Penn State University
CSE 564 Complexity of Combinatorial Problems

Instructor: Pei Wu

Module
8

Probabilistically checkable proofs (III): quadratic equations via sum-check

The previous PCP lectures introduced the PCP model and studied the BLR linearity test, which verifies (with three queries) that a function behaves like a linear function over $\mathbb{F}_2^n$. In this lecture we switch gears from "local" consistency tests (linearity) to a more global algebraic verification tool: the *sum-check protocol*. Sum-check lets a verifier certify that a low-degree polynomial sums to zero over a large product set H^t using only a polylogarithmic number of queries.
The main application here is a PCP for a quadratic feasibility problem over an arbitrary field $\mathbb{F}$ (not necessarily $\mathbb{F}_2$). The proof combines two ideas: (i) compressing many quadratic constraints into a *single* quadratic constraint using a small amount of randomness ("polynomial hashing"), and (ii) arithmetizing the quadratic constraint and verifying it using sum-check on a low-degree extension of the alleged assignment. Finally, we record the missing ingredient needed to turn this into a full PCP: a *low-degree test* to ensure that the prover's oracle is actually close to a low-degree polynomial.

8.1 Main theorem and the two issues

We work over a finite field $\mathbb{F}$. Consider a system of m quadratic equations in n variables over $\mathbb{F}$, represented as polynomials

$$Q_1(x_1, \ldots, x_n),\ Q_2(x_1, \ldots, x_n),\ \ldots,\ Q_m(x_1, \ldots, x_n) \in \mathbb{F}[x_1, \ldots, x_n],$$

each of total degree at most 2. The satisfiability problem asks whether there exists an assignment $a = (a_1, \ldots, a_n) \in \mathbb{F}^n$ such that $Q_i(a) = 0$ for all $i \in [m]$.

THEOREM 8.1 (quadratic equations have a polylog-query PCP). *The quadratic feasibility problem over $\mathbb{F}$ has a PCP verifier using $\log n$ random bits and $\mathrm{poly}(\log n)$ queries over alphabet $\Sigma = \mathbb{F}$, with proof length*

$$\ell \;=\; |\mathbb{F}|^{\,O(\log n / \log\log n)}.$$

(In particular, choosing $|\mathbb{F}| = \mathrm{poly}(\log n)$ makes $\ell = \mathrm{poly}(n)$.)

The notes highlight two conceptual obstacles that must be addressed to prove Theorem 8.1.

- **Issue 1 (many constraints):** We start from m quadratic constraints $Q_1, \ldots, Q_m$, but the PCP verifier wants to check only a *constant* (or at most polylogarithmic) amount of information. How can we reduce checking all constraints to checking a *single* quadratic equation, using little randomness?
- **Issue 2 (feasibility of one constraint):** Given a single quadratic equation Q, how can a verifier check (with few queries) the NP statement "$\exists a \in \mathbb{F}^n$ such that $Q(a) = 0$"?

The rest of the lecture addresses Issue 1 and Issue 2 in turn.

8.2 Issue 1: combining many quadratic equations

A first thought is: pick a random equation $Q \leftarrow \{Q_1, \ldots, Q_m\}$ and check whether it is satisfiable. This fails because even if the system is globally unsatisfiable, each *individual* constraint Q_i might still have a satisfying assignment. Similarly, picking a uniformly random *linear* combination $\sum_i \alpha_i Q_i$ is not obviously helpful: the combined equation might have solutions even when the full system has none.

A polynomial combination (warm-up) The notes suggest a more structured random combination. Choose a random field element $r \in \mathbb{F}$ and form the polynomial

$$Q_r(x) \;:=\; \sum_{i=1}^{m} r^i \, Q_i(x).$$

If there exists a common satisfying assignment $a^\star$ with $Q_i(a^\star) = 0$ for all i, then $Q_r(a^\star) = 0$ for every r (perfect completeness).

