Lecture 4: Zero Knowledge

Henry Corrigan-Gibbs

CS355 – Spring 2019

Plan

April 10, 2019

- Recap: Interactive proofs
- Zero knowledge
    * What it is
    * Why it's useful
    * How we define it
- Example: ZK Proof for HAMCYCLE

Reminders

→ HW 1 due Friday at 5pm via Gradescope
  ↳ Come to OH today!
→ Late day policy

Today

- We will be discussing the most beautiful idea in all of CS. Maybe of all time?
  ↖ Controversial but still true :)
- Zero knowledge – How to prove to you that I know something (e.g. ϕ is SAT) without leaking anything else to you (SAT assignment)
- Amazingly clever, also useful in many crypto protocols.

→ Lesson: Importance of definitions.
  Original ZK paper is important b/c of defn of ZK, not because of the specific constructions.
  ↳ Defn is $> 1/2$ the battle

→ Paper rejected 3 (I think) times before published
  ↳ Lesson?
  Goldwasser, Micali, Rackoff (STOC '85)

Recap: Interactive proofs

On Monday, Florian introduced interactive proofs

Goal of a proof: Convince $\underbrace{\text{someone}}_{\text{"the verifier"}}$ of $\underbrace{\text{something}}_{\text{"statement"}}$

In complexity theory, we consider statements of the form:

$$\text{"} \, x \in \mathcal{L} \, \text{"}$$

instance language

Examples: "N is the product of exactly two primes"

$N \in \{pq \mid \text{primes } p, q\}$

"The Pythagorean Thm is true."

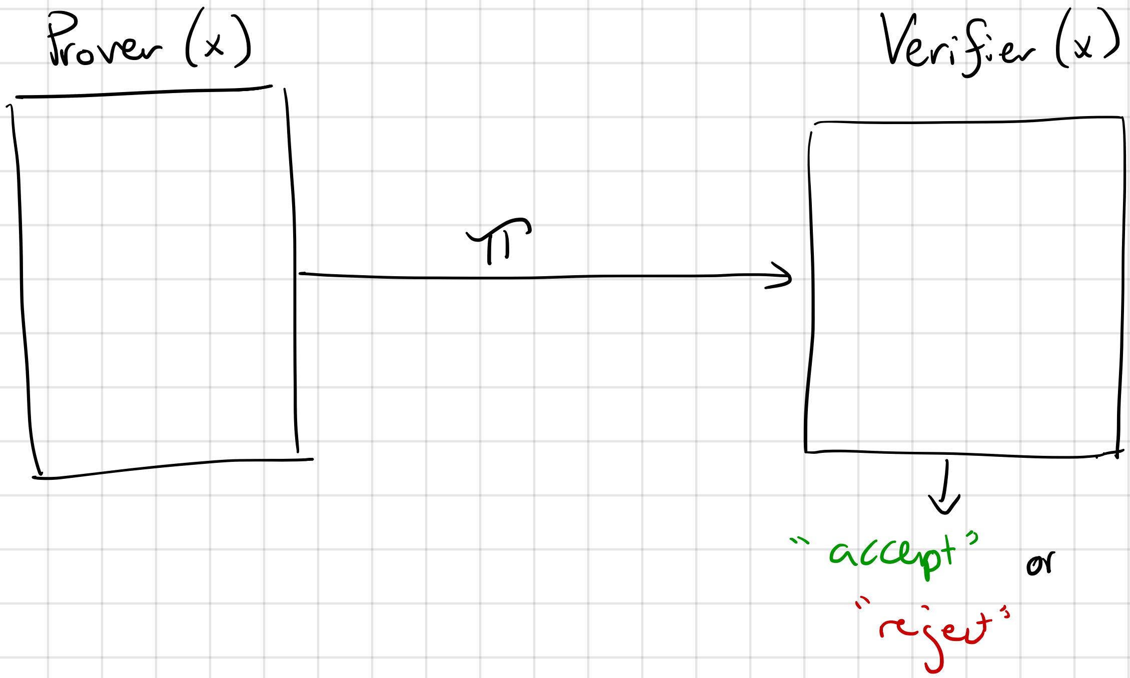
$\text{PYTHM} \in \{\text{true statements in some formal system}\}$

"ϕ is an unsatisfiable SAT formula"

$\phi \in \{\text{set of unsatisfiable SAT instances}\}$"

Recap

Conventional Proof



- π might be hard to find (exponential time?)
- π should be easy to check (polynomial time, deterministic verifier)

Say that we want to prove that $x \in \mathcal{L}$.

