Last time: We started showing that UNSAT $\in$ IP.

By Arithmetization, we reduced the problem of checking if $\varphi \in$ UNSAT
(3-CNF)

to checking

$$\sum_{b_1, b_2, \ldots, b_n \in \{0,1\}} P_\varphi(b_1, b_2, \ldots, b_n) = 0$$

where $P_\varphi(x_1, x_2, \ldots, x_n)$

→ is a polynomial of deg $\leq 3m$ (m = # of clauses in φ), and

→ can be evaluated in time poly($|\varphi|$) on a given input.

The same protocol that we will now see will also help us solve a more general problem:

$\#SAT_D = \{ (\varphi, K) \mid \varphi$ a 3CNF formula with exactly K satisfying assignments $\}$

$[\varphi \in UNSAT \iff (\varphi, 0) \in \#SAT_D]$

$(\varphi, K) \in \#SAT_D$

$$\Updownarrow$$

$$\sum_{b_1, \ldots, b_n \in \{0,1\}} P_\varphi(b_1, \ldots, b_n) = K.$$

SUMCHECK protocol:

Want to check: $\sum_{b_1,\dots,b_n \in \{0,1\}} g(b_1,\dots,b_n) = K$

where g has degree $\leq d$ & can be evaluated efficiently.

→ If $n=1$, check $g(0)+g(1)=K$ & accept if so. Reject otherwise.

→ If $n>1$, let
$$h(x_1) = \sum_{b_2,\dots,b_n \in \{0,1\}} g(x_1,b_2,\dots,b_n)$$
(univariate polynomial of degree $\leq d$)

(i) Prover sends $s(x_1)$ which is <u>supposedly</u> $h(x_1)$.

(ii) Verifier checks that $s(0)+s(1)=K$ & rejects if this is not true.

(iii) Verifier chooses a random $a_1 \in \{1, \dots, M\}$ & sends to Prover, asking the Prover to show

$$(*): \sum_{b_2, \dots, b_n \in \{0,1\}} g(a_1, b_2, \dots, b_n) = s(a_1)$$

Note: $(*)$ is the same kind of claim about the polynomial

$$g(a_1, x_2, \dots, x_n)$$

which

→ has degree $\leq d$.

→ can also be evaluated in time poly($|\varphi|$).

→ depends on $n-1$ variables

Correctness:

Completeness: Assume that

$$\sum_{b_1,\ldots,b_n \in \{0,1\}} g(b_1, b_2, \ldots, b_n) = K$$

[i.e the prover's claim is true]

Then, if the prover sends

$$s(x_1) = h(x_1)$$

then the Verifier accepts in Step (i)

& recursively asks the prover to show

$$\sum_{b_2,\ldots,b_n \in \{0,1\}} g(a_1, b_2, \ldots, b_n) = h(a_1)$$

which is another true claim.

Continuing this way, an honest prover can convince the verifier with prob. 1.

<u>Soundness</u>: Now we assume that the prover's claim is false.

i.e $$\sum_{b_1, \ldots, b_n \in \{0,1\}} g(b_1, \ldots, b_n) \neq K.$$

This time, we want to show that no matter <u>what</u> the prover sends, the verifier rejects with good probability

<u>Lemma</u>: The verifier rejects with probability $\left(1 - \frac{d}{M}\right)^{n-1}$.

<u>Proof</u>: [Induction]

<u>n=1</u>: Easy because the verifier checks the claim directly.

$\underline{\underline{n>1}}$: If the prover sends

$$s(x_1) = h(x_1)$$

then the Verifier computes

$$s(0) + s(1) = h(0) + h(1)$$

$$= \sum_{b_1,\dots,b_n \in \{0,1\}} g(b_1, b_2, \dots, b_n) \neq K.$$

& immediately rejects with probability 1. ↳ (step (ii))

If the prover sends

$$s(x_1) \neq h(x_1)$$

such that $s(0) + s(1) = K$, then the Verifier does not reject in step (ii).

In step (iii), the verifier asks the prover to show

$$(*) \quad \underbrace{\sum_{b_2,\dots,b_n} g(a_1,b_2,\dots,b_n)}_{\text{same as } h(a_1)} = s(a_1)$$

$(*)$ is false as long as

$h(a_1) \neq s(a_1)$.

$$\Pr_{a_1}\left[h(a_1) - s(a_1) \neq 0\right] \geq 1 - \frac{d}{M} \quad \text{(I)}$$

because the two distinct polynomials h, s of degree $\leq d$ can agree at $\leq d$ points

[This is the base case of the DeMillo-Lipton-Schwartz-Zippel lemma]

Assuming $h(a_1) \neq s(a_1)$, the prover has to prove another false claim (*) this time with $n-1$ variables. By induction, the proba-bility the verifier rejects is $\geq$

$$\left(1-\frac{d}{M}\right)^{n-2} \longrightarrow (\mathrm{II})$$

Thus, the overall probability of rejection is

$$\geq \left(1-\frac{d}{M}\right) \cdot \left(1-\frac{d}{M}\right)^{n-2} = \left(1-\frac{d}{M}\right)^{n-1}$$

This proves the inductive step & hence the lemma.

