Penn State University
CSE 564 Complexity of Combinatorial Problems

Instructor: Pei Wu

Module

4

The sum-check protocol

The last lecture introduced interactive proofs and the public-coin classes MA and AM. We recalled two useful "collapse" phenomena: (i) any *constant* number of public-coin rounds collapses to a single round, i.e. $\mathsf{AM}[k] = \mathsf{AM}$ for constant k; and (ii) private coins can be made public with only two extra rounds, $\mathsf{IP}[k] \subseteq \mathsf{AM}[k+2]$. A first consequence is that AM is widely believed to be a rather small class: if $\mathsf{coNP} \subseteq \mathsf{AM}$, then the polynomial hierarchy collapses.

The main new technical tool in this lecture is *arithmetization*: replacing a Boolean formula by a low-degree polynomial over a field. This viewpoint leads to the *sum-check protocol*, an interactive protocol for verifying a claimed sum of a polynomial over the Boolean hypercube. As an application (due to Lund–Fortnow–Karloff–Nisan), we obtain an interactive proof for the counting problem $\#\mathsf{SAT}$, and hence for coNP.

Finally, we begin the proof of Shamir's theorem $\mathsf{IP} = \mathsf{PSPACE}$. We reduce membership in TQBF to checking the value of an arithmetized polynomial with alternating "OR" and "AND" operators. A naive protocol breaks because degrees blow up; the fix is a degree-reduction step based on the identity $x_i^2 = x_i$ on Boolean inputs, implemented via a *linearization* operator.

4.1 A warning sign: if coNP is in AM, PH collapses

We start with the following (standard) collapse implication.

Theorem 4.1 (Boppana–Håstad–Zachos). *If $\mathsf{coNP} \subseteq \mathsf{AM}$, then the polynomial hierarchy collapses; in particular $\Sigma_2 = \Pi_2$.*

Proof sketch from the notes. Assume $\mathsf{coNP} \subseteq \mathsf{AM}$. Let $L \in \Sigma_2$. By definition, there exists a polynomial-time deterministic verifier V such that

$$x \in L \iff \exists \pi_1 \, \forall \pi_2 \;\; V(x, \pi_1, \pi_2) = 1.$$

Fix a candidate π_1. The inner predicate

$$\forall \pi_2 \;\; V(x, \pi_1, \pi_2) = 1$$

is a coNP-type statement (it is the complement of "$\exists \pi_2$ such that $V(x, \pi_1, \pi_2) = 0$", which is in NP). By the assumption $\mathsf{coNP} \subseteq \mathsf{AM}$, this inner statement has an AM protocol.

Therefore, membership in L has a MAM protocol: Merlin first sends π_1, and then Merlin and Arthur run the AM protocol for the coNP substatement. Since constant-round public-coin interaction collapses (recall $\mathsf{AM}[k] = \mathsf{AM}$ for constant k), we get $\mathsf{MAM} \subseteq \mathsf{AM}$, hence $L \in \mathsf{AM}$.

Finally, $\mathsf{AM} \subseteq \Pi_2$ (a fact that uses perfect completeness), so $L \in \Pi_2$. We have shown $\Sigma_2 \subseteq \Pi_2$, and the reverse containment $\Pi_2 \subseteq \Sigma_2$ is trivial, so $\Sigma_2 = \Pi_2$. This implies a collapse of the entire polynomial hierarchy. □

The moral is that it is unlikely that $\mathsf{coNP} \subseteq \mathsf{AM}$. In contrast, interaction with *private* coins turns out to be strong enough to capture coNP (and much more).

4.2 Arithmetization of Boolean formulas

A key idea in modern interactive proofs is to replace a Boolean computation by an algebraic one.

Boolean formulas. Let $\psi(x_1, \ldots, x_n)$ be a Boolean formula with $\wedge, \vee, \neg$ gates. We will map it to a polynomial P_ψ over a field $\mathbb{F}$ such that $P_\psi(x) = \psi(x)$ for all Boolean inputs $x \in \{0,1\}^n$.

