Focus thus far in the course: protecting communication (e.g., message confidentiality and message integrity)

Remainder of course: protecting computations

Zero-knowledge: a defining idea at the heart of theoretical cryptography

↳ Idea will seem very counter-intuitive, but surprisingly powerful ← with surprising implications (DSA/ECDSA signatures based on ZK!)

↳ Showcases the importance and power of definitions (e.g., "What does it mean to know something?")

We begin by introducing the notion of a "proof system"

- Goal: A prover wants to convince a verifier that some statement is true

  e.g., "This Sudoku puzzle has a unique solution"
  "The number N is a product of two prime numbers p and q"
  "I know the discrete log of h base g"

  } these are all examples of statements

We model this as follows:

the verifier is assumed to be an efficient algorithm

prover (x) $\xrightarrow{\pi}$ verifier (x) → $b \in \{0,1\}$ — given statement x and proof π, verifier decides whether to accept or reject

x: statement that the prover is trying to prove (known to both prover and verifier)

↳ We will write $\mathcal{L}$ to denote the set of true statements (called a "language")

π: the proof of x

Properties we care about:

- Completeness: Honest prover should be able to convince honest verifier of true statements

$$\forall x \in \mathcal{L}: \Pr[\pi \leftarrow P(x) : V(x,\pi) = 1] = 1$$

- Soundness: Dishonest prover cannot convince honest verifier of false statement

$$\forall x \notin \mathcal{L}: \Pr[\pi \leftarrow P(x) : V(x,\pi) = 1] < 1/3$$

[Could relax requirement to allow for some error]

Important: We are not restricting to efficient provers (for now)

Typically, proofs are "one-shot" (i.e., single message from prover to verifier) and the verifier's decision algorithm is deterministic

↳ Languages with these types of proof systems precisely coincide with NP (proof of statement x is to send NP witness w)

Recall that NP is the class of languages where there is a deterministic solution-checker:

$$\mathcal{L} \in NP \iff \exists \text{ efficiently-computable relation } R \text{ s.t.}$$

$$x \in \mathcal{L} \iff \exists w \in \{0,1\}^{|x|} : R(x,w) = 1$$

statement (x), language ($\mathcal{L}$), witness (w), NP relation (R)

Proof system for NP:

prover (x) $\xrightarrow{w}$ verifier (x)

accept if $R(x,w) = 1$

Perfect completeness + soundness

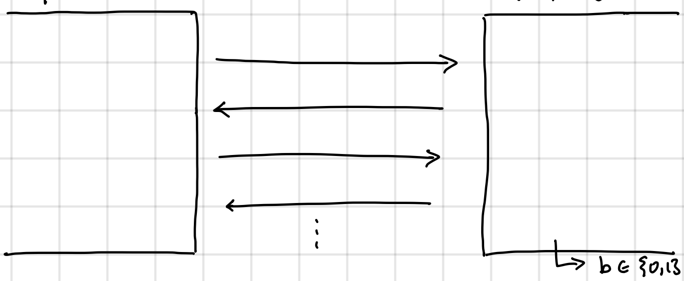
Going beyond NP: we augment the model as follows

- Add randomness: the verifier can be a randomized algorithm
- Add interaction: verifier can ask "questions" to the prover

Interactive proof systems [Goldwasser-Micali-Rackoff]:

prover (x) ← may be inefficient

efficient and randomized → verifier (x)



Verifier randomness is critical. Otherwise, class of languages that can be recognized collapses to NP. (See HW5).

Interactive proof should satisfy completeness + soundness (as defined earlier)

We define IP[k] to denote class of languages where there is an interactive proof with k messages.
We write $IP = IP[poly(n)]$ where n is the statement length
(i.e., IP is the class of languages with an interactive proof with polynomially-many rounds)

Special case: Arthur (verifier) - Merlin (prover) proofs: verifier randomness is public and known to the prover ← interactive proofs: verifier can rely on secret randomness

$AM[k]$: AM proof with k messages, class $AM = AM[2]$ (two-message public-coin proofs)
for constant k, $AM[k] = AM = AM[2]$ (constant message = 2 message)
↳ equivalent to BP·NP (class of languages with randomized reduction from 3-SAT)
$B \leq_r C \iff \exists$ efficient M: $\forall x : \Pr[C(M(x)) = B(x)] \geq 2/3$
↳ randomized reduction

Theorem (Goldwasser-Sipser): For every $k \in \mathbb{N}$, $IP[k] \subseteq AM[k+2]$
(Any private-coin interactive proof can be simulated by a public-coin interactive proof with two extra rounds)

What is the power of IP?

- For constant number of messages, seems comparable to NP ($IP[k]$ collapses to AM (= BP·NP) for constant $k \in \mathbb{N}$)
- Going from constant to polynomial number of rounds is 'significant'

Theorem. (Lund-Fortnow-Karloff-Nisan '90, Shamir '90) $IP = PSPACE$. ← the set of languages that can be checked in polynomial space

Proof (Idea). We will prove a weaker statement which illustrates all of the main techniques of the proof.
Let 3col be the graph 3-coloring problem
- Given graph $G = (V, E)$, can we color the nodes so two adjacent nodes have different colors? [NP complete]

Let #3col be the problem of counting the number of 3-colorings of a graph.
We will show $\#3col \in IP$ (this implies for instance that $coNP \subseteq IP$ since #3col is coNP-hard)
coNP: problems where No instances can be efficiently checked
[if number of colorings is 0, then G is a No instance]
↳ #3col is #P-complete (Toda's theorem: $PH \subseteq P^{\#P}$)
↳ counting the number of witnesses to a polynomial-time relation

