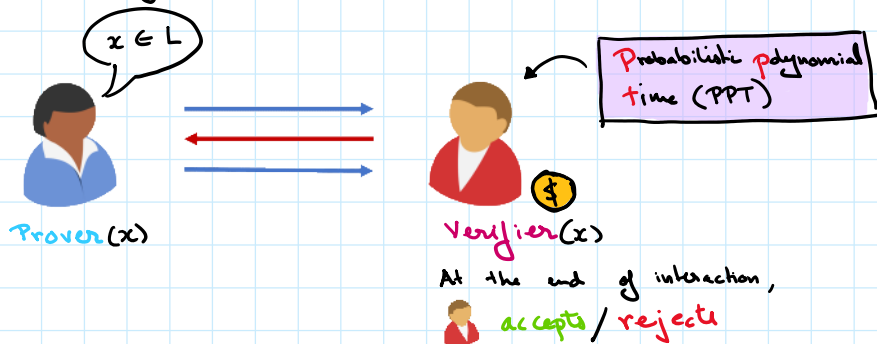


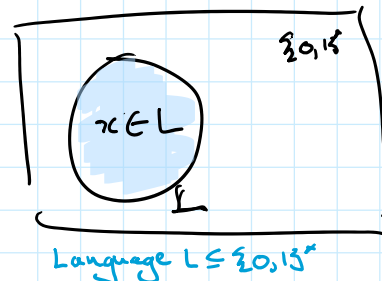
Recap

Interactive Proofs



① Completeness

In the honest execution of the protocol,
 if $x \in L$
 $\Pr[\text{Verifier}(x) \text{ accepts}] = 1$



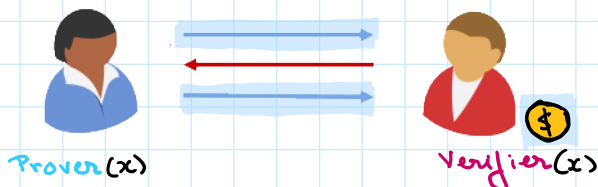
② Soundness

if $x \notin L$, then for every Prover(x)
 $\Pr[\text{Verifier}(x) \text{ accepts}] \leq \text{negl}(|x|)$


