PCP Theorem
[PCP Theorem is] the most important result in complexity theory since Cook’s Theorem.

Ingo Wegener, 2005


The 1st proof is algebraic, the 2nd one is combinatorial and non-elementary.
Two ways to view the PCP Theorem:

- It is a result about locally testable proof systems.
- It is a result about hardness of approximation.

PCP = Probabilistically Checkable Proof
Synopsis

1. Approximation Algorithm
2. Two Views of PCP Theorem
3. Equivalence of the Two Views
4. Inapproximability
5. The Fourier Transform Technique
6. Efficient Conversion of NP Certificate to PCP Proof
7. Proof of PCP Theorem
8. Threshold Result by Håstad’s 3-Bit PCP Theorem
9. Proof of Håstad’s 3-Bit PCP Theorem
10. Historical Remark
Approximation Algorithm
Since the discovery of NP-completeness in 1972, researchers had been looking for approximate solutions to NP-hard optimization problems, with little success.

The discovery of PCP Theorem in 1992 explains the difficulty.
Suppose $\rho : \mathbb{N} \to (0, 1)$. A $\rho$-approximation algorithm $A$ for a maximum, respectively minimum optimization problem satisfies

$$\frac{A(x)}{\text{Max}(x)} \geq \rho(|x|),$$

respectively

$$\frac{\text{Min}(x)}{A(x)} \geq \rho(|x|)$$

for all $x$. 
SubSet-Sum

Given $m$ items of sizes $s_1, s_2, \ldots, s_m$, and a positive integer $C$, find a subset of the items that maximizes the total sum of their sizes without exceeding the capacity $C$.

- There is a well-known dynamic programming algorithm.
- Using the algorithm and a parameter $\epsilon$ a $(1-\epsilon)$-approximation algorithm can be designed that runs in $O \left( \left( \frac{1}{\epsilon} - 1 \right) \cdot n^2 \right)$ time.
- We say that SubSetSum has an FPTAS.
Let $U = \{u_1, u_2, \ldots, u_m\}$ be the set of items to be packed in a knapsack of size $C$. For $1 \leq j \leq m$, let $s_j$ and $v_j$ be the size and value of the $j$-th item, respectively.

The objective is to fill the knapsack with items in $U$ whose total size is at most $C$ and such that their total value is maximum.

- There is a similar dynamic programming algorithm.
- Using the algorithm and a parameter $\epsilon$ one can design a $(1-\epsilon)$-approximation algorithm of $O \left( \left( \frac{1}{\epsilon} - 1 \right) \cdot n^{\frac{1}{\epsilon}} \right)$ time.
- We say that KnapSack has a PTAS.
Max-3SAT

For each 3CNF $\varphi$, the value of $\varphi$, denoted by $\text{val}(\varphi)$, is the maximum fraction of clauses that can be satisfied by an assignment to the variables of $\varphi$.

- $\varphi$ is satisfiable if and only if $\text{val}(\varphi) = 1$.

Max-3SAT is the problem of finding the maximum $\text{val}(\varphi)$.

- A simple greedy algorithm for Max-3SAT is $\frac{1}{2}$-approximate.
- We say that Max-3SAT is in APX.

By definition, $\text{FPTAS} \subseteq \text{PTAS} \subseteq \text{APX} \subseteq \text{OPT}$. We will see that the inclusions are strict assuming $P \neq \text{NP}$.
Max-IS

\[ \text{Min-VC} + \text{Max-IS} = m. \]

A \( \frac{1}{2} \)-approximation algorithm for Min-VC.

1. Pick up a remaining edge and collect the two end nodes.
2. Remove all edges adjacent to the two nodes.
3. Goto Step 1 if there is at least one remaining edge.

▶ Is Min-VC in PTAS?
▶ Is Max-IS in APX?
A breakthrough in the study of approximation algorithm was achieved in early 1990's.

[1991]. There is no $2^\log^{1-\epsilon}(n)$-approximation algorithm for Max-IS unless SAT $\in$ SUBEXP.

[1992]. Max-IS is not in APX if $P \neq NP$.


Two Views of PCP Theorem
Surprisingly, $\text{IP} = \text{PSPACE}$. More surprisingly, $\text{MIP} = \text{NEXP}$.

The latter can be interpreted as saying that nondeterminism can be traded off for randomness + interaction.
Suppose $L$ is an NP problem and $x$ is an input string.

1. Prover provides a proof $\pi$ of polynomial length.
2. Verifier uses at most logarithmic many random bits, and makes a constant number of queries on $\pi$.

▶ A query $i$ is a location of logarithmic length. The answer to query $i$ is $\pi(i)$.
▶ Queries are written on a special address/oracle tape.
▶ We assume that verifier is nonadaptive in that its selection of queries is based only on input and random string.
Suppose $L$ is a language and $q, r : \mathbb{N} \rightarrow \mathbb{N}$.

