Computational Complexity: Lecture 27.

Agenda: - Zero knowledge for IP.
- Zero knowledge proofs of knowledge.

Recap: - Zero knowledge interactive proofs.

Defn: L has a ZK IP if there is a protocol $V \leftrightarrow P$ s.t.:

[Completeness] $x \in L \Rightarrow \Pr[V \leftrightarrow P(x) = acc.] \geq 2/3$

[Soundness] $x \notin L \Rightarrow \forall P^*: \Pr[V \leftrightarrow P^*(x) = acc] \leq 1/3$.

[Zero knowledge] $\forall V^*$, there is a ZPP simulator S^* s.t. $\{S^*(x)\} \equiv \{View_{V^*}(V^* \leftrightarrow P(x))\}$. $x \in L$.

(perfect ZK, statistical ZK, computational ZK).

Examples:

▷ GNI: $\{(G_1, G_2) : G_1 \not\cong G_2\}$.

Ver: Pick $b \in_R \{1,2\}$
Pick $\sigma \in_R S_n$
$H := \sigma(G_b)$ $\xrightarrow{\quad H \quad}$ Prover.

Acc if $c = b$. $\xleftarrow{\quad c \quad}$

Simulator for V^*:

- Pick b, σ acc to V^*.
- Return view : $\left(b, \sigma, \begin{array}{c} \xrightarrow{H} \\ \xleftarrow{b} \end{array} \right)$

- GI: $\{(G_1, G_2) : G_1 \cong G_2\}$.

Prover: $(\tau : G_1 \to G_2)$

Picks $\sigma \in_R S_n$.

$H = \sigma(G_2)$

Verifier

Picks $c \in_R \{1,2\}$ $\xleftarrow{\quad H \quad}$

$\xrightarrow{\quad c \quad}$

If $c = 2$, returns $\pi = \sigma$

Else, returns $\pi = \sigma\tau$

Acc if $\xleftarrow{\quad \pi \quad}$

$\pi(G_c) = H$.

Simulator for V^*:

- Pick $b \in_R \{1, 2\}$. Pick $\sigma \in_R S_n$.
- $H = \sigma(G_b)$
- Simulate V^* by pretending to send H to V^*.
  V^* will send c.
  If $c = b$, return view: $\begin{array}{c} \xleftarrow{H} \\ \xrightarrow{c} \\ \xleftarrow{\sigma} \end{array}$
  Else: try again.

Thm [Goldreich-Micali-Wigderson] CZK for all of NP.

Obs: The honest prover is only a BPP with witness.

Sketch for 3-colouring:

Prover knows a $\sigma : [n] \to \{1,2,3\}$.

- Prover picks $\pi \in_R S_3$.
- $P \xrightarrow[\text{commitment to } \pi\sigma]{B_1, \dots, B_n} V$

Don't respond if $(i,j) \notin E$.

$\xleftarrow{(i,j)}$

$\xrightarrow{key_i, \; key_j}$

Repeat $10n^2$ times.

$\square$.

Commitment: $\text{Commit}(k, \text{value}) = c$.

- (Perfectly binding) $\text{Commit}(k_1, v_1) = \text{Commit}(k_2, v_2) \Rightarrow v_1 = v_2$.

  (No adversary can later "change" the value after commiting to it).

- (Computationally hiding).

  $$\{\text{Commit}(\cdot, x)\} \underset{c.i}{\equiv} \{\text{Commit}(\cdot, y)\}.$$

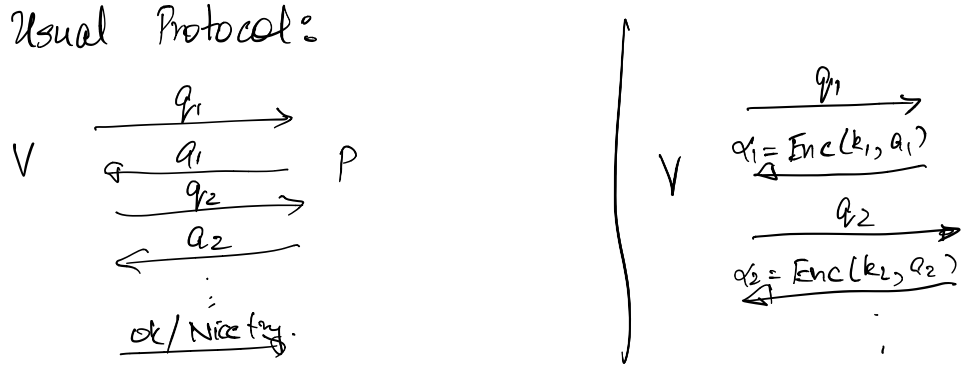
  (before revealing the key, no p.p.t machine should be able to "guess" value from commitment).

Qn: Can we make any IP into a zero knowledge IP protocol?

[BenOr - Goldreich - Goldwasser - Håstad - Kilian - Micali - Rogaway]

Yes!

Usual Protocol:



$Ver(q_1\, a_1\, q_2\, a_2 \ldots\ldots)$

Two obvious issues:

1) How will verifier know what to ask in round 2 if verifier doesn't know what the prover sent in round 1?

IP = AM[poly]. Arthur doesn't need to understand prev messages to send a random string.

2) How will the verifier know whether or not to accept?

G.

Ask Prover for keys.

" $\exists k_1, .., k_m$ s.t $Ver(q_1, a_1, .., q_m, a_m)$
where $a_i = Dec(k_i, \alpha_i)$ "

This is in NP! [GMW].

