Foundations of Probabilistic Proofs

A course by **Alessandro Chiesa**

Lecture 01

Intro to IPs

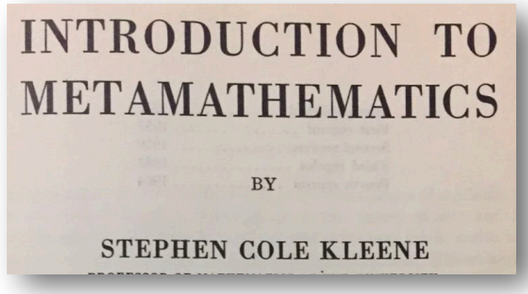
What are Mathematical Proofs?

One answer: an approximation of understanding.

"A proof is whatever convinces me." (Shimon Even, 1978) [Stories about Shimon Even by Oded Goldreich]

Logic formalizes the notion of "proof" so it can be studied using mathematics itself.

In particular this leads to METAMATHEMATICS.



A formal system is a tuple (alphabet, grammar, axioms, inference rules).

A (true) theorem is a sentence that has a (valid) proof:

a list of sentences whose last sentence is the theorem such that every sentence is an axiom, an assumption, or derived from prior sentences.

a logical derivation from axioms & assumptions

Fundamental question: HILBERT'S ENTSCHEIDUNGSPROBLEM

is there a procedure to determine if a sentence is a theorem?

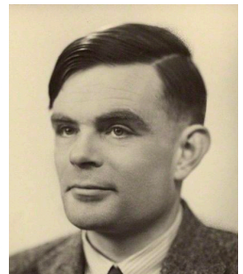
NO no such procedure exists

1931: Gödel
- IT1: every strong enough formal system is not complete (∀ sentence x, x or $\bar{x}$ has a proof)
- IT2: every strong enough formal system cannot prove its own consistency (∀ sentence x, at most one of x or $\bar{x}$ has a proof)

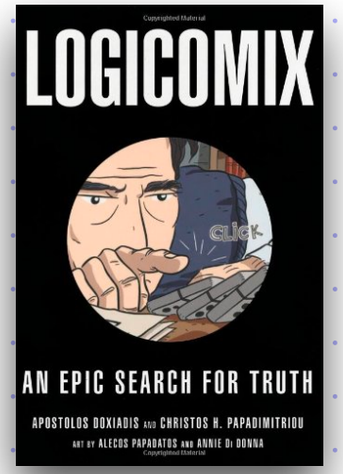
1935: Church (via λ-calculus)

1936: Turing (via Turing machines)





Proofs and Computation



Mathematical proofs were implicitly always about computation.

Formulating (and answering) the Entscheidungsproblem made this explicit.

Computation is formalized via Turing machines (and other equivalent models).

This enables the study of COMPUTABILITY THEORY and COMPLEXITY THEORY.

Undecidability of a language rules out any mathematical proof.

Languages with NONDETERMINISTIC WITNESSES correspond to theorems with mathematical proofs:

A language L is in $\mathrm{NTIME}[T]$ iff $\exists$ verifier machine V that runs in time $T(|x|)$ s.t.

- completeness: $\forall$ theorem instance $x \in L$ $\exists$ $T(|x|)$-size proof witness w π $V(x,\pi)=1$ $M(x,w)=1$
- soundness: $\forall$ theorem instance $x \notin L$ $\forall$ $T(|x|)$-size proof witness w π $V(x,\pi)=0$ $M(x,w)=0$

The NTIME-hierarchy theorem tells us that checking a witness may take arbitrarily long:

theorem: $\mathrm{NTIME}[o(T)] \subsetneq \mathrm{NTIME}[T]$ [precisely: $\mathrm{NTIME}[f(n)] \subsetneq \mathrm{NTIME}[g(n)]$ $\forall$ time-constructible f, g with $f(n+1) = o(g(n))$]

Hence EFFICIENTLY-VERIFIABLE witnesses better capture mathematical proofs:

$NP = \bigcup_{c \in \mathbb{N}} \mathrm{NTIME}[n^c]$ nondeterministic polynomial time

Efficient Mathematical Proofs = NP

The definition of the complexity class NP (nondeterministic polynomial time):

$L \in NP \iff \exists$ polynomial-time verifier V / decider D s.t.