$L$ has an $(r(n), q(n))$-PCP verifier if a P-time verifier $V$ exists satisfying the following.

- **Efficiency.** On input $x$ and given access to any location of a proof $\pi$ of length $\leq q(n)2^{r(n)}$, the verifier $V$ uses at most $r(n)$ random bits and makes at most $q(n)$ nonadaptive queries to the proof $\pi$ before it outputs ‘1’ or ‘0’.
- **Completeness.** If $x \in L$, then $\exists \pi. \Pr[V^\pi(x) = 1] = 1$.
- **Soundness.** If $x \notin L$, then $\forall \pi. \Pr[V^\pi(x) = 1] \leq 1/2$.

$V^\pi(x)$ denotes the random variable with $x$ and $\pi$ fixed.
1. Proof length ≤ \( q(n)2^{r(n)} \).
   ▶ At most \( q(n)2^{r(n)} \) locations can be queried by verifier.

2. \( L \in \text{NTIME}(q(n)2^{O(r(n))}) \).
   ▶ An algorithm guesses a proof of length \( q(n)2^{r(n)} \).
   ▶ It executes deterministically \( 2^{r(n)} \) times the verifier’s algorithm.
   ▶ The total running time is bounded by \( q(n)2^{O(r(n))} \).

Both random bits and query time are resources. An \((r(n), q(n))\)-PCP verifier has
   ▶ randomness complexity \( r(n) \) and
   ▶ query complexity \( q(n) \).

Sometimes one is concerned with proof complexity \( q(n)2^{r(n)} \).
A language is in $\text{PCP}(r(n), q(n))$ if it has a $(cr(n), dq(n))$-PCP verifier for some $c, d$.

$\text{PCP}(r(n), q(n)) \subseteq \text{NTIME}(q(n)2^{O(r(n))})$.

- $\text{PCP}(0, \log) = \text{P}$.
- $\text{PCP}(0, \text{poly}) = \text{NP}$.
- $\text{PCP}(\log, \text{poly}) = \text{NP}$.
1. $\text{PCP}(\text{poly}, \text{poly}) \subseteq \text{NEXP}$.
2. $\text{PCP}(\text{log}, \text{log}) \subseteq \text{NP}$.
3. $\text{PCP}(\text{log}, 1) \subseteq \text{NP}$.

In three influential papers in the history of PCP, it is proved that the above ‘$\subseteq$’ can be strengthened to ‘$=$’.
The PCP Theorem

**PCP Theorem.** $\text{NP} = \text{PCP}(\log, 1)$.

Every NP-problem has specifically chosen certificates whose correctness can be verified probabilistically by checking only 3 bits.
Example

\[ \text{GNI} \in \text{PCP}(\text{poly}, 1). \]

- Suppose both \( G_0 \) and \( G_1 \) have \( n \) vertices.
- Proofs of size \( 2^{n^2} \) are indexed by adjacent matrix representations.
  - If the location, a string of size \( n^2 \), represents a graph isomorphic to \( G_i \), it has value \( i \).
- The verifier picks up \( b \in \{0, 1\} \) at random, produces a random permutation of \( G_b \), and queries the bit of the proof at the corresponding location.
Can we scale down **PCP Theorem** further?

---

**Fact.** If $\text{NP} \subseteq \text{PCP}(o(\log), o(\log))$, then $P = \text{NP}$.
**Theorem.** $\text{PCP}(\text{poly}, 1) = \text{NEXP}$. 
Hardness of Approximation Viewpoint

For many NP-hard optimization problems, computing approximate solutions is no easier than computing the exact solutions.
The PCP Theorem, Hardness of Approximation

**PCP Theorem.** There exists $\rho < 1$ such that for every $L \in \text{NP}$ there is a P-time computable function $f : L \rightarrow 3\text{SAT}$ such that

- $x \in L \Rightarrow \text{val}(f(x)) = 1,$
- $x \notin L \Rightarrow \text{val}(f(x)) < \rho.$

▶ Figure out the significance of the theorem by letting $L = 3\text{SAT}.$
The PCP Theorem, Hardness of Approximation

**PCP Theorem** cannot be proved using Cook-Levin reduction.

- $\text{val}(f(x))$ tends to 1 even if $x \notin L$.

“The intuitive reason is that computation is an inherently unstable, non-robust mathematical object, in the sense that it can be turned from non-accepting to accepting by changes that would be insignificant in any reasonable metric.”

Papadimitriou and Yannakakis, 1988
Corollary. There exists some $\rho < 1$ such that if there is a P-time $\rho$-approximation algorithm for Max-3SAT then $P = NP$.

- The $\rho$-approximation algorithm for Max-3SAT is NP-hard.
Equivalence of the Two Views
CSP, Constraint Satisfaction Problem

If \( q \) is a natural number, then a \( q \)CSP instance \( \varphi \) with \( n \) variables is a collection of constraints \( \varphi_1, \ldots, \varphi_m : \{0, 1\}^n \rightarrow \{0, 1\} \) such that for each \( i \in [m] \) the function \( \varphi_i \) depends on \( q \) of its input locations.