Gate-by-gate translation. We interpret literals as low-degree polynomials:

$$x_i \;\mapsto\; x_i, \qquad \neg x_i \;\mapsto\; 1 - x_i.$$

For internal gates, the notes use the following arithmetizations:

$$\begin{aligned} \mathrm{OR}(a,b,c) &= 1 - (1-a)(1-b)(1-c), \\ \mathrm{AND}(a,b,c) &= a \cdot b \cdot c. \end{aligned}$$

(For a binary gate, drop the unused argument(s).) Notice that on $\{0,1\}$ inputs these expressions compute the correct Boolean operations. Also, degrees behave well for AND: $\deg(\mathrm{AND}(a,b,c)) = \deg(a) + \deg(b) + \deg(c)$.

FACT 4.2 (agreement on the Boolean cube). *For every $x \in \{0,1\}^n$, we have $\psi(x) = 1$ if and only if $P_\psi(x) = 1$.*

Proof. By structural induction on the formula. Each arithmetized gate matches its Boolean counterpart on Boolean inputs. □

An immediate corollary is that satisfiability reduces to a sum.

FACT 4.3 (satisfiability as a sum). *ψ is satisfiable if and only if*

$$\sum_{x \in \{0,1\}^n} P_\psi(x) > 0.$$

Moreover, $\deg(P_\psi) = \mathrm{poly}(n)$ *and* $\sum_{x \in \{0,1\}^n} P_\psi(x) \leq 2^n$.

Proof. By Fact 4.2, $P_\psi(x) \in \{0,1\}$ on Boolean inputs and equals $\psi(x)$. Thus the sum counts satisfying assignments. The degree bound follows because each gate increases degree by at most an additive amount, and the formula size is polynomial. □

4.3 The sum-check protocol and #SAT $\in$ IP

The sum-check protocol is a generic way to verify a claimed value $\sum_{x \in \{0,1\}^n} P(x)$ for a low-degree polynomial P.

Statement of the protocol.

Fix a field $\mathbb{F}$ and a polynomial $P : \mathbb{F}^n \to \mathbb{F}$ of (total) degree at most d. The prover claims that

$$S \stackrel{?}{=} \sum_{x \in \{0,1\}^n} P(x).$$

The verifier can evaluate $P(r_1, \ldots, r_n)$ at any point in $\mathbb{F}^n$ in polynomial time (for our application, $P = P_\psi$ is given explicitly as an arithmetic formula).

Protocol (sum-check). The interaction lasts n rounds. We maintain a "current claim" $S_0 := S$. For $i = 1, 2, \ldots, n$:

1. The prover sends a univariate polynomial $\widetilde{Q}_i(z) \in \mathbb{F}[z]$.
2. The verifier checks that $\deg(\widetilde{Q}_i) \le d$ (or the appropriate per-variable degree bound) and that
$$\widetilde{Q}_i(0) + \widetilde{Q}_i(1) = S_{i-1}.$$
If the check fails, reject.
3. The verifier samples $\alpha_i \leftarrow \mathbb{F}$ uniformly at random and sets $S_i := \widetilde{Q}_i(\alpha_i)$. The verifier sends α_i to the prover.

After round n, the verifier evaluates $P(\alpha_1, \ldots, \alpha_n)$ itself and accepts iff $S_n = P(\alpha_1, \ldots, \alpha_n)$.

Honest prover. The notes describe the honest prover's message in terms of polynomials $Q_1, \ldots, Q_n$ defined by partial summation:

$$\begin{aligned}
Q_1(z) &:= \sum_{x_2, \ldots, x_n \in \{0,1\}} P(z, x_2, \ldots, x_n), \\
Q_2(z) &:= \sum_{x_3, \ldots, x_n \in \{0,1\}} P(\alpha_1, z, x_3, \ldots, x_n), \\
&\ \ \vdots \\
Q_n(z) &:= P(\alpha_1, \ldots, \alpha_{n-1}, z).
\end{aligned}$$

It is immediate that $Q_1(0) + Q_1(1) = \sum_{x \in \{0,1\}^n} P(x)$ and that the consistency checks hold:

$$Q_{i-1}(\alpha_{i-1}) = Q_i(0) + Q_i(1) \qquad (i = 2, \ldots, n),$$

and finally $Q_n(\alpha_n) = P(\alpha_1, \ldots, \alpha_n)$.

Soundness via Schwartz–Zippel.