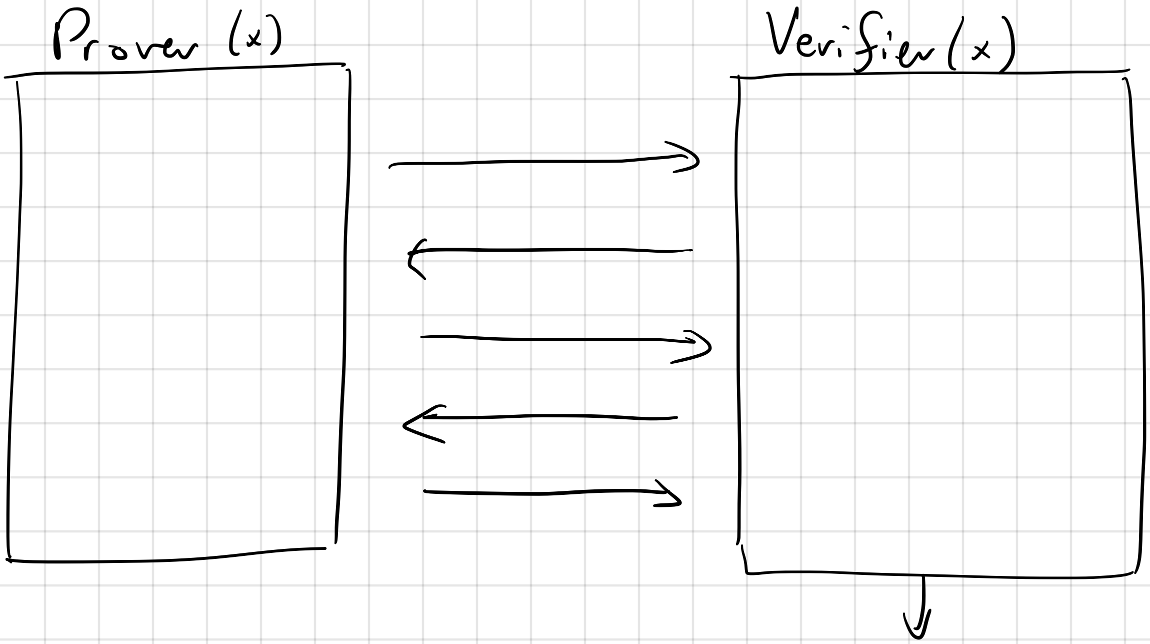
Can do so w/ a conventional proof $\iff$ $\mathcal{L}$ is an NP language

e.g. to prove that ϕ is SAT, P sends satisfying assignment to V.

Blum NP = "nifty proof"

Recap What if we allow P & V to interact?

What if V can use randomness?



"accept" or "reject"

Can increase to 1.

Properties we want

1. Completeness $\forall x \in \mathcal{L}$ $\Pr[\langle P, V\rangle(x) = \text{"accept"}] \geq 2/3$

2. Soundness $\forall x \notin \mathcal{L}$ $\forall P^*$ $\Pr[\langle P^*, V\rangle(x) = \text{"accept"}] \leq 1/3$.

Can reduce to negligible w/ repetition.

Q: Why is interaction useful?

A1: (On monday)
IP captures a larger class of problems.
↳ PSPACE ... prove to you that a graph is NOT 3-COLORABLE!

A2: (Today)
Interactive proofs can have a third surprising property.

