MIT 6.875

Foundations of Cryptography
Lecture 18

Zero Knowledge Proofs

An Interactive Protocol (P,V) is a **perfect/statistical/computational zero-knowledge** proof system for a language L if it is

a. Complete
b. Sound and
c. Zero knowledge: **for every PPT V^***, there exists a (expected) poly time simulator S s.t. for every $x \in L$, the following two distributions are identical/statistically close/computationally close:

$$1.\ view_{V^*}(P, V^*)(x) \qquad 2.\ S(x, 1^\lambda)$$

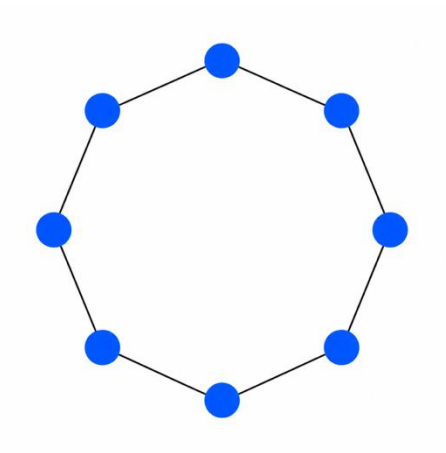
Efficient Prover (given a Witness)

In the protocols for quadratic residuosity and for graph isomorphism, the (honest) prover is actually polynomial-time ***given the NP witness*** (the square root of y in the case of QR, and the isomorphism in the case of graph-iso.)

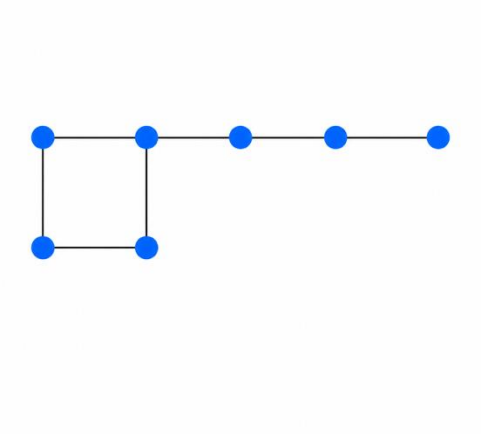
Soundness is nevertheless against any, even computationally unbounded, prover P^*.

But what when there isn't necessarily an NP witness?

Example: G_1 is NOT isomorphic to G_2



Graph G



Graph H



Prover



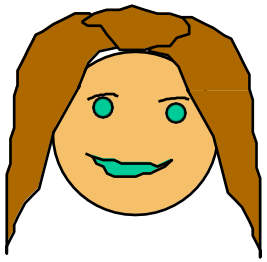
Verifier

Shortest classical proof:
exponential n!

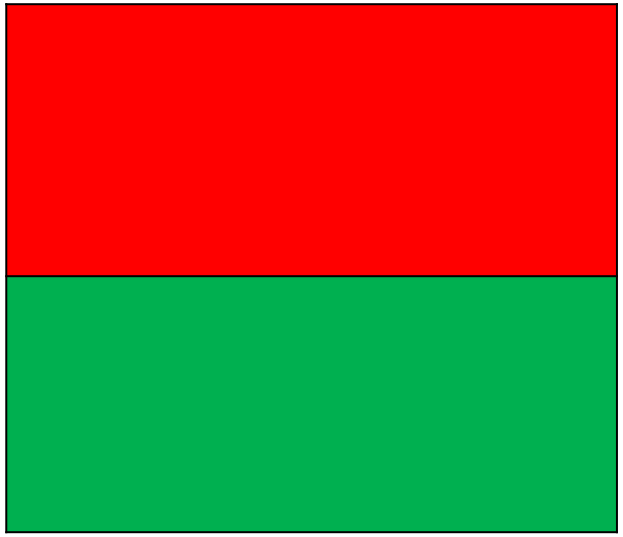
But **CAN** convince with an efficient interactive proof

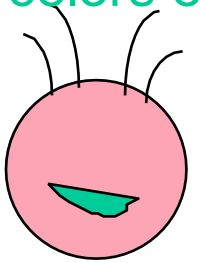
Zero Knowledge Proof

Prove to color blind bob that colors exist



Claim: there are 2 colors on this page,





Coin=**head**

Bob color blind

Bob tosses coin

If Heads keep red on top. If Tails flip to green on top.

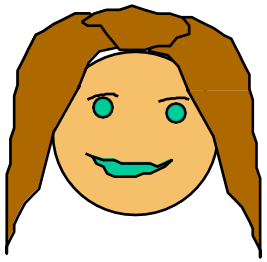
Bob sends resulting page

Alice guesses coin'

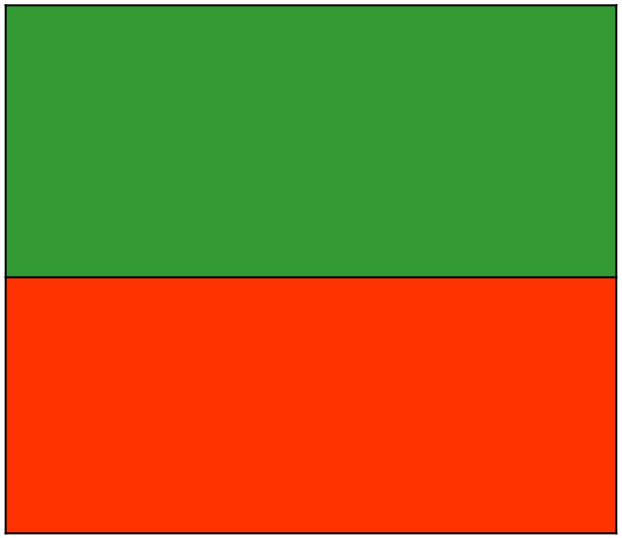
Reject, if Alice is wrong,
Otherwise repeat

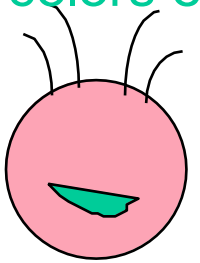
Zero Knowledge Proof

Prove to color blind bob that colors exist



Claim: there are 2 colors on this page,





Coin=**tail**

Bob color blind

Bob tosses coin

If Heads keep red on top. If Tails flip to green on top.

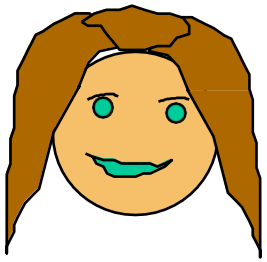
Bob sends resulting page

Alice Alice guesses coin'

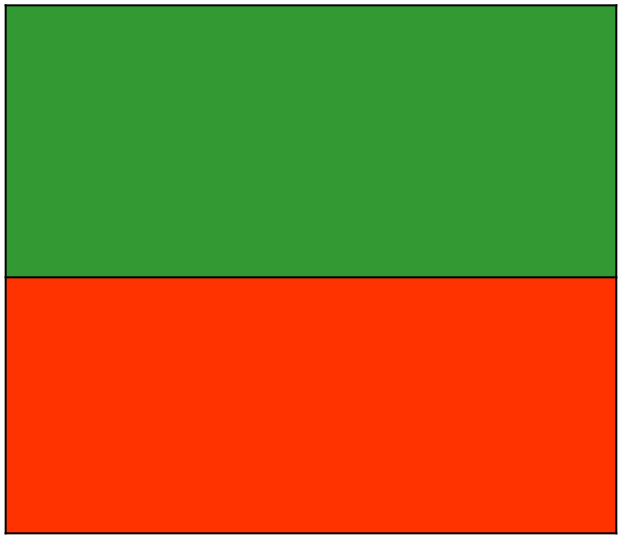
Reject, if Alice is wrong,
Otherwise repeat

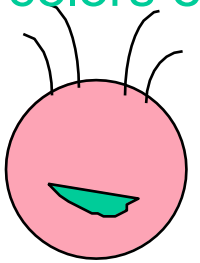
Zero Knowledge Proof

Prove to color blind bob that colors exist



Claim: there are 2 colors on this page,





Coin=**tail**

Bob color blind

Bob tosses coin
If Heads keep red on top. If Tails flip to green on top.

Bob sends resulting page

Alice Alice guesses $coin_1'$

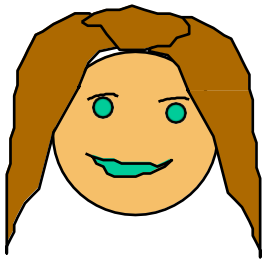
Bob sends resulting page

Alice Alice guesses $coin_2'$

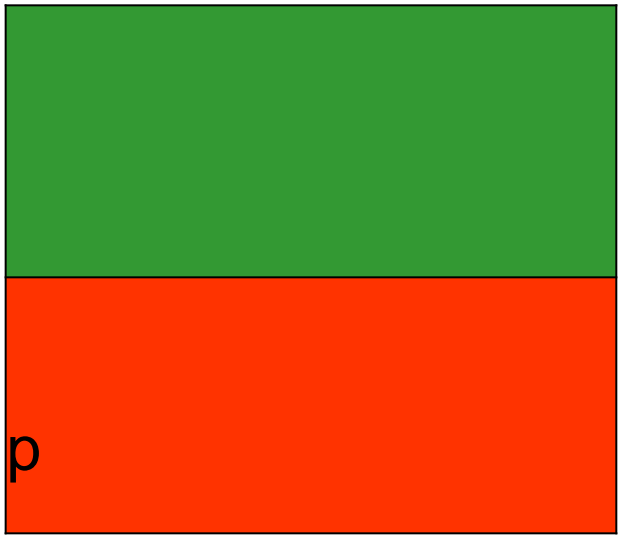
After λ trials, if Alice never wrong, then accept

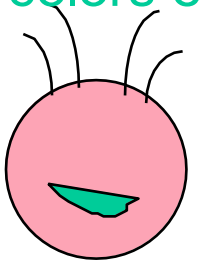
Zero Knowledge Proof

Prove to color blind bob that colors exist



Claim: there are 2 colors on this page, red to p





Bob color blind



Bob tosses coin

If Heads keep red on top.

Tails flip to green on top

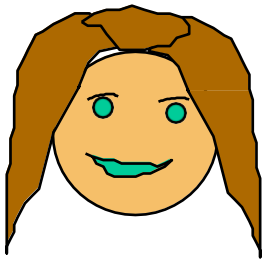
Completeness: if there are 2 colors, Bob will always accept

Soundness: if there is only 1 color, then

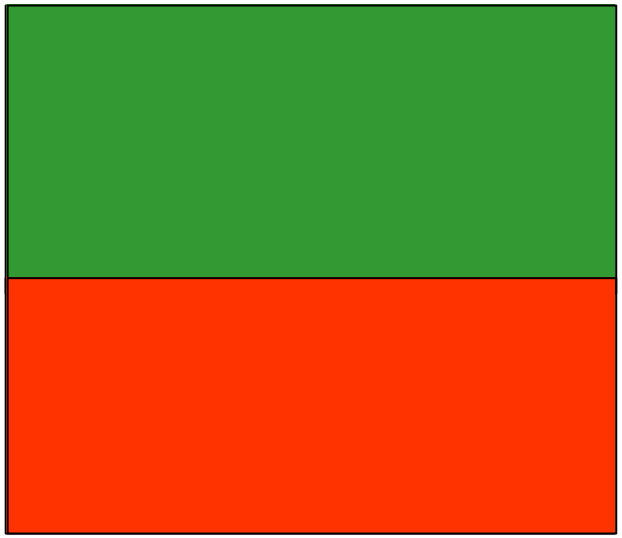
Probability that after 100 iterations Bob will reject > $1-1/2^{100}$

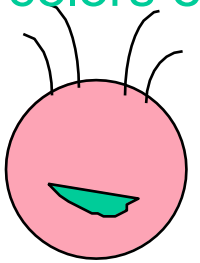
Zero Knowledge Proof

Prove to color blind bob that colors exist



Claims there are 2 colors on this page





Bob color blind

Bob tosses coin to decide if to flip page:
Heads keep red on top.
Tails flip to green on top.

Bob sends resulting page

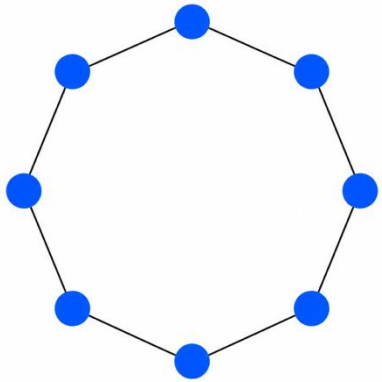
Alice names color on top=$coin_1$

Bob send resulting page

Alice names color on top=$coin_2$

View of Bob= {(Page, coin)}

ZK Proof for Graph Non-Isomorphism
(Honest Verifier)



Graph G



Graph H



Prover

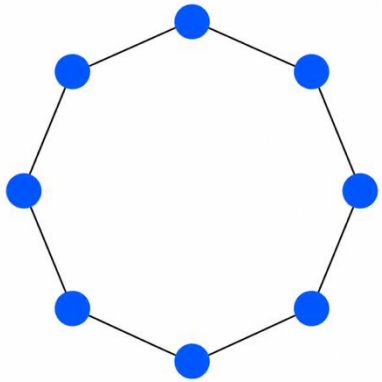
$$K = \rho(G)$$



Verifier

- flip coin c
- pick random ρ
- If c=0 let $K = \rho(G)$
- If c=1 let $K = \rho(H)$

ZK Proof for Graph Non-Isomorphism
(Honest Verifier)



Graph G



Graph H



Prover

$K = \rho(G)$



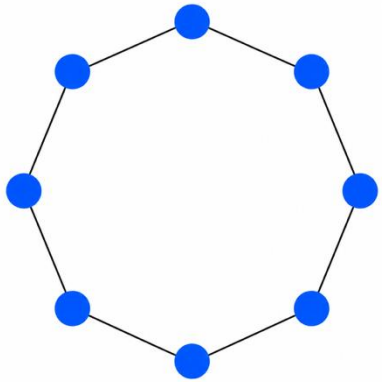
Verifier

- flip coin c
- pick random ρ
- If c=0 let $K = \rho(G)$
- If c=1 let $K = \rho(H)$

if K isomorphic to G
Then b=0
if K isomorphic to H
Else b=1

bit b

ZK Proof for Graph Non-Isomorphism
(Honest Verifier)



Graph G



Graph H



Prover

$K = \rho(G)$



Verifier

- flip coin c
- pick random ρ
- If c=0 let $K = \rho(G)$
- If c=1 let $K = \rho(H)$

if K isomorphic to G
Then b=0
if K isomorphic to H
Else b=1

bit b

Output ACCEPT

iff b = c

GNI Interactive Proof

- completeness:
  - if G not isomorphic to H then K is isomorphic to exactly one of (G, H)
  - Powerful prover can choose correct b=c
- soundness:

  if G is isomorphic to H then prover sees same distribution on K

  for c = 0 and on c = 1, namely K yields no information on c

  $\Rightarrow \forall\ P^*$ Prob[prover P* outputs b=c]<=1/2

- Soundness: After λ repetitions,

if (G,H)$\in ISO$, then $Prob[(P, V)[(G,H)]=accept] \leqq 1/2^{l}$

Honest Verifier Zero Knowledge

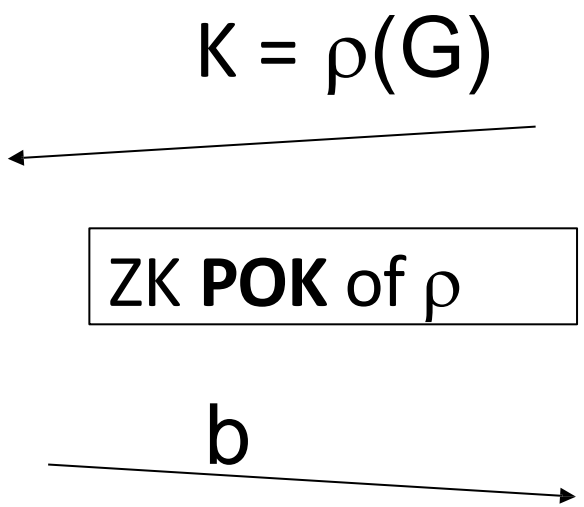
This is obviously honest verifier zero-knowledge (when the graphs are isomorphic):

--All the verifier gets to see is the coin c he tossed.

But, is it zero-knowledge for **all** verifiers?

-No. cheating V can use P to find out if K is isomorphic to G or isomorphic to H for K of its choice.

-Instead, the verifier proves in ZK to the prover that **"he knows"** ρ s.t either $K=\rho(G)$ or $K=\rho(H)$



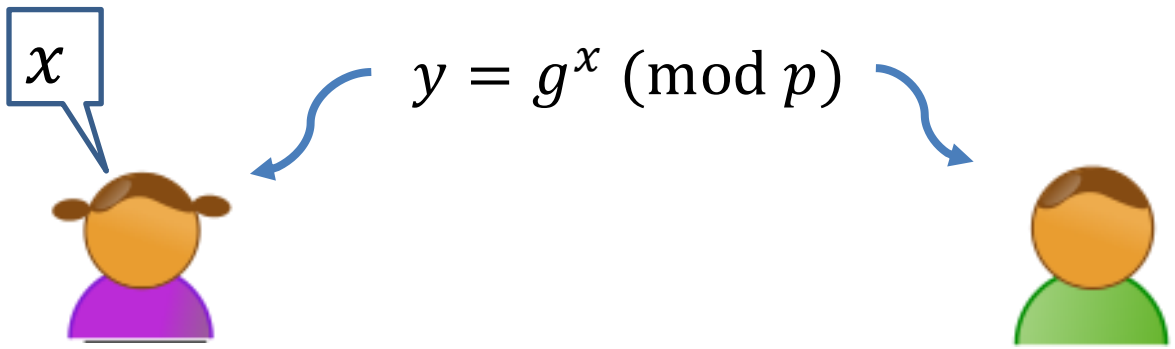
Zero Knowledge
Proofs of Knowledge (POK)

So far: Decision Problems

$$y \in L \text{ or } y \notin L$$

(e.g. y is a quadratic residue mod N or it is not)

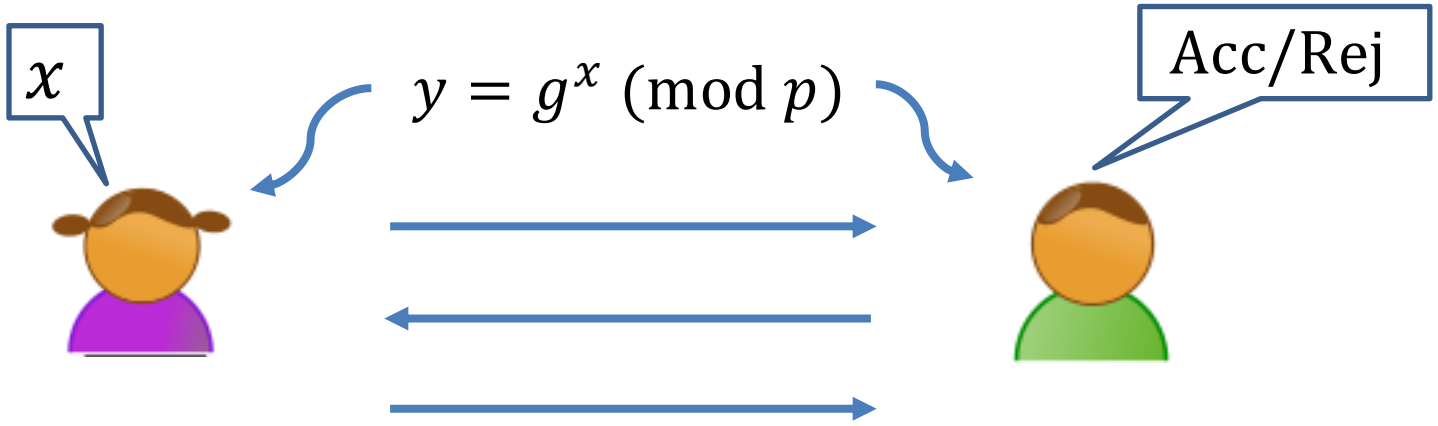
Here is a different scenario:



Discrete log of y always exists (assuming g is a generator)…

Alice wants to convince Bob that ***she knows a solution*** to a problem, e.g. that she knows the discrete log of y

So far: Decision Problems

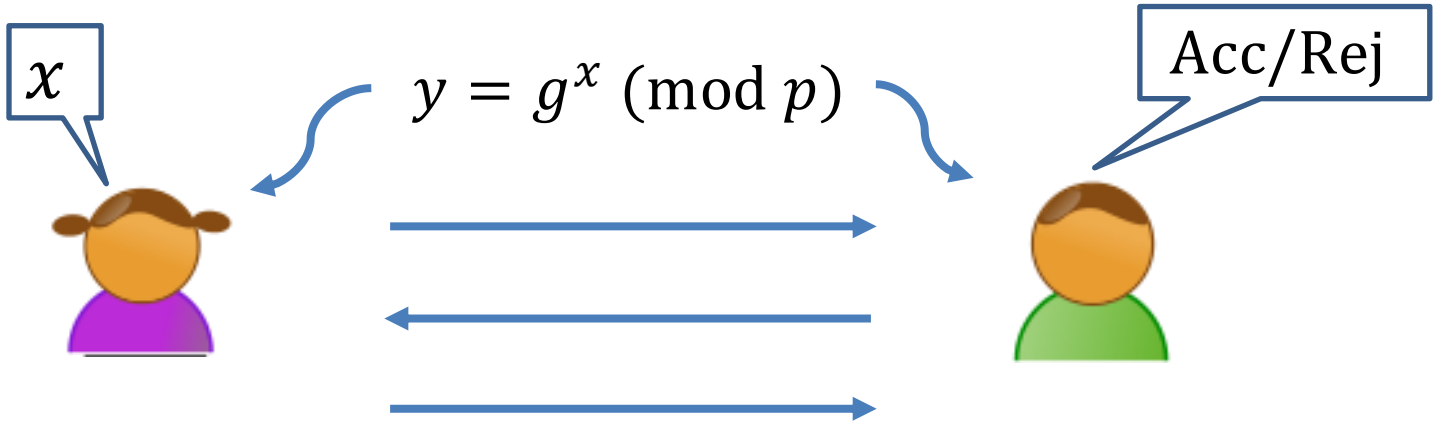


Completeness: When Alice and Bob run the protocol where Alice has input x, Bob outputs *accept*.

Soundness? How to define it?

Zero Knowledge: There is a simulator that, given only y, outputs a view of Bob that is indistinguishable from his view in an interaction with Alice.

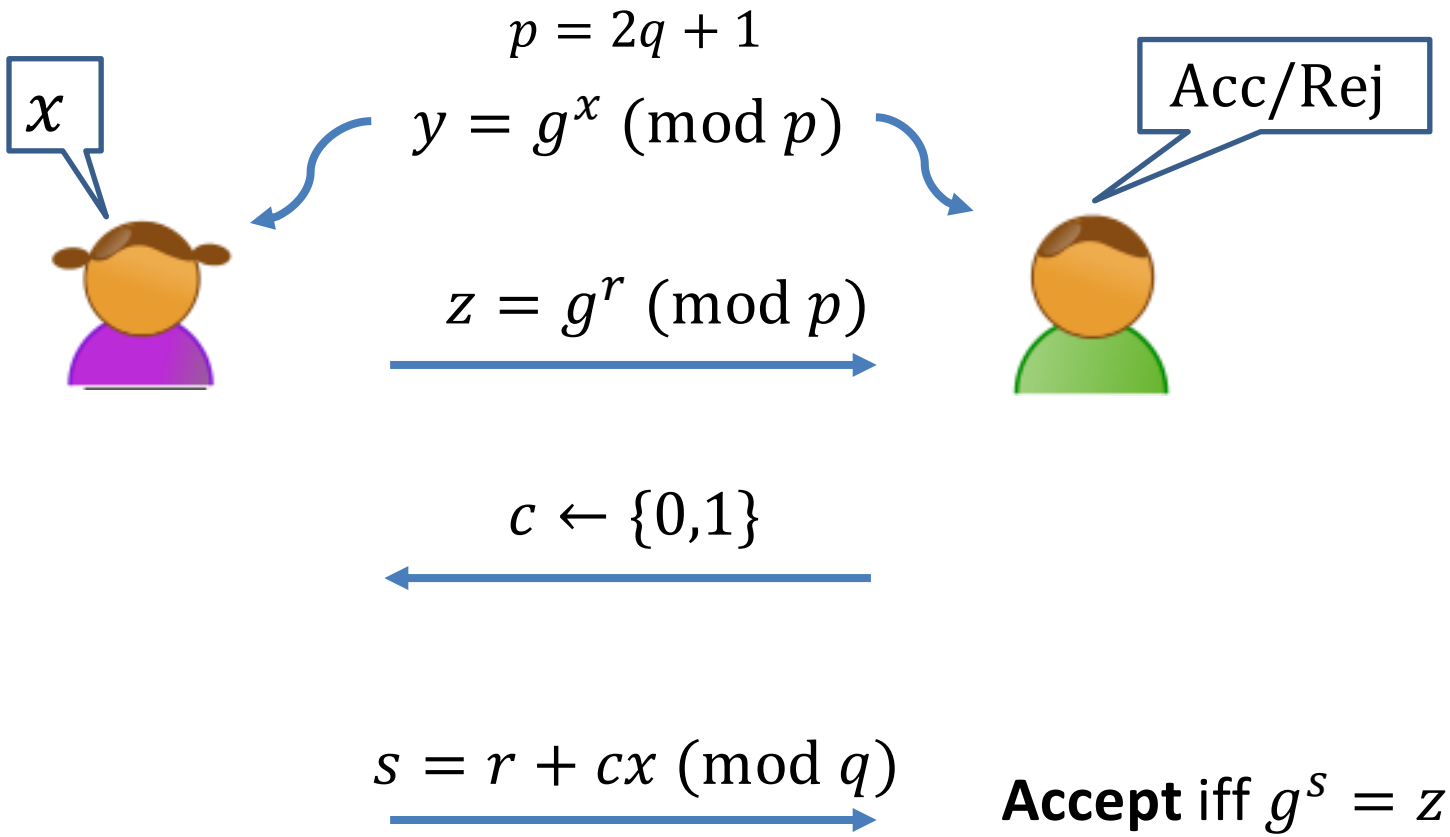
Proof of Knowledge



If Alice knows x, there must be a way to "extract it from her".

I will not define an extractor formally but will show you an example (see Goldreich's book for more)

ZK Proof of Knowledge of Discrete Log



Accept iff $g^s = z \cdot y^c$

Completeness and Zero Knowledge: Exercise.

Proof of Knowledge: Extractor



P^*

$y = g^x \pmod p$

$z = g^r \pmod p$

$c = 0$ $\quad$ $c = 1$

s_0 $\quad$ s_1

Assume P^* convinces the verifier with prob. $> \frac{1}{2} + 1/poly$

Extractor runs P^* to get a z.

Runs P^* with $c = 0$ and gets s_0

Rewinds P^* to the first message.

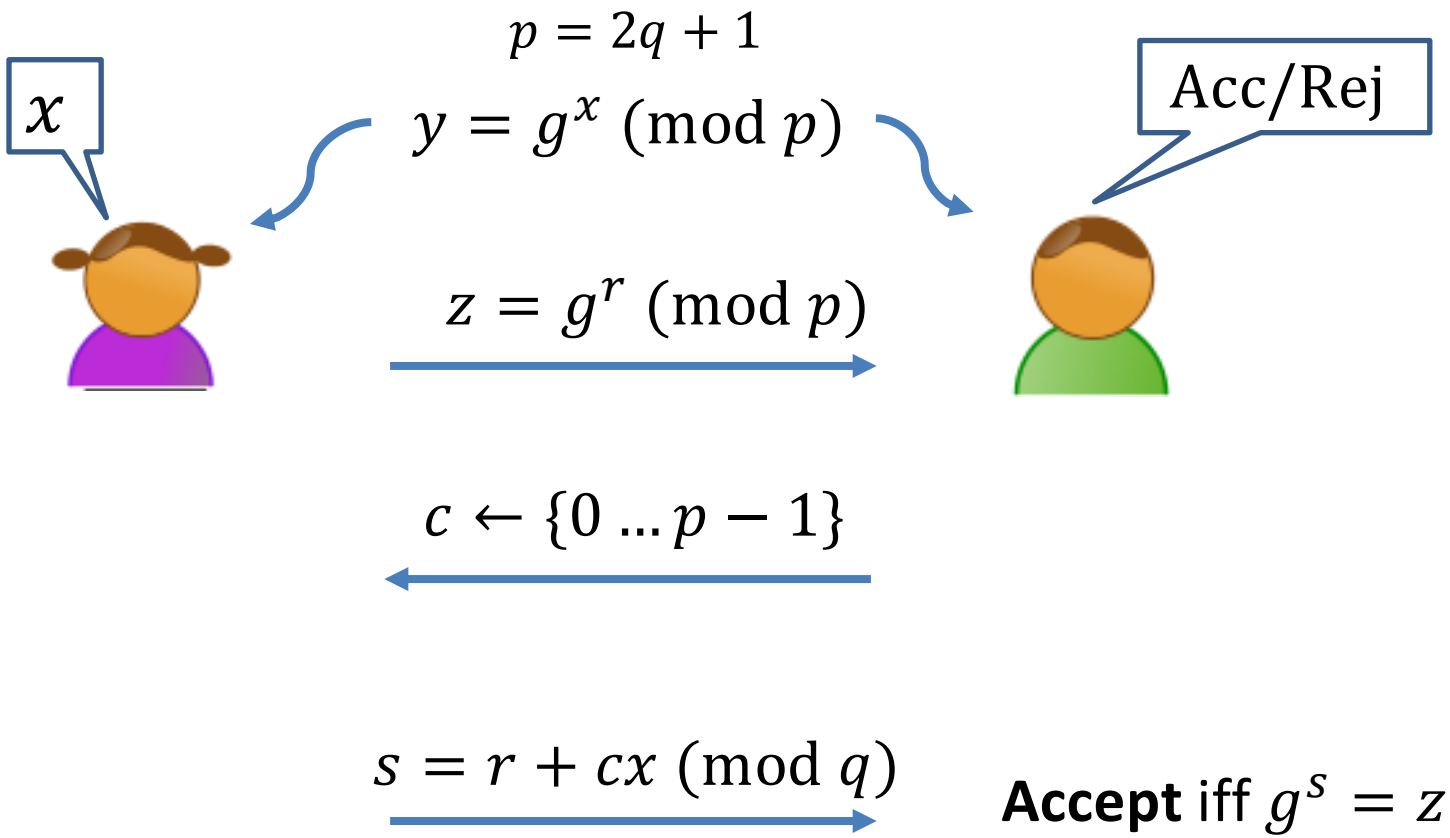
Runs P^* with $c = 1$ and gets s_1

$g^{s_0} = z$ and $g^{s_1} = zy$ w.p. $1/poly$

$g^{s_1 - s_0} = y$.

So, $s_1 - s_0$ is the discrete log of y.

Take 2: ZK Proof of Knowledge of Discrete Log



Completeness and Zero Knowledge: Exercise.

Proof of Knowledge: Extractor



P^*

$y = g^x \pmod p$

$z = g^r \pmod p$

$c = 0$ $\quad$ $c = 1$

s_0 $\quad$ s_1

Assume P^* convinces the verifier with prob. $> \frac{1}{p} + 1/poly$

Extractor runs P^* to get a z.

Runs P^* with $c = 0$ and gets s_0

Rewinds P^* to the first message.

Runs P^* with random $c \neq 0$ and gets s_1

$g^{s_0} = z$ and $g^{s_1} = zyc$ w.p. $1/poly$

So, $c^{-1}(s_1 - s_0)$ is the discrete log of y.

Zero Knowledge vs. Proof of Knowledge

Zero knowledge is a property of the prover against malicious verifiers. A prover P reveals zero knowledge if for all V^* …

Soundness and Proof of knowledge are properties of the verifier against malicious provers. A verifier V is sound (resp. satisfied PoK) if for all P^* …

Is Privacy of Verifiers coins essential?

No, It's all about sizes of sets
[GoldwasserSipser 86]

ISO(G) =Graphs Isomorphic to G

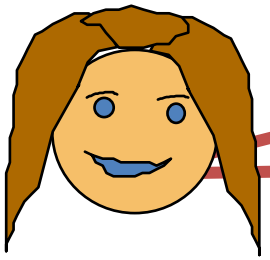
ISO(H) = Graphs Isomorphic to H

If $G \approx H$, then $|\ ISO(G) \cup ISO(H)| = n!$

If $G \not\approx H$, then $|\ ISO(G) \cup ISO(H)| = 2n!$

Not Quite,
Up to
Automorphisms

To show $G \not\approx H$:



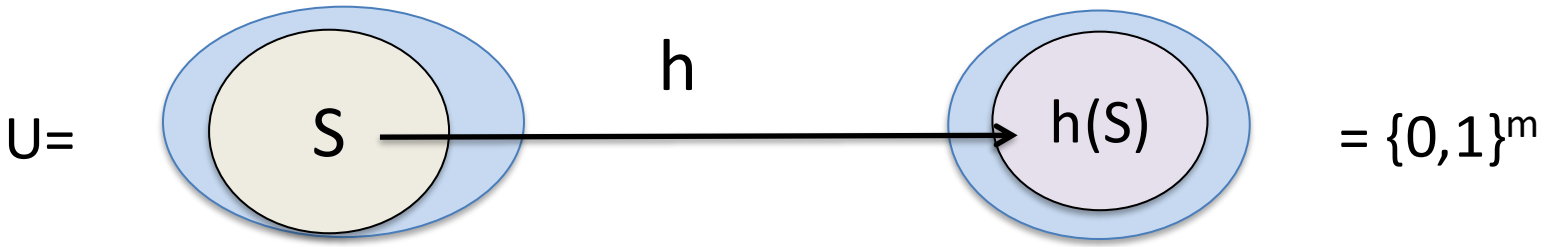
I will prove that

$|\ ISO(G_0) \cup ISO(G_1)| > n!$

How to prove set S is large?

Naïve: $S \subseteq U$. Verifier samples U (e.g. n-node graphs), counts # hits in S? works only if density of S in U is large.

Idea: Let $h: U \Rightarrow \{0,1\}^m$ be "random enough" function

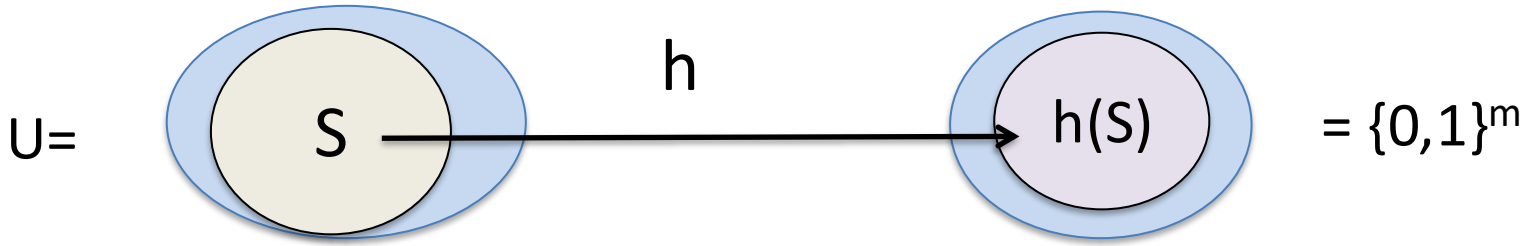


The larger |S| is, the more likely h(S) will hit target y in $\{0,1\}^m$

How to prove set S is large?

Naïve: $S \subseteq U$. Verifier samples U (e.g. n-node graphs), counts # hits in S? works only if density of S in U is large.

Idea: Let $h: U \Rightarrow \{0,1\}^m$ be "random enough" function



- V: sends random h, $y \in \{0.1\}^m$.
- P: must find $z \in S$ s.t. $h(z)=y$
  Send to V both z and proof that $z \in S$.
- V: accept iff $h(z)=y$ & verifies that $z \in S$

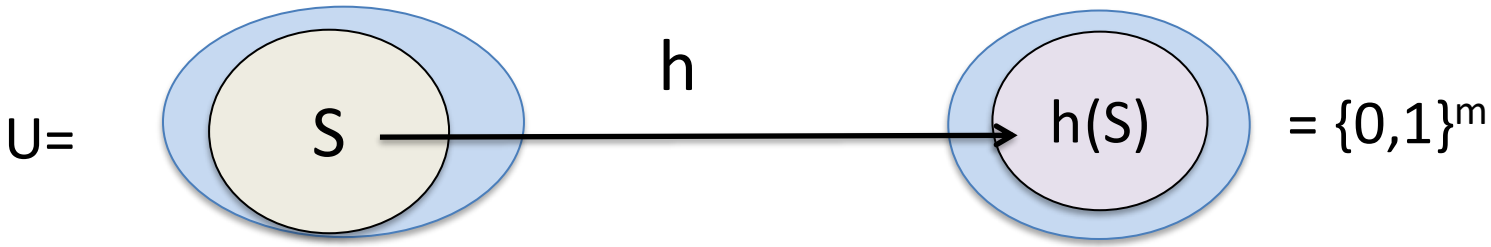
Completeness:
if $|S| \geq c2^m$,
prob[V accept] $\geq c/(1+c)$

Soundness:
if $|S| \leq s2^m$, for all P*
prob[V accept] $\leq s$

How to prove set S is large?

Naïve: $S \subseteq U$. Verifier samples U (e.g. n-node graphs), counts # hits in S? works only if density of S in U is large.

Idea: Let $h: U \Rightarrow \{0,1\}^m$ be "random enough" function



- How do you specify h that's is random enough?

Universal Hash Functions

Definition

An efficiently samplable family $\mathcal{H}$ of functions $h : \{0,1\}^n \to \{0,1\}^m$ is universal if, for every pair of distinct inputs $x \neq y$:

$$\Pr_{h \leftarrow \mathcal{H}}[\, h(x) = h(y) \,] \leq 2^{-m}$$

Explicit construction

$$h_{A,b}(x) = Ax + b \pmod 2, \quad A \leftarrow \mathbb{F}_2^{mxn}, \quad b \leftarrow \mathbb{F}_2^{m}$$

Polynomial-time samplable and evaluable;
no hardness assumption required.

Do all NP languages have Perfect ZK proofs?

We showed two languages with perfect ZK proofs. Can we show this for *all* NP languages?

Theorem [Fortnow'89, Aiello-Hastad'87] No, unless bizarre stuff happens in complexity theory (technically: the polynomial hierarchy collapses.)

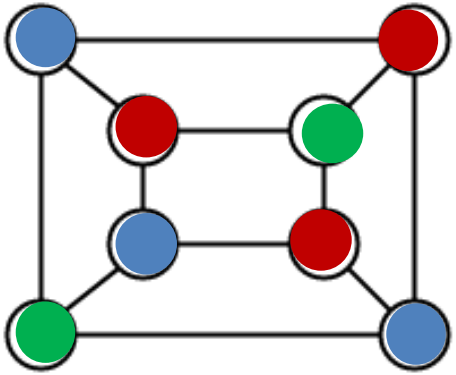
Do all NP languages have ZK proofs?

Nevertheless, today, we will show:

Theorem [Goldreich-Micali-Wigderson'87] Assuming one-way functions exist, all of NP has computational zero-knowledge proofs.

This theorem it tells us that everything that can be proved (in the sense of having a short witness) can be proved in zero knowledge!

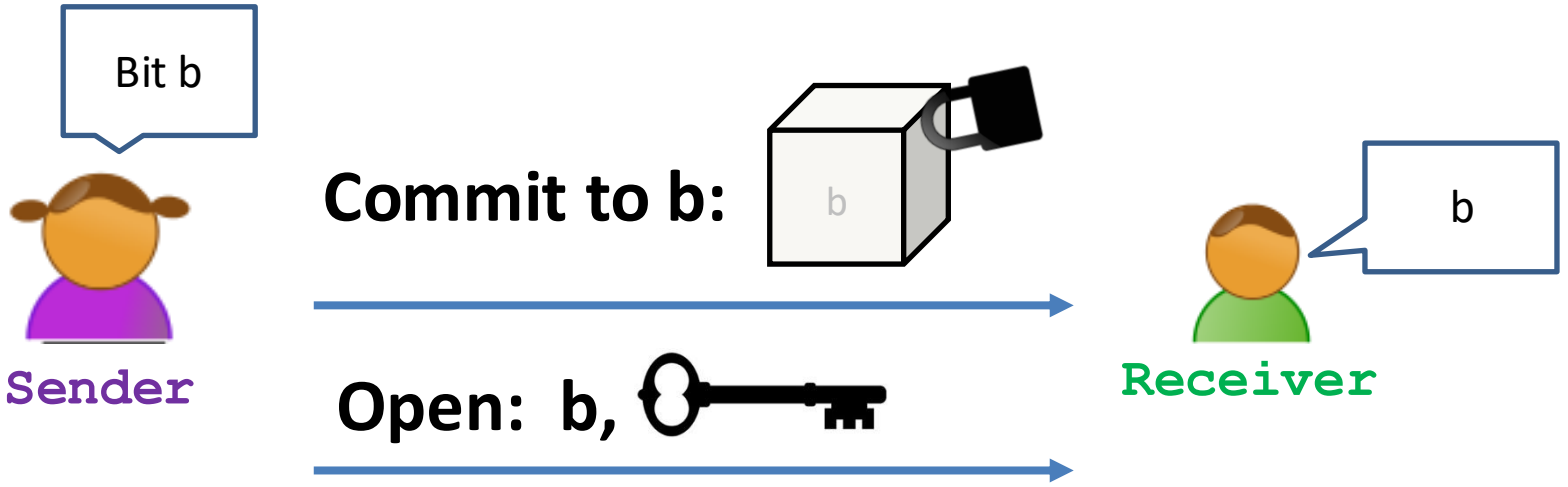
Zero Knowledge Proof for 3-Coloring



NP-Complete Problem:

Every other problem in NP can be reduced to it.

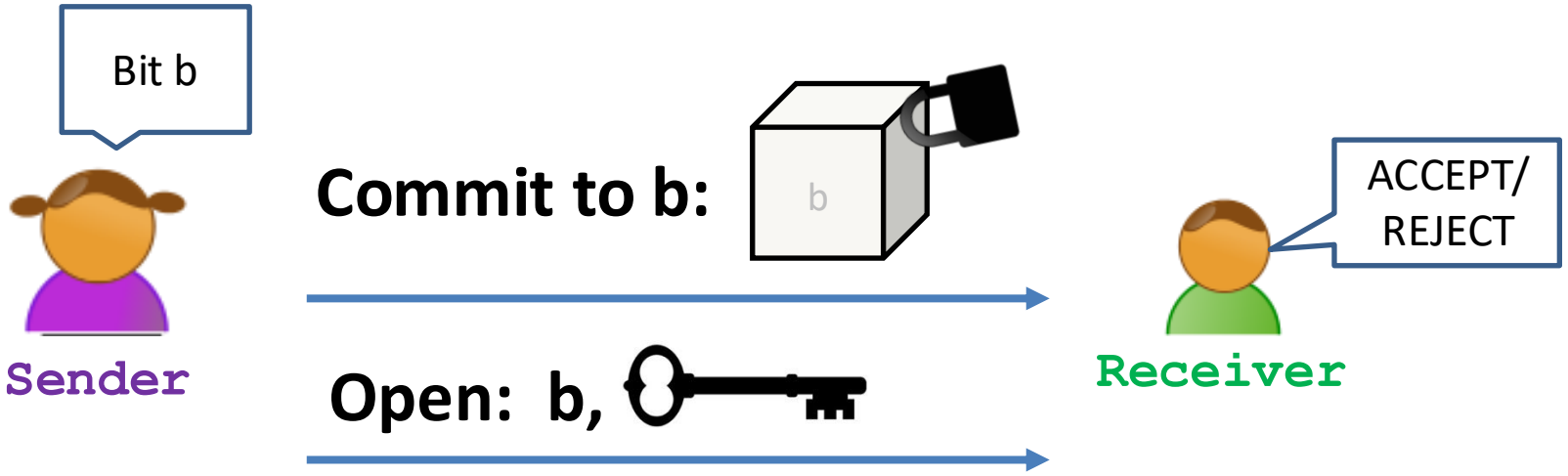
We need a commitment scheme (aka a "promise hiding scheme" from pset 1).



1. **Hiding:** The locked box should completely hide b.

2. **Binding:** Sender shouldn't be able to open to 1-b.

In pset 1, you implemented a commitment scheme using PRGs. We will later show another construction using one-way permutations.

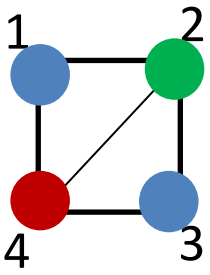


1. **Hiding:** The locked box should completely hide b.

2. **Binding:** Sender shouldn't be able to open to 1-b.

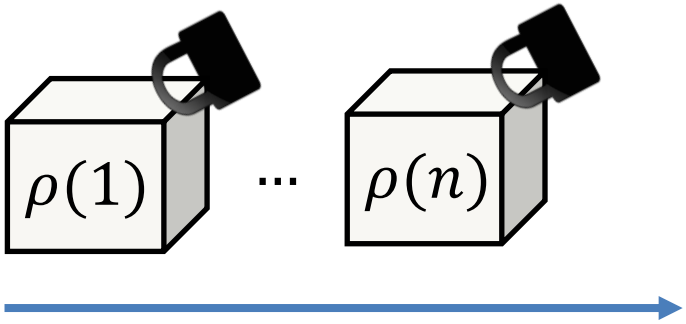
Zero Knowledge Proof for 3COL

Graph G
=(V,E)





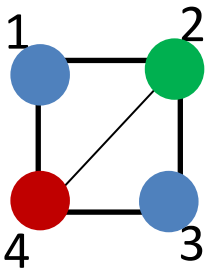
Come up with a random permutation of the colors $\rho: V \to \{R, B, G\}$

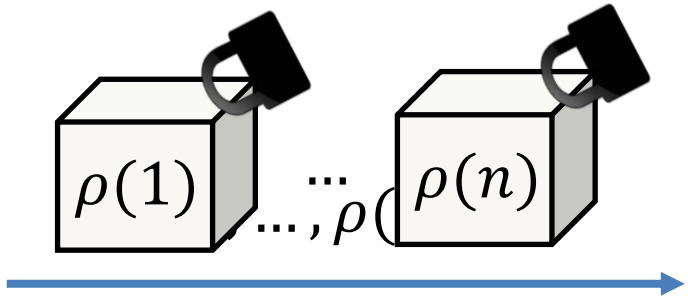




Zero Knowledge Proof for 3COL

Graph G
=(V,E)





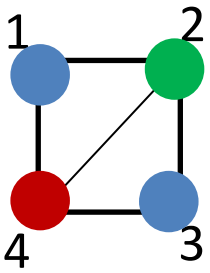


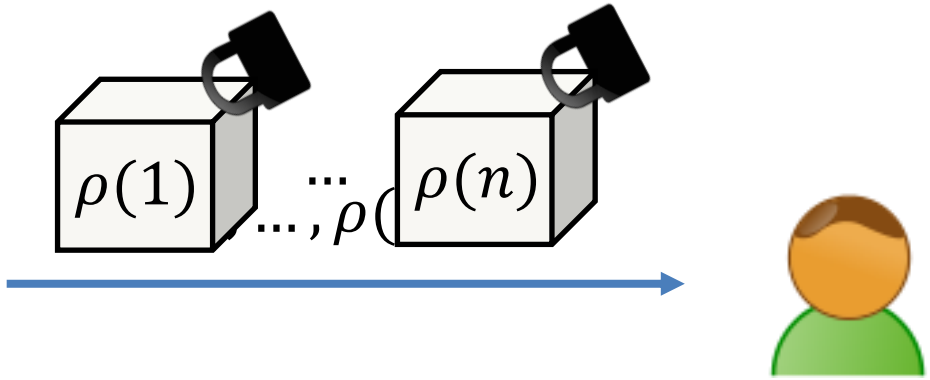
random edge (i, j)



Zero Knowledge Proof for 3COL

Graph G
=(V,E)





random edge (i, j)

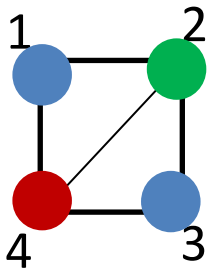
open $\rho(i)$ and $\rho(j)$



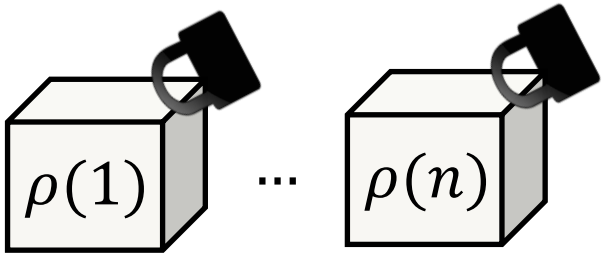
1. Check the openings
2. Check: $\rho(i), \rho(j) \in \{R, B, G\}$
3. Check: $\rho(i) \neq \rho(j)$.

Zero Knowledge Proof for 3COL

Graph G
=(V,E)









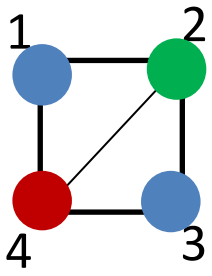
random edge (i, j)

open $\rho(i)$ and $\rho(j)$

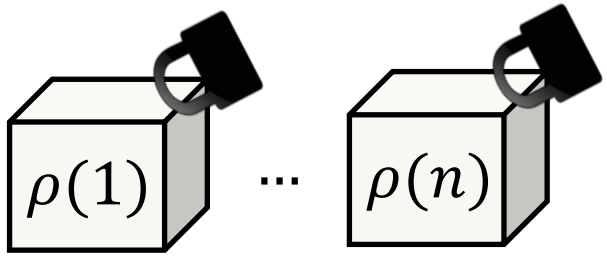
Completeness: Exercise.

Zero Knowledge Proof for 3COL

Graph G
=(V,E)









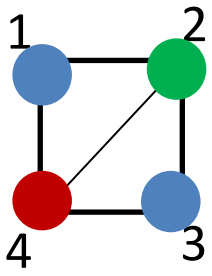
random edge (i, j)

open $\rho(i)$ and $\rho(j)$

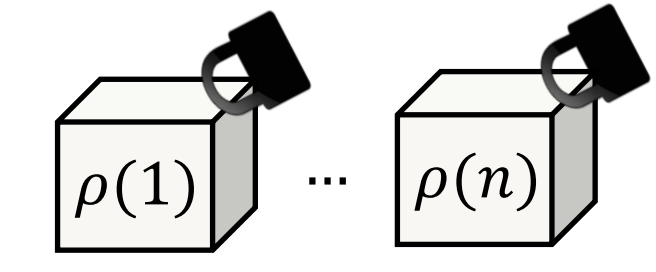
Soundness: If the graph is not 3COL, in every 3-coloring (that P commits to), there is some edge whose end-points have the same color. V will catch this edge and reject with probability $\geq 1/|E|$.

Zero Knowledge Proof for 3COL

Graph G
=(V,E)









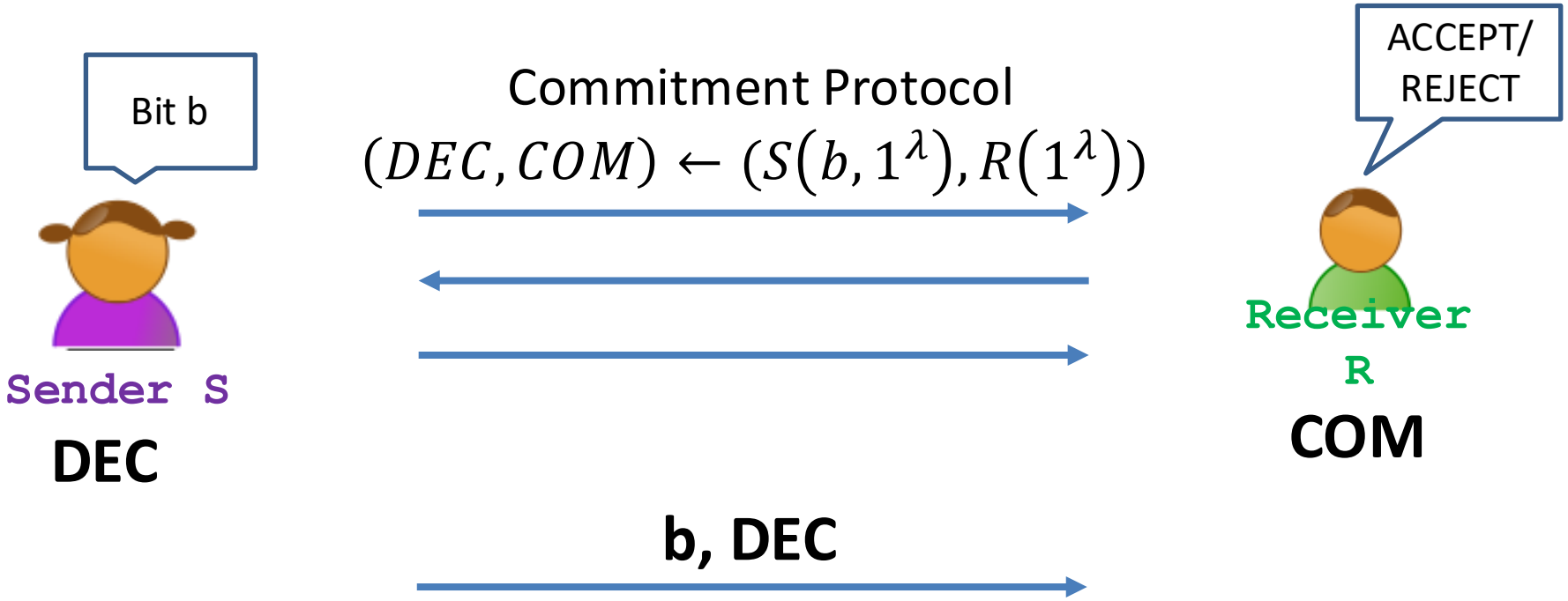
random edge (i, j)

open $\rho(i)$ and $\rho(j)$

Repeat $|E| \cdot \lambda$ times to get the verifier to accept with probability $\leq (1 - 1/|E|)^{|E| \cdot \lambda} \leq 2^{-\lambda}$

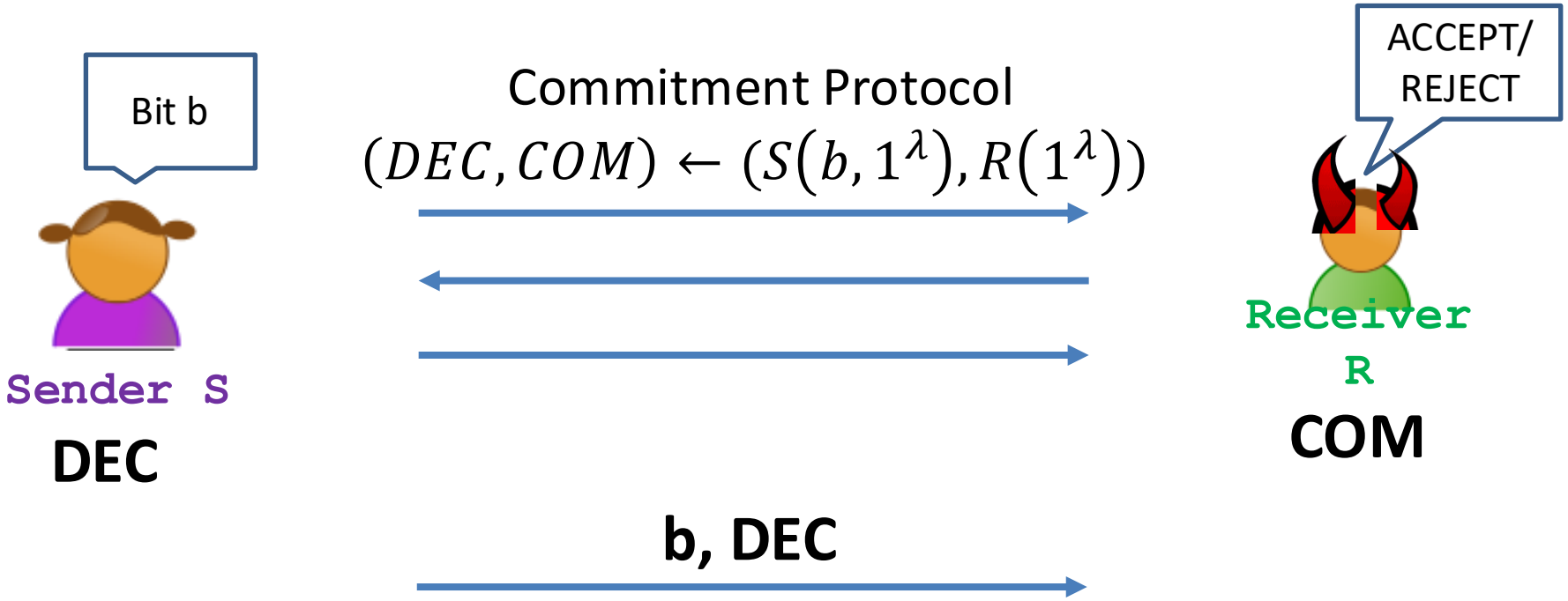
Commitment Schemes

Commitment Schemes



1. Completeness: R always accepts in an honest execution.

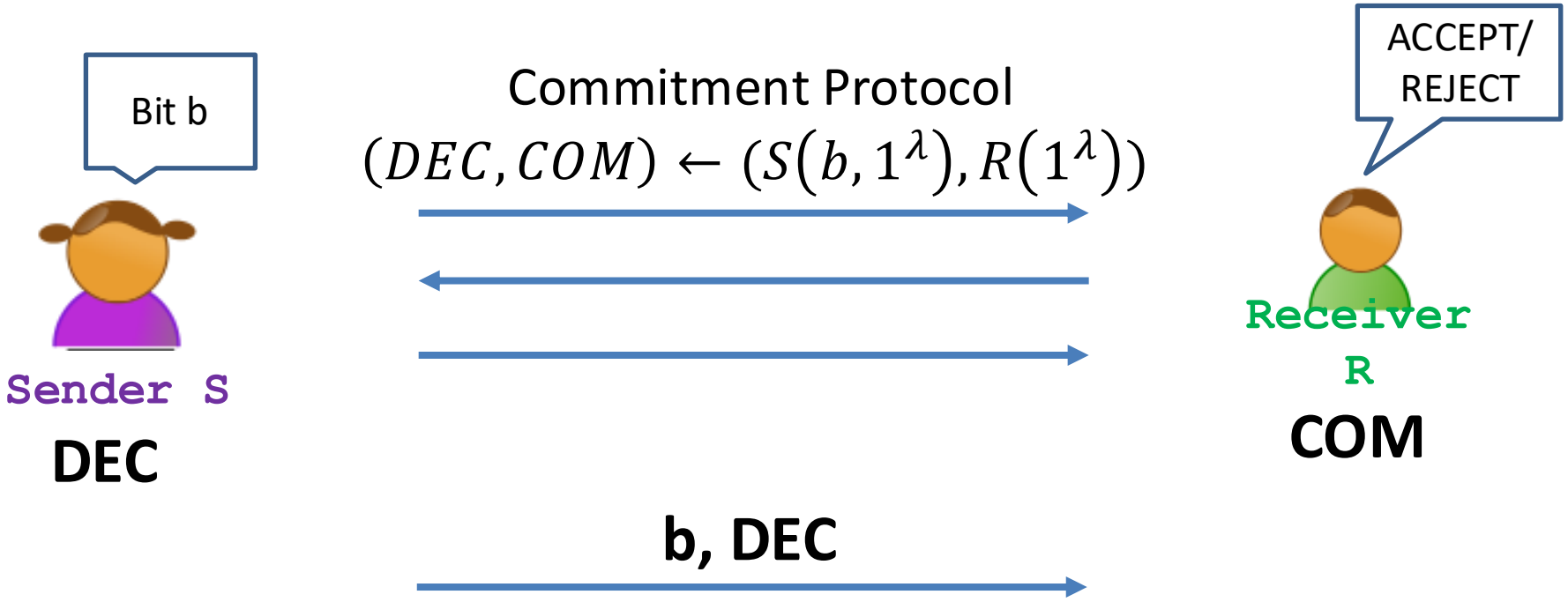
Commitment Schemes



2. Computational Hiding: For every possibly malicious (PPT) R^*,

$$view_{R^*}(S(0), R^*) \approx_c view_{R^*}(S(1), R^*)$$

Commitment Schemes



3. Perfect Binding: For every possibly malicious S^*, let COM be the receiver's output in an execution of (S^*, R). There is no pair of decommitments (DEC_0, DEC_1) s.t. R accepts both $(\text{com}, 0, DEC_0)$ and $(\text{com}, 1, DEC_1)$.

A Commitment Scheme from any OWP



Sender S

$DEC = r$

$COM = (f(r), HCB(r) \oplus b)$



Receiver R

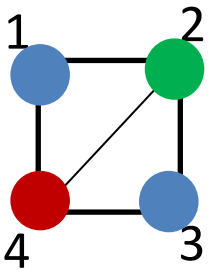
Let $COM = (x, y)$.
Check that
1. $f(r) = x$ and
2. $HCB(r) \oplus b = y$

$OPEN: (b, r)$

1. **Completeness:** Exercise.
2. **Comp. Hiding:** by the hardcore bit property.
3. **Perfect Binding:** because f is a permutation.

Back to ZK Proof for 3COL

Graph G
=(V,E)





$\{Com(\rho(k); r_k)\}_{k=1}^{n}$



random edge (i, j)

send openings $\rho(i), r_i$ and $\rho(j), r_j$

Why is this zero-knowledge?

Simulator S works as follows:

1. First pick a random edge (i^*, j^*)

   Color vertices i^* and j^* with random, different colors

   Color all other vertices red.

$\{Com(\rho(k); r_k)\}_{k=1}^{n}$ →

2. Feed the commitments of the colors to V^* and get edge (i, j)

← edge (i, j)



3. If $(i, j) \neq (i^*, j^*)$, go back and repeat.

send openings r_i and r_j →

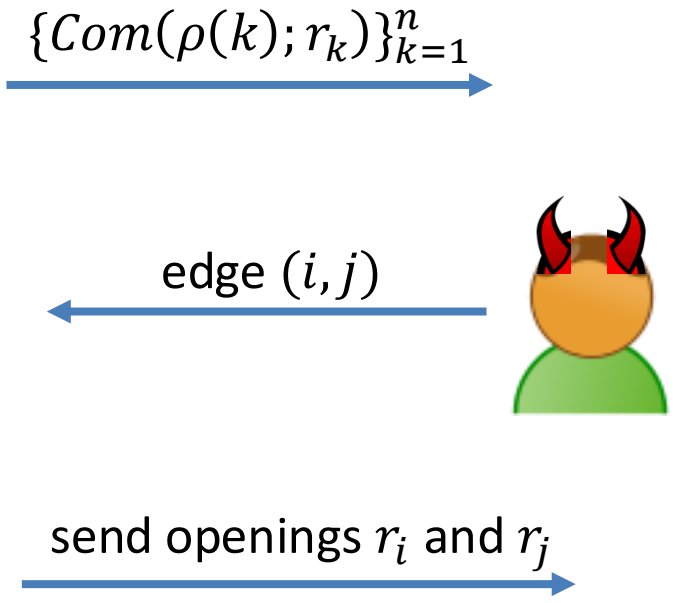
4. If $(i, j) = (i^*, j^*)$, output the commitments and openings r_i and r_j as the simulated transcript.

Why is this zero-knowledge?

Lemma:

(1) Assuming the commitment is hiding, S runs in expected polynomial-time.

(2) When S outputs a view, it is comp. indist. from the view of V^* in a real execution.



Why is this zero-knowledge?

Simulator S works as follows (call this Hybrid 0)

1. First pick a random edge (i^*, j^*)

   Color vertices i^* and j^* with random, different colors

   Color all other vertices red.

$\{Com(\rho(k); r_k)\}_{k=1}^{n}$ →

2. Feed the commitments of the colors to V^* and get edge (i, j)

← edge (i, j)



3. If $(i, j) \neq (i^*, j^*)$, go back and repeat.

send openings r_i and r_j →

4. If $(i, j) = (i^*, j^*)$, output the commitments and openings r_i and r_j as the simulated transcript.

Why is this zero-knowledge?

Not-a-Simulator S works as follows (call this Hybrid 1)

1. First pick a random edge (i^*, j^*)

   Permute a legal coloring and color all vertices correctly.

$\{Com(\rho(k); r_k)\}_{k=1}^{n}$ →

2. Feed the commitments of the colors to V^* and get edge (i, j)

← edge (i, j)



3. If $(i, j) \neq (i^*, j^*)$, go back and repeat.

send openings r_i and r_j →

4. If $(i, j) = (i^*, j^*)$, output the commitments and openings r_i and r_j as the simulated transcript.

Why is this zero-knowledge?

Claim: Hybrids 0 and 1 are computationally indistinguishable, assuming the commitment scheme is computationally hiding.

Proof: By contradiction. Show a reduction that breaks the hiding property of the commitment scheme, assuming there is a distinguisher between hybrids 0 and 1.

Why is this zero-knowledge?

Not-a-Simulator S works as follows (call this Hybrid 1)

1. First pick a random edge (i^*, j^*)

   Permute a legal coloring and color all vertices correctly.

$\{Com(\rho(k); r_k)\}_{k=1}^{n}$ →

2. Feed the commitments of the colors to V^* and get edge (i, j)

← edge (i, j)



3. If $(i, j) \neq (i^*, j^*)$, go back and repeat.

send openings r_i and r_j →

4. If $(i, j) = (i^*, j^*)$, output the commitments and openings r_i and r_j as the simulated transcript.

Why is this zero-knowledge?

Here is the real view of V* (Hybrid 2)

1. ~~First pick a random edge (i^*, j^*)~~

   Permute a legal coloring and color all edges correctly.

$\{Com(\rho(k); r_k)\}_{k=1}^{n}$ →

2. Feed the commitments of the colors to V^* and get edge (i, j)

← edge (i, j)



3. ~~If $(i, j) \neq (i^*, j^*)$, go back and repeat.~~

send openings r_i and r_j →

4. ~~If $(i, j) = (i^*, j^*)$,~~ output the commitments and openings r_i and r_j as the transcript.

Why is this zero-knowledge?

Claim: Hybrids 1 and 2 are identical.

Hybrid 1 merely samples from the same distribution as Hybrid 2 and, with probability $1 - 1/|E|$, decides to throw it away and resample.

Put together:

Theorem: The 3COL protocol is zero knowledge.

Examples of NP Assertions

- **My public key is well-formed** (e.g. in RSA, the public key is N, a product of two primes together with an e that is relatively prime to $\varphi(N)$.)
- **Encrypted bitcoin (or Zcash): "I have enough money to pay you."** (e.g. I will publish an encryption of my bank account and prove to you that my balance is $\geq \$X$.)
- **Running programs on encrypted inputs:** Given Enc(x) and y, prove that y = PROG(x).

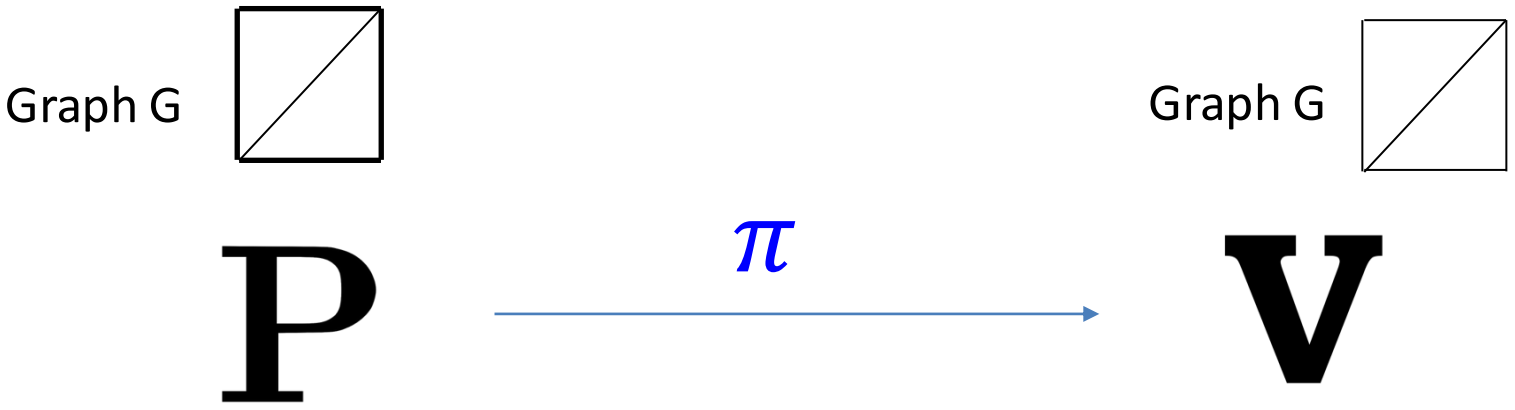
Examples of NP Assertions

- **Running programs on encrypted inputs:** Given Enc(x) and y, prove that y = PROG(x).

More generally: A tool to enforce honest behavior without revealing information.

Interaction is Necessary for ZK

Suppose there *were* a non-interactive ZK proof system for 3COL.

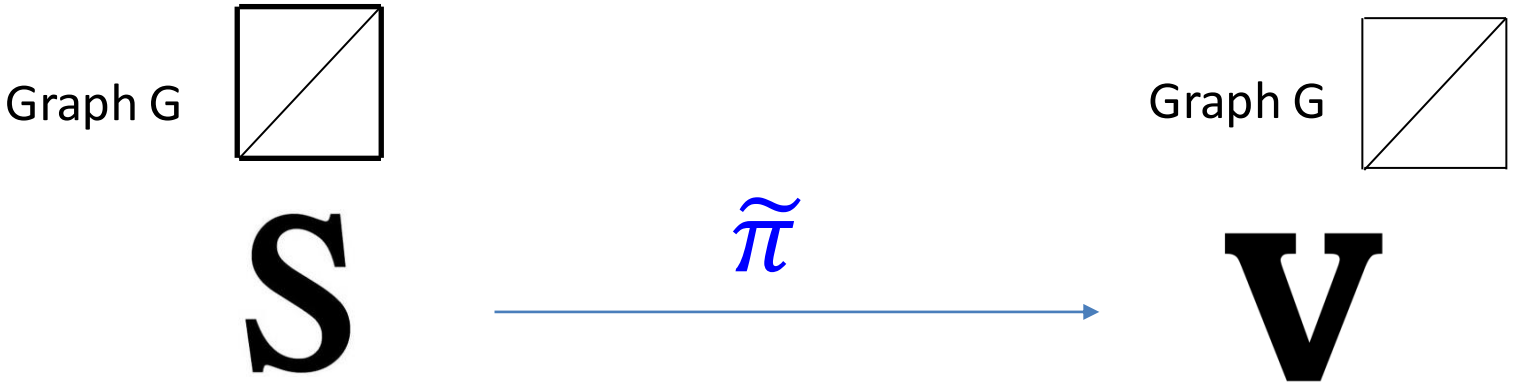


Step 1. When G is in 3COL, V accepts the proof π.

(Completeness)

Interaction is Necessary for ZK

Suppose there *were* a non-interactive ZK proof system for 3COL.

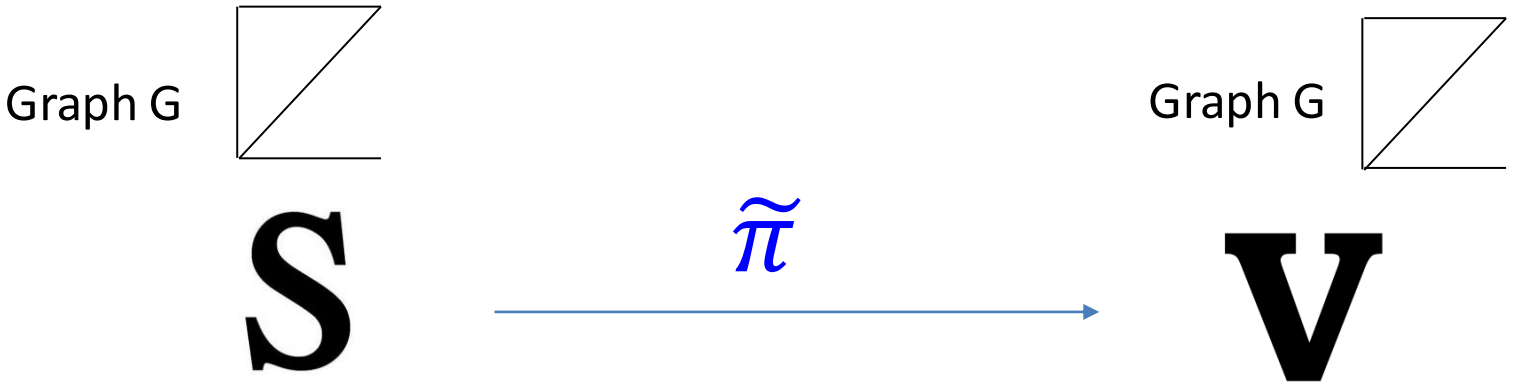


Step 2. **PPT** Simulator S, **given only G in 3COL**, produces an indistinguishable proof $\tilde{\pi}$ (Zero Knowledge).

In particular, V accepts $\tilde{\pi}$.

Interaction is Necessary for ZK

Suppose there *were* a non-interactive ZK proof system for 3COL.

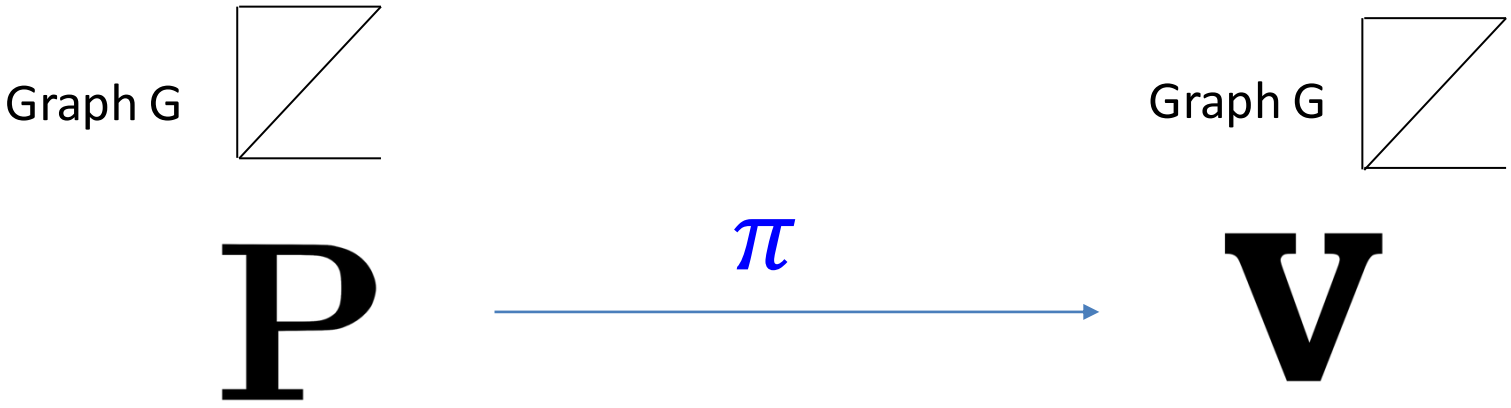


Step 3. Imagine running the Simulator S on a $G \notin$ 3COL. It produces a proof $\tilde{\pi}$ which the verifier still accepts!

(WHY?! Because S and V are PPT. They together cannot tell if the input graph is 3COL or not)

Interaction is Necessary for ZK

Suppose there *were* a non-interactive ZK proof system for 3COL.



Step 4. **Therefore, S is a cheating prover!**

Produces a proof for a $G \notin$ 3COL that the verifier nevertheless accepts.

Ergo, the proof system is NOT SOUND!

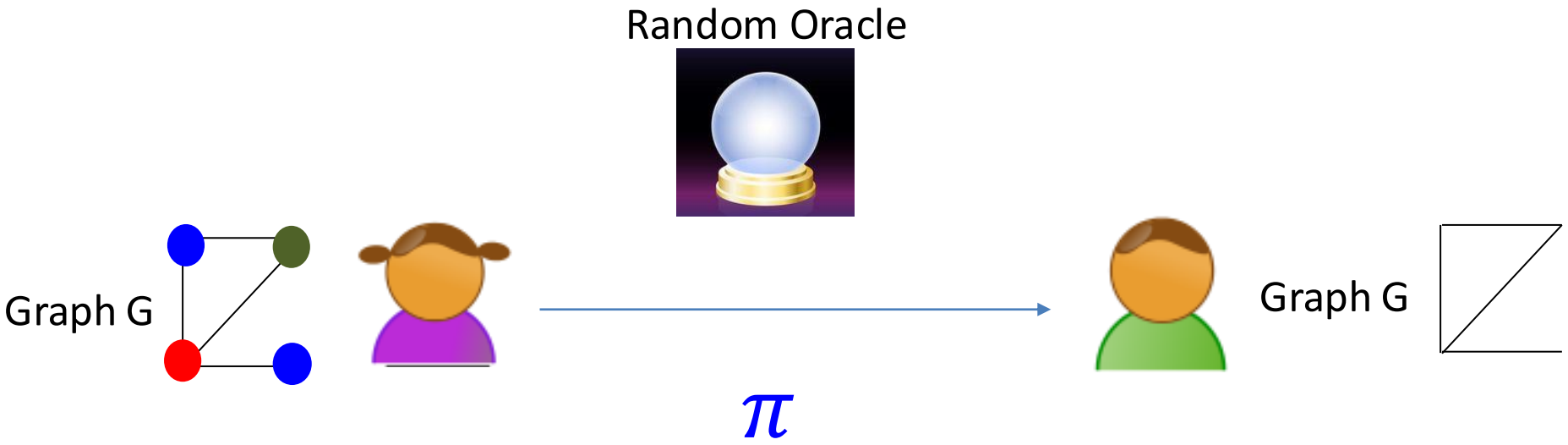


THE END

Or, is it?

Two Roads to Non-Interactive ZK (NIZK)

1. Random Oracle Model & Fiat-Shamir Transform.



2. Common Random String Model.

NEXT TIME

Topic 2:

The Power of Interactive Proofs

What can we prove with interaction?

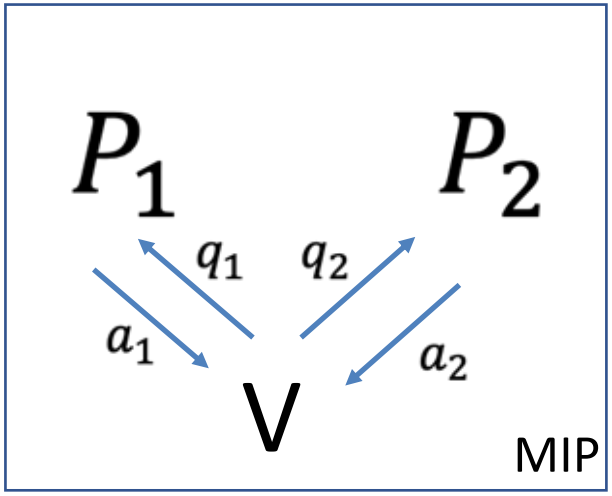
The Power of Interactive Proofs

Theorem [Nisan'90, Lund-Fornow-Karloff-Nisan'90] There is an interactive proof for the statement that the number of satisfying assignments to a formula is a given number (this complexity class is called $\#P$).

Theorem [Shamir'90] $IP = PSPACE$.

The Power of Interactive Proofs

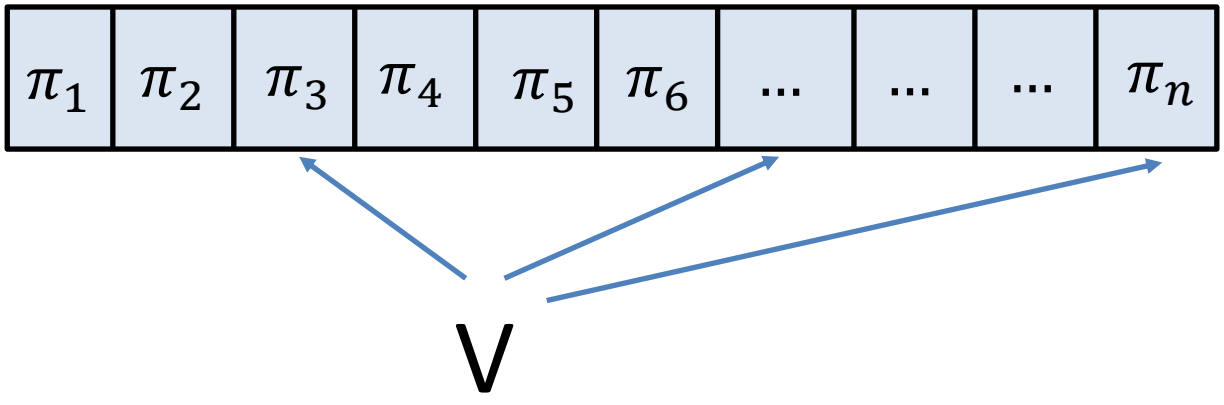
Definition of multi-prover interactive proofs [BenOr-Goldwasser-Kilian-Wigderson'88]



Theorem [Babai-Fornow-Lund'90] $\mathrm{M}IP = NEXP$.

The Power of Interactive Proofs

Definition of probabilistically checkable proofs [Arora-Safra'92, Feige-Goldwasser-Lovasz-Safra-Szegedy'91]



Theorem [Arora-Lund-Motwani-Sudan-Szegedy'92]

$$\mathrm{PCP}(3) = NP.$$

E-mail and the unexpected power of interaction

László Babai *
Eötvös University, Budapest
and
The University of Chicago

Abstract

This is a true fable about Merlin, the infinitely intelligent but never trusted magician; and Arthur, the reasonable but impatient sovereign with an occasional unorthodox request; about the concept of an efficient proof; about polynomials and interpolation, electronic mail, coin flipping, and the *incredible power of interaction*.

About MIP, IP, $\#P$, $PSPACE$, $NEXPTIME$, and new techniques that do not relativize. About fast progress, fierce competition, and e-mail ethics.

1 How did Merlin end up in the cave?

In the court of King Arthur[1] there lived 150 knights and 150 ladies. "Why not 150 married couples," the King contemplated one rainy afternoon, and action followed the thought. He asked the Royal Secret Agent (RSA) to draw up a diagram with all the 300 names, indicating bonds of mutual interest between lady and knight by a red line; and the lack thereof, by

Of course not even a tiny fraction could fit in the throne room, but Arthur wouldn't even wait till the room filled up. He dismissed Merlin's procedure ("obviously, you overlooked a case") and ordered him to come back with a solution the next day. Arthur's diaries reveal another thought that was on his mind: "The lifetime of the universe wouldn't suffice to check all that crud. That's how the old fox wants to fool me."

Merlin *knew* that he was right, and he knew also that Arthur was reasonable. All Merlin had to do was to convince him, *in five minutes*, that there was no solution.

Fortuitously, in the cafeteria he bumped into an unassuming character dressed in brand new blue jeans. An East Bloc visitor, the man humbly introduced himself as Dénes König, number one expert on perfect matchings. "Frobenius also claims this title," he added without bitterness. "Are you perhaps interested in my mini-max theory?" Having, at last, found a willing listener, the visiting scholar forgot his French fries and the free ketchup, and began a passionate lecture about bipartite graphs, maximum matchings