→ "vanishes" faster than any $\frac{1}{\text{poly}}$ i.e. small

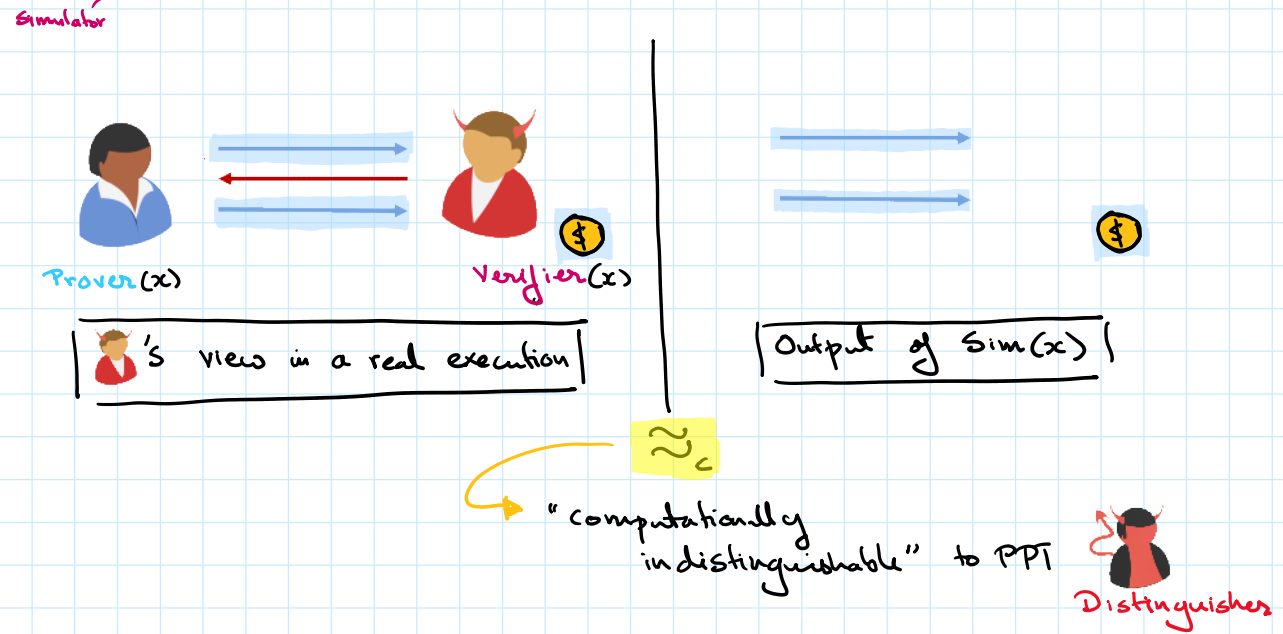
③ Zero-knowledge

Verifier(x)'s view of the protocol execution



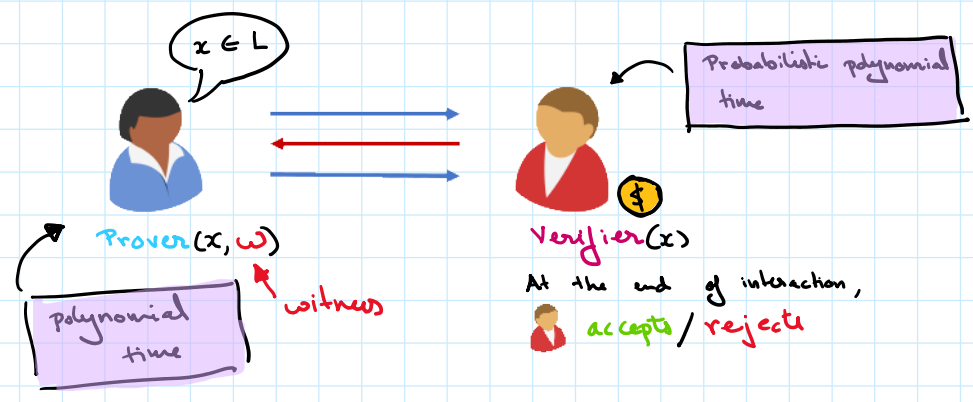
Every interaction with the prover can generate a different view

For every , there exists a PPT algorithm Sim such that for every $x \in L$




Relaxation to definition

① Efficient provers for NP

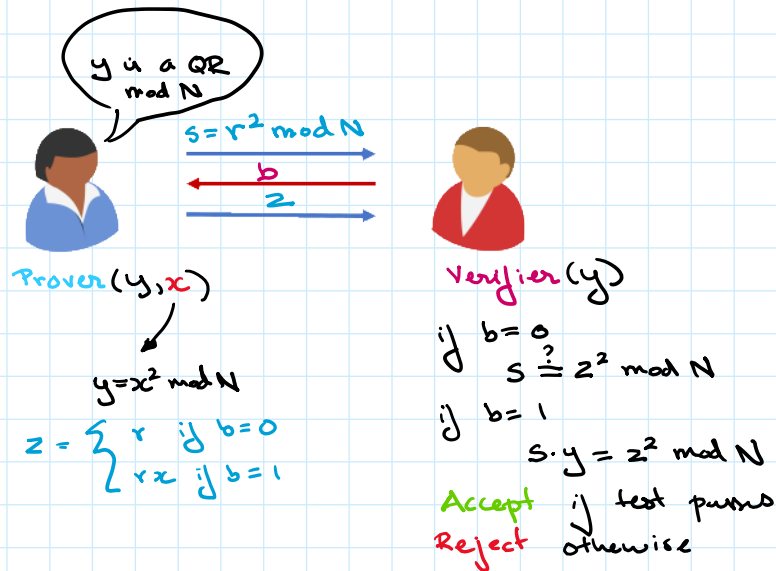


② Computational Soundness (Arguments)