We call \( q \) the arity of \( \varphi \), and \( m \) the size of \( \varphi \).

An assignment \( u \in \{0, 1\}^n \) satisfies a constraint \( \varphi_i \) if \( \varphi_i(u) = 1 \). Let

\[
\text{val}(\varphi) = \max_{u \in \{0, 1\}^n} \left\{ \frac{\sum_{i=1}^n \varphi_i(u)}{m} \right\}.
\]

We say that \( \varphi \) is satisfiable if \( \text{val}(\varphi) = 1 \).

\( q \)CSP is a generalization of 3SAT.
1. We assume that $n \leq qm$.

2. Since every $\varphi_i$ can be described by a formula of size $q2^q$, and every variable can be coded up by $\log n$ bits, a $q$CSP instance can be described by $O(mq2^q \log n)$ bits.

3. The greedy algorithm for MAX-3SAT can be applied to MAX$q$CSP to produce an assignment satisfying $\geq \frac{\text{val}(\varphi)}{2^q} m$ constraints.
Suppose \(q \in \mathbb{N}\) and \(\rho \leq 1\).

Let \(\rho\text{-GAP}q\text{CSP}\) be the promise problem of determining if a \(q\text{CSP}\) instance \(\varphi\) satisfies either (1) \(\text{val}(\varphi) = 1\) or (2) \(\text{val}(\varphi) < \rho\).

We say that \(\rho\text{-GAP}q\text{CSP}\) is \textbf{NP-hard} if for every NP-problem \(L\) some P-time computable function \(f : L \rightarrow \rho\text{-GAP}q\text{CSP}\) exists such that

\[
\begin{align*}
    x \in L & \implies \text{val}(f(x)) = 1, \\
    x \notin L & \implies \text{val}(f(x)) < \rho.
\end{align*}
\]

\textbf{PCP Theorem.} There exists some \(\rho \in (0, 1)\) such that \(\rho\text{-GAP}3\text{SAT}\) is NP-hard.
**PCP Theorem.** There exist $q \in \mathbb{N}$ and $\rho \in (0, 1)$ such that $\rho$-GAP$q$CSP is NP-hard.
Equivalence Proof

**PCP Theorem** ⇒ **PCP Theorem**.

This is essentially the Cook-Levin reduction.

1. Suppose $\textbf{NP} \subseteq \textbf{PCP}(\log, 1)$. Then 3SAT has a PCP verifier $V$ that makes $q$ queries using $c \log n$ random bits.

2. Given input $x$ with $|x| = n$ and random string $r \in \{0, 1\}^{c \log n}$, $V(x, r)$ is a Boolean function of type $\{0, 1\}^q \rightarrow \{0, 1\}$.

3. $\varphi = \{V(x, r)\}_{r \in \{0, 1\}^{c \log n}}$ is a P-size qCSP instance.
   - By completeness, $x \in 3\text{SAT} \Rightarrow \text{val}(\varphi) = 1$.
   - By soundness, $x \not\in 3\text{SAT} \Rightarrow \text{val}(\varphi) \leq \frac{1}{2}$.

4. The map from 3SAT to $\frac{1}{2}$-GAPqCSP is P-time computable.
   - $V$ runs in P-time.
Suppose $L \in \mathbf{NP}$ and $\rho$-GAP$q$CSP is NP-hard for some $q \in \mathbf{N}$, $\rho < 1$. By assumption there is some P-time reduction $f : L \rightarrow \rho$-GAP$q$CSP.

1. The verifier for $L$ works as follows:
   - On input $x$, compute the $q$CSP instance $f(x) = \{\varphi_i\}_{i \in [m]}$.
   - Given a proof $\pi$, which is an assignment to the variables, it randomly chooses $i \in [m]$ and checks if $\varphi_i$ is satisfied by reading the relevant $q$ bits of the proof.

2. If $x \in L$, the verifier always accepts; otherwise it accepts with probability $< \rho$. 

**PCP Theorem $\iff$ PCP Theorem.**
Equivalence Proof

**PCP Theorem ⇔ PCP Theorem.**

This is very much like the equivalence between SAT and 3SAT.

1. Let $\epsilon > 0$ and $q \in \mathbb{N}$ be such that $(1-\epsilon)$-GAP$q$CSP is NP-hard.
2. Let $\varphi = \{\varphi_i\}_{i=1}^m$ be a $q$CSP instance with $n$ variables.
3. Each $\varphi_i$ is the conjunction of at most $2^q$ clauses, each being the disjunction of at most $q$ literals.
4. If all assignments fail at least an $\epsilon$ fragment of the constraints of $\varphi$, then all assignments fail at least a $\frac{\epsilon}{q2^q}$ fragment of the clauses of the 3SAT instance.
The equivalence of the proof view and the inapproximability view is essentially due to the Cook-Levin Theorem for PTM.

