Interactive proofs

Interaction

Let $f, g : (\{0,1\}^*)^+ \to \{0,1\}^*$. A k-round interaction of f and g on input $x \in \{0,1\}^*$ is the sequence $a_1, \ldots, a_k \in \{0,1\}^*$, where

$$
\begin{aligned}
a_1 &= f(x) \\
a_2 &= g(x, a_1) \\
a_3 &= f(x, a_1, a_2) \\
&\ldots\ldots\ldots
\end{aligned}
$$

The f-output of the interaction is $out_f^k\langle f, g\rangle(x) = f(x, a_1, \ldots, a_k)$. The g-output is $out_g^k\langle f, g\rangle(x) = g(x, a_1, \ldots, a_k)$.

The class dIP

A language L belongs to class dIP if

- exists a polynomially bounded $k : \mathbb{N} \to \mathbb{N}$, and
- exists poly-time (in first argument) algorithm $V : (\{0,1\}^*)^+ \to \{0,1\}^*$, such that
- exists function $P : (\{0,1\}^*)^+ \to \{0,1\}^*$, such that for all $x \in$ L
  - $out_V^{k(|x|)}\langle V, P\rangle(x) = 1$ **(Completeness)**
- for all functions $P : (\{0,1\}^*)^+ \to \{0,1\}^*$ and all $x \notin L$:
  - $out_V^{k(|x|)}\langle V, P\rangle(x) = 0$ **(Soundness)**.

Theorem. dIP = NP. **Proof.** certificate $\approx$ transcript

Randomized verifier

Let V be randomized: $a_k = V(x, \alpha, a_1, \ldots, a_{k-1})$.

A language L belongs to class IP$[\lambda n.k(n)]$ if

- exists a poly-time (in first argument) randomized algorithm $V : (\{0,1\}^*)^+ \to \{0,1\}^*$, such that
- exists function $P : (\{0,1\}^*)^+ \to \{0,1\}^*$, such that for all $x \in$ L
  - $\Pr[out_V^{k(|x|)}\langle V, P\rangle(x) = 1] \geq 2/3$ **(Completeness)**
- for all functions $P : (\{0,1\}^*)^+ \to \{0,1\}^*$ and all $x \notin L$:
  - $\Pr[out_V^{k(|x|)}\langle V, P\rangle(x) = 1] \leq 1/3$ **(Soundness)**.

Define IP $= \bigcup_{c\in\mathbb{N}}$ IP$[\lambda n.n^c]$.

Example: graph non-isomorphism is in IP$[2]$

- Input: two graphs G_1, G_2. Claim: $G_1 \not\cong G_2$.
- Protocol:
  - V randomly picks $i \in_R \{1, 2\}$.
  - V sends to P a random permutation of G_i.
  - P responds with i'. V checks that $i = i'$.

Arthur-Merlin protocols

A language L belongs to class AM$[\lambda n.k(n)]$ if

- $L \in$ IP$[\lambda n.k(n)]$ and the verifier V works as follows:
  - whenever V has to send a message to P, it generates a random bit-string α and sends it.
  - V does not generate any more random bits.

(public-coin protocols)

Define AM $=$ AM$[2]$.

Class MA defined similarly, but prover sends the first message.

Graph non-isomorphism with public coins

Let G_1 and G_2 have n vertices. Consider the set

$$S = \{\langle H, \pi \rangle \mid (H \cong G_1 \vee H \cong G_2) \wedge \pi(H) = H\}$$

- If $G_1 \cong G_2$ then $|S| = n!$. If $G_1 \ncong G_2$ then $|S| = 2n!$.
- **Set lower bound protocol** for S and a number K has the following result:
  - if $|S| \geq K$ then verifier accepts with high probability.
  - if $|S| \leq K/2$ then verifier rejects with high probability.
- Use this protocol for $(S, 2n!)$.

