Consider following example: Suppose prover wants to convince verifier that $N = pq$ where p, q are prime (and secret).

prover (N, p, q) verifier (N)

$$\xrightarrow{\pi = (p,q)}$$

$\downarrow$

accept if $N = pq$ and reject otherwise

Proof is certainly complete and sound, but now verifier also learned the factorization of N... (may not be desirable if prover was trying to convince verifier that N is a proper RSA modulus (for a cryptographic scheme) without revealing factorization in the process

↳ In some sense, this proof conveys information to the verifier [i.e., verifier learns something it did not know before seeing the proof]

Zero-knowledge: ensure that verifier does not learn anything (other than the fact that the statement is true)

How do we define "zero-knowledge"? We will introduce a notion of a "simulator."

Definition. An interactive proof system $\langle P, V \rangle$ for a language $\mathcal{L}$ is zero-knowledge if for all efficient (and possibly malicious) verifiers V^*, there exists an efficient simulator S such that for all $x \in \mathcal{L}$:

$$\underbrace{\mathrm{View}_{V^*}(\langle P, V \rangle (x))}_{\substack{\text{random variable denoting the set of messages} \\ \text{sent and received by } V^* \text{ when interacting with the prover } P \text{ on input } x}} \overset{c}{\approx} S(x)$$

What does this definition mean?

$\mathrm{View}_{V^*}(P \longleftrightarrow V^*(x))$: this is what V^* sees in the interactive proof protocol with P

$S(x)$: this is a function that only depends on the statement x, which V^* already has

If these two distributions are indistinguishable, then anything that V^* could have learned by talking to P, it could have learned just by invoking the simulator itself, and the simulator output only depends on x, which V^* already knows

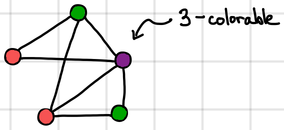
↳ In other words, anything V^* could have learned (i.e., computed) after interacting with P, it could have learned without ever talking to P!

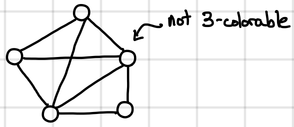
Very remarkable definition!

More remarkable: Using cryptographic commitments, then every language $\mathcal{L} \in \mathrm{IP}$ has a zero-knowledge proof system. ← can in fact be constructed from OWFs

↳ Namely, anything that can be proved can be proved in zero-knowledge!

We will show this theorem for NP languages. Here it suffices to construct a single zero-knowledge proof system for an NP-complete language. We will consider the language of graph 3-colorability.





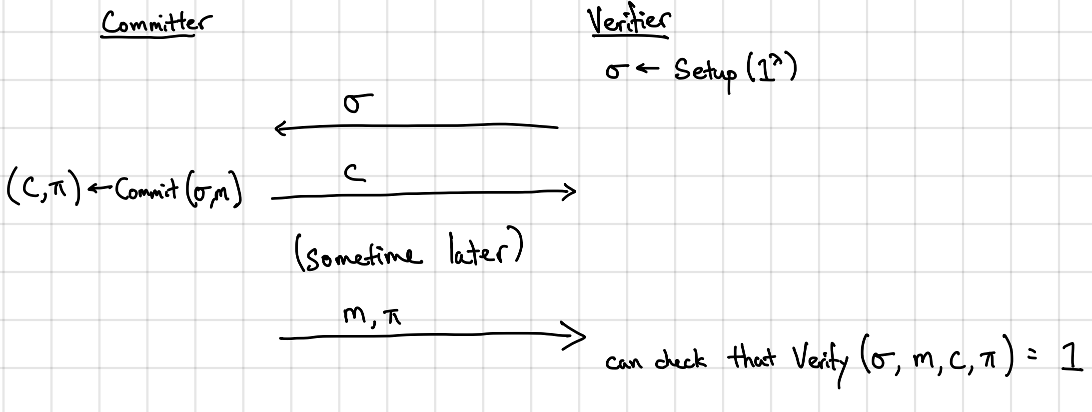
3-coloring: given a graph G, can you color the vertices so that no adjacent nodes have the same color?

cryptographic analog of a sealed "envelope"

We will need a commitment scheme. A (non-interactive) commitment scheme consists of three algorithms (Setup, Commit, Open):

- Setup$(1^\lambda) \to \sigma$: Outputs a common reference string (used to generate/validate commitments) σ
- Commit$(\sigma, m) \to (c, \pi)$: Takes the CRS σ and message m and outputs a commitment c and opening π
- Verify$(\sigma, m, c, \pi) \to 0/1$: Checks if c is a valid commitment to m (given π)

Typical setup:

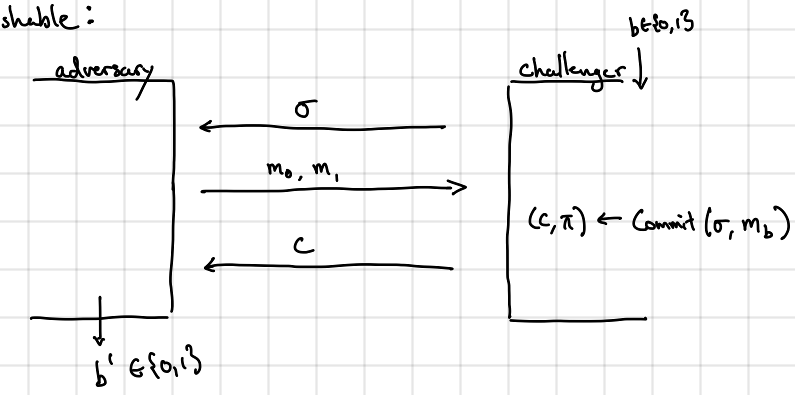


Requirements: [see HW5 for construction from OWFs]

- Correctness: for all messages m:

$$\Pr[\sigma \leftarrow \text{Setup}(1^\lambda);\ (c,\pi) \leftarrow \text{Commit}(\sigma, m);\ \text{Verify}(\sigma, c, m, \pi) = 1] = 1$$

- Hiding: for all common reference strings $\sigma \in \{0,1\}^n$ and all efficient A, following distributions are computationally indistinguishable:

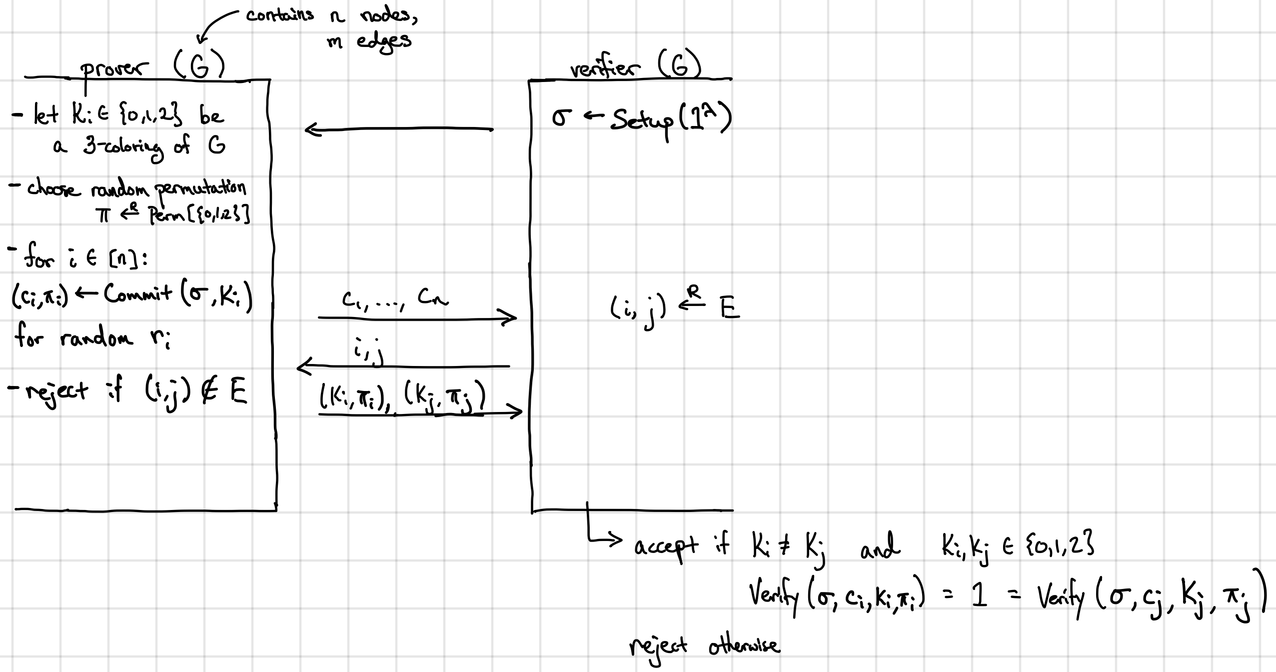


$$\left|\Pr[b' = 1 \mid b = 0] - \Pr[b' = 1 \mid b = 1]\right| = \text{negl}(\lambda)$$

- Binding: for all adversaries A, if $\sigma \leftarrow \text{Setup}(1^\lambda)$, then

$$\Pr[(m_0, m_1, c, \pi_0, \pi_1) \leftarrow A :\ m_0 \neq m_1 \text{ and } \text{Verify}(\sigma, c, m_0, \pi_0) = 1 = \text{Verify}(\sigma, c, m_1, \pi_1)] = \text{negl}(\lambda)$$

