CAS CS 535: Complexity Theory

Lecturer: Mark Bun Fall 2023

Lecture Notes 22:

$\mathbf{IP} = \mathbf{PSPACE}$

Reading.

- Arora-Barak § 8.3

Last time: Arthur-Merlin (Public Coin) Proofs

An Abbreviated History. See Babai's entertaining retrospective, "E-mail and the unexpected power of interaction" for more.

Mid-80's Goldwasser-Micali-Rackoff introduce $\mathbf{IP}$ and zero-knowledge proofs and study the quadratic non-residuosity problem. Babai independently introduces $\mathbf{AM}$ and public coin proofs, initially to study matrix group problems.

1987 Goldreich-Micali-Wigderson show that $\mathsf{GNI} \in \mathbf{IP}$.

1987 Negative results: Boppana-Håstad-Sipser show $\mathbf{coNP} \not\subseteq \mathbf{AM}$ unless $\mathbf{PH}$ collapses. Fortnow-Sipser give an oracle A such that $\mathbf{coNP}^A \not\subseteq \mathbf{IP}^A$. So interactive proofs might not be much more powerful than static ones, or at least, proving so *requires non-relativizing techniques*.

11/27/89 Nisan: $\mathbf{P}^{\#\mathbf{P}} \subseteq \mathbf{MIP}$, i.e., the permanent has a *multi-prover* interactive proof.

12/13/89 Lund-Fortnow-Karloff-Nisan: $\mathbf{P}^{\#\mathbf{P}} \subseteq \mathbf{IP}$.

12/26/89 Shamir: $\mathbf{PSPACE} = \mathbf{IP}$.

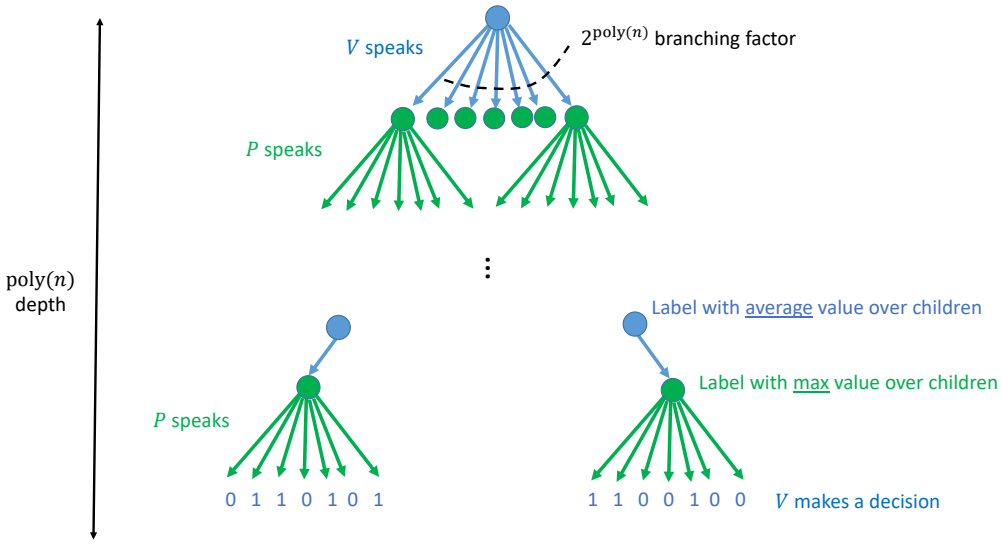
1 $\mathbf{IP} \subseteq \mathbf{PSPACE}$

This is the "easy" direction. Here's the idea. Let $L \in \mathbf{IP}$ with poly-time verifier V. We'll show that we can compute, in polynomial space, the value of

$$\max_{P^*} \Pr[\langle P^*, V \rangle(x) = 1].$$

It then suffices to check if this values is at least $2/3$, in which case we accept, or at most $1/3$ in which case we reject.