Set lower bound protocol for (S, K)

- Let $S \subseteq \{0,1\}^m$. Let k be such that $2^{k-2} \leq K < 2^{k-1}$.
- Let $\mathcal{H}_{m,k}$ be a pairwise independent hash function family from $\{0,1\}^m$ to $\{0,1\}^k$.
  - For any $x, x' \in \{0,1\}^m$: distribution of $h(x)\|h(x')$ is uniform, when $h \leftarrow \mathcal{H}_{m,k}$.
- Verifier randomly picks and sends $h \leftarrow \mathcal{H}_{m,k}$ and $y \in_R \{0,1\}^k$.
- Prover responds with $x \in S$ (and with x's certificate), such that $h(x) = y$. Verifier checks.

Let $p = K/2^k$. If $|S| \leq K/2$ then $|h(S)| \leq (p/2) \cdot 2^k$ and the probability of existence of x is at most $p/2$.
If $|S| \geq K$ then...

If $K \leq |S| \leq 2^{k-1}$ then...

Let E_x be the event $h(x) = y$. All probabilities are wrt. choice of h and y.

$$\Pr[\exists x \in S : h(x) = y] = \Pr[\bigcup_{x \in S} E_x] \overset{(*)}{\geq} \sum_{x \in S} \Pr[E_x] - \frac{1}{2} \sum_{x \neq x' \in S} \Pr[E_x \cap E_{x'}]$$
$$\geq \frac{|S|}{2^k} - \frac{1}{2} \cdot \frac{|S|^2}{2^{2k}} \geq \frac{3}{4} p \ .$$

(*) Induction over $|S|$.

We see that the acceptance probabilities for $|S| \geq K$ and $|S| \leq K/2$ are significantly different. Repetition makes the difference as large as necessary.

Generalization: $\mathsf{IP}[k] \subseteq \mathsf{AM}[k+2]$.

Perfect completeness

Theorem. If $L \in$ AM then exists verifier V, such that

- if $x \in L$ then exists prover P, such that $\Pr[out_V\langle V, P\rangle(x) = 1] = 1$;
- if $x \notin L$ then for all provers P, $\Pr[out_V\langle V, P\rangle(x) = 1] \leq 1/3$.

Proof is similar to BPP $\subseteq \Sigma_2^p$. Let V' be a verifier with exponentially small completeness and soundness errors. Then

- exist bitstrings $u_1, \ldots, u_k$, such that for each α and for each $x \in L$:
- V' accepts with at least one random string $\alpha \oplus u_1, \ldots, \alpha \oplus u_k$.

Corollaries of perfect completeness

Theorem. AM $\subseteq \Pi_2^p$.

Theorem. If graph isomorphism is NP-complete then $\Sigma_2^p = \Pi_2^p$.

Proof. Let

- f be poly-time reduction from Boolean formulas to pairs of graphs, such that $\forall \vec{x} \varphi(\vec{x}) \equiv$ true iff $f(\varphi)_1 \ncong f(\varphi)_2$;
- V be a verifier for graph non-isomorphism with public coins, perfect completeness, and soundness error better than 2^{-n}.

$\exists \vec{x} \forall \vec{y} \varphi(\vec{x}, \vec{y}) \equiv$ true iff $\forall \alpha \, \exists \vec{x} \, \exists msg_P : V(f(\varphi(\vec{x}, \cdot)), \alpha, msg_P)$

MA $\subseteq$ AM

Theorem. MA $\subseteq$ AM.

- **Proof.** Let L have MA-proof where verifier has perfect completeness and $2^{-p(|x|)-1}$ soundness error.
  - $p(|x|)$ — length of Merlin's message on input x.
  - Soundness error reduced by Arthur checking Merlin's claim multiple times.
- Then V is also an AM-verifier for L.
  - Completeness — still perfect.
  - Soundness — error at most $1/2$.

Corollary AM$[k+1] \subseteq$ AM$[k]$. The finite levels of AM collapse.

IP $\subseteq$ AM

- Let $L \in \mathsf{IP}[T]$.
- Let V be such, that for $x \in \{0,1\}^n$,
  - exactly $2 \cdot T(n)$ messages are sent; each having length $m(n)$.
  - exactly $\ell(n)$ random bits are used by V.
- Let P be a suitable prover.
  - Let $V \| P$ have exponentially small error probability.
- Let $w \in \{0,1\}^n$ be the string that V and P consider.