The crucial algebraic fact used in the analysis is the following.

Lemma 4.4 (Schwartz–Zippel). *Let $R \in \mathbb{F}[z]$ be a nonzero univariate polynomial of degree at most d. Then*

$$\Pr_{a \leftarrow \mathbb{F}}[R(a) = 0] \le \frac{d}{|\mathbb{F}|}.$$

Proof. A nonzero univariate polynomial of degree d has at most d roots in a field. ∎

THEOREM 4.5 (soundness of sum-check). *If $S \neq \sum_{x\in\{0,1\}^n} P(x)$, then any (possibly cheating) prover causes the verifier to accept with probability at most $n \cdot d/|\mathbb{F}|$.*

Proof idea (as in the notes). Consider the first round where the prover's polynomial $\widetilde{Q}_i$ differs from the honest polynomial Q_i (defined above). If the prover ever sends $\widetilde{Q}_n \neq Q_n$, then to pass the final check it must happen that $\widetilde{Q}_n(\alpha_n) = Q_n(\alpha_n)$, which occurs with probability at most $d/|\mathbb{F}|$ by Lemma 4.4. Conditioning on $\widetilde{Q}_n = Q_n$, if $\widetilde{Q}_{n-1} \neq Q_{n-1}$, then to pass round $n-1$ we need $\widetilde{Q}_{n-1}(\alpha_{n-1}) = Q_{n-1}(\alpha_{n-1})$, again a degree-$\leq d$ event of probability at most $d/|\mathbb{F}|$. Proceeding backwards and union-bounding over the n possible first "lies" gives an overall acceptance probability at most $n \cdot d/|\mathbb{F}|$.

The handwritten notes instantiate this with a field of size roughly $|\mathbb{F}| = 2^n$ and $d = \mathrm{poly}(n)$, yielding soundness $2^{-\Omega(n)}$. ∎

Application: #$\mathsf{SAT} \in \mathsf{IP}$.

THEOREM 4.6 (Lund–Fortnow–Karloff–Nisan). *There is an interactive proof for verifying the value of #SAT. Equivalently, the language*

$$\#\mathsf{SAT} \;=\; \{(\psi, T) : T = |\{x \in \{0,1\}^n : \psi(x) = 1\}|\}$$

is in IP.

Proof sketch. Given (ψ, T), let P_ψ be the arithmetization polynomial. By Fact 4.3, we have $T = \sum_{x\in\{0,1\}^n} P_\psi(x)$. Arthur runs the sum-check protocol with claimed sum $S = T$ and polynomial $P = P_\psi$. Arthur can evaluate $P_\psi(\alpha_1, \ldots, \alpha_n)$ in polynomial time since P_ψ is given as an explicit arithmetic formula. Completeness follows from the honest-prover construction, and soundness follows from Theorem 4.5.

As a corollary, $\mathsf{SAT} \in \mathsf{IP}$ (accept iff $T > 0$), and therefore $\mathsf{coNP} \subseteq \mathsf{IP}$. ∎

4.4 Toward Shamir's theorem $\mathsf{IP} = \mathsf{PSPACE}$

The sum-check protocol is one of the main ingredients in Shamir's celebrated result that interactive proofs capture polynomial space.

THEOREM 4.7 (Shamir). *IP = PSPACE.*

Beginning of the proof as in the notes. We already proved $\mathsf{IP} \subseteq \mathsf{PSPACE}$ in Lecture 3. For the other direction it suffices to show that the canonical PSPACE- complete problem TQBF has an interactive proof.

An instance of TQBF is a quantified Boolean formula

$$\Phi \;=\; \exists x_1\ \forall x_2\ \exists x_3\ \cdots\ \forall x_n\ \varphi(x_1, \ldots, x_n),$$

and $\Phi \in \mathsf{TQBF}$ iff this evaluates to true. As before, we arithmetize φ to a polynomial P_φ.

