**CS 579: Computational Complexity** Fall 2025

# Lecture 08 - September 18

*Prof. Fernando Granha Jeronimo* *Super Scribes: Connor Mowry & Abhinav Angirekula*

## 1 Introduction and Recap

This lecture featured a flipped classroom approach where students worked in small groups to prove the Baby PCP Theorem. Before diving into the proof construction, we briefly recapped the key definitions and motivation.

### 1.1 PCP Definition Recap

**Definition 1** (PCP)**.** A language $L \in \mathbf{PCP}[r(n), q(n)]$ if there exists a probabilistic polynomial-time verifier that:

- Uses $r(n)$ random bits
- Makes $q(n)$ queries to a proof $\pi$ (possibly of exponential length)
- **Completeness:** If $x \in L$, then $\exists \pi$ such that the verifier accepts with probability 1
- **Soundness:** If $x \notin L$, then $\forall \pi$, the verifier accepts with probability $\leq 1/2$

### 1.2 CSP Examples

We reviewed several constraint satisfaction problems that will be relevant:

**Example** (Standard CSPs)**.**

- **MAX-CUT:** Given a graph, partition vertices to maximize edges crossing the cut
- **3-COLORING:** Assign one of three colors to vertices so adjacent vertices differ
- **3-SAT:** Boolean formula with 3-literal clauses to satisfy

### 1.3 The PCP Theorems

**Theorem 2** (Full PCP Theorem)**.**

$$\mathbf{NP} \subseteq \mathbf{PCP}[O(\log n), O(1)]$$

**Theorem 3** (Scaled-up Version)**.**

$$\mathbf{NEXP} \subseteq \mathbf{PCP}[\mathrm{poly}(n), O(1)]$$

**Theorem 4** (Baby PCP Theorem - Today's Goal)**.**

$$\mathbf{NP} \subseteq \mathbf{PCP}[\mathrm{poly}(n), O(1)]$$

The Baby PCP Theorem is weaker than the full PCP theorem but already highly nontrivial. It allows polynomial randomness instead of logarithmic, making it more accessible to prove while still demonstrating key techniques.

## 2 Tools for the Proof

### 2.1 Error-Correcting Codes

The proof critically relies on codes with special properties. We need:

**Definition 5** (Required Code Properties)**.** A code $C : \mathbb{F}_2^k \to \mathbb{F}_2^n$ suitable for our PCP construction should be:

- **Linear:** The encoding function is linear
- **Locally Testable:** Can test membership with few queries
- **Good distance:** Relative distance $\delta = \Omega(1)$
- **Locally Decodable/Correctable:** Can recover/correct bits with few queries

**Proposition 6** (Hadamard Code Properties)**.** *The Hadamard code has all required properties:*

- *Block length:* $2^k$ *(exponential)*
- *Relative distance:* $1/2$
- *Locally testable with 3 queries (BLR test)*
- *Locally correctable with 2 queries*

### 2.2 Quadratic Equations over $\mathbb{F}_2$

We'll reduce from an NP-complete problem involving quadratic equations:

**Definition 7** (Quadratic Equations Problem)**.** Given a system of $m$ quadratic equations over $\mathbb{F}_2^n$:

$$\sum_{i,j} a_{ij}^{(1)} x_i x_j = b_1 \tag{1}$$

$$\sum_{i,j} a_{ij}^{(2)} x_i x_j = b_2 \tag{2}$$

$$\vdots \tag{3}$$

$$\sum_{i,j} a_{ij}^{(m)} x_i x_j = b_m \tag{4}$$

where $a_{ij}^{(\ell)}, b_\ell \in \mathbb{F}_2$. Deciding if this system has a solution is NP-complete.

This can be written in matrix form as:

$$Av = b$$

where $v \in \mathbb{F}_2^{n^2}$ contains all products $x_i x_j$, and $A \in \mathbb{F}_2^{m \times n^2}$.

**Remark.** The reduction from Circuit-SAT to quadratic equations is standard: encode each gate's computation as quadratic constraints.

## 3 Proof Construction: The Baby PCP

The proof proceeded through three main hints that guided the group work:

### 3.1 Hint 1: Basic Structure

**Claim.** *If we have $x, y \in \mathbb{F}_2^n$, then for a random $r \in \mathbb{F}_2^n$:*

$$\langle r, x - y \rangle \neq 0 \text{ with probability } \frac{1}{2} \text{ if } x \neq y$$

*When $x = y$, we have $\langle r, x - y \rangle = 0$ with probability 1.*

This gives us a way to check consistency of assignments using random linear combinations.

### 3.2 Hint 2: Hadamard Encoding

Instead of asking for the solution $v = x \otimes x$ directly, we ask for:

- The Hadamard encoding of $x$: $\text{Had}(x)$
- The Hadamard encoding of $x \otimes x$: $\text{Had}(x \otimes x)$

This exponentially increases the proof size but enables local testing and correction.