Full proof: Given a protocol $\Pi : V \Leftrightarrow P$ for L.
Assume WLOG:

- This is an AM[poly] protocol.
  Verifier only sends public, random coins.
  Acceptance is deterministic.
- Prover is also deterministic.
- Both verifier & prover send length ℓ messages each round.
- Perfect completeness.

Completeness & soundness is straightforward.

Simulator for V^*:

- Run V^* and get q_1.
- Send $\alpha_1 = \mathrm{Enc}_{k_1}(0^\ell)$ to V^*. Get q_2 from V^*
- Send $\alpha_2 = \mathrm{Enc}_{k_2}(0^\ell)$ to V^*. Get q_3 from V^*.

  $\vdots$

- Build the SAT instance. $\exists k_1, \ldots, k_m : V^*(q_1, \ldots, q_m, a_1, \ldots, a_m)$ with $a_i = \mathrm{Dec}_{k_i}(\alpha_i)$.
- Reduce the instance to 3-COL to get a graph G.
- Run the ZK simulator for G.

(wave hands) $\square$

Proofs of knowledge:

Can a protocol somehow convince a verifier that the prover must <u>know</u> something? Or <u>have</u> a witness?

" The only way this prover can convince me w.h.p is if the prover <u>knows</u> a witness."

Example: GI

Verifier		Prover: $(\tau : G_1 \to G_2)$
	$\xleftarrow{\quad H \quad}$	$\sigma \in_R S_n$ $\quad H = \sigma(G_2)$
$b \in_R \{1,2\}$	$\xrightarrow{\quad b \quad}$	If $b=1$ $\quad \pi = \sigma \cdot \tau$; $b=2$ $\quad \pi = \sigma$
	$\xleftarrow{\quad \pi \quad}$
Check if $H = \pi(G_b)$	$\xrightarrow{\text{Ok / Nice try.}}$

Suppose P^* makes verifier accept w.p $\geq 3/4$ does it mean P^* "knows" an isomorphism?

[Bellare-Goldreich] : " You should be able to "extract" a witness from such a prover".

$\Downarrow$

knowledge extractor.

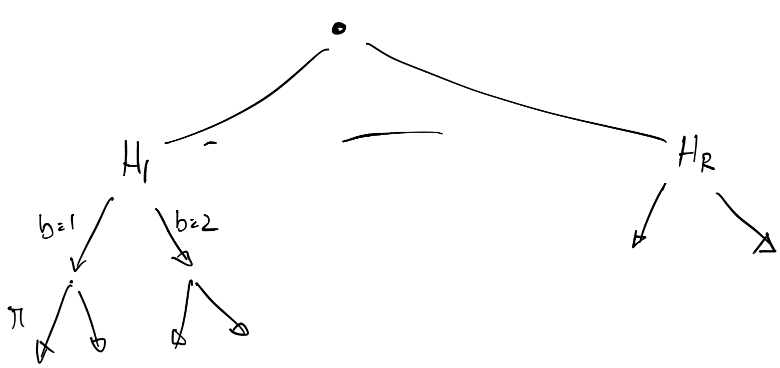
Knowledge extractor : A machine M is a "knowledge extractor".
if
- M is a randomised expected poly time oracle machine. with "rewindable BB access" to P^*,
- If P^* makes verifier accept w.p >> threshold. then M given access to P^* can produce w.

Lemma: There is a knowledge extractor for GI. with threshold 1/2.

Pf: M: Run P^* to get the first message H.
"Save the state"
Send b=1 and get π_1 from P^*.
Rewind to saved state
Send b=2 and get π_2 from P^*
Check if $\pi_2^{-1}\pi_1(G_1)=G_2$. If yes, return. Else, retry.

P^* succ w.p $\geq 3/4$

Why does this run in expected poly time? How many repeats?



H is "good" if $\Pr\left[\begin{array}{c}P^* \text{ makes ver. acc} \\ \text{after saying } H\end{array}\right] \geq 2/3$

Claim: $\Pr[P^*\text{'s first message } H \text{ is "good"}] \geq 1/4$.

Pf:

good	bad
δ	$1-\delta$
≤ 1	$\leq 2/3$

$$\delta \cdot 1 + (1-\delta)\cdot\frac{2}{3} = \frac{\delta}{3} + \frac{2}{3} \geq \frac{3}{4}$$

$$\Rightarrow \delta \geq 1/4. \qquad \square$$

$p = \Pr[\text{Ver acc. after } (H, 1)]$ $\quad q = \Pr[\text{Ver. acc after } (H, 2)]$

$$\frac{p+q}{2} \geq \frac{2}{3} \Rightarrow p+q \geq \frac{4}{3}$$

$$\Rightarrow p, q \geq 1/3.$$

$\Pr[\text{Ver acc after } (H,1) \ \& \ (H,2)] \geq 1/9$ $\quad \square$.

$\Rightarrow$ M breaks out w.p $\geq 1/36$.

What about all of NP?

Blum's protocol for Hamiltonicity:

Prover: (knows a Hamilton cycle in G).

Picks $\sigma \in_R S_n$. $H = \sigma(G)$. Puts all n^2 bits in locked boxes and sends these to verifier.

Verifier: With prob $1/2$:

- Show me how the locked boxes contain a shuffling of G.

With prob $1/2$:

- Reveal the length n path.

Further reading:- "On Σ-protocols" by Ivan Damgård.

- Non-interactive zero knowledge