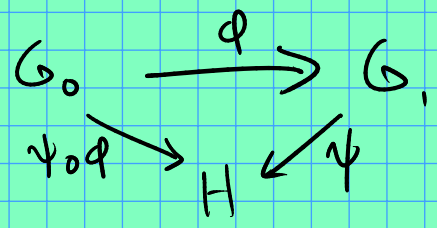
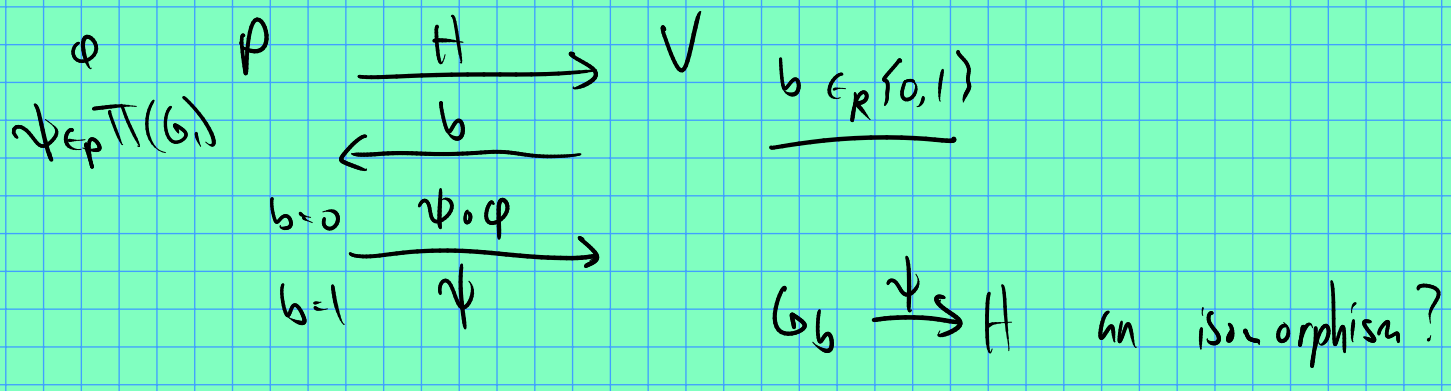
From last time: ZK proof for Graph isomorphism.



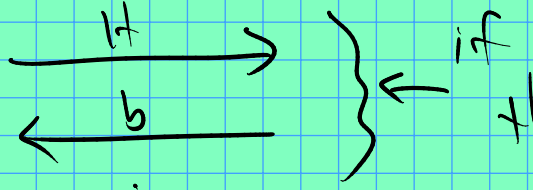


Last time we saw how to simulate an honest verifier.

What if V doesn't follow the protocol?!

Say V^* chooses $b = \text{L.S.B.}(AES_K(H))$. Then what?

Old transcripts might be statistically far from these!



Note: makes sense for simulator to have access to V^* (and any secrets it holds).

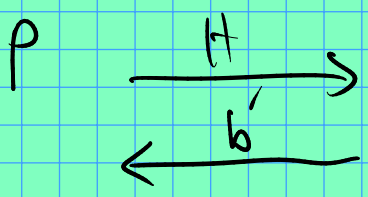
Solution: just guess $b \in_R \{0,1\}$! if we're wrong, $b \neq V^*(H)$, then start over (fresh bit b, new isomorphic copy $H \simeq G_b$)

Expected # of tries? only 2. why?

H has no information about our bit b!

Simulation: ① $b \in_R \{0,1\}$. ② Choose $\psi \in_R \Pi(G_b)$.

③ Set $H = \psi(G_b)$.



if $b' \neq b$, start over @ ①

b, b' independent, so $b = b'$ w/ prob $1/2$

(if $G_0 \simeq G_1$)

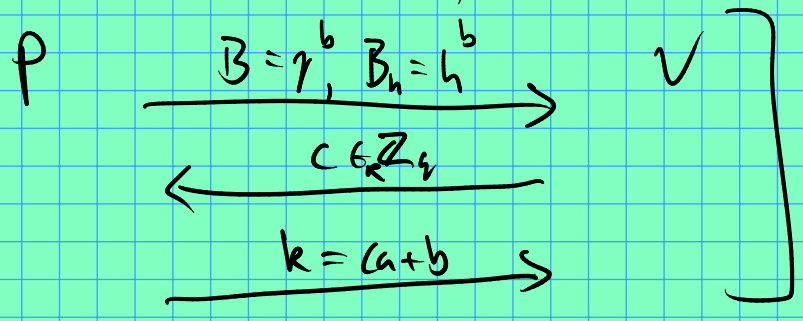
$\Pi(G_0) = \Pi(G_1)$

H H

Analysis: not hard to show transcripts produced in this way will be identically distributed to real transcripts of $P \leftrightarrow V^*$

(See Ahi + Rafael, pdf-page 135)
(Section 4.6)

Note: not clear how to do this for the ZK protocol for equality of discrete logs!