To do this, imagine (as a thought experiment) constructing the following tree:



The label on node represents the success probability of the best prover strategy given the history of the protocol up to that point. So the value at the root gives the success probability of the best prover strategy overall. By evaluating the node labels via a post-order traversal, we can compute the value at the root in polynomial space.

2 Arithmetization

Recall from Toda's Theorem that

$$\mathbf{PH} \subseteq \mathbf{P}^{\#\mathbf{P}} \subseteq \mathbf{PSPACE}.$$

We'll work our way up to our goal by first showing the weaker (but still powerful) statement that $\mathbf{P}^{\#\mathbf{P}} \subseteq \mathbf{IP}$. To do this, it suffice to give an interactive proof for the following decisional version of the #SAT problem:

$$\#\mathsf{SAT}_{\mathsf{D}} = \{\langle \varphi, k \rangle \mid \text{CNF } \varphi \text{ has exactly } k \text{ satisfying assignments}\}.$$

The first step is to think about this problem in a more algebraic way. Observe that

$$\begin{aligned}
\varphi \text{ has } k \text{ satisfying assignments} &\iff \sum_{x_1 \in \{0,1\}} \sum_{x_2 \in \{0,1\}} \cdots \sum_{x_n \in \{0,1\}} \varphi(x_1, \ldots, x_n) = k \\
&\iff \sum_{x_1 \in \{0,1\}} \sum_{x_2 \in \{0,1\}} \cdots \sum_{x_n \in \{0,1\}} \varphi(x_1, \ldots, x_n) = k \mod p \text{ for prime } p \in [2^n, 2^{n+1}] \\
&\iff \sum_{x_1 \in \{0,1\}} \sum_{x_2 \in \{0,1\}} \cdots \sum_{x_n \in \{0,1\}} F(x_1, \ldots, x_n) = k \mod p
\end{aligned}$$

where F is a polynomial over the integers and $F(x_1, \ldots, x_n) = \varphi(x_1, \ldots, x_n)$ for all $x \in \{0,1\}^n$.

We can construct such an integer polynomial by arithmetizing the formula φ gatewise:

- Replace $x \wedge y$ with xy

- Replace $\neg x$ with $1 - x$
- Replace $x \vee y = \overline{\overline{x} \wedge \overline{y}}$ with $1 - (1 - x)(1 - y)$.

Example 1. The arithmetization of the formula $\varphi(x) = (x_1 \vee \overline{x_2} \vee x_3) \wedge (x_2 \vee x_4 \vee \overline{x_5})$ is $F(x) = (1 - (1 - x_1)x_2(1 - x_3))(1 - (1 - x_1)(1 - x_4)x_5)$.

Properties of SAT arithmetiziation: If φ has n variables and m clauses, then:

- F is computed by an arithmetic formula of size $O(m + n)$
- $\deg F = O(m) = O(n^3)$
- The conversion to an arithmetic formula takes polynomial time.

Thus, to solve $\#\mathsf{SAT}_\mathsf{D}$, it suffices to give an interactive protocol solving the more general problem:

Theorem 2. *There exists a poly-time interactive proof for the following problem. Given as input an arithmetic circuit F of degree d, a prime p, and an integer k:*

Completeness *If $\sum_x F(x) = k \mod p$, then $\Pr[\langle P, V\rangle(F, p, k) = 1] = 1$.*

Soundness *If $\sum_x F(x) \neq k \mod p$, then $\Pr[\langle P, V\rangle(F, p, k) = 1] \leq \frac{dn}{p}$.*

3 Sumcheck Protocol

The "sumcheck protocol" is a recursively defined interactive protocol for the above problem. Each step of the recursion decreases the number of variables n by 1.

Base case ($n = 1$): The verifier checks itself that $k_1 = \sum_{x_1 \in \{0,1\}} F_1(x_1)$.