Quantifiers as algebraic operators. On Boolean inputs, a universal quantifier corresponds to an $\wedge$ of two restrictions, and an existential quantifier corresponds to an $\vee$. Thus, if we define operators on polynomials by

$$(\Pi_{x_i} P)(x_1, \dots, x_{i-1}) := P(x_1, \dots, x_{i-1}, 0) \cdot P(x_1, \dots, x_{i-1}, 1),$$
$$(\Sigma^{\vee}_{x_i} P)(x_1, \dots, x_{i-1}) := 1 - \big(1 - P(x_1, \dots, x_{i-1}, 0)\big)\big(1 - P(x_1, \dots, x_{i-1}, 1)\big),$$

then evaluating Φ corresponds to iteratively applying Π for $\forall$ and $\Sigma^{\vee}$ for $\exists$, starting from P_{φ}. For example, for the last quantifier $\forall x_n$ the notes write

$$P_n \;=\; P_{\varphi}(x_1, \dots, x_{n-1}, 0) \cdot P_{\varphi}(x_1, \dots, x_{n-1}, 1),$$

and for an existential step they write

$$P_2 \;=\; 1 - \big(1 - P_4(x_1, x_2, 0)\big)\big(1 - P_4(x_1, x_2, 1)\big) \;=\; \mathrm{OR}\big(P_4(x_1, x_2, 0),\, P_4(x_1, x_2, 1)\big).$$

Obvious (but flawed) attempt. One could try to generalize sum-check as follows: Merlin sends the current polynomial Q_1 (say $Q_1 = P_2$), Arthur chooses a random field element α_1, Merlin responds with Q_2 obtained by partially substituting ($Q_2 = P_3(\alpha_1, \cdot)$), and so on, until the verifier reduces to checking $P_{\varphi}(\alpha_1, \dots, \alpha_n)$ at a random point. The notes summarize the obstacle as: *"the degree is too big; Merlin can't send Q"*. Indeed, composing Π and $\Sigma^{\vee}$ can blow up the total degree exponentially in n.

Degree reduction via linearization. The fix relies on the fact that all these operators are intended to agree with the Boolean semantics only on inputs $x \in \{0,1\}^n$, where $x_i^2 = x_i$. We can force a polynomial to become *linear* in a chosen variable without changing its values on Boolean inputs.

Define the *linearization operator* L_i acting on polynomials P by

$$(L_i P)(x_1, \dots, x_n) := (1 - x_i)\, P(x_1, \dots, x_{i-1}, 0, x_{i+1}, \dots, x_n) + x_i\, P(x_1, \dots, x_{i-1}, 1, x_{i+1}, \dots, x_n).$$

On the Boolean cube, $L_i P$ agrees with P (it is exactly the unique linear interpolation between the values at $x_i = 0$ and $x_i = 1$). Moreover, $L_i P$ has degree at most 1 in the variable x_i.

The remainder of Shamir's protocol alternates quantifier steps (Π_{x_i} for $\forall x_i$ and $\Sigma^{\vee}_{x_i}$ for $\exists x_i$) with linearization steps L_i to keep degrees bounded, and then uses the same "random evaluation" idea as in sum-check to prevent cheating.

Concretely, the notes describe a chain of intermediate *claims* $S_1, S_2, \dots$ obtained by applying a long sequence of operators to the base polynomial P_{φ} (quantifier operators interleaved with L_i's). Each claim can be expressed as the value of a *univariate* polynomial once the verifier has fixed some variables to random field elements. In a typical round, Merlin sends a univariate polynomial Q, and Arthur checks the appropriate operator identity using only evaluations at 0 and 1:

- If the next operator is Π_{x_i}, check $\Pi_{x_i} Q = Q(0) \cdot Q(1)$.
- If the next operator is $\Sigma^{\vee}_{x_i}$ (the OR operator), check $\Sigma^{\vee}_{x_i} Q = 1 - (1 - Q(0))(1 - Q(1))$.
- If the next operator is L_i, check the linear interpolation identity

  $$(L_i Q)(\alpha) = (1 - \alpha) Q(0) + \alpha Q(1)$$

  at the verifier's chosen point $\alpha \in \mathbb{F}$.

After a successful check, Arthur samples a fresh random field element and reduces to the next claim by evaluating the prover's polynomial at that point.

The last check is an explicit evaluation of P_φ at a random point $(\alpha_1, \ldots, \alpha_n) \in \mathbb{F}^n$, which Arthur can compute directly from the arithmetized formula. $\square$