<table>
<thead>
<tr>
<th><strong>Proof View</strong></th>
<th><strong>Inapproximability View</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>PCP verifier $V$</td>
<td>CSP instance $\varphi$</td>
</tr>
<tr>
<td>PCP proof $\pi$</td>
<td>assignment to variables $u$</td>
</tr>
<tr>
<td>proof length $</td>
<td>\pi</td>
</tr>
<tr>
<td>number of queries $q$</td>
<td>arity of constraints $q$</td>
</tr>
<tr>
<td>number of random bits $r$</td>
<td>logarithm of number of constraints $\log m$</td>
</tr>
<tr>
<td>soundness parameter $\epsilon$</td>
<td>maximum fraction of violated constraints of no instance</td>
</tr>
<tr>
<td><strong>NP $\subseteq$ PCP($\log, 1$)</strong></td>
<td>$\rho$-GAP$q$CSP is NP-hard</td>
</tr>
</tbody>
</table>
Inapproximability
Min-VC and Max-IS are inherently different from the perspective of approximation.

- \( \text{Min-VC} + \text{Max-IS} = n. \)
- \( \rho \)-approximation algorithm of Max-IS \( \Rightarrow \frac{n-IS}{n-\rho IS} \)-approximation algorithm of Min-VC.
Lemma. There is a P-time computable function $f$ from 3CNF formulas to graphs that maps a formula $\varphi$ to an $n$-vertex graph $f(\varphi)$ whose independent set is of size $\text{val}(\varphi)^{\frac{n}{7}}$.

The standard Karp reduction from 3SAT to Max-IS is as follows:

- Each clause is translated to a clique of 7 nodes, each node represents a (partial) assignment that validates the clause.
- Two nodes from two different cliques are connected if and only if they conflict.

A 3CNF formula $\varphi$ consisting of $m$ clauses is translated to a graph with $7m$ nodes, and an assignment satisfying $l$ clauses of $\varphi$ if and only if the graph has an independent set of size $l$. 
Theorem. The following statements are valid.

1. \( \exists \rho' < 1. \) \( \rho' \)-approximation to Min-VC is NP-hard, and

2. \( \forall \rho < 1. \) \( \rho \)-approximation to Max-IS is NP-hard.

\[ \exists \rho. \rho \text{-approximation to Max-IS is NP-hard.} \] * By PCP Theorem, \( \rho \)-approximation to Max-3SAT is NP-hard for some \( \rho \). So by Lemma \( \rho \)-approximation to Max-IS is NP-hard.

1. Referring to the map of Lemma, the minimum vertex cover has size \( n - \text{val}(\varphi) \frac{n}{7} \).

Let \( \rho' = \frac{6}{7-\rho} \). Suppose Min-VC had a \( \rho' \)-approximation algorithm.

- If \( \text{val}(\varphi) = 1 \), it would produce a vertex cover of size \( \leq \frac{1}{\rho'}(n - \frac{n}{7}) = n - \rho \frac{n}{7} \).
- If \( \text{val}(\varphi) < \rho \), the minimum vertex cover has size \( \geq n - \rho \frac{n}{7} \). The algorithm must return a vertex cover of size \( > n - \rho \frac{n}{7} \).

The first proposition is established. The second will be proved by making use of \([\_]\)*.
2. Assume that Max-IS were $\rho_0$-approximate. Let $k$ satisfy $\rho_0 \binom{I_S}{k} > \binom{\rho I_S}{k}$.

1. Suppose $G$ is the input graph. Construct $G^k$ as follows:
   - The vertices are $k$-size subsets of $V_G$;
   - Two vertices $S_1, S_2$ are disconnected if $S_1 \cup S_2$ is an independent set of $G$.

2. Apply the $\rho_0$-approximation algorithm to $G^k$, and derive an independent set of $G$ from the output of the algorithm.

\begin{itemize}
  \item The largest independent set of $G^k$ is of size $\binom{I_S}{k}$, where $I_S$ is the maximum independent set of $G$.
  \item The output is an independent set of size $\geq \rho_0 \binom{I_S}{k} > \binom{\rho I_S}{k}$.
  \item An independent set of size $\rho I_S$ can be derived. This is a contradiction.
\end{itemize}
The Fourier Transform Technique
A Boolean function $f : \{0, 1\}^n \rightarrow \{0, 1\}$ is a **linear function** if

$$f(x + y) = f(x) + f(y),$$

where “$+$” is the exclusive-or operator.
Fourier Transform over $GF(2)^n$

Boolean functions can be studied using Fourier transform over $GF(2)^n$.

We shall use $\{+1, -1\}$ instead of $\{0, 1\}$ in this section.

