\
\mathsf{P}_{M'} & \xleftarrow{r_1} & \mathsf{V}_{A'} \\
\mathsf{P}_{M'} & \xrightarrow{E(m_2, \rho_2)} & \mathsf{V}_{A'} \\
\mathsf{P}_{M'} & \xleftarrow{r_2} & \mathsf{V}_{A'} \\
 & \cdots & \\
\mathsf{P}_{M'} & \xrightarrow{E(m_j, \rho_j)} & \mathsf{V}_{A'}
\end{array}
$$

4. This concludes the AM phase of the new protocol. At this point, $\mathsf{V}_{A'}$ should try to verify in the same way that V_A does. However, since the messages m_i are all encrypted, this is impossible. Instead, the verifier now constructs the statement

σ = "If I decrypted all of the encrypted messages that you sent me, the original Arthur verification strategy accepts on the resulting messages."

More formally, consider the language

$$\text{THUMBS-UP} = \{(\alpha_1, r_1, \alpha_2, ..., \alpha_j) \mid \exists\ (m_1, r_1, ..., m_j) \text{ s.t. } \mathsf{V}_A(m_1, r_1, ..., m_j) \text{ accepts, and there exists, for each } i, \text{ a } \rho_i \text{ s.t.} E(m_i, \rho_i) = \alpha_i \text{ for } i = 1...j\}$$

Note that this language THUMBS-UP is an NP-language since the messages m_i and random bits ρ_i used in encryption serve as a short witness of membership. Therefore, by the Cook-Levin theorem, there is a polynomial-time transformation that produces a graph G that is 3-colorable if and only if above statement σ is in THUMBS-UP. Both the verifier and the prover run this transformation and produce such a graph, G.

5. Since NP $\subseteq$ ZK, $\mathsf{P}_{M'}$ gives a zero-knowledge proof that G is three-colorable to $\mathsf{V}_{A'}$. This may take some more rounds of interaction.

In order to show that this new protocol is zero-knowledge, we consider three things. For the first two, notice that our protocol $(\mathsf{P}_{M'}, \mathsf{V}_{A'})$ inherits completeness from the original AM protocol. Soundness is almost inherited directly as well. For the first part of the protocol, there exists at most probability 1/2 that the AM protocol for L admits a set of (m_i, r_i) messages which convince V_A to accept. Therefore, there is at most a 1/2 chance that the resulting transcript belongs to THUMBS-UP, and therefore, the final 3-colorable is colorable with at most probability 1/2. Using the protocol for 3-coloring from last lecture, this means that overall soundness is at most 1/2.

To show that the protocol is zero knowledge, consider the following simulator:

1. S generates a probabilistic encryption function (G, E, D).
2. S initializes the verifier's random tape with random bits and begins the protocol.
3. At each stage in the AM part of the protocol, S sends $E(0, \rho_i)$ to the verifier where ρ_i are random coins.
4. At the end of the AM phase, the simulator constructs the resulting graph from the THUMBS-UP reduction, and then proceeds to fake a proof of colorability to the verifier using the simulator from the 3-coloring protocol.

Notice that S runs in polynomial time. However, in this case, the simulator is attempting to prove a patently false theorem. In our proof for 3-colorability, the simulator generated indistinguishable transcripts for theorems that were true. Is the transcript that our simulator generates still indistinguishable from the transcripts of $\mathsf{P}_{M'}$?

Query: When can a simulator fake the proof of a false statement?

Answer: If a set of false statements is indistinguishable (to a polynomial time machine) from a set of true statements, then running these false statements through a polynomial time simulator produces transcripts that are still indistinguishable from transcripts generated by the real prover on statements from the true set. If this were not true, then the simulator

would in fact be a distinguisher of the input sets, contradicting the assumption that they were indistinguishable.

In our case, if we use a probabilistic encryption scheme that the professor has endorsed, then by definition no PPT machine can distinguish encryptions of zero, $E(0, \rho)$, from encryptions of messages $E(m_i, \rho')$. Therefore, the AM phase of the transcript is indistinguishable from the real prover's transcript. By the same token, the sets of graphs produced from the reduction are indistinguishable, and the transcripts produced by the 3-col simulator are indistinguishable.