Some notation

- x_i — verifier's messages. y_i — prover's messages.
- A partial transcript: $t = (x_1, y_1, x_2, y_2, \ldots)$, ending with some x_i or y_i.
- $\mathrm{ACC}(w, t)$ — the set of all such $r \in \{0,1\}^{\ell(n)}$, where the transcript of $V(r) \| P$ on input w starts with t and ends with V accepting.

Arthur's protocol

- **Round 0.** Receive $b_1 \in \mathbb{N}$ from Merlin.
- **Round** i **(**$1 \leq i \leq T(n)$**).** Select (and send to Merlin)
  - random linear functions $h_1, \ldots, h_{\ell(n)} : \{0,1\}^{m(n)} \to \{0,1\}^{b_i+1}$
  - random strings $z_1, \ldots, z_{\ell(n)^2} \in \{0,1\}^{b_i+1}$
- Receive $b_{i+1} \in \mathbb{N}$ and $x_i, y_i \in \{0,1\}^{m(n)}$. Check that $\exists j, k : h_j(x_i) = z_k$.
- **Round** $i = T(n) + 1$**.** Select (and send to Merlin)
  - random linear functions $h_1, \ldots, h_{\ell(n)} : \{0,1\}^{\ell(n)} \to \{0,1\}^{b_i+1}$
  - random strings $z_1, \ldots, z_{\ell(n)^2} \in \{0,1\}^{b_i+1}$
- Receive $r \in \{0,1\}^{\ell(n)}$ Check that $\exists j, k : h_j(r) = z_k$.
- Check that $x_1, y_1, \ldots, x_{T(n)}, y_{T(n)}$ is an accepting transcript for $V(w; r)$.
- Check that $\sum_{i=1}^{T(n)} b_i \geq \ell(n) - T(n) \log \ell(n)$.

Merlin's protocol ($w \in L$)

- **Finding** b_i
  - Let t be current transcript. Let $\mathcal{ACC}_d = \{x \mid 2^{d-1} < |\mathrm{ACC}(w,t,x)| \leq 2^d\}$.
  - Let $d_{\max} = \arg\ \max_d \bigcup_{x \in \mathcal{ACC}_d} \mathrm{ACC}(w,t,x)$.
  - Let $b_i = 2 + \lceil \log d_{\max} \rceil$.
  - $b_{T(n)+1}$ comes directly from the size of $\mathrm{ACC}(w,t)$.
- **Finding** x_i, y_i
  - Pick x_i from the set $\mathcal{ACC}_{d_{\max}}$ defined before.
  - Let $y_i = P(w,t,x_i)$.

During the protocol, $|\mathrm{ACC}(w,t,x_i,y_i)| \approx |\mathrm{ACC}(w,t)|/2^{b_i}$.

IP $\subseteq$ PSPACE

- Consider all possible executions of the verifier V on input x, when ranging over
  - all possible inputs from the prover;
  - all possible randomness strings.
- Compute the acceptance probability when the prover uses its best strategy.
- Note that the prover can do the same computations and pick the best messages to send to the verifier.

Corollary. We can assume that our provers work in polynomial space.

Arithmetization

For a Boolean formula $\varphi(x_1, \ldots, x_n)$ define a n-variable polynomial P_φ as follows:

$$\begin{aligned} P_{x_i} &= x_i \\ P_{\neg\varphi} &= 1 - P_\varphi \\ P_{\varphi_1 \wedge \varphi_2} &= P_{\varphi_1} \cdot P_{\varphi_2} \\ P_{\varphi_1 \vee \varphi_2} &= 1 - (1 - P_{\varphi_1})(1 - P_{\varphi_2}) \end{aligned}$$

- If $x_1, \ldots, x_n \in \{0, 1\}$ then $\varphi(x_1, \ldots, x_n) = P_\varphi(x_1, \ldots, x_n)$;
- P_φ is easy to evaluate from $x_1, \ldots, x_n \in R$ and φ.
  - R — some ring.
- Degree of $O(P_\varphi)$ is $O(|\varphi|)$.

PSPACE $\subseteq$ IP

Consider the problem
$\#\mathrm{SAT}_D = \{\langle \varphi, K \rangle \mid \varphi \text{ has exactly } K \text{ satisfying valuations}\}$.
Lemma. $\#\mathrm{SAT}_D \in \mathrm{IP}$.