if $x \notin L$, then for every  PPT

$\Pr[\text{accepts}] \leq \text{negl}(|x|)$



ZK Protocol for QR



Proof of Knowledge

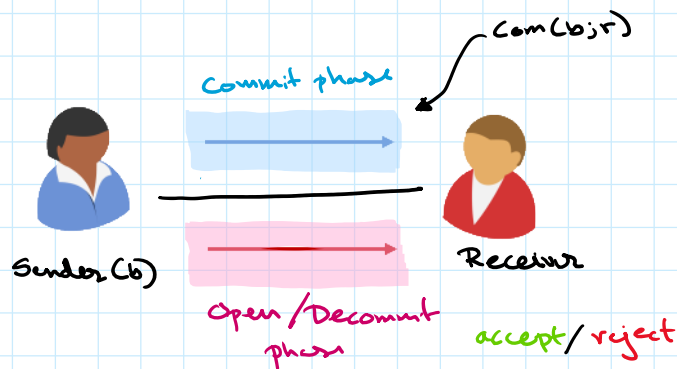
If a proof is accepting, prover "knows" the witness.

An interactive proof for an NP relation R is a proof of knowledge if \exists a PPT extractor E s.t.

For any , if **accept**, then E  (x) outputs witness w for x , i.e. $(x, w) \in R$

Zero-Knowledge for all of NP

Tool: Commitment (Digital analogue of locked boxes)



① Hiding the committed value is hidden

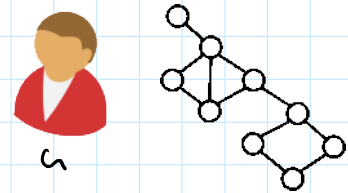
$\text{Com}(0;r) \approx_c \text{Com}(1;r)$
cannot tell a commitment of 0 from commitment of 1

② Binding there doesn't exist two different openings

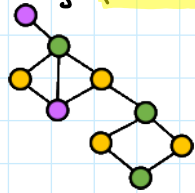
$$\Pr [\exists r_1, r_2 \neq r \cdot \text{Com}(0;r_1) = \text{Com}(1;r_2)] < \text{negl}$$

Decommitment is typically the randomness r and value b
→ Receiver can recompute $\text{Com}(b;r)$ to check if the decommitment is valid

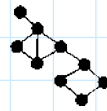
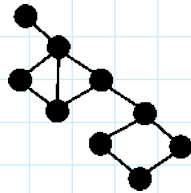
Graph 3-coloring ZKP



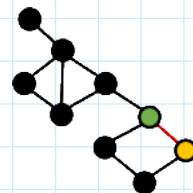
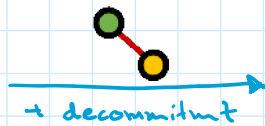
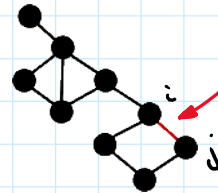
Step 1: Randomly **permute** the colors



Step 2: **Commit** to new coloring



Pick a random edge



Check

- ① Decommit is valid to (i,j)
- ② i & j have distinct colors

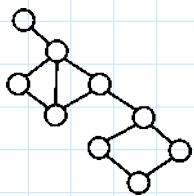
Accept if checks pass

- ① Completeness
- ② Soundness
- ③ Zero knowledge

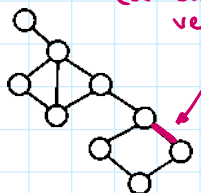
covered in class

Works for malicious verifiers

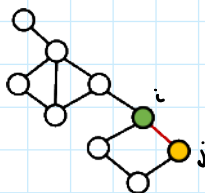
Simulator



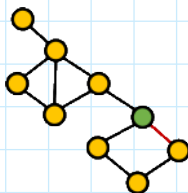
Step 1: Pick an edge at random
(cross the edge
verifier will pick)



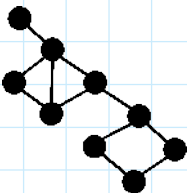
Step 2 Color edge distinct colors




Step 3 "Color" the rest



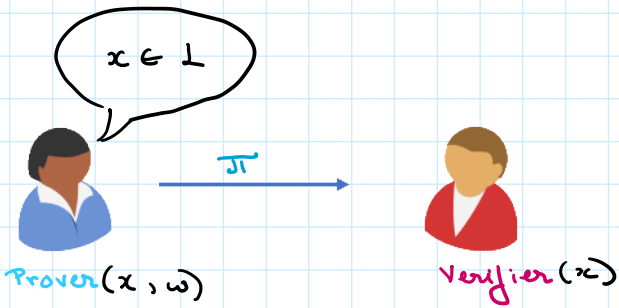
Step 4: Commit to graph



Step 5: Run  on 

i, j  picks (i, j) "done" can open the commitment and complete the transcript
else repeat from Step 1