The formal way to prove this is to assume, by contradiction, that there is a distinguisher $\Delta(\cdot, \cdot)$ for the transcripts from our simulator and $\mathsf{P}_{M'}$ and then use this distinguisher to break the encryption scheme used in the first step by distinguishing encryptions of two message values.

■

This technique of encrypting the messages is a very powerful one that we will see in more examples.

There are a few points to consider about this proof. The first comment is that we need to be careful about our definitions for completeness. Previously, we have agreed that our definition for completeness tolerates a negligible probability of rejecting strings that are in a language. That is, a protocol need only accept legitimate words with probability $1 - \nu(k)$. However, if we stick with this leeway in our definition for AM games, then our proof above has a slight problem.

In our proof, the simulator mimics the prover by sending $E(0)$ as during each round, and then fakes the 3-coloring proof of the resulting graph. However, if completeness does not have probability 1, then there might be certain unfortunate random choices that lead to failure even for strings in the language L. Suppose a malicious verifier knew of specific random choices that cause the original AM protocol to fail on a string in L. This malicious verifier can use these values to distinguish between a simulator and the prover by sending these unlucky coins. Our simulator would blissfully send its garbage $E(0)$ and then unknowingly output a transcript that convinces the verifier of the theorem. The original prover, however, would be unable to convince the verifier on these bad coins, and so there is a clear way to distinguish the two.

The easiest way to get around this problem is to assume that the completeness for the AM (IP) language has probability 1. On the other hand, we have also quoted a result that states that an IP protocol which has completeness $1 - \nu(k)$ can be transformed to have completeness 1.

The other point is to note that since we are using encryption (as commitments in this particular case), all of our results are computational zero knowledge. Can we really have perfect zero knowledge and completeness which holds with probability one?

3 Efficiency

In this section, we consider the round efficiency of zero knowledge proofs. The round efficiency in general trumps the bit complexity or computing time required for the proofs. Consider the zero-knowledge protocol for 3-coloring. Recall that in one complete interaction

of the protocol, there is a $1 - 1/|E|$ probability that the prover can cheat the verifier where E is the set of edges in the graph. It is a simple matter of repeating this proof $|E|$ times in order to get a probability of cheating to be less than $(1 - 1\|E|)^{|E|} < 1/2$, and therefore the number of rounds is really quite large. As an alternative, consider the following protocol for Hamiltonian path first presented by Blum. This protocol achieves soundness of $1/2$ directly in three rounds.

Consider a graph $G = (V, E)$ and recall that a Hamiltonian path is a path through G that touches every node exactly once. In this protocol,

1. The prover first selects a random permutation π and permutes the original graph $\pi(G) = H$.

2. The prover then generates a probabilistic encryption scheme (G, E, D) and sends an encryption of the permutation and an encryption of the edges to the verifier. Let $\rho_{u,v}$ be the random coins used to encrypt the edge $(u, v) \in E$.

$$\mathsf{P} \overset{E(\pi,\rho),\ \{E((u,v),\rho_{u,v})\ \forall\ (u,v)\in E\}}{\longrightarrow} \mathsf{V}$$

3. The verifier flips a random coin and sends it to the Prover.

$$\mathsf{P} \overset{r}{\longleftarrow} \mathsf{V}$$

4. If the coin is zero, the prover decrypts what was sent in the first message by sending the key and the random coins used in encryption.

$$\text{if } r = 0\ \mathsf{P} \overset{\{D,\ \rho_{u,v}\forall\ (u,v)\in E\}}{\longrightarrow} \mathsf{V}$$

   Essentially, the prover proves that the messages sent in the first round were correct. If the verifier sends a one, then the prover sends the decryptions of the $|V|$ edges of the Hamiltonian path in the permuted graph H.

$$\text{if } r = 1\ \mathsf{P} \overset{\text{Ham path in } H}{\longrightarrow} \mathsf{V}$$

The completeness of the protocol is standard.

For soundness, suppose G does not have a Hamiltonian path. If P acts honestly in step 1, there will be no cycle to present if V sends one. On the other hand, if H has a Hamiltonian path and G does not, then the prover will not be able to respond to the first case properly. By considering these four cases, if G does not have a Hamiltonian path, then the prover is caught with probability $1/2$. Note that the entire protocol is only three rounds.