A ZK protocol for graph 3-coloring:



contains n nodes, m edges

prover (G)

- let $K_i \in \{0,1,2\}$ be a 3-coloring of G
- choose random permutation $\pi \xleftarrow{R} \mathrm{Perm}[\{0,1,2\}]$
- for $i \in [n]$:
  $(c_i, \pi_i) \leftarrow \mathrm{Commit}(\sigma, K_i)$
  for random r_i
- reject if $(i,j) \notin E$

verifier (G)

$\sigma \leftarrow \mathrm{Setup}(1^\lambda)$

$(i,j) \xleftarrow{R} E$

accept if $K_i \neq K_j$ and $K_i, K_j \in \{0,1,2\}$
$\mathrm{Verify}(\sigma, c_i, K_i, \pi_i) = 1 = \mathrm{Verify}(\sigma, c_j, K_j, \pi_j)$

reject otherwise

Intuitively: Prover commits to a coloring of the graph
Verifier challenges prover to reveal coloring of a single edge
Prover reveals the coloring on the chosen edge and opens the entries in the commitment

Completeness: By inspection [if coloring is valid, prover can always answer the challenge correctly]

except with prob. 1 − negl.

Soundness: Suppose G is not 3-colorable. Let $K_1, \ldots, K_n$ be the coloring the prover committed to. If the commitment scheme is statistically binding, $c_1, \ldots, c_n$ uniquely determine $K_1, \ldots, K_n$. Since G is not 3-colorable, there is an edge $(i,j) \in E$ where $K_i = K_j$ or $i \notin \{0,1,2\}$ or $j \notin \{0,1,2\}$. [Otherwise, G is 3-colorable with coloring $K_1, \ldots, K_n$.] Since the verifier chooses an edge to check at random, the verifier will choose (i,j) with probability $1/|E|$. Thus, if G is not 3-colorable,

$$\Pr[\text{verifier rejects}] \geq \frac{1}{|E|}$$

Thus, this protocol provides soundness $1 - \frac{1}{|E|}$. We can repeat this protocol $O(|E|^2)$ times sequentially to reduce soundness error to

$$\Pr[\text{verifier accepts proof of false statement}] \leq \left(1 - \frac{1}{|E|}\right)^{|E|^2} \leq e^{-|E|} = e^{-m} \quad [\text{since } 1 + x \leq e^x]$$

__Zero Knowledge__: We need to construct a simulator that outputs a valid transcript given only the graph G as input.

Let V^* be a (possibly malicious) verifier. Construct simulator S as follows:

1. Run V^* to get σ^*.
2. Choose $K_i \leftarrow \{0,1,2\}$ for all $i \in [n]$.
   Let $(c_i, \pi_i) \leftarrow \text{Commit}(\sigma^*, K_i)$
   Give $(c_1, \dots, c_n)$ to V^*.

   } Simulator does not know coloring so it commits to a random one

3. V^* outputs an edge $(i,j) \in E$
4. If $K_i \neq K_j$, then S outputs (K_i, K_j, π_i, π_j).
   Otherwise, restart and try again (if fails λ times, then abort)

Simulator succeeds with probability $2/3$ (over choice of $K_1, \dots, K_n$). Thus, simulator produces a valid transcript with prob. $1 - \frac{1}{3^\lambda} = 1 - \text{negl}(\lambda)$ after λ attempts. It suffices to show that simulated transcript is indistinguishable from a real transcript.

- __Real scheme__: prover opens K_i, K_j where $K_i, K_j \xleftarrow{R} \{0,1,2\}$ [since prover randomly permutes the colors]
- __Simulation__: K_i and K_j sampled uniformly from $\{0,1,2\}$ and conditioned on $K_i \neq K_j$, distributions are identical

In addition, (i,j) output by V^* in the simulation is distributed correctly since commitment scheme is computationally-hiding (e.g. V^* behaves essentially the same given commitments to a random coloring as it does given commitment to a valid coloring

If we repeat this protocol (for soundness amplification), simulator simulate one transcript at a time

__Summary__: Every language in NP has a zero-knowledge proof (assuming existence of OWFs)

Can be used to obtain ZK proof for IP:

recall: $\text{IP}[k] \subseteq \text{AM}[k+2]$

(Without loss of generality, suppose proof is public-coin – e.g., an Arthur-Merlin proof)

To construct ZK proof for $\mathcal{L} \in \text{IP}$, proceed as follows:

1) Replace prover's message with a computationally-hiding and statistical binding commitment to message
2) Verifier just send its random coins as in the AM protocol
3) Prover proves in zero-knowledge at the very end that the set of messages it committed to would cause the verifier to accept

   ↳ this is an NP statement [witness is the commitment openings and messages, relation checks openings to commitment and that verifier accepts the transcript]

__Implication__: Everything that can be proven (IP) can be proven in zero knowledge!