Zero-Knowledge Proofs.

Informally: want verifier to learn nothing else than $x \in L$.
How do we make this formal?
We'll ask that some other TM as powerful as verifier can generate same transcript.

def. L has $2k$ proof in $\exists P, V$ like $IP[k]$ st (P unlimited, V random poly-time)

$\to x \in L \to \Pr[a_{2k+1} = 1] > 2/3$
$\to x \notin L \to \forall P' \ \Pr[a_{2k+1} = 0] > 2/3$
$\to \forall V' \exists S'$ st $\forall x \in L$ a'_{2k+1} identically distr. to $S'(x)$.
(a'_{2k+1} with V'.)

S' randomized TM either running in expected poly time or in strict poly time and allowed to answer "Don't Know" with small prob.

obs V' is allowed to output whichever function of the transcript it wants, not necessarily $x \in L$, but behaves like V for $a_1 \dots a_{k-1}$.

can also ask that a_{2k+1} and $S'(x)$ are statistically close or are computationally indistinguishable.

e.g. zero-k protocol for GI.

V: say nothing.
P: sample unif random $\pi \in S_n$, send $\pi(G_1) = H$.
V: sample $i \in \{1,2\}$ unif; send i.
P: send π^{-1} if $i=1$ or $\pi^{-1} \circ \tau^{-1}$ if $i=2$ ($= \sigma$)
V: check $\sigma(H) = G_i$. ($\tau : G_1 \to G_2$.)

correctness ok.

why is this $2k$?

consider V'. output is $a_{2k+1}' = V(G_1, G_2, H, i, \sigma)$.
(fixed, unif., fixed st $\sigma(H) = G_i$.)

S' does: sample unif i'.
sample unif π and send $H = \pi(G_i)$ ← not G_1!
receive i.
if $i \neq i'$: abort; repeat
if $i = i'$: send π^{-1}; finish simulation of V'.

(in order to simulate V' we first need to give it some H, and use the same i that it says, o/w it may complain).
obs with prob $1/2$ we will choose the right i', hence we expect running time at most $T(V') \cdot \sum 1/2^k$.

let us go back to normal interactive proofs. notice that in protocol for graph non-iso it is very important that prover does not know which graph verifier picked, o/w could cheat.
iow: prover's randomness needs to stay private.
what can we do with public randomness?

def $AM[k]$: interactive proofs where Verifier only sends random messages (and is det. o/w).

usually $AM = AM[2]$; $MA = MA[2]$ if Prover sends first message.
(M = Merlin = Prover; A = Arthur = Verifier).

because $AM[k] = AM[2]$!
(nontrivial, we'll see).

we'll see how to do non-gi with public randomness.
in fact sth stronger is true:

th: (Goldwasser–Sipser) $IP[k] \subseteq AM[k+2]$.
intuition:
assume we have IP proto, let r be V's randomness, let S_x be accepting strings. if $x \in L$ then $|S_x| \geq \frac{2}{3} \cdot 2^m$, while if $x \notin L$ then $|S_x| \leq \frac{1}{3} \cdot 2^m$.
can we distinguish between these two cases?

Yes, if we can also check membership efficiently.

lemma: Let $S \subseteq \{0,1\}^n$ be a set st prover can certify that $x \in S$. Let $k \in \mathbb{N}$. There is an AM proto st
$|S| \geq k \Rightarrow V$ accepts w/pr. $\geq 2/3$
$|S| \leq k/2 \Rightarrow V$ rejects w/pr. $\geq 2/3$.

proof: use hashing to map S to a set of size $\approx 2k$.
if $|S| \geq k$ we'll cover maj. set; if $|S| \leq k/2$ not.

formally: $\mathcal{H} = \{h_i\}$, $h_i : \{0,1\}^n \to \{0,1\}^k$.

$\mathcal{H}$ pairwise indep. hash if $\forall x \neq x' \in \text{dom}$; $\forall y, y' \in \text{im}$;
$\Pr_{h_i \in \mathcal{H}}[h_i(x) = y \wedge h_i(x') = y'] = 2^{-2k}$.
claim: $\mathcal{H}$ exists. //
obs implies uniform hashing:
$\Pr_{h_i \in \mathcal{H}}[h_i(x) = y] = \sum_{y'} \Pr[\wedge] = 2^k \cdot 2^{-2k} = 2^{-k}$.
protocol: let $k = \lfloor \log(2k) \rfloor$ (i.e. $2^{k+1} < 2k \leq 2^k$).

V samples h and y.
P sends x st $h(x) = y$ and cert. that $x \in S$. (or $\not\exists$ if $\not\exists$).
V checks $h(x) = y$ and $x \in S$.