Properties we want

3. Zero Knowledge V "learns nothing" from her interaction with P, except that $x \in \mathcal{L}$.

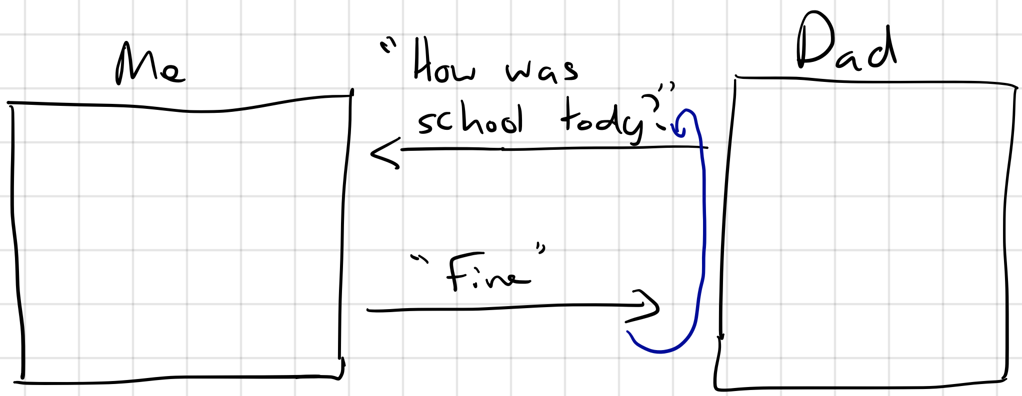
Huh? What does this even mean?

Application: Can prove to you that I executed some protocol correctly without revealing any of my secrets.

Defn of ZK used to define security in many protocols
↳ want to show that "nothing leaks"

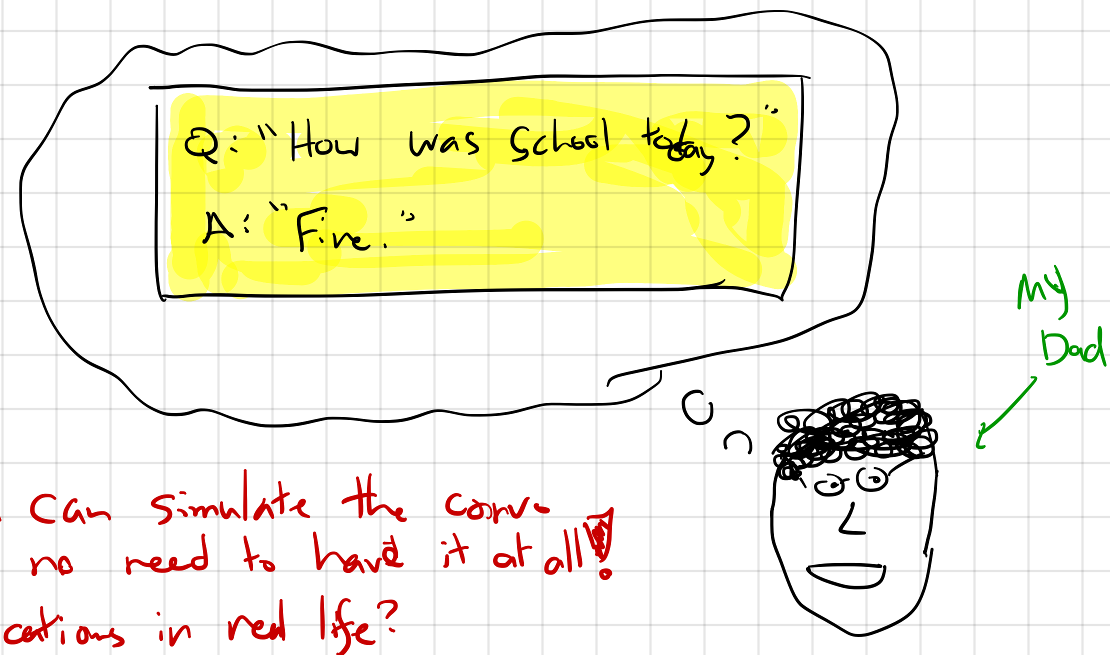
Q: What does it mean to "learn nothing" from an interaction?