So we have shown:

UNSAT & more generally.

$$\#SAT_D \in IP.$$

$\Rightarrow$ co-NP $\subseteq$ IP

A similar idea also works for PSPACE, starting with TQBF instead of UNSAT [one more trick needed :)]

Zero-Knowledge Proofs

So far, we have the situation of an unbounded Prover that wants to convince the Verifier of something. Verifier asks the prover for a certificate that the prover can compute.

But what if the prover does not want to reveal the certificate?

Relevant for Crypto!

Eg: certificate = password
& Prover is a user of a systems that requires Prover to authenticate.

Let L be a language in NP.

A Zero-Knowledge Proof for L is a pair of probabilistic poly time TMs P & V along with a protocol between them such that:

① Completeness: If $x \in L$ & P has a valid certificate y showing this, then $\Pr_{P,V}[V \text{ accepts}] \geq 2/3$.

② Soundness: If $x \notin L$, then even if P is replaced by an arbitrary computationally unbounded P^*

$$\Pr_{V}[V \text{ accepts}] \leq 1/3$$

(3) Perfect Zero-Knowledge: For any probabilistic TM V^* (possibly not V) there is a probabilistic TM S^* (simulator) s.t.

when $x \in L$ & P has a valid certificate y for this,

the output of $S^*(x)$ is

identically distributed to

the transcript of the protocol

between $P(x, y)$ & $V^*(x)$

Intuitive meaning of (3):

The Verifier learns nothing about y from the protocol even if the Verifier deviates from the prescribed strategy V, as long as the prover sticks to the prescribed strategy P.

Example: Graph Isomorphism (GI)

Input: G, H with vertex set $\{1, \dots, n\}$

Question: Is there a bijection

$$\pi : \{1, \dots, n\} \to \{1, \dots, n\}$$

that turns G into H?

Certificate: π.

We will show that GI has a Zero-Knowledge Proof.

Let $\mathcal{G}_0$ = set of graphs isomorphic to G

$\mathcal{G}_1$ = " to H

Recall: $G \cong H \Rightarrow \mathcal{G}_0 = \mathcal{G}_1 = \mathcal{G}$

(last time) $G \not\cong H \Rightarrow \mathcal{G}_0 \cap \mathcal{G}_1 = \emptyset$.

Protocol (with prescribed strategies)

P has input (G, H, π) where π shows $G \cong H$.

V has input (G, H)

(1) P picks a uniformly random permutation $\pi' : \{1, \ldots, n\} \to \{1, \ldots, n\}$

& applies it to H.

P then sends this graph R to V.

(2) V choose uniformly at random b $\{0, 1\}$ & sends to P;

asking P to show R is isomorphic to G (if $b=0$) or H (if $b=1$).

(3) P responds with π' if $b=1$ & $\pi \circ \pi'$ if $b=0$.

[Protocol continues on Next Page...]

(4) V accepts if & only if P's response is a valid bijection between R & G (if $b=0$) or R & H (if $b=1$).

Completeness: If $G \cong H$ & P has a valid isomorphism π & both P, V follow the protocol, then V accepts with probability 1.

Soundness: If $G \not\cong H$, then the prover must follow some other strategy P^*.

No matter what graph R is sent in step ①, R cannot be in both $\mathcal{G}_0$ & $\mathcal{G}_1$ [because they are dis-joint]

So with probability at least $1/2$, in step ③, V chooses b for which P^* has no valid response.

Zero Knowledge: First assume that both Prover & Verifier follow the prescribed strategies. Then the output of the protocol is

$$(R, b, \pi')$$

where

→ R is a random graph in G.

→ b is a random but independent of R.

→ $b = 0 \Rightarrow \pi'$ is an isomorphism between R & G

$b = 1 \Rightarrow \pi'$ is an isomorphism between R & H.

Simulator S

→ Choose a random bit $b \in \{0,1\}$

→ If $b=0$, choose a random bijection

$\pi' : \{1, \ldots, n\} \to \{1, \ldots, n\}$

& apply it to G to get R

→ If $b=1$, do the same with H

Output (R, b, π').

Now, what if Verifier deviates
& follows strategy V^*?

S^*

→ Choose (R, b, π') as before.
→ Run Verifier V^* on R & get bit b'.
→ If $b' = b$, output (R, b, π').
→ Otherwise, repeat.

Why does this work?
No matter what the value of b, R is always uniformly distributed over G. This means that R is

independent of b.

Hence, $V^*(R) = b$ with prob $\frac{1}{2}$.

Repeating this many times gives an algorithm that succeeds with high probability.

In the end, the simulator S^* is not exactly a poly-time algo but rather an expected poly-time algorithm.

[I lied (a bit) in the definition :)]