- $0 \leftrightarrow (-1)^0 = 1$ and $1 \leftrightarrow (-1)^1 = -1$.
- $\{0, 1\}^n$ is turned into $\{\pm 1\}^n$.
- "exclusive-or" is turned into "multiplication".
Fourier Transform over $GF(2)^n$

The $2^n$-dimensional Hilbert space $\mathbb{R}^{\{\pm1\}^n}$ is defined as follows: For $f, g \in \mathbb{R}^{\{\pm1\}^n}$,

1. $(f + g)(x) = f(x) + g(x)$,
2. $(cf)(x) = cf(x)$, and
3. expectation inner product: $\langle f, g \rangle = E_{x \in \{\pm1\}^n}[f(x)g(x)]$.

Standard orthogonal basis: $\{e_x\}_{x \in \{\pm1\}^n}$. 

---

Computational Complexity, by Fu Yuxi
Fourier Transform over $GF(2)^n$

Fourier Basis: $\{\chi_\alpha\}_{\alpha \subseteq [n]}$, where $\chi_\alpha(x) = \prod_{i \in \alpha} x_i$.

1. $\chi_{\emptyset} = 1$.
2. We will see that Fourier basis functions are the same as the linear functions.
3. Inner product $\langle \chi_\alpha, \chi_\beta \rangle = 1$ if $\alpha = \beta$ and $= 0$ otherwise.

Fourier basis is orthonormal.

- $\langle \chi_\alpha, \chi_\alpha \rangle = \mathbb{E}_{x \in \{\pm 1\}^n}[\chi_\alpha(x)\chi_\alpha(x)] = \mathbb{E}_{x \in \{\pm 1\}^n}[\chi_\alpha(x \cdot x)] = \mathbb{E}_{x \in \{\pm 1\}^n}[1] = 1$.
- $\langle \chi_\alpha, \chi_\beta \rangle = \mathbb{E}_{x \in \{\pm 1\}^n}[\chi_\alpha(x)\chi_\beta(x)] = 0$ if $\alpha \neq \beta$.

The notation $y \cdot z$ stands for the dot product of $y$ and $z$. 

Computational Complexity, by Fu Yuxi
Fourier Transform over $GF(2)^n$

\[ f = \sum_{\alpha \subseteq [n]} \hat{f}_\alpha \chi_\alpha \text{ for every } f \in \mathbb{R}^{\{\pm 1\}^n}, \text{ where } \hat{f}_\alpha \text{ is the } \alpha\text{th Fourier coefficient of } f. \]

**Lemma.** (i) $\langle f, g \rangle = \sum_{\alpha \subseteq [n]} \hat{f}_\alpha \hat{g}_\alpha$. (ii) (Parsevals Identity) $\langle f, f \rangle = \sum_{\alpha \subseteq [n]} \hat{f}_\alpha^2$.

**Proof.**

\[ \langle f, g \rangle = \langle \sum_{\alpha \subseteq [n]} \hat{f}_\alpha \chi_\alpha, \sum_{\beta \subseteq [n]} \hat{g}_\beta \chi_\beta \rangle = \sum_{\alpha, \beta \subseteq [n]} \hat{f}_\alpha \hat{g}_\beta \langle \chi_\alpha, \chi_\beta \rangle = \sum_{\alpha \subseteq [n]} \hat{f}_\alpha \hat{g}_\alpha. \]

\[\square\]
Fourier Transform over $GF(2)^n$

Example.

1. Majority function of 3 variables $= \frac{1}{2} u_1 + \frac{1}{2} u_2 + \frac{1}{2} u_3 - \frac{1}{2} u_1 u_2 u_3$.

2. Projection function $\lambda x_1 \ldots x_n.x_i$. Here $\hat{f}_\alpha$ is 1 if $\alpha = \{i\}$ and is 0 if $\alpha \neq \{i\}$.
Fourier Transform over $GF(2)^n$

**Theorem.** Suppose $f : \{\pm 1\}^n \to \{\pm 1\}$ satisfies $\Pr_{x,y}[f(x \cdot y) = f(x)f(y)] \geq \frac{1}{2} + \epsilon$. Then there is some $\alpha \subseteq [n]$ such that $\hat{f}_\alpha \geq 2\epsilon$.

The assumption is equivalent to $E_{x,y}[f(x \cdot y)f(x)f(y)] \geq \frac{1}{2} + \epsilon - \left(\frac{1}{2} - \epsilon\right) = 2\epsilon$. Now

$$2\epsilon \leq E_{x,y}[f(x \cdot y)f(x)f(y)] = E_{x,y}[(\sum_\alpha \hat{f}_\alpha \chi_\alpha(x \cdot y))(\sum_\beta \hat{f}_\beta \chi_\beta(x))(\sum_\gamma \hat{f}_\gamma \chi_\gamma(y))]$$

$$= E_{x,y}[(\sum_\alpha \hat{f}_\alpha \hat{f}_\beta \hat{f}_\gamma \chi_\alpha(x)\chi_\beta(y)\chi_\gamma(x))]$$

$$= \sum_{\alpha,\beta,\gamma} \hat{f}_\alpha \hat{f}_\beta \hat{f}_\gamma E_{x,y}[\chi_\alpha(x)\chi_\beta(y)]E_y[\chi_\gamma(y)]$$

$$= \sum_{\alpha,\beta,\gamma} \hat{f}_\alpha \hat{f}_\beta \hat{f}_\gamma E_{x}[\chi_\alpha(x)\chi_\beta(x)]E_y[\chi_\gamma(y)]$$

$$= \sum_{\alpha} \hat{f}_\alpha^3 \leq \left(\max_\alpha \hat{f}_\alpha\right) \sum_{\alpha} \hat{f}_\alpha^2 = \max_\alpha \hat{f}_\alpha.$$  