Ex. Me in 7th grade



Ex. Military spokesperson.

INTUITION: If V can easily write down a transcript of its interaction with P, then V hasn't learned anything useful from P.



If you can simulate the conv. transcript, no need to have it at all!

⟶ Applications in real life?

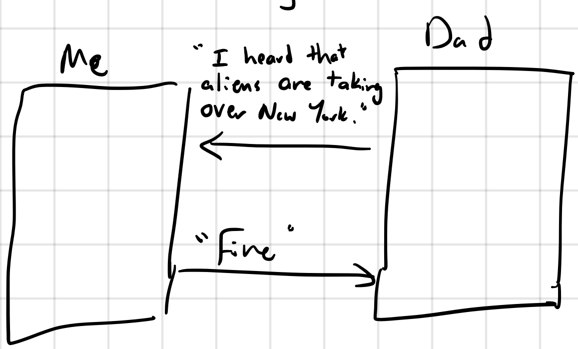
The surprising thing is that there is a very clean way to formalize this intuition

3. Zero Knowledge: $\forall$ efficient V^*, $\exists$ efficient Sim s.t. $\forall x \in \mathcal{L}$

$$\left\{ \mathrm{View}_{V^*}\left[P(x) \leftrightarrow V^*(x) \right] \right\} \approx \left\{ \mathrm{Sim}(x) \right\}$$

There are diff flavors of ZK
- perfect $=$
- statistical $\approx_s$
- computational $\approx_c$

Intuition:
- Whatever V can learn by interacting w/ P it can learn sitting at home by running Sim.
- Holds even if V^* is malicious?



- Key to remember: Input to Sim essentially captures what the (P,V) interaction leaks.

[There is an annoying technical issue that comes up when you want to run a ZK protocol many times.
→ "Auxiliary-input ZK"
See Goldreich §4.3.3]

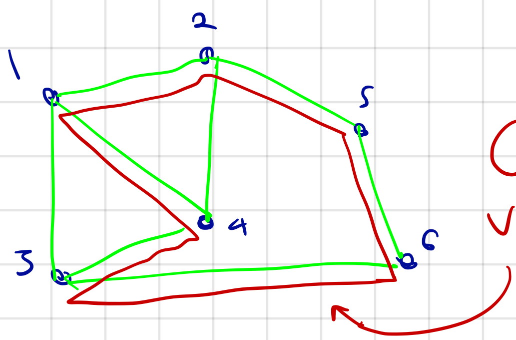
ZK Protocol for Hamiltonian Cycle [Blum '87 (?)]

- HamCycle is an NP-Complete problem
  - ↳ Anything provable (in NP) is provable in ZK — {Goldreich Micali Wigderson '87}
  - ↳ Reduce to HamCycle instance, use this protocol.

In theory, can prove to you that I know an 0day in iOS without revealing it to you! And so on....

Reminder: Defn of Ham Cycle

$G = (V, E)$ undirected graph



Cycle in graph that visits each vertex once

See Knuth (linked from course website) for fun history of this problem.

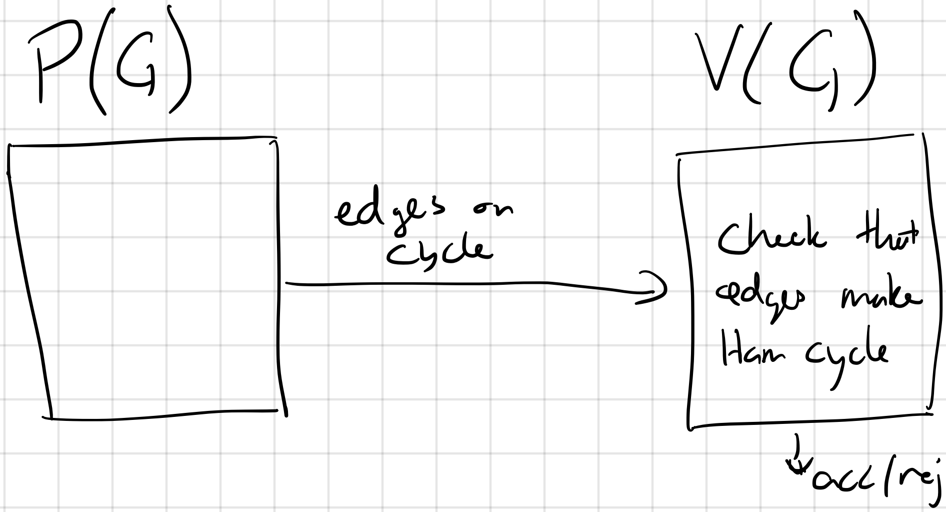
$$\text{HAMCYCLE} = \{ G \mid G \text{ has a Hamiltonian cycle} \}$$