### 3.3 Hint 3: Checking Linear Constraints

**Claim.** *Given $Av = b \in \mathbb{F}_2^m$ where $v = x \otimes x$, we can verify this by checking:*

$$\langle r, Av \rangle = \langle r, b \rangle$$

*for random $r \in \mathbb{F}_2^m$.*

*This is equivalent to checking:*

$$\langle r^\top A, v \rangle = \langle r, b \rangle$$

## 4 The Complete Protocol

### 4.1 Proof Format

The proof $\pi$ consists of two parts:

1. $\pi_1$: Purported Hadamard encoding of solution $x \in \mathbb{F}_2^n$
2. $\pi_2$: Purported Hadamard encoding of $x \otimes x \in \mathbb{F}_2^{n^2}$

### 4.2 Verification Algorithm

**Algorithm 1** Baby PCP Verifier

**Input:** Quadratic system $(A, b)$, proof $\pi = (\pi_1, \pi_2)$
**Randomness:** $O(n^2)$ bits

**Step 1:** Test if $\pi_1$ is close to a Hadamard codeword
Sample $r, s \in \mathbb{F}_2^n$ randomly
Check: $\pi_1(r) + \pi_1(s) = \pi_1(r + s)$

**Step 2:** Test if $\pi_2$ is close to a Hadamard codeword
Sample $u, v \in \mathbb{F}_2^{n^2}$ randomly
Check: $\pi_2(u) + \pi_2(v) = \pi_2(u + v)$

**Step 3:** Test consistency between $\pi_1$ and $\pi_2$
Sample $r \in \mathbb{F}_2^n, s \in \mathbb{F}_2^n$ randomly
Query $\pi_1(r), \pi_1(s), \pi_2(r \otimes s)$
Check: $\pi_1(r) \cdot \pi_1(s) = \pi_2(r \otimes s)$

**Step 4:** Verify the quadratic equations
Sample $t \in \mathbb{F}_2^m$ randomly
Compute $q = A^\top t \in \mathbb{F}_2^{n^2}$
Query $\pi_2(q)$
Check: $\pi_2(q) = \langle t, b \rangle$

**Accept** if all checks pass

**Remark.** Some of the steps above are repeated constant many times to ensure stronger guarantees.

### 4.3 Analysis Sketch

**Theorem 8** (Completeness)**.** *If the quadratic system has a solution $x^*$, then the prover can provide:*

- $\pi_1 = \mathit{Had}(x^*)$
- $\pi_2 = \mathit{Had}(x^* \otimes x^*)$

*These encodings will pass all tests with probability 1.*

**Theorem 9** (Soundness)**.** *If the system has no solution, then for any proof $\pi$:*

- *If $\pi_1, \pi_2$ are far from valid Hadamard codewords, the linearity tests fail with good probability*
- *If they are close to codewords encoding $x$ and $v$ respectively, but $v \neq x \otimes x$, the consistency test fails*
- *If $v = x \otimes x$ but $Av \neq b$, the equation verification fails with probability $\geq 1/2$*

*Overall, the verifier rejects with probability $\geq 1/2$.*

# 5 Key Insights from the Workshop

## 5.1 Why This Construction Works

The construction cleverly combines several ideas:

1. **Arithmetization:** Converting the NP problem to algebraic equations allows probabilistic checking
2. **Error-correcting codes:** Hadamard encoding provides redundancy enabling local testing
3. **Self-correction:** Even with errors, we can recover correct values locally
4. **Tensor product structure:** Checking $v = x \otimes x$ reduces to checking random linear combinations

## 5.2 From Baby PCP to Full PCP

The Baby PCP uses polynomial randomness because:

- Hadamard encoding has exponential blowup
- We sample random vectors of dimension $n$ or $n^2$

The full PCP theorem achieves logarithmic randomness through:

- More efficient codes (e.g., Reed-Muller codes)
- Proof composition techniques
- Recursive reduction of randomness

# 6 Student Discussion Points

During the group work, several important observations emerged:

**Remark** (On Code Choice)**.** Students recognized that the Hadamard code, despite its exponential blowup, provides exactly the properties needed: it's linear, locally testable with the BLR test, and enables checking that $f(x) + f(y) = f(x + y)$ with just 3 queries.

**Remark** (On Problem Reduction)**.** The choice of quadratic equations over $\mathbb{F}_2$ as the NP-complete problem is crucial—it naturally connects to the tensor product structure and allows clean algebraic manipulation.

# 7 Conclusion

The Baby PCP Theorem, while weaker than the full PCP Theorem, demonstrates all the key ideas:

- Probabilistic verification of deterministic statements
- Trading proof length for query complexity
- Using error-correcting codes for robustness
- Algebraic techniques for constraint verification

This hands-on approach to proving the theorem helps build intuition for the more sophisticated techniques used in the full PCP Theorem and its applications to hardness of approximation.