The last equality is due to the fact that $f$ is Boolean.
Fourier Transform over $GF(2)^n$

For Boolean functions $f, g : \{\pm 1\}^n \rightarrow \{\pm 1\}$, the inner product $\langle f, g \rangle$ is

$$|\{x \mid f(x) = g(x)\}| - |\{x \mid f(x) \neq g(x)\}|.$$ 

Referring to the Theorem, let $\hat{f}_\alpha \geq 2\epsilon$ be the largest Fourier coefficient of $f$. Then

$$\langle f, \chi_\alpha \rangle \geq 2\epsilon.$$ 

In other words $f$ coincides with the basis function $\chi_\alpha$ on $\geq \frac{1}{2} + \epsilon$ fraction of inputs.
Efficient Conversion of NP Certificate to PCP Proof
Proofs of PCP Theorems involve some interesting ways of encoding NP-certificates and the associated methods of checking if a string is a valid encoding.

One idea is to amplify any error that appears in an NP-certificate. We shall showcase how it works by looking at a problem to which the amplification power of Walsh-Hadamard Code can be exploited.

**Theorem.** $\mathbf{NP} \subseteq \mathbf{PCP}(\text{poly}(n), 1)$. 
Walsh-Hadamard Code

The Walsh-Hadamard function $\text{WH} : \{0, 1\}^n \rightarrow \{0, 1\}^{2^n}$ encodes a string of length $n$ by a function in $n$ variables over $\text{GF}(2)$:

$$\text{WH}(u) : x \mapsto u \odot x,$$

where $u \odot x = \sum_{i=1}^{n} u_i x_i \pmod{2}$.

Random Subsum Principle.

- If $u \neq v$ then for exactly half the choices of $x$, $u \odot x \neq v \odot x$. 
Walsh-Hadamard Codeword

We say that $f$ is a Walsh-Hadamard codeword if $f = WH(u)$ for some $u \in \{0, 1\}^n$.

1. Walsh-Hadamard codewords are precisely the linear functions. This is because a linear function $f$ is the same as $WH(f)$, where

$$f = \begin{pmatrix} f(e_1) \\ f(e_2) \\ \vdots \\ f(e_n) \end{pmatrix}.$$ 

2. Walsh-Hadamard codewords are essentially the Fourier basis functions.
Let $\rho \in [0, 1]$. The functions $f, g : \{0, 1\}^n \to \{0, 1\}$ are $\rho$-close if

$$\Pr_{x \in \{0, 1\}^n} [f(x) = g(x)] \geq \rho.$$ 

**Theorem.** Let $f : \{0, 1\}^n \to \{0, 1\}$ be such that for some $\rho > \frac{1}{2}$,

$$\Pr_{x,y \in \{0,1\}^n} [f(x + y) = f(x) + f(y)] \geq \rho.$$ 

Then $f$ is $\rho$-close to a linear function.

**Proof.**

Let $\rho = \frac{1}{2} + \epsilon$. This follows immediately from the theorem of the previous section.
Local Testing of Walsh-Hadamard Codeword

A local test of $f$ checks if $f$ is a Walsh-Hadamard codeword by making a constant number of queries.

- It accepts every linear function, and
- it rejects every function that is far from being linear with high probability.

For $\delta \in (0, 1/2)$ a $(1 - \delta)$-linearity test rejects with probability $\geq \frac{1}{2}$ any function not $(1 - \delta)$-close to a linear function by testing

$$f(x + y) = f(x) + f(y)$$

randomly for $\frac{1}{\delta}$ times. 

The error probability is $\leq (1 - \delta)^{\frac{1}{\delta}} \approx \frac{1}{e} < \frac{1}{2}$. 

Computational Complexity, by Fu Yuxi
Local Decoding

Suppose $\delta < \frac{1}{4}$ and $f$ is $(1 - \delta)$-close to some linear function $\tilde{f}$.

Given $x$ one can learn $\tilde{f}(x)$ by making only two queries to $f$.