For a *fixed* assignment a, if not all $Q_i(a)$ are zero then

$$p_a(r) := Q_r(a) = \sum_{i=1}^{m} r^i Q_i(a)$$

is a nonzero univariate polynomial of degree at most m, and therefore

$$\Pr_{r \in \mathbb{F}}[Q_r(a) = 0] \;\le\; \frac{m}{|\mathbb{F}|}.$$

This is a Schwartz–Zippel type bound.

However, this does *not* immediately imply a bound on $\Pr_r[\exists a \; Q_r(a) = 0]$, because the satisfying assignment a could depend on r. The notes therefore move to a refined construction that uses multiple random field elements and keeps the effective "degree" small.

A refined combination with fewer random bits Let $H \subseteq \mathbb{F}$ be a subset of size $|H| = O(\log m)$, and write

$$k \;:=\; \frac{\log m}{\log |H|} \;=\; O\!\left(\frac{\log m}{\log \log m}\right).$$

We can index the m constraints by the grid H^k (since $|H|^k = m$), i.e., we relabel them as $\{Q_{j_1,\ldots,j_k}\}_{(j_1,\ldots,j_k)\in H^k}$. Sample $r_1, \ldots, r_k \in \mathbb{F}$ uniformly at random and define

$$Q_{r_1,\ldots,r_k}(x) \;:=\; \sum_{(j_1,\ldots,j_k)\in H^k} \left(\prod_{t=1}^{k} r_t^{j_t}\right) Q_{j_1,\ldots,j_k}(x). \tag{8.1}$$

The number of monomials $\prod_{t=1}^{k} r_t^{j_t}$ with each $j_t \in H$ is $|H|^k = m$.

- **Completeness:** If $a^\star$ satisfies all constraints, then $Q_{r_1,\ldots,r_k}(a^\star) = 0$ for all $(r_1, \ldots, r_k)$.
- **Soundness (as stated in the notes):** If the system has no common satisfying assignment, then

$$\forall a \in \mathbb{F}^n, \; \Pr_{r_1,\ldots,r_k\in\mathbb{F}}\big[Q_{r_1,\ldots,r_k}(a) = 0\big] \;\le\; \frac{|H|^k}{|\mathbb{F}|} \;=\; \frac{m}{|\mathbb{F}|}. \tag{8.2}$$

A key point for parameters is the randomness cost: sampling $r_1, \ldots, r_k \in \mathbb{F}$ uses $k \log |\mathbb{F}|$ random bits, which for $|\mathbb{F}| = \text{poly}(\log m)$ is

$$k \log |\mathbb{F}| = O\!\left(\frac{\log m}{\log\log m} \cdot \log\log m\right) = O(\log m).$$

8.3 Issue 2: feasibility of one quadratic equation

We now focus on checking the statement: For an assignment $a \in \mathbb{F}^n$ committed by the prover,

$$Q(a) = 0,$$

where Q is a single quadratic polynomial. Write (as in the notes) the quadratic form

$$Q(a) = \sum_{i,j\in[n]} C_{ij}\, a_i a_j \;+\; \sum_{i\in[n]} C_i\, a_i \;+\; C,$$

for coefficients $C_{ij}, C_i, C \in \mathbb{F}$. For simplicity the notes focus on the quadratic part $\sum_{i,j} C_{ij} a_i a_j$; the linear and constant terms can be handled in the same way by folding them into the sum-check expression.

The goal is to reduce checking $Q(a) = 0$ to a *sum-check protocol* instance.

8.4 Step 1: low-degree extensions

We set up the arithmetization so that the verifier can access the alleged assignment through evaluations of a low-degree polynomial.

Embedding $[n]$ into a small grid. Choose a set $H \subseteq \mathbb{F}$ with $|H| = \Theta(\log n)$, and let

$$k \;:=\; \frac{\log n}{\log |H|} \;=\; \Theta\!\left(\frac{\log n}{\log\log n}\right) \quad \text{so that} \quad |H|^k = n.$$

Fix a bijection $\tau : H^k \leftrightarrow [n]$. Given an assignment $a \in \mathbb{F}^n$, define the function $\widehat{a} : H^k \to \mathbb{F}$ by

$$\widehat{a}(y) := a_{\tau(y)} \qquad (y \in H^k).$$

Low-degree extension. Let $\widetilde{a} : \mathbb{F}^k \to \mathbb{F}$ be the unique polynomial that agrees with $\widehat{a}$ on H^k and has *individual degree* $< |H|$ in every variable. Equivalently, $\widetilde{a}$ is obtained by multivariate Lagrange interpolation over the grid H^k.

FACT 8.2 (individual degree bound for $\widetilde{a}$). *The low-degree extension $\widetilde{a} : \mathbb{F}^k \to \mathbb{F}$ has individual degree $\deg_{x_t}(\widetilde{a}) \le |H|-1$ for each coordinate $t \in [k]$. In particular, $\deg_{x_t}(\widetilde{a}) = O(\log n)$.*

Proof from the notes (Lagrange basis). For $y = (y_1, \ldots, y_k) \in H^k$, define the univariate Lagrange basis polynomials

$$L_{y_t}(x_t) := \prod_{z \in H \setminus \{y_t\}} \frac{x_t - z}{y_t - z}.$$

This polynomial has degree $|H| - 1$ and satisfies $L_{y_t}(y_t) = 1$ and $L_{y_t}(z) = 0$ for all $z \in H \setminus \{y_t\}$. The multivariate interpolation formula is

$$\widetilde{a}(x) = \sum_{y \in H^k} \left(\prod_{t=1}^{k} L_{y_t}(x_t) \right) \widehat{a}(y).$$

Each variable x_t appears only inside $L_{y_t}(x_t)$, which has degree $|H| - 1$. Hence $\deg_{x_t}(\widetilde{a}) \le |H| - 1$ for every t. □

Low-degree extension of the coefficients. Similarly, view the coefficient matrix as a function $C : [n] \times [n] \to \mathbb{F}$ and transfer it to a function on $H^k \times H^k$ via τ. Let $\widetilde{C} : \mathbb{F}^k \times \mathbb{F}^k \to \mathbb{F}$ be its low-degree extension (individual degree $< |H|$ in each of the $2k$ variables).

8.5 Step 2: reducing $Q(a) = 0$ to sum-check

Define the polynomial in $2k$ variables

$$\widetilde{q}(x, y) \;:=\; \widetilde{C}(x, y) \cdot \widetilde{a}(x) \cdot \widetilde{a}(y), \qquad x, y \in \mathbb{F}^k.$$

Then the quadratic form can be written as a sum over the grid:

$$Q(a) = \sum_{x \in H^k} \sum_{y \in H^k} \widetilde{C}(x, y)\, \widehat{a}(x)\, \widehat{a}(y) = \sum_{x \in H^k} \sum_{y \in H^k} \widetilde{q}(x, y).$$

Therefore, checking $Q(a) = 0$ reduces to verifying the claim

$$\sum_{x \in H^k} \sum_{y \in H^k} \widetilde{q}(x, y) \;=\; 0. \tag{8.3}$$

Degree bookkeeping. Each variable has individual degree

$$\deg_{x_t}(\widetilde{q}) \le \deg_{x_t}(\widetilde{C}) + \deg_{x_t}(\widetilde{a}) \le O(|H|) = O(\log n),$$

and similarly for the y-variables. Thus the *total* degree over all $2k$ variables is at most

$$\deg(\widetilde{q}) \;\le\; 2k \cdot O(|H|) = O\left(\frac{\log^2 n}{\log \log n} \right).$$

The sum-check protocol (sketch). The sum-check protocol is an interactive proof for (8.3). It proceeds for $2k$ rounds, one per variable in $(x_1, \dots, x_k, y_1, \dots, y_k)$. To avoid confusing these with the original quadratic constraints Q_i, we denote the prover's round-t message by g_t (the handwritten notes label these $q_1, q_2, \dots$).

