Lecture 9: IP and ZK

Today we enter third "unit" of the class
- Unit 1 (Lectures 1-4): Foundations
- Unit 2 (Lectures 5-8): Cryptanalysis & ECC
- Unit 3 (Lectures 9-13): Zero-Knowledge

<u>Outline</u>:
- Proofs
- Interactive Proofs
- Zero Knowledge
- ZKP for Hamiltonian Cycles

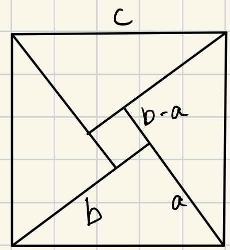
Q: What is a proof?

→ It demonstrates truth... or

prover → SOMEONE convinces

verifier → SOMEONE ELSE that

statement → SOMETHING is true



$$4\left(\tfrac{1}{2}\right)ba + (b-a)^2 = c^2$$
$$a^2 + b^2 = c^2$$

A convention: statements as membership

"~~x is prime~~"

$$\underbrace{x \in L_{primes}}_{\text{statement}}$$

set of primes

instance

language

Formally (from complexity theory):

A language is a set of strings $L \subseteq \{0,1\}^*$
A statement takes the form $x \in L$

Examples:

- "15 is biprime" → $15 \in \{pq : p, q \in L_{prime}\}$
- "ϕ is satisfiable" → $\phi \in \{\text{formula } \phi \mid \exists x \ \phi(x) = 1\}$
- "ϕ is unsatisfiable" → $\phi \in \{\text{formula } \phi \mid \forall x \ \phi(x) = 0\}$
- "G has a Hamiltonian cycle" → $G \in \{\text{all graphs with Ham. cycle}\}$
- "ϕ_{PYTHAG} is true" → ...

Not all statements are equally hard to prove!

Non-Interactive Proofs

Classically, a proof is simply read (non-interactive)

Fix a language L. A prover P wants to create a proof π to convince a verifier that $x \in L$

unbounded $\rightarrow$ $P(x) \xrightarrow{\quad \pi \quad} V(x)$ $\leftarrow$ polytime

Verify$(x, \pi) \rightarrow \{0,1\}$

deterministic

Key Properties:
- Completeness: $\forall x \in L$, Verify$(x, \pi) = 1$
- Soundness: $\forall x \notin L$, Verify$(x, \pi) = 0$

If we have completeness and soundness $\Rightarrow L \in NP$
(π is an NP witness)

NP Complexity Class:

↳ Informally, a language $L \in NP$ if statement $x \in L$ can be proven with a non-interactive proof

↳ Formally, there exists an efficient algorithm $M(\cdot,\cdot)$ s.t. $x \in L \longleftrightarrow \exists w \in \{0,1\}^{poly(|x|)}$ s.t. $M(x,w) = 1$

Ex: To prove $\phi \in SAT$, P sends satisfying assignment to V.

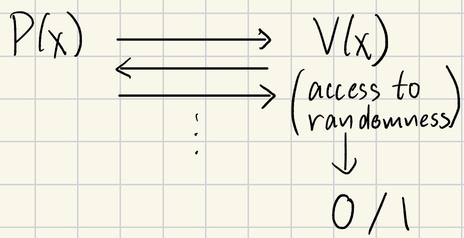
More generally, P sends w and V checks $M(x,w) = 1$

How can we prove more?

By changing the model!

↳ in a court, people ask questions!

Interactive Proofs [Goldwasser, Micali]



Interactive TMs:
- Can be modeled as a TM that implements a next message fcn:

$\text{next}(i, \text{msg}_i, \text{state}_i) \to (\text{msg}_i, \text{state}_i)$

round # — msg recv — prev state — msg sent — next state

initial state: $\text{next}(0, x, 0) \to (-, \text{init st})$

- P, V are randomized, interactive Turing machines
  - ↳ # rounds, message length, V time : poly
  - ↳ P : unbounded

Key Properties:

denotes output of V when P, V interact given x

- Completeness: $\forall x \in L$, $\Pr[\underbrace{\langle P, V\rangle(x)} = 1] \geq 2/3$ — can amplify to 1-negl w/ repetition
- Soundness: $\forall x \notin L$, $\forall P^*$, $\Pr[\langle P^*, V\rangle(x) = 1] \leq 1/3$ — can amplify to negl w/ repetition

P is honest prover, and soundness should hold for <u>any</u> malicious prover, not just the honest one

What do we get from interaction?

1. IP captures much broader class of problems than NP. In fact, IP = PSPACE!
2. Even for NP statements, interaction can allow proving a statement with communication $< |w|$
3. Interaction enables a surprising new property: ZERO-KNOWLEDGE...