Cook-Levin Theorem? Showed existence of NP complete problems.

Idea: Show that any efficient witness verification machine can be modeled/simulated by a boolean formula.

(reminder: Languages $L \subseteq \{0,1\}^*$ have the property that
$\exists$ an (efficient / poly time) verification algo V s.t.
$\forall x \in L$, $\exists\ w_x$ s.t.
$V(x, w_x) = 1$.
Further, $\forall x \notin L$, $\forall w_x$,
$V(x, w_x) = 0$.)

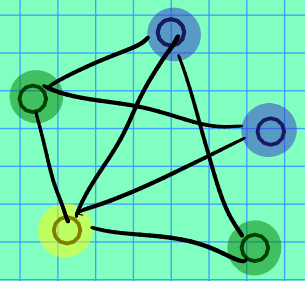
Zero Knowledge Proofs for NP. (Boaz + Rafael Sec. 4.7)

Idea: ① demonstrate a ZK proof for an NP complete problem L.
② for any other NP language L', to prove $x' \in L'$,
prepare instance $x \in L$ and show it is in L.

$$x = r(x')$$

$r(x') \in L \iff x' \in L'$

For ①, we'll use the Graph 3-Coloring problem.

$G =$ 

For a simple graph G,
can you assign each vertex i
a color $c_i \in \{0, 1, 2\}$
s.t. $(i,j) \in E(G)$
$\Rightarrow c_i \neq c_j$?

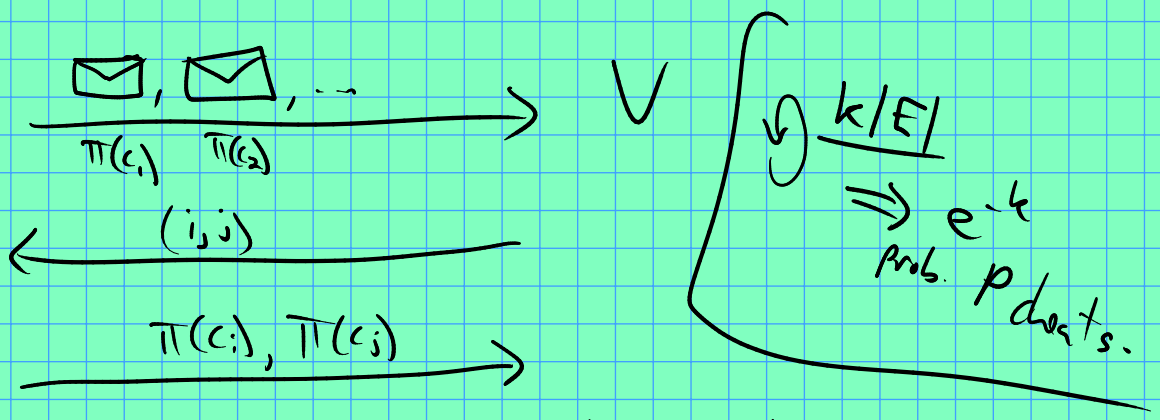
Idea for ZK proof for Graph 3-colorability:

P chooses random permutation of the colors, π.
Then commits to $(i, \pi(c_i))$ for each vertex $i \in G$.

V chooses a random edge $(i,j) \in E(G)$ and sends to P.

P reveals $\pi(c_i)$, $\pi(c_j)$.

V accepts $\Leftrightarrow \pi(c_i) \neq \pi(c_j)$.



If graph is not 3 colorable, what is the probability that P can cheat V? (in terms of $|E| = \#$ of edges in G)

Let's say P only has to have one edge in violation.
So only 1 of $|E|$ edges have the same color vertexes.

Hence $\Pr[P \text{ cheats}] = 1 - \frac{1}{|E|}$

$1 + x \approx e^x$. in fact, $1 + x \leq e^x$.
(for small x)

So, $\left(1 - \frac{1}{|E|}\right) \leq e^{-1/|E|}$.

Hence if we repeat protocol above some # of times proportional to $|E|$, we're fine!

$\Pr[P \text{ cheats in } k|E| \text{ consecutive rounds}]$

$$= \left(1 - \frac{1}{|E|}\right)^{k|E|} \leq e^{-k|E|/|E|} = e^{-k}.$$

Completeness is ... straightforward. ✓

Soundness? Sure, if we repeat $k|E|$ rounds,

$$\Pr[P \text{ cheats}] \leq e^{-k}.$$

ZK? Yes. See graph isomorphism proof for basic idea. Details in Abhi + Rafael if you want them.