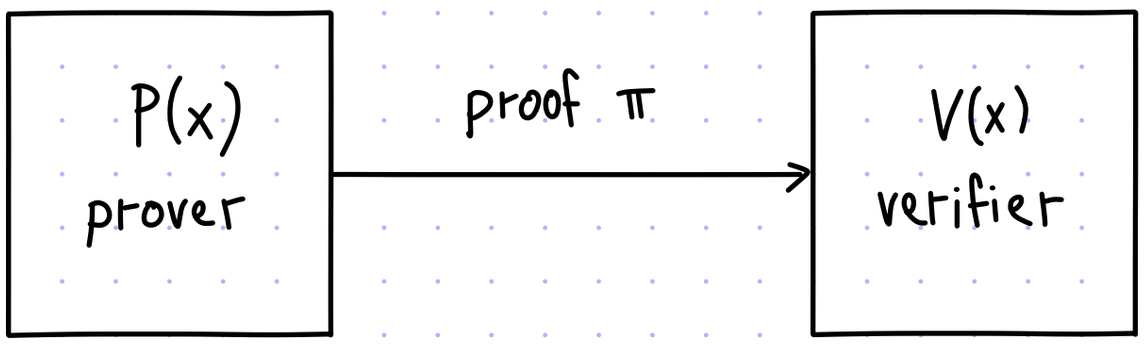
completeness: ① $\forall$ theorem instance $x \in L$ $\exists$ proof π poly($|x|$)-size witness w $V(x,\pi)=1$ / $D(x,w)=1$

soundness: ② $\forall$ theorem instance $x \notin L$ $\forall$ proof π poly($|x|$)-size witness w $V(x,\pi)=0$ / $D(x,w)=0$

Example: $L = SAT$

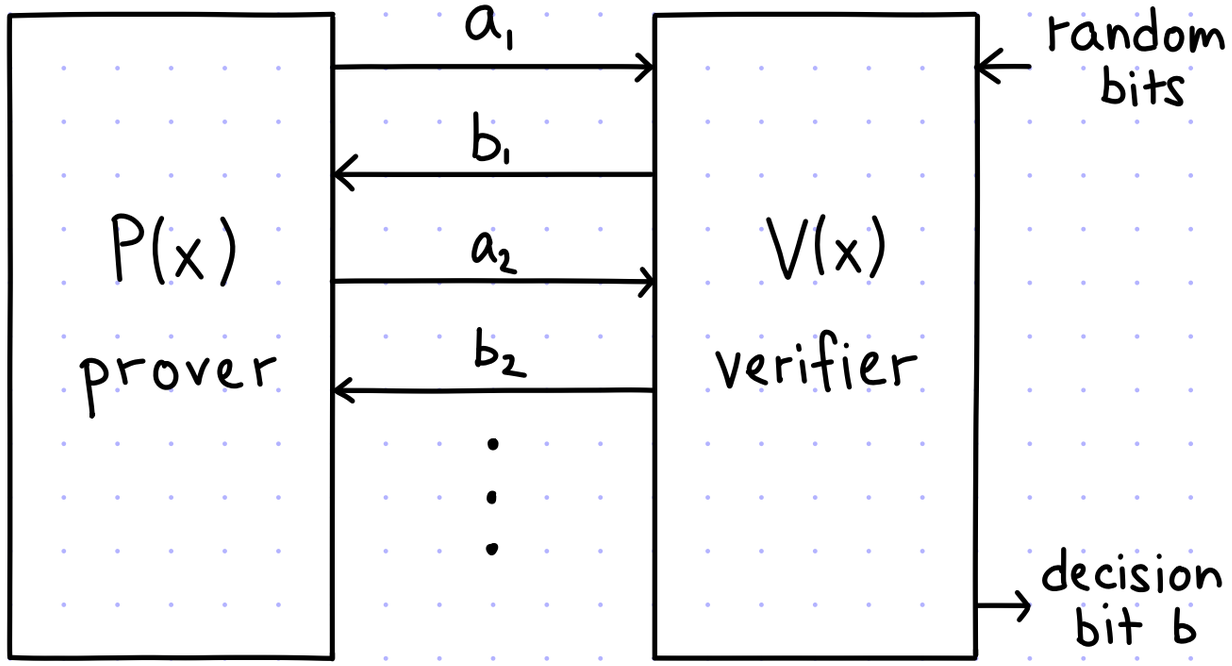
- x is a boolean formula $\varphi(x_1,\dots,x_n)$
- w is an assignment $(a_1,\dots,a_n) \in \{0,1\}^n$
- D checks that $\varphi(a_1,\dots,a_n) = \text{true}$

Hence NP captures traditional (efficient) mathematical proofs:



Interactive Proofs

A revolutionary idea: the verifier may use randomness and interact with the prover.