Zero Knowledge

Conceptually, a proof that shows $x \in L$ and reveals nothing else

Examples:
- Given ϕ, prove that $\phi \in SAT$ without revealing the satisfying assignment
- Prove x is the correct output of some algorithm without revealing my secret inputs to the algorithm

How do we define "reveals nothing else" $\rightarrow$ how do we define knowledge?
- Say you have $N = pq$ and also the factor p. Do you know q? Yes b/c you can calculate q efficiently!
- Say you have an encryption of x $Enc_{pk}(x)$. Do you know x? Intuitively no b/c you can't efficiently recover x from $Enc_{pk}(x)$

$\Rightarrow$ KNOWLEDGE is what you can compute efficiently

Intuition: if any info a dishonest verifier can derive from the protocol transcript could have been efficiently derived from x, the protocol is zero-knowledge!

Zero Knowledge: (P, V) is ZK is $\forall$ PPT V^*, $\exists$ PPT Sim, $\forall x \in L$,

$$\{View_{V^*}[\langle P, V^* \rangle(x)]\} =_* \{Sim(x)\}$$

$*$ computational, statistical, or perfect

Needs to be true for any (potentially malicious) verifier V^*, not just honest verifier V. If we write definition with V instead of V^*, it is called "honest verifier ZK" or HVZK

- $View_{V^*}[\langle P, V\rangle(x)]$ is what V^* sees when interacting with P
- $Sim(x)$ is the algorithm that writes down the transcript without interacting with P

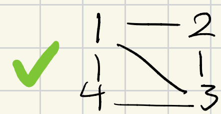
* Remember: input to $Sim(\cdot)$ essentially captures what the (P, V) interaction leaks because that's the information the verifier is allowed to use when writing down the transcript

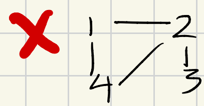
How do we achieve ZK? What languages have ZK proofs?

↳ today, we will prove that there is a ZK proof protocol for every language in NP!

Approach: give a ZK protocol for one NP-complete problem (HAMCYCLE), then a ZK protocol for any other NP language is just to reduce the instance to this language and use the same protocol

ZKP for Hamiltonian Cycle





Def: A Hamiltonian Cycle visits every node in a graph exactly once

Let HAM be the set (language) of graphs w/ a Hamiltonian Cycle

Trivial IP for HAM:

$$\begin{array}{ccc} P(G) & & V(G) \\ \text{finds a cycle } c \in G & \xrightarrow{\quad c \quad} & \text{checks } c \to 0/1 \end{array}$$

1) Complete? Yes!
2) Sound? Yes!
3) ZK? Probably not...
↳ if P=NP, then V can compute the edges on the cycle itself, so this would be ZK!

A ZKP for HAM

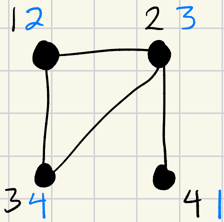
First, a sketch:

$P(G)$		$V(G)$
$c \leftarrow \text{FindCycle}(G)$		
$\sigma \xleftarrow{\$}$ permutation on vertices		
Commit to σ		
Commit to $\sigma(G)$		
	$\xrightarrow{\text{commitments}}$	
	$\xleftarrow{\ b\ }$	$b \xleftarrow{\$} \{0,1\}$
	if $b=0$, open $\sigma, \sigma(G)$ $\longrightarrow$	// shows σ (not c)
	if $b=1$, open subset of $\sigma(G)$ that is $\sigma(c)$	// shows $\sigma(c)$ (not σ)

Now, in detail:

Let Commit be computationally hiding and binding
Let G have n vertices $[n] = \{1, 2, \dots n\}$ and an adjacency matrix $M \in \{0,1\}^{n \times n}$ $\quad G \triangleq (n, M)$
For a permutation σ on $[n]$, $\sigma(M)$ is M' where $\forall i,j,\ M'_{\sigma(i),\sigma(j)} = M_{ij}$

$$M = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 1 & 1 & 0 \\ 2 & 1 & 0 & 1 & 1 \\ 3 & 1 & 1 & 0 & 0 \\ 4 & 0 & 1 & 0 & 0 \end{array}$$



i	1	2	3	4
$\sigma(i)$	2	3	4	1

$$M' = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 0 & 1 & 0 \\ 2 & 0 & 0 & 1 & 1 \\ 3 & 1 & 1 & 0 & 1 \\ 4 & 0 & 1 & 1 & 0 \end{array}$$

A cycle ℓ is a list of $n+1$ vertices s.t.
- $\forall i \in [n] \quad M_{\ell_i, \ell_{i+1}} = 1$ and $\ell_{n+1} = \ell_1$
- $|\{\ell_i : i \in [n]\}| = n$ (Hamiltonian)

$P((n, M), c)$ $\qquad\qquad$ $V((n, M))$

$\sigma \xleftarrow{\$} \text{Perms}[[n]] \quad M' \leftarrow \sigma(M)$

$\forall i,j \in [n],\ r_{ij} \xleftarrow{\$} R$

$\quad c_{ij} \leftarrow \text{Commit}(M'_{ij}, r_{ij})$

$\forall i \in [n],\ s_i \xleftarrow{\$} R$

$\quad d_i \leftarrow \text{Commit}(\sigma(i), s_i)$

$\xrightarrow{\text{all } c_{ij}, d_i}$ $\qquad b \xleftarrow{\$} \{0,1\}$

$\xleftarrow{b}$

if $b = 0$: $\quad \xrightarrow{\sigma,\ \text{all } r_{ij}, s_i}$

σ is a permutation?