- In round 1, the prover sends a univariate polynomial $g_1(z)$ that is supposed to equal $\sum_{x_2,\dots,x_k \in H} \sum_{y \in H^k} \widetilde{q}(z, x_2, \dots, x_k, y)$. The verifier checks $\sum_{z \in H} g_1(z) = 0$ and then chooses a random $r_1 \in \mathbb{F}$.
- In round $t \geq 2$, after the verifier has chosen $r_1, \dots, r_{t-1}$, the prover sends a univariate polynomial $g_t(z)$ intended to equal the sum of $\widetilde{q}$ over the remaining $2k - t$ variables with the first $t - 1$ variables fixed to $r_1, \dots, r_{t-1}$. The verifier checks the consistency condition
$$\sum_{z \in H} g_t(z) \;=\; g_{t-1}(r_{t-1}),$$
and then samples a fresh random $r_t \in \mathbb{F}$.
- In the final round $2k$, the verifier checks $g_{2k}(r_{2k}) = \widetilde{q}(r_1, \dots, r_{2k})$. This last equality can be verified by querying $\widetilde{C}$ and $\widetilde{a}$ at the needed points and multiplying the returned field elements:
$$\widetilde{q}(r_1, \dots, r_{2k}) = \widetilde{C}(r_1, \dots, r_{2k}) \cdot \widetilde{a}(r_1, \dots, r_k) \cdot \widetilde{a}(r_{k+1}, \dots, r_{2k}).$$

From interactive proof to PCP. A PCP proof can contain a *static transcript tree* of the sum-check interaction: for every possible verifier randomness string $(r_1, \dots, r_{2k})$, the proof stores the corresponding prover messages $g_1, \dots, g_{2k}$. The verifier samples a random path and reads only the nodes on that path. Since the branching factor is $|\mathbb{F}|$ and the depth is $2k$, the tree has $|\mathbb{F}|^{2k}$ nodes, which is $|\mathbb{F}|^{O(\log n / \log\log n)}$ with our choice of k. Each node stores a univariate polynomial of degree $O(|H|) = O(\log n)$, which can be encoded by its $O(\log n)$ coefficients (or by its values on H).

8.6 Soundness and the need for a low-degree test

Assume for the moment that the oracle $\widetilde{a}$ queried by the verifier is indeed the correct low-degree extension of some assignment a. Then the standard soundness analysis of sum-check yields that a cheating prover can only make the verifier accept (8.3) with probability at most roughly
$$\frac{(\#\text{ rounds}) \cdot (\text{degree bound})}{|\mathbb{F}|} = O\!\left(\frac{k|H|}{|\mathbb{F}|}\right) = O\!\left(\frac{\log^2 n}{|\mathbb{F}| \log\log n}\right).$$
The notes indicate choosing $|\mathbb{F}|$ large enough so that this is at most $1/2$ (for example $|\mathbb{F}| \geq \text{poly}(\log n)$ suffices). Completeness is perfect: if the equation is truly satisfiable and the prover responds consistently, all checks pass.

Of course, a malicious prover is not forced to provide an actual low-degree extension $\widetilde{a}$. To complete the PCP, the verifier must also *test* that the purported oracle function is close to a low-degree polynomial. This is exactly the role of low-degree testing.

THEOREM 8.3 (low-degree test (statement from the notes)). *Given oracle access to a function $f : \mathbb{F}^m \to \mathbb{F}$, there exists a randomized q-query algorithm T such that:*

- ***Completeness:*** *If f is a polynomial of total degree at most d, then* $\Pr[T \text{ accepts}] = 1$.
- ***Soundness:*** *If f is ε-far (in relative Hamming distance) from every degree-d polynomial, then* $\Pr[T \text{ accepts}] \leq 1/2$.
- ***Query complexity:*** $q = \text{poly}(d)$.

REMARK 8.4. The notes end by relating low-degree testing to coding-theoretic notions such as locally decodable codes (LDC), locally correctable codes (LCC), and locally testable codes (LTC). In PCP constructions, low-degree extensions form a code, and low-degree tests are the local tests that certify proximity to that code.