(unbounded) honest prover — (efficient) honest verifier

An **interactive proof** for a language L is a pair (P, V) such that:

① completeness: $\forall x \in L \quad \Pr_{r_p, r_v}\left[\langle P(x; r_p), V(x; r_v)\rangle = 1\right] = 1.$

② soundness: $\forall x \notin L \quad \forall \tilde{P} \quad \Pr_{r_v}\left[\langle \tilde{P}, V(x; r_v)\rangle = 1\right] \leq 1/2.$

↑ the "best" randomness can be hardcoded in $\tilde{P}$

more generally:

completeness $\geq 1 - \varepsilon_c$

soundness $\leq \varepsilon_s$

and it suffices to have $1 - \varepsilon_c - \varepsilon_s \geq \frac{1}{\text{poly}}$

What Is The Power Of Interactive Proofs?

Somewhat degenerate if we leverage only interaction or only randomness:

randomness \ interaction	no	yes
no	NP	NP
yes	MA	IP

believed to equal NP (strong PRGs → MA = NP)

Hence we should leverage interaction and randomness simultaneously.

We wish to understand:

Which languages have interactive proofs?

Are there any beyond NP (and MA)?

Isomorphisms Between Graphs

Let $G_0 = (V, E_0)$ and $G_1 = (V, E_1)$ be two graphs on vertices V.

def: $G_0 \equiv G_1$ (G_0 & G_1 are isomorphic) if $\exists$ permutation $\pi : V \to V$ s.t.

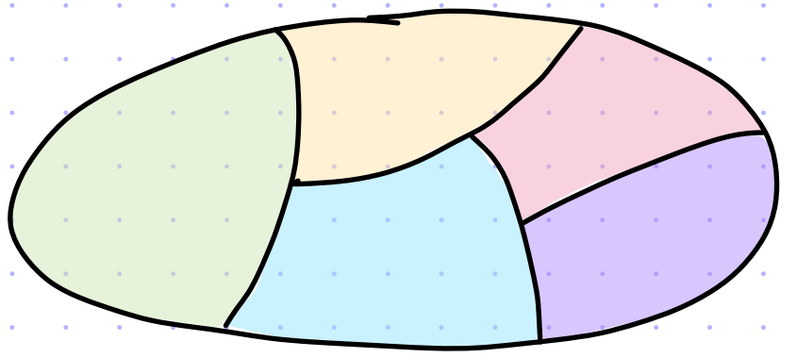
$$(u,v) \in E_0 \iff (\pi(u), \pi(v)) \in E_1 \ .$$

If so, we write $G_1 = \pi(G_0)$.

The isomorphism relation is an EQUIVALENCE RELATION:

- it is reflexive: $G \equiv G$ (via the identity permutation)
- it is symmetric: $G_1 = \pi(G_0) \leftrightarrow G_0 = \pi^{-1}(G_1)$
- it is transitive: $G_1 = \pi_1(G_0) \wedge G_2 = \pi_2(G_1) \to G_2 = (\pi_2 \circ \pi_1)(G_0)$

Hence the relation partitions all graphs into EQUIVALENCE CLASSES:

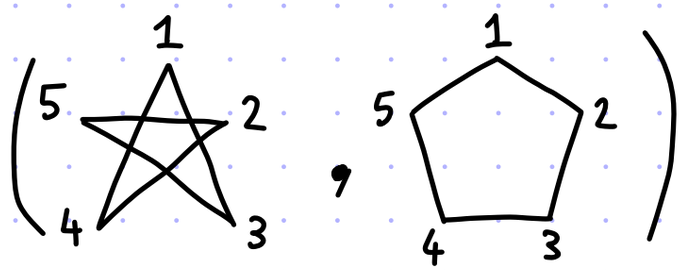


Languages for Graph Isomorphism

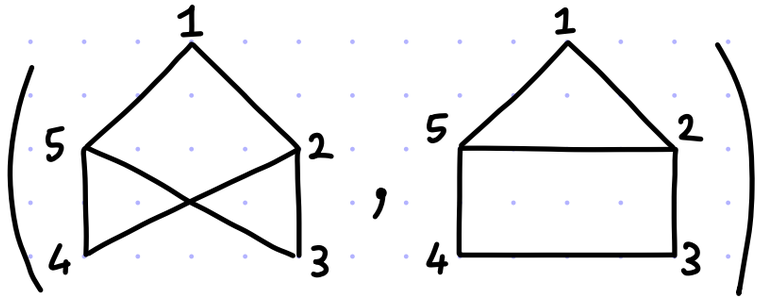
The graph isomorphism relation induces two languages:

$$GI := \{ (G_0, G_1) \mid G_0 \equiv G_1 \} \qquad GNI := \{ (G_0, G_1) \mid G_0 \not\equiv G_1 \}$$

Examples:

- 

$\in GI$. Why? $\pi = \{1 \to 1, 3 \to 2, 5 \to 3, 2 \to 4, 4 \to 5\}$

- 

$\in GNI$. Why? G_1 has a triangle but G_0 does not

More generally, $GI \in NP$, and so $GNI \in coNP$.
But it is not known if GI (or GNI) is in P.

Q: How to prove that $G_0 \not\equiv G_1$?

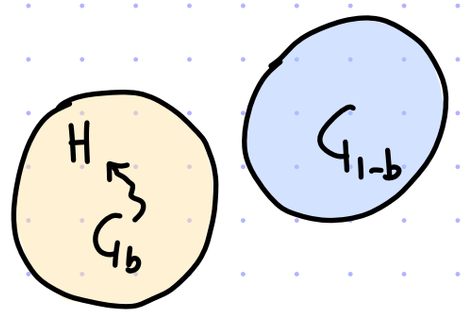
In general, it is harder to see than in the above example.

Interactive Proof for Graph Non-Isomorphism [1/2]

theorem: $GNI \in IP$

$P(G_0, G_1)$		$V(G_0, G_1)$
		$b \leftarrow \{0,1\}$
		$\pi \leftarrow \{\text{permutations on vertices}\}$
find $\tilde{b}$ s.t.	$\xleftarrow{H}$	$H := \pi(G_b)$
$H \equiv G_{\tilde{b}}$	$\xrightarrow{\tilde{b}}$	$\tilde{b} \stackrel{?}{=} b$

Completeness: Suppose that $(G_0, G_1) \in GNI$ (i.e., $G_0 \not\equiv G_1$).
Then G_b and G_{1-b} are in different equivalence classes:



The honest prover determines b by finding out which graph H is isomorphic to.

NOTE: for now we ignore the efficiency of the honest prover.

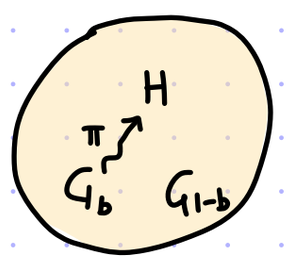
Interactive Proof for Graph Non-Isomorphism [2/2]

theorem: $GNI \in IP$

$P(G_0, G_1)$		$V(G_0, G_1)$
		$b \leftarrow \{0,1\}$
		$\pi \leftarrow \{$ permutations on vertices$\}$
find $\tilde{b}$ s.t.	$\xleftarrow{H}$	$H := \pi(G_b)$
$H \equiv G_{\tilde{b}}$	$\xrightarrow{\tilde{b}}$	$\tilde{b} \stackrel{?}{=} b$

Soundness: Suppose that $(G_0, G_1) \notin GNI$ (i.e., $G_0 \equiv G_1$).

The random variable $\pi(G_b)$ is identical to $\pi(G_{1-b})$:



Hence $H := \pi(G_b)$ and b are independent.

We conclude that $Pr[\tilde{b} = b] = 1/2$ regardless of how a malicious prover chooses $\tilde{b}$ based on H.

NOTE: it is crucial that b is secret ... but later we learn how to avoid "private randomness"

Upper Bound on IP

theorem: $IP \subseteq PSPACE$ ← languages decidable in polynomial space

Let $L \in IP$, and let (P,V) be an IP for L.

We show that $L \in PSPACE$.

Fix an instance x and define $q_x := \max_{\tilde{P}} \Pr\left[\langle \tilde{P}, V(x;r)\rangle = 1\right]$. maximum winning probability

If $x \in L$ then $q_x = 1$.
If $x \notin L$ then $q_x \leq 1/2$.
} It suffices to compute q_x in polynomial space.

PROBLEM: we cannot iterate over ALL provers $\tilde{P}$ because
this includes those that require large space to run

IDEA: any interaction transcript has polynomial size (the IP verifier reads it)
so we CAN iterate over all transcripts in polynomial space

We show that the optimal prover strategy is computable
in polynomial space, and so is the probability q_x.

Upper Bound on IP

The optimal prover is defined as follows:

$P^*(x, (a_1, b_1, \ldots, a_i, b_i))$ outputs a^*_{i+1} that maximizes the probability that $P^*(x)$ convinces $V(x)$ conditioned on the first i rounds being $(a_1, b_1, \ldots, a_i, b_i)$.

claim: $P^* \in \text{PSPACE} \rightarrow q_x \in \text{PSPACE}$

proof: The optimality of P^* implies that $q_x = \frac{\sum_{r \in R} d(x,r)}{|R|}$ where:

- R are the possible random strings of the IP verifier V,
- $d(x,r)$ is the decision bit of $V(x;r)$ when interacting with P^*.

Note that $d(x,r)$ is computable in polynomial space:

$a_1^* := P^*(x, \perp)$ $\quad a_2^* := P^*(x, (a_1^*, b_1))$ $\quad \ldots$ $\quad a_k^* := P^*(x, (a_1^*, b_1, \ldots, a_{k-1}^*, b_{k-1}))$

$b_1 := V(x, r, a_1^*)$ $\quad b_2 := V(x, r, a_1^*, a_2^*)$ $\quad b_k := V(x, r, a_1^*, \ldots, a_k^*)$

Indeed:
- each invocation of P^* is in polynomial space (by assumption)
- each invocation of V is in polynomial time (by definition)

Hence

$A(x) :=$
1. Initialize $c := 0$.
2. For each $r \in R$, $c := c + d(x,r)$.
3. Output $c/|R|$.

runs in polynomial space. ■

Upper Bound on IP

claim: $P^* \in PSPACE$

proof: Given a transcript $tr = (a_1, b_1, \ldots, a_i, b_i)$ of i rounds, define:

$$R[x, tr] := \left\{ r \in R \;\middle|\; \begin{array}{l} b_1 = V(x, r, a_1) \\ b_2 = V(x, r, a_1, a_2) \\ \vdots \\ b_i = V(x, r, a_1, \ldots, a_i) \end{array} \right\}$$

set of random strings consistent with (x, tr)

Membership in $R[x, tr]$ can be checked in polynomial time, and thus polynomial space.

The proof is by (reverse) induction on $i \in \{0, 1, \ldots, k-1\}$.

Base case is $i = k-1$.

By definition of P^*,

$$P^*(x, tr) = P^*(x, (a_1, b_1, \ldots, a_{k-1}, b_{k-1})) = \arg\max_{a_k} \Pr_{r \in R[x,tr]} \left[V(x, r, a_1, \ldots, a_{k-1}, a_k) = 1 \right]$$

We can iterate over all prover messages a_k and all random strings r in polynomial space. (By reusing space for every a_k and r.)

Upper Bound on IP [4/4]

claim: $P^* \in PSPACE$

proof: (continued)

Inductive case is $i < k-1$. (We assume that $P^* \in PSPACE$ for $|tr| > i$.)

By definition of P^*,

$$P^*(x, tr) = P^*(x, (a_1, b_1, \ldots, a_i, b_i)) = \arg\max_{a_{i+1}} \Pr_{r \in R[x,tr]} \left[V(x, r, a_1, \ldots, a_i, a_{i+1}, a^*_{i+2}, \ldots, a^*_k) = 1 \right]$$

where $a^*_{i+2}, \ldots, a^*_k$ are optimal prover messages for (x, tr, r, a_{i+1}):

$b_{i+1} := V(x, r, a_1, \ldots, a_i, a_{i+1})$ in polynomial time

$a^*_{i+2} := P^*(x, (a_1, b_1, \ldots, a_i, b_i, a_{i+1}, b_{i+1}))$ in polynomial space (inductive assumption)

$b_{i+2} := V(x, r, a_1, \ldots, a_i, a_{i+1}, a^*_{i+2})$ in polynomial time

$\vdots$

$a^*_k := P^*(x, (a_1, b_1, \ldots, a_i, b_i, a_{i+1}, b_{i+1}, a^*_{i+2}, b_{i+2}, \ldots, a^*_{k-1}, b_{k-1}))$ in polynomial space (inductive assumption)

We can iterate over all prover messages a_{i+1} and all random strings r.

We conclude that P^* is computable in polynomial space. ■

Error Reduction [1/3]

How to reduce the completeness error ε_c and/or soundness error ε_s of an IP?

Idea: Run the IP t times in sequence and (somehow) decide based on that.

For every IP prover $\tilde{P}_t$, define the random variables $(Z_i(x,\tilde{P}_t))_{i\in[t]}$:= "i-th IP verifier accepts".

The RVs are independently and identically distributed.

Moreover:
- $x \in L \to \forall\, i\in[t] \quad \Pr[Z_i(x,P_t)=1] \geq 1-\varepsilon_c$.
- $x \notin L \to \forall\, i\in[t] \quad \forall\, \tilde{P}_t \quad \Pr[Z_i(x,\tilde{P}_t)=1] \leq \varepsilon_s$.

The case of PERFECT COMPLETENESS ($\varepsilon_c = 0$):

$V_{AND}(x;(y_1,\dots,y_t)) := \bigwedge_{i\in[t]} V(x;y_i)$

- perfect completeness is preserved: $\varepsilon_c' := 0$

$$\Pr[\langle P_t(x), V_{AND}(x;(y_1,\dots,y_t))\rangle = 0] = \Pr[\bigwedge_{i\in[t]} Z_i(x,P_t)=0] = 1 - \prod_{i\in[t]} \Pr[Z_i(x,P)=1] = 1-1 = 0$$

- soundness error decays exponentially: $\varepsilon_s' := \varepsilon_s^t$

$$\Pr[\langle \tilde{P}_t, V_{AND}(x;(y_1,\dots,y_t))\rangle = 1] = \Pr[\bigwedge_{i\in[t]} Z_i(x,\tilde{P}_t)=1] = \prod_{i\in[t]} \Pr[Z_i(x,\tilde{P}_t)=1] \leq \prod_{i\in[t]} \varepsilon_s$$

Error Reduction [2/3]

How to reduce the completeness error ε_c and/or soundness error ε_s of an IP?

Idea: Run the IP t times in sequence and (somehow) decide based on that.

What about the case of IMPERFECT COMPLETENESS ($\varepsilon_c > 0$)?

The AND verifier increases the completeness error: $\varepsilon_c' := 1-(1-\varepsilon_c)^t \leq t \cdot \varepsilon_c$.

➜ This option demands small ε_c to begin with.

Alternatively, consider the OR verifier: $V_{OR}(x;(\rho_1,\dots,\rho_t)) := \bigvee_{i\in[t]} V(x;\rho_i)$

- completeness error decays exponentially: $\varepsilon_c' := \varepsilon_c^t$

$$\Pr\left[\langle P_t(x), V_{OR}(x;(\rho_1,\dots,\rho_t))\rangle = 0\right] = \Pr\left[\bigvee_{i\in[t]} z_i(x,P_t)=0\right] = \prod_{i\in[t]} \Pr[z_i(x,P_t)=0] \leq \prod_{i\in[t]} \varepsilon_c .$$

- soundness error increases: $\varepsilon_s' := 1-(1-\varepsilon_s)^t \leq t\cdot\varepsilon_s$

$$\Pr\left[\langle \tilde{P}_t, V_{OR}(x;(\rho_1,\dots,\rho_t))\rangle = 1\right] = \Pr\left[\bigvee_{i\in[t]} z_i(x,\tilde{P}_t)=1\right] = 1-\prod_{i\in[t]} \Pr[z_i(x,\tilde{P}_t)=0] \leq 1-\prod_{i\in[t]}(1-\varepsilon_s) .$$

➜ This option demands small ε_s to begin with.

What if neither ε_c nor ε_s is small (e.g. $\varepsilon_c = 1/3$ and $\varepsilon_s = 1/3$)?

Error Reduction [3/3]

How to reduce the completeness error ε_c and/or soundness error ε_s of an IP?

Idea: Run the IP t times in sequence and (somehow) decide based on that.

The random variable $Z(x,\tilde{P}) := \frac{1}{t}\sum_{i\in[t]} Z_i(x,\tilde{P})$ has mean $\mathbb{E}[Z(x,\tilde{P})] = \mathbb{E}[Z_i(x,\tilde{P})]$.

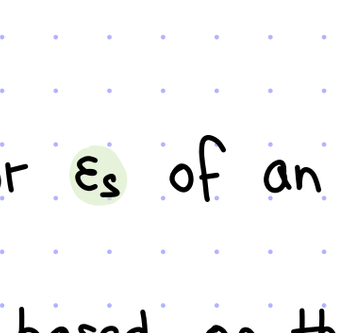
By a Chernoff bound, $Z(x,\tilde{P})$ concentrates around its mean:

$$\forall\, \gamma \geq 0 \quad \Pr\left[\left|Z(x,\tilde{P}) - \mathbb{E}[Z(x,\tilde{P})]\right| \geq \gamma\right] \leq 2\cdot e^{-\frac{t\gamma^2}{4}}.$$

This motivates the MAJ verifier:

$V_{MAJ}(x;(\rho_1,\ldots,\rho_t)) :=$

1. Set the treshold $\tau := \frac{1-\varepsilon_c+\varepsilon_s}{2}$.
2. For every $i\in[t]$, compute $b_i := V(x;\rho_i)$.
3. If $\frac{1}{t}\sum_{i\in[t]} b_i \geq \tau$ output 1; else output 0.



- $x\in L \to \Pr[\langle P(x), V_{MAJ}(x)\rangle = 0] = \Pr[Z(x,P) < \tau] \leq \Pr\left[\left|Z(x,P) - \mathbb{E}[Z(x,P)]\right| \geq \frac{1-\varepsilon_c-\varepsilon_s}{2}\right] \leq 2\cdot e^{-t\cdot(1-\varepsilon_c-\varepsilon_s)^2}$.
  ($\mathbb{E}[Z(x,P)] \geq 1-\varepsilon_c$; Chernoff bound)
- $x\notin L \to \Pr[\langle \tilde{P}, V_{MAJ}(x)\rangle = 1] = \Pr[Z(x,\tilde{P}) \geq \tau] \leq \Pr\left[\left|Z(x,\tilde{P}) - \mathbb{E}[Z(x,\tilde{P})]\right| \geq \frac{1-\varepsilon_c-\varepsilon_s}{2}\right] \leq 2\cdot e^{-t\cdot(1-\varepsilon_c-\varepsilon_s)^2}$.
  ($\mathbb{E}[Z(x,\tilde{P})] \leq \varepsilon_s$; Chernoff bound)

Hence if $t \geq \frac{\ln 2/\varepsilon}{(1-\varepsilon_c-\varepsilon_s)^2}$ then $\varepsilon_c', \varepsilon_s' \leq \varepsilon$.

Bibliography

Miscellaneous

- Introduction to metamathematics, by Stephen Cole Kleene.
- Gödel, Escher, Bach: an eternal golden braid by Douglas Hofstadter.
- The annotated turing, by Charles Petzold.
- Logicomix. Apostolos Doxiadis, Christos Papadimitriou, Alecos Papadatos, Annie Di Donna.
- (▸Proofs, secrets, and computation), by Silvio Micali.

Definitions

- [GMR 1985]: The knowledge complexity of interactive proof-systems, by Shafi Goldwasser, Silvio Micali, Charles Rackof.
- [Babai 1985]: Trading group theory for randomness by László Babai.
- [BM 1986]: Arthur–Merlin games: a randomized proof system, and a hierarchy of complexity classes, by László Babai, Shlomo Moran.

Introduces interactive proofs

Public-coin interactive proofs

Introduces AM and MA complexity

Technical sections

- [IW 1997]: P = BPP if E requires exponential circuits: derandomizing the XOR lemma, by Russel Impagliazzo, Avi Wigderson.
- [GMW 1987]: How to play any mental game or a completeness theorem for protocols with honest majority, by Oded Goldreich, Silvio Micali, Avi Wigderson.
- Hoeffding and Chernoff concentration inequalities.

AM = NP (plausibly)