$d_i \stackrel{?}{=} \text{Commit}(\sigma(i), s_i)$

$M' \leftarrow \sigma(M)$

$c_{ij} \stackrel{?}{=} \text{Commit}(M'_{ij}, r_{ij})$

if $b = 1$:

$\ell' \leftarrow [\sigma(i) : i \in \ell]$ $\quad \xrightarrow{\ell',\ \forall i \in [n],\ r_{\ell'_i, \ell'_{i+1}} \text{ and } M'_{\ell'_i \ell'_{i+1}}}$

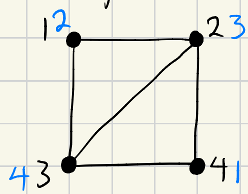
$c_{\ell'_i \ell'_{i+1}} \stackrel{?}{=} \text{Commit}(M'_{\ell'_i \ell'_{i+1}}, r_{\ell'_i \ell'_{i+1}})$

ℓ' is a cycle?

An example run of the protocol:

$n = 4$

$$M = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$



$\ell = [1, 2, 4, 3, 1]$

Commit

$\sigma = \{1 \to 2, 2 \to 3, 3 \to 4, 4 \to 1\}$ + $[s_1, s_2, s_3, s_4] \xleftarrow{\$} R^4$

$$M' = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix} \quad \begin{matrix}\text{Commit} \\ + \end{matrix} \quad \begin{bmatrix} r_{11} & r_{12} & r_{13} & r_{14} \\ r_{21} & r_{22} & r_{23} & r_{24} \\ r_{31} & r_{32} & r_{33} & r_{34} \\ r_{41} & r_{42} & r_{43} & r_{44} \end{bmatrix} \xleftarrow{\$} R^{4\times 4}$$

reveal when $b=0$

$\ell' = [2, 3, 1, 4, 2]$

reveal when $b=1$

Complete: ✓

Sound: If $M \notin HAM$, $\forall \sigma$, $\sigma(M) \notin HAM$.

So, if P commits to $\sigma(M)$:

if $b=1$ (prob 50%) $\Rightarrow$ V rejects

If P commits to something else

if $b=0$ (prob 50%) $\Rightarrow$ V rejects

(or, a binding break)

soundness: $\geq 1/2$ - binding error

To prove ZK, we must define Sim:

let $$\tilde{M} = \begin{bmatrix} 0 & 1 & & & & \\ & 0 & 1 & & & \\ & & 0 & 1 & & \\ & & & \ddots & \ddots & \\ & & & & 0 & 1 \\ 1 & & & & & 0 \end{bmatrix}$$

cycle graph

Sim(n, M):

$b \xleftarrow{\$} \{0, 1\}$

$\sigma \xleftarrow{\$} \text{Perms}[[n]]$

if $b = 0$:

$M' \leftarrow \sigma(M)$

if $b = 1$:

$M' \leftarrow \sigma(\tilde{M})$, $\ell = [1, 2, \ldots, n, 1]$, $\ell' \leftarrow \sigma(\ell)$

// commit as in protocol

$b' \leftarrow V^*(M, \text{commits})$; if $b \neq b'$, restart (for dishonest V^*)

// open as in protocol

if $b=0$, ℓ / ℓ' are not needed

if $b=1$, relationship $M' = \sigma(M)$ isn't checked

Output transcript

Note: this is only HVZK (since $b \xleftarrow{\$} \{0,1\}$)

- a malicious V^* might bias b (so add b')
  - ↳ if b' and b are independent:
    $\Pr[\text{restart}] = 50\% \Rightarrow \lambda$ reps for a $2^{-\lambda}$ failure rate
  - ↳ if not:
    b' is correlated w/ b is correlated w/ the msgs
    $\Rightarrow$ attack on commitment hiding!

Now, we must show

$$\{\text{View}_{V^*}[\langle P, V^* \rangle(G)]\} =_c \{\text{Sim}(G)\}$$

$$\{(G, c_{ij}, d_i, b, \text{openings})\}$$

- all of these are distributed exactly as in the real protocol
- some of these are opened: they're also distributed exactly as in the real protocol
- the rest are un-opened: if they can be distinguished, we have an attack on hiding.
  (hybrid argument over all unopened commitments)