Proof. $\#\varphi = \sum_{b_1 \in \{0,1\}} \cdots \sum_{b_n \in \{0,1\}} P_\varphi(b_1, \ldots, b_n) \leq 2^n$.

- Let p be a $(n+1)$-bit prime. Everything below is *mod* p.
- For a polynomial $g(x_1, \ldots, x_m)$, denote
  $proj_g(x) = \sum_{b_2 \in \{0,1\}} \cdots \sum_{b_m \in \{0,1\}} g(x, b_2, \ldots, b_m)$
- Given m-variable d-degree polynomial g and number K, the **Sumcheck**-protocol verifies that

$$K = \sum_{b_1 \in \{0,1\}} \cdots \sum_{b_m \in \{0,1\}} g(b_1, \ldots, b_m) \ .$$

Sumcheck

- If $m = 1$ then V checks whether $g(0) + g(1) = K$
- If $m > 1$ then P sends a polynomial $s(x)$ to V.
  - presumably $s = proj_g$
- V checks that $s(0) + s(1) = K$.
- V picks a random number $a \in \mathbb{Z}_p$ and sends it to P.
- **Sumcheck**: $s(a) = \sum_{b_2 \in \{0,1\}} \cdots \sum_{b_n \in \{0,1\}} g(a, b_2, \ldots, b_m)$

Claim. Sumcheck has perfect completeness and soundness at least $(1 - d/p)^n$.

Proof. Induction over the number of variables.

TQBF as polynomial

- $\exists x_1 \forall x_2 \cdots \varphi(x_1, x_2, \ldots)$ is true iff $\sum_{b_1 \in \{0,1\}} \prod_{b_2 \in \{0,1\}} \cdots P_\varphi(b_1, b_2, \ldots) > 0$.
- This number can be at most 2^{2^n}. We'll compute it *modulo* some p.
  - Prover picks p, verifier checks primality.
- We could also do **Prodcheck**, but degrees of polynomials are too large.
- We rewrite $\exists \forall \exists \cdots \varphi$, such that the polynomials s will have low degree.
- We get a formula that is not in prenex form, but that's OK.
  - prenex — all quantifiers in the beginning.

Rewriting quantified φ

- A TQ formula is simple if at most one $\forall$ is between each variable and its binding place.
- $Qx \cdots \forall y \psi(x) \equiv Qx \cdots \forall y \exists x'(x = x' \wedge \psi(x'))$.
- Any formula can be made simple by at most squaring its number of variables.
- Let negations be only in front of variables.
- Change arithmetization: $P_{\varphi_1 \vee \varphi_2} = P_{\varphi_1} + P_{\varphi_2}$.
- **Lemma.** For any values of free variables of φ: $P_\varphi(b_1, \ldots, b_k) > 0$ iff $\varphi(b_1, \ldots, b_k)$ is true.
- The protocol for verifying $P_\varphi \stackrel{?}{=} K$:

The protocol

- If $\varphi \equiv \varphi_1 \operatorname{op} \varphi_2$ then P sends both values $K_1 = P_{\varphi_1}$ and $K_2 = P_{\varphi_2}$. Verifier checks that $K = K_1 \otimes K_2$ and then run the protocol for $K_1 \stackrel{?}{=} P_{\varphi_i}$.
- If $\varphi \equiv \exists x \varphi'$ or $\varphi \equiv \forall x \varphi'$ then P sends the polynomial $s(x) = P_{\varphi'}(x)$ to V.
  - Its degree is at most $2|\varphi'|$.
- V verifies $s(0) \otimes s(1) \stackrel{?}{=} K$. Then pick number a and run the protocol for $P_{\varphi'}(a) \stackrel{?}{=} s(a)$.

A relationship between different classes

Theorem. If PSPACE $\subseteq$ P/$poly$ then PSPACE $=$ MA.

Proof. The prover for TQBF is in PSPACE. Protocol:

- In round 1, P sends to V the circuit for TQBF prover.
- V does the "interactive" proof with the help of this circuit.