Non-interactive Zero Knowledge

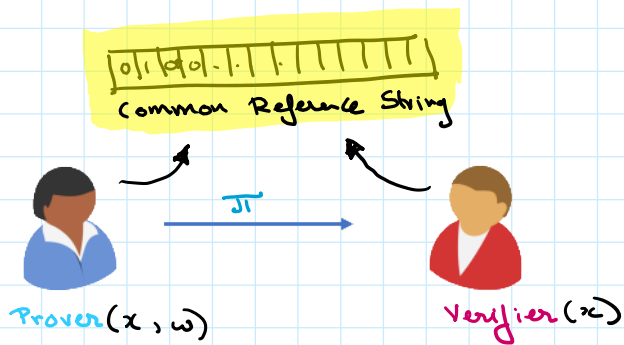


Impossible for anything "interesting"

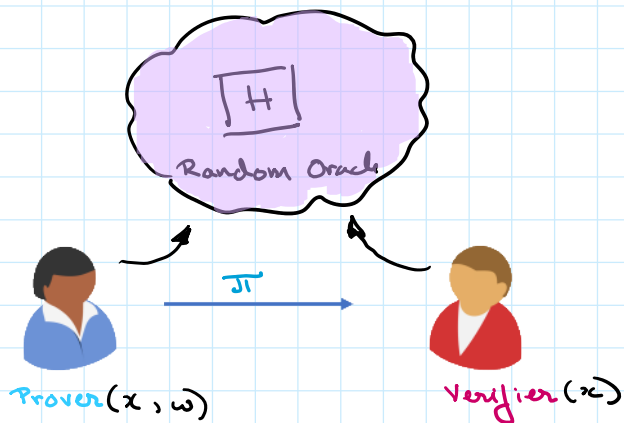
↳ anything the verifier could do by themselves

Change model and possible

① CRS



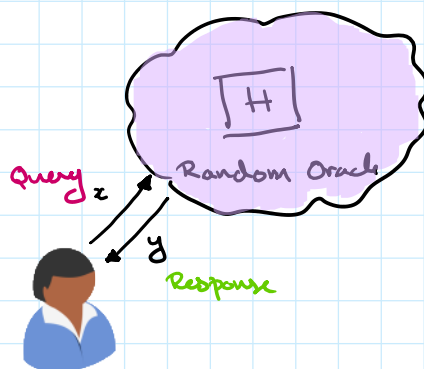
② Random Oracle (RO)



Random Oracle Model

[Bellare-Rogaway '93]

Model a hash function e.g. SHA256(\cdot) as if it were a truly random function H .



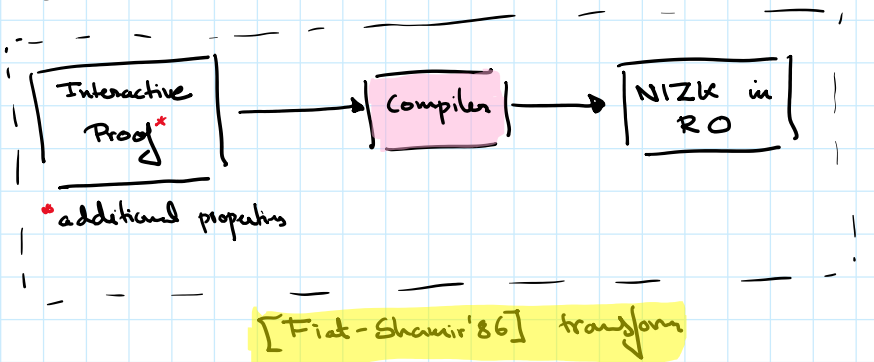
- We prove things to be secure in this **idealized model** in theory.
- In practice, we replace the random oracle with a fixed hash function and treat security as **heuristic**.