Recursive case:

- Define the univariate polynomial

$$g(t) = \sum_{x_1, \ldots, x_{n-1} \in \{0,1\}^{n-1}} F_n(x_1, \ldots, x_{n-1}, t).$$

- Prover sends the description of a polynomial $g^*(t)$ (say, specified by d coefficients, using $O(nd)$ bits) that it claims is equal to $g(t)$.
- Verifier checks that $g^*(0) + g^*(1) = k_n \mod p$ (rejecting if not.) Then recursively call the same protocol on $\langle F_{n-1}, p, k_{n-1}\rangle$ where
  - $r \in \{0, 1, \ldots, p - 1\}$ is uniformly random
  - $k_{n-1} = g^*(r)$
  - $F_{n-1}(x_1, \ldots, x_{n-1}) = F(x_1, \ldots, x_{n-1}, r)$.

3.1 Analysis

Completeness: If the claim is correct, then the prover can send $g(t)$ in every round, and the verifier always accepts.

Soundness: If the claim is false, we'll show that in each round, either:

1. V catches P^* with probability 1, or
2. With probability at least $1 - \frac{d}{p}$, we have that P^* is forced into having to try to prove a false claim in the next round.

By induction and a union bound, we have that the probability that V catches P^* is at least $1 - n \cdot \frac{d}{p}$.

To analyze one round: Suppose $\sum_x F(x) \neq k \mod p$. Then there are two cases:

Case 1: The prover sends the actual polynomial $g(t)$. Then $g(0) + g(1) \neq k \mod p$, so the verifier catches the lie with probability 1.

Case 2: The prover sends some other polynomial $g^* \neq g$. Since they have degree at most d, g^* agrees with g on at most d points, so the probability that the next claim the prover will have to work with is true is

$$\Pr_r[g^*(r) = g(r)] \leq \frac{d}{p}.$$

4 PSPACE $\subseteq$ IP

To show that **PSPACE** $\subseteq$ **IP**, we'll sketch how to modify the sumcheck protocol can handle the **PSPACE**-complete problem TQBF. Let

$$\Psi = \forall x_1 \exists x_2 \ldots \exists x_n \varphi(x_1, \ldots, x_n)$$

be a fully quantified Boolean formula, and assume WLOG that each x_i represents a single bit.

Idea 1: Observe that

$$\Psi \in \mathsf{TQBF} \iff \prod_{x_1 \in \{0,1\}} \sum_{x_2 \in \{0,1\}} \cdots \sum_{x_n \in \{0,1\}} F(x_1, \ldots, x_n) \neq 0,$$

where F is the arithmetization of φ. One could then apply the sumcheck protocol, with the modification that when handling a $\forall$ quantifier, one should check the product $g^*(0) \cdot g^*(1)$ instead of the sum.

The problem with this is that polynomial multiplication increases degree, resulting in a final polynomial with degree as high as $2^{\Omega(n)}$. In general, this is too long for the prover to transmit to the verifier.

Idea 2: We only care that our polynomial representation holds on Boolean inputs, where $x_i^2 = x_i$ for every variable i. (And thus, $x_i^k = x_i$ for every k.) So we can replace our polynomials with their "multilinearizations" which agree with them on all Boolean inputs but have individual degree 1 on every variable.

More precisely, define the multilinearization operator L_i by

$$(L_i F)(x) = x_i F(x_1, \ldots, x_{i-1}, 1, x_{i+1}, \ldots, x_n) + (1 - x_i) F(x_1, \ldots, x_{i-1}, 0, x_{i+1}, \ldots, x_n)$$

This doesn't change the value of F on Boolean inputs, but has the effect of reducing the individual degree on variable x_i down to 1. Thus, we have

$$\Psi \in \mathsf{TQBF} \iff \prod_{x_1 \in \{0,1\}} L_1 \sum_{x_2 \in \{0,1\}} L_1 L_2 \cdots \sum_{x_n \in \{0,1\}} L_1 L_2 \ldots L_n F(x_1, \ldots, x_n) \neq 0,$$

which (modulo a large prime) we can verify using the sumcheck protocol to peel off one Π, Σ or L operator in each round.