Step 1 (Arithmetization): We will construct a polynomial P_G that outputs 1 on a valid coloring and 0 otherwise.
- Let $G = (V, E)$ be the graph. For each vertex $u \in V$, let $x_u \in \{0,1,2\}$ be the associated color.
$|V| = n \quad |E| = m$

- Consider the polynomial

$$\hat{P}_G(x_1,\dots,x_n) = \prod_{(u,v)\in E} (x_u - x_v)$$

Suppose $(x_1,\dots,x_n)$ is an invalid coloring. Then, for some $(u,v)\in E$, $x_u = x_v$, and $P_G(x_1,\dots,x_n) = 0$.
Suppose $(x_1,\dots,x_n)$ is a valid coloring. Then, for all $(u,v)\in E$, $x_u - x_v \in \{-2,-1,1,2\}$.

Define $f:\mathbb{R}\to\mathbb{R}$ be a polynomial where $f(0)=0$ and $f(-2)=f(-1)=f(1)=f(2)=1$.
e.g., $f(x) = \frac{5}{4}x^2 - \frac{1}{4}x^4$ satisfies the desired properties

- Define $P_G(x_1,\dots,x_n) = \prod_{(u,v)\in E} f(x_u - x_v)$
  - For an invalid coloring: $P_G(x_1,\dots,x_n) = 0$
  - For a valid coloring: $P_G(x_1,\dots,x_n) = 1$

$\Longrightarrow$ Number of valid colorings: $\sum_{x_1\in\{0,1,2\}}\sum_{x_2\in\{0,1,2\}}\cdots\sum_{x_n\in\{0,1,2\}} P_G(x_1,\dots,x_n)$

Goal: interactive proof to check sum of this polynomial

$$K = \sum_{x_1\in\{0,1,2\}}\sum_{x_2\in\{0,1,2\}}\cdots\sum_{x_n\in\{0,1,2\}} P_G(x_1,\dots,x_n)$$

Step 2 (Sumcheck protocol): Instead of working over $\mathbb{R}$, we will work over $\mathbb{Z}_p$ (for prime p)
[if $p > 3^n$, this is guaranteed to be correct]

Approach: Prover first computes polynomial

$$P_1(x) = \sum_{x_2\in\{0,1,2\}}\sum_{x_3\in\{0,1,2\}}\cdots\sum_{x_n\in\{0,1,2\}}\prod_{(u,v)\in E} f(x_u - x_v) \qquad (*)$$

- This is a polynomial with degree $d \le 4m$ since $\deg(f) = 4$
- Polynomial is univariate so can be described by at most $4m+1$ coefficients
- Prover can send P_1 to verifier ($4m+1$ coefficients) and verifier can check that
  $K = \sum_{x_1\in\{0,1,2\}} P_1(x_1)$

This can be checked efficiently! But what if prover cheats and sends $\tilde{P}_1$ that does not satisfy $(*)$.
- Verifier needs to check validity of $\tilde{P}_1$.

Idea: sample $r \xleftarrow{R} \mathbb{Z}_p$ and ask prover to prove that

$$\tilde{P}_1(r_1) = \sum_{x_2\in\{0,1,2\}}\cdots\sum_{x_n\in\{0,1,2\}} P_G(r_1, x_2,\dots,x_n)$$

two observations: if $(*)$ holds, then this is another instance of sumcheck with one fewer variable
if $(*)$ does not hold, then this statement is false unless r is a root of $\tilde{P}_1 - P_1$. This polynomial is not identically zero, so it has at most $\deg(\tilde{P}_1 - P_1) \le 4m$ roots.

$$\Pr[r_1 \xleftarrow{R} \mathbb{Z}_p : \tilde{P}_1(r_1) = P_1(r_1)] \le \frac{4m}{p}$$

$\therefore$ with prob. $1 - 4m/p$, prover now has to prove a false statement using sumcheck

Continue this process until we have a univariate polynomial:

$$\tilde{P}_{n-1}(r_{n-1}) = \sum_{x_n\in\{0,1,2\}} P_G(r_1,\dots,r_{n-1},x_n)$$

this is a polynomial of degree $4m$ in 1 variable so the verifier can directly check it if the statement is false $\Rightarrow$ verifier always rejects

otherwise, verifier always accepts

Can now argue soundness inductively:

- for false statement on n variables: verifier rejects false statement w.p. at least $(1-4m/p)^n$
  - trivial for case where $n = 1$
  - for general case on n variables: $r_n \xleftarrow{R} \mathbb{Z}_p$ is not a root of $P_n - \tilde{P}_n$ w.p. $1 - 4m/p$, in which case prover must show false statement on $n-1$ variables

$\Rightarrow$ soundness $(1-4m/p)(1-4m/p)^{n-1} = (1-4m/p)^n$

Choose $p > 4mn$ so soundness holds with constant prob.

<u>Implication</u>: $\#3col \in IP$ so $coNP \subseteq IP$.

$\forall x_1 \exists x_2 \forall x_3 \cdots \exists x_n \underbrace{\phi}_{\text{Boolean formula}}(x_1, \ldots, x_n)$

Approach directly generalizes to the total quantifies Boolean formula (TQBF) problem which is complete for PSPACE

$\Rightarrow$ PSPACE $\subseteq$ IP $\Rightarrow$ IP = PSPACE

↳ arithmetize (with linearization), followed by sumcheck

Sumcheck protocols very useful for verifying <u>polynomial-time</u> computations with small communication

"interactive proofs for muggles" [Goldwasser-Kalai-Rothblum '08]

] key building block for succinct arguments and verifiable computation