Commitments in the RO Model

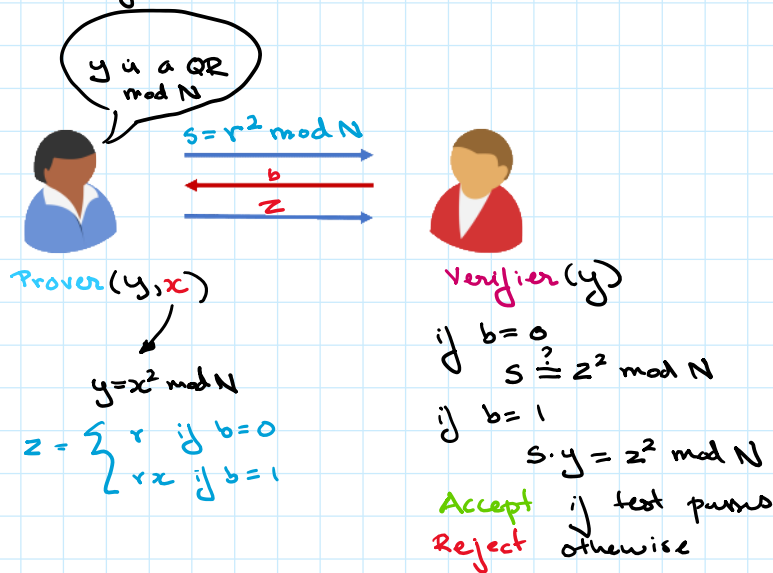
$$\text{com}(m;r) = H(m||r)$$

- ① Hiding - Random function hides its input
- ② Binding - Hard to find colliding inputs for a random function

NIZK in the RO model



Recall: Protocol for QR



Properties

- 1] Verifier has no hidden state → only sends random bits
- 2] Given two transcripts with different challenges.

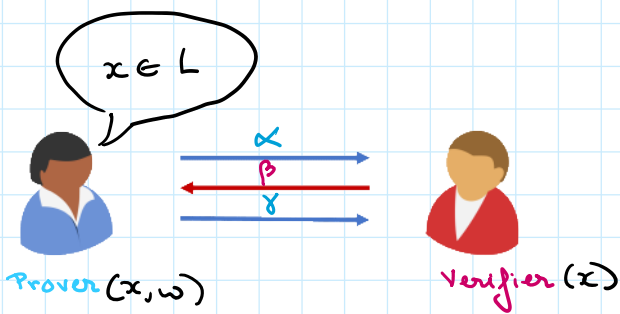


can extract the witness x . "Special" soundness


- 3] Given the challenge b , Sim outputs a perfectly indistinguishable view. "Special" Honest Verifier ZK (HVZK)

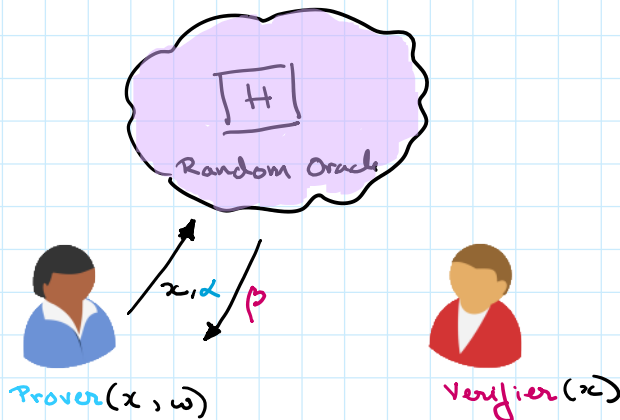
Sigma Protocols

Efficient ZK proofs for many interesting languages




- ① Special soundness ② Special HVZK
 Fiat-Shamir transform ③ public coin

Replace  with RO in a Sigma Protocol



- ① Compute α
 ② $\beta = H(x, \alpha)$
 ③ Compute γ
- $\pi = (\alpha, \beta, \gamma)$

Check if $H(x, \alpha) \stackrel{?}{=} \beta$
 Accept if (α, β, γ) accepts

H behaves like an honest 

① Completeness - Follows from completeness of Sigma protocol

② Soundness Intuition: Prover cannot guess β when it computes α and Proof of Knowledge

③ Zero-Knowledge - Follows from HVZK of Sigma protocol

→ formally proving requires the ability to "program" the random oracle, i.e. the simulator & extractor can set output of H at certain inputs.