Adjacency Matrix

$$A = \begin{array}{c|cccccc|} & 1 & 2 & 3 & 4 & 5 & 6 \\ \hline 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 2 & 1 & 0 & 0 & 1 & 1 & 0 \\ 3 & 1 & 0 & 0 & 1 & 0 & 1 \\ 4 & 1 & 1 & 1 & 0 & 0 & 0 \\ 5 & 0 & 1 & 0 & 0 & 0 & 1 \\ 6 & 0 & 0 & 1 & 0 & 1 & 0 \\ \hline \end{array}$$

$$A_{i,j} = \begin{cases} 1 & \text{if } (i,j) \in E(G) \\ 0 & \text{o.w.} \end{cases}$$

Trivial Protocol



Not **Zero knowledge**
(under reasonable assumptions...)

ZK Protocol (Blum)

Following Blum, we'll imagine that P can send V "locked boxes," which we implement w/ cryptographic commitments.

Prover (G) | Verifier (G)

* Put each of the n vertices $v_1, \dots, v_n$ into n boxes $B_1, \dots, B_n$ in random order.

* Into box B_{ij}, put $\begin{cases} 1 & \text{if vertices in } B_i \text{ and } B_j \text{ are adjacent in } G \\ 0. & \text{o.w.} \end{cases}$

B_i's = relabeling of vertices

B_{ij}'s = adj. matrix under relabeling

Send the $n + \binom{n}{2}$ boxes $\longrightarrow$

Flip a coin $b \xleftarrow{R} \{0,1\}$

If $b=0$: "Show me G!"

If $b=1$: "Show me the cycle!"

$\longleftarrow$

If $b=0$: Unlock all boxes.

If $b=1$: Unlock only boxes corresponding to Ham Cycle in G.

Keys $\longrightarrow$

Check:

$b=0$ Got a perm of adj. matrix

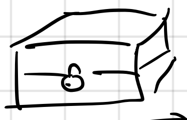
$b=1$: Got a cycle

Accept if so.

Some Comments

Box containing msg m

Some particular type of hash fn.

Imagine:  $= \underline{H(m,r)}$

Key  $= \underline{(m,r)}$

Properties

1. Complete. ✓
2. Sound.
   If $G \notin \text{HamCycle}$, then no matter what P^* puts in boxes, V will reject w.p $\geq 1/2$.
3. Zero knowledge. We construct eff Sim.

   $\text{Sim}(G \in \text{HamCycle})$

   - Guess $\hat{b} \xleftarrow{R} \{0,1\}$.
   - If $\hat{b}=0$, put random perm of Adj mat in Boxes
   - $\hat{b}=1$, put random perm of cycle in Boxes.
   - Run $b \leftarrow V^*(G, \text{Boxes})$
   - If $b \neq \hat{b}$, Abort.
   - Else, open boxes per V^*'s request
   - Output $(G, \text{Boxes}, b, \text{Keys to boxes})$ as transcript.

[N.B. When we replace ideal box w/ a real commitment, we get a protocol that's only computational ZK.]

Life lessons to remember

* If you can efficiently simulate an interaction, you haven't learned anything useful from it.
  ↪ Ideally doesn't apply to this lecture.

* Input to simulator ≈ what leaks.
* Anything that has a traditional (NP) proof also has a zero knowledge proof system.