1. Choose $x' \in \mathbb{R}\{0, 1\}^n$;
2. Set $x'' = x + x'$;
3. Output $f(x') + f(x'')$.

By union bound $\tilde{f}(x) = \tilde{f}(x') + \tilde{f}(x'') = f(x') + f(x'')$ holds with probability $\geq 1 - 2\delta$. 
Quadratic Equation in GF(2)

Suppose $A$ is an $m \times n^2$ matrix and $b$ is an $m$-dimensional vector with values in GF(2). Let $(A, b) \in \text{QUADEQ}$ if there is an $n$-dimensional vector $u$ such that

$$A(u \otimes u) = b,$$

where $u \otimes u$ is the tensor product of $u$.

$u \otimes u = (u_1 u_1, \ldots, u_1 u_n, \ldots, u_n u_1, \ldots, u_n u_n)^\dagger$. 

---

Computational Complexity, by Fu Yuxi

PCP Theorem
Quadratic Equation in $GF(2)$

An instance of QUADEQ over $u_1, u_2, u_3, u_4, u_5$:

\[ u_1 u_2 + u_3 u_4 + u_1 u_5 = 1 \]
\[ u_1 u_1 + u_2 u_3 + u_1 u_4 = 0 \]

A satisfying assignment is $(0, 0, 1, 1, 0)$. 
Theorem 60

QUADEQ is NP-Complete

CKT-SAT $\leq^K$ QUADEQ.

- The inputs and the outputs are turned into variables.
- Boolean equality $x \lor y = z$ relating the inputs to the output is turned into algebraic equality $(1 - x)(1 - y) = 1 - z$ in GF(2), which is equivalent to $xx + yy + xy + zz = 0$.
- $\neg x = z$ is turned into $xx + zz = 1$.
- $x \land y = z$ is turned into $xy + zz = 0$. 
From NP Certificate to PCP Proof

A certificate for \((A, b)\) is an \(n\)-dimensional vector \(u\) witnessing \((A, b) \in \text{QUADEQ}\).

- To check if \(u\) is a solution, one reads the \(n\) bits of \(u\) and checks the \(m\) equations.

We convert an NP-certificate \(u\) to the PCP-proof \(\text{WH}(u)\text{WH}(u \otimes u)\).

- The proof is a string of length \(2^n + 2^{n^2}\).
- Using the proof it is straightforward to verify probabilistically if \((A, b) \in \text{QUADEQ}\).
Step 1. Verify that \( f, g \) are linear functions.

1. Perform a 0.999-linearity test on \( f, g \).

If successful we may assume that \( f(r) = u \odot r \) and \( g(z) = w \odot z \).
Verifier for QUADEQ

Step 2. Verify that $g$ encodes $(u \otimes u) \odot \_$. 

1. Get independent random $r, r'$.
2. Reject if $f(r)f(r') \neq g(r \otimes r')$.
3. Repeat the test 10 times.

- In a correct proof $f(r)f(r') = (\sum_i u_i r_i)(\sum_j u_j r'_j) = \sum_{i,j} u_i u_j r_i r'_j = g(r \otimes r')$.
- Assume $w \neq u \otimes u$. Let matrices $W$ and $U$ be $w$ and respectively $u \otimes u$. One has
  - $g(r \otimes r') = w \odot (r \otimes r') = \sum_{i,j} w_{ij} r_i r'_j = r W r'$, and
  - $f(r)f(r') = (u \otimes r)(u \otimes r') = (\sum_i u_i r_i)(\sum_j u_j r'_j) = r U r'$.

  $r W, r U$ differ for at least $\frac{1}{2}$ of $r$'s; and $r W r', r U r'$ differ for at least $\frac{1}{4}$ of $(r, r')$'s.
- The overall probability of rejection is at least $1 - (\frac{3}{4})^{10} > 0.9$. 
Verifier for QUADEQ

Step 3. Verify that $g$ encodes a solution.

1. Take a random subset $S$ of $[m]$.
2. Reject if $g(\sum_{k \in S} A_{k,-}) \neq \sum_{k \in S} b_k$.

There is enough time to check $A(u \otimes u) = b$ by checking, for every $k \in [m]$, the equality $g(A_{k,-}) = b_k$.

However since $m$ is part of the input, the number of queries, which must be a constant, should not depend on $m$.

If $\{k \in [m] \mid g(A_{k,-}) \neq b_k\} \neq \emptyset$, then

$$\Pr_{S \subseteq \mathbb{R}[m]}[|S \cap \{k \in [m] \mid g(A_{k,-}) \neq b_k\}| \text{ is odd}] = \frac{1}{2}.$$ 

Note that $g(\sum_{k \in S} A_{k,-}) = \sum_{k \in S} g(A_{k,-})$ by linearity.
Suppose the PCP verifier for QUADEQ makes $q_0$ queries.

It follows from the completeness of QUADEQ that all NP problems have PCP verifiers that toss coins for a polynomial number of time and make precisely $q_0$ queries.
Proof of PCP Theorem
CSP with Nonbinary Alphabet

$q_{CSP_W}$ is analogous to $q_{CSP}$ except that the alphabet is $[W]$ instead of $\{0, 1\}$. The constraints are functions of type $[W]^q \rightarrow \{0, 1\}$.