Applications

Can "enforce" honest behavior for a wide range of algorithms / computations!

Can simplify protocol design: design for "honest but curious" participants, and then enforce honest behavior.

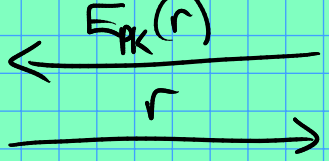
Think about threshold decryption: we introduced ZK proofs to enforce honest behavior (using the Pedersen proof for $=$ of dlogs (dss).

Turns out this is fairly general.

Example: authentication using RSA public key.

$PK = n, e$

p, q
$pq = n$



What if V doesn't know r?! (say message was actually encrypted email for P!)

Alternative: use ZK for P to prove it knows p, q s.t. $pq = n = PK$.

Then V learns nothing except that the Prover really does know the secret key for PK.

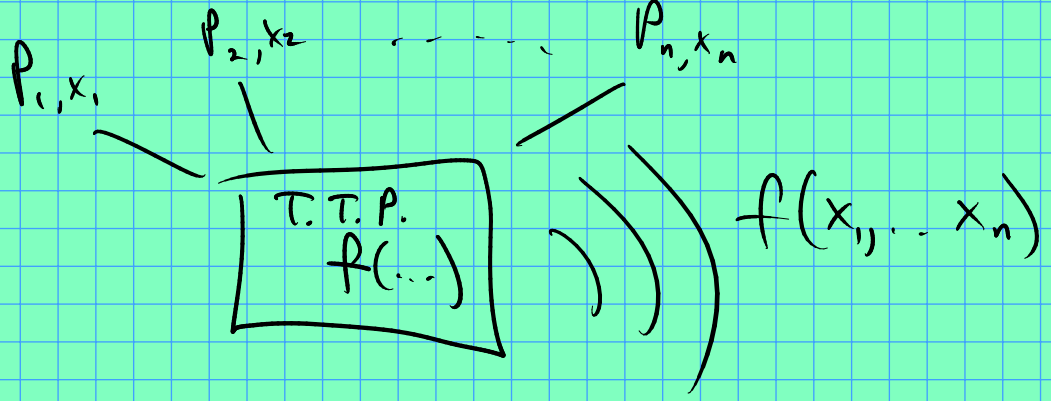
Cool things we didn't get to cover:

Secure Multiparty computation:

compute a function $f(x_1, \ldots, x_n)$ on inputs x_i held by party P_i, w/o revealing the inputs!

Example: Yao's Millionaires Problem.

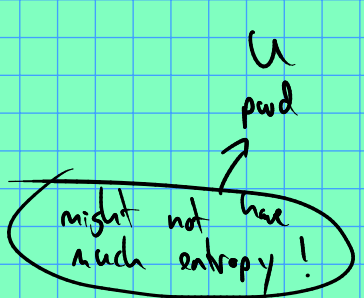
n_1 P_1 $n_1 \stackrel{?}{\leq} n_2$ P_2 n_2

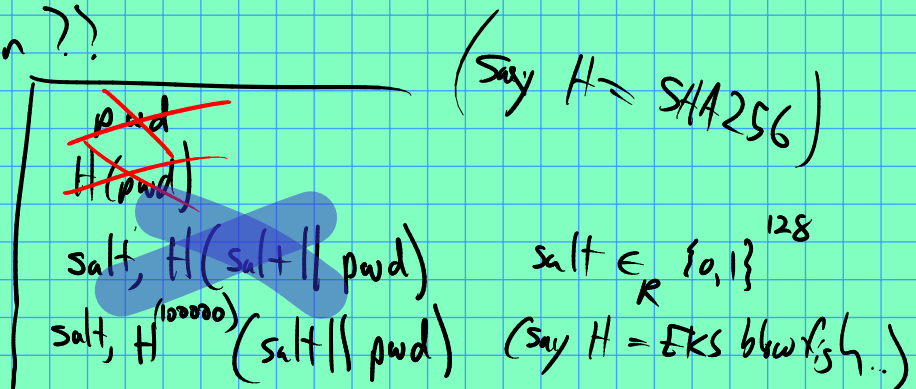


Possible without the T. T. P.!

Biometric Authentication / Fuzzy Sketches:

First: password authentication?!





for $x \in$ ⟨dictionary / list of common passwords⟩
compute $H(x)$.

Note: Salt prevents offline attack (pre computation is useless)

What about biometrics (e.g. finger print reader)?
Issue: scan will have small variations each time!
"Cute" trick using ECC can resolve this
for some metrics. See "Fuzzy Sketches".