The zero-knowledge simulator for the protocol follows the standard pattern from the simple isomorphism proof. The simulator can set the verifier's random tape, pick an encryption scheme, and guess whether to generate a new graph H with a known Hamiltonian path, or encrypt the original graph G, and proceed. If the simulator guesses incorrectly, then it can rewind. As before, it will succeed with high probability after a constant number of tries.

3.1 Alternative protocol

As an alternative, consider the following protocol: In the first step, the prover selects a permutation $\pi(G) = H$ and sends H to the verifier. Like before, the verifiers flips a coin and sends it to the prover. If zero, the prover sends π. If one, the prover returns a Hamiltonian cycle in H.

Is this protocol zero knowledge? Intuitively, it seems like no since the verifier learns of a cycle in H. However, proving that a protocol is not zero knowledge is difficult! The problem with our original simulator is that it does not know the Hamiltonian path in G, and so it cannot respond when the verifier sends a one. If it attempts to send an H which has a known cycle of the appropriate length, we must prove that the verifier cannot distinguish this graph from a legitimate permutation of G. In particular, this would require proving that ISO is not in P– a long-standing problem in complexity theory.

But this argument only applies to our original simulator. How can we show that there is no such simulator for this alternative protocol? Again, this question is equivalent to determining whether $ISO \in$ P. If there was such a simulator, then no verifier could distinguish its transcript from the original prover's transcript in polynomial time, and so in particular, no verifier can determine if $H \sim G$ in polynomial time.

3.2 Parallel

Another way to increase round efficiency is to try to run n interactions of a protocol in parallel. That is, if a traditional zero knowledge protocol for language L works as follows:

$$\begin{array}{lcl} \mathsf{P} & \xrightarrow{\alpha} & \mathsf{V} \\ \mathsf{P} & \xleftarrow{\beta} & \mathsf{V} \\ \mathsf{P} & \xrightarrow{\gamma} & \mathsf{V} \end{array}$$

then instead, we do

$$\begin{array}{lcl} \mathsf{P} & \xrightarrow{\alpha_1,\ldots,\alpha_n} & \mathsf{V} \\ \mathsf{P} & \xleftarrow{\beta_1,\ldots,\beta_n} & \mathsf{V} \\ \mathsf{P} & \xrightarrow{\gamma_1,\ldots,\gamma_n} & \mathsf{V} \end{array}$$

Is it a proof system? Is it complete? For every x in the language, there really is a good proof for it, even n of them. Therefore, if the original system was complete, the parallel version is as well. Soundness? The prover has to send all the proofs at once, so this imposes more constraints, which cannot help the prover. The verifier can always simulate the view in isolation. So if the original protocol was sound, the verifier's responses can be based on the isolated view and soundness can be preserved. One might argue that in the final step, the prover gets an advantage in the sense that the prover gets to see all of the β at once, but this is not an advantage. An honest verifier would send independent β_i and so there is no advantage for the prover to see them all at once.

Query: Can we prove in general parallel composition pumps the soundness to 2^{-n}? This is the case with the 3COL protocol, but is it true in general?

However, when trying to prove that the parallel composition is still zero-knowledge, there is a bigger problem. Using the original simulation strategy does not work. The original simulator would require an exponential number of rewinds in order to generate a legitimate transcript. Suppose a verifier V' sends back a hash of the messages received in the first round. How will a simulator correctly respond? The original simulator simply picks at the beginning whether to guess the "cut" or the "choice" when it initializes the verifier's random tape. If it has guessed incorrectly, it rewinds the verifier and guesses again. It should only have to guess a few times before it stumbles on an interaction which works. In the parallel version, the simulator will have to make n guesses after it initializes the verifier. With high probability, there will be one position in which the random β_i sent back from the verifier differs from whether the simulator had chosen the "cut" or the "choice" for that i. Hence, the simulator would need to run exponentially long in order to get a transcript which works in all positions.

This begs a fundamental question of whether zero knowledge should inherently parallelize or not. The professor deeply felt that sequential composition should be part of the definition; but is ambivalent about whether parallel composition is an inherent property.

References

[GS86] S. Goldwasser, M. Sipser. Private coins versus public coins in interactive proof systems. Proceedings of the eighteenth annual ACM symposium on Theory of computing. pp. 59-68, 1986.