For $\rho \in (0, 1)$, we define the promise problem $\rho\text{-GAP}q_{CSP_W}$ analogous to $\rho\text{-GAP}q_{CSP}$. 
3COL is a case of $2\text{CSP}_3$. 
PCP Theorem states that $\rho$-GAP$q$CSP is NP-hard for some $q, \rho$.

The proof we shall describe is based on the following:

1. If $\varphi$ of $m$ constraints is unsatisfied, then $\text{val}(\varphi) \leq 1 - \frac{1}{m}$.
2. There is a construction that increases the gap.

The idea is to start with an NP-problem, then apply Step 2 for $\log m$ times.
Let \( f \) be a function mapping CSP instances to CSP instances. It is a \textit{CL-reduction} (complete linear-blowup reduction) if it is P-time computable and the following are valid for every CSP instance \( \varphi \).

- **Completeness.** If \( \varphi \) is satisfiable then \( f(\varphi) \) is satisfiable.
- **Linear Blowup.** If \( \varphi \) has \( m \) constraints, \( f(\varphi) \) has no more than \( Cm \) constraints over a new alphabet \( W \).
  - \( C, W \) depend on neither the number of constraints nor the number of variables of \( \varphi \).
**Main Lemma.** There exist constants $q_0 \geq 3$, $\epsilon_0 > 0$ and CL-reduction $f$ such that for every $q_0$CSP instance $\varphi$ and every $\epsilon < \epsilon_0$, $f(\varphi)$ is a $q_0$CSP instance satisfying

$$\text{val}(\varphi) \leq 1 - \epsilon \Rightarrow \text{val}(f(\varphi)) \leq 1 - 2\epsilon.$$ 

<table>
<thead>
<tr>
<th>$q_0$CSP Instance</th>
<th>Arity</th>
<th>Alphabet</th>
<th>Constraint</th>
<th>Gap</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\varphi$</td>
<td>$q_0$</td>
<td>$\text{binary}$</td>
<td>$m$</td>
<td>$1 - \epsilon$</td>
</tr>
<tr>
<td>$\Downarrow$</td>
<td>$\Downarrow$</td>
<td>$\Downarrow$</td>
<td>$\Downarrow$</td>
<td>$\Downarrow$</td>
</tr>
<tr>
<td>$f(\varphi)$</td>
<td>$q_0$</td>
<td>$\text{binary}$</td>
<td>$Cm$</td>
<td>$1 - 2\epsilon$</td>
</tr>
</tbody>
</table>

In the following proof of the PCP Theorem, $q_0$, $\epsilon_0$, $\ell$, $W$, $W'$, $d$, $t$ are all constants. They depend neither on the number $n$ of variables nor on the number $m$ of constraints.
Proof of PCP Theorem

Let $q_0 \geq 3$ and $\epsilon_0 > 0$ be given by the Main Lemma. A CL-reduction from $q_0$CSP to $(1-2\epsilon_0)$-GAP$q_0$CSP is obtained as follows:

1. $q_0$CSP is NP-hard.
2. For a $q_0$CSP instance $\varphi$ with $m$ constraints, apply Main Lemma for $\log m$ times to amplify the gap. We get an instance $\psi$.
3. If $\varphi$ is satisfiable, then $\psi$ is satisfiable. Otherwise according to Main Lemma

$$\text{val}(\psi) \leq 1 - 2^{\log m} \cdot \frac{1}{m} \leq 1 - 2\epsilon_0.$$

4. $|\psi| \leq C^{\log m}m = \text{poly}(|\varphi|)$. Conclude that $(1-2\epsilon_0)$-GAP$q_0$CSP is NP-hard.

$C$ depends on two constants, $q_0$ and 2 (the size of alphabet).
Main Lemma is proved in three steps.

1. **Prove that every** $q$**CSP** instance can be turned into a “nice” $q$**CSP** \(_W\) instance.

2. **Gap Amplification.** Construct a CL-reduction $f$ that increases both the gap and the alphabet size of a “nice” $q$**CSP** instance. [Dinur’s proof]

3. **Alphabet Reduction.** Construct a CL-reduction $g$ that decreases the alphabet size to 2 with a modest reduction in the gap. [Proof of Arora et al.]
**Gap Amplification.** For all numbers \( \ell, q \), there are number \( W, \epsilon_0 \in (0, 1) \) and a CL-reduction \( g_{\ell, q} \) such that for every \( q \text{CSP} \) instance \( \varphi, \psi = g_{\ell, q}(\varphi) \) is a \( 2\text{CSP}_W \) instance that satisfies the following for all \( \epsilon < \epsilon_0 \).

\[
\text{val}(\varphi) \leq 1 - \epsilon \Rightarrow \text{val}(\psi) \leq 1 - \ell \epsilon.
\]

**Alphabet Reduction.** There exist a constant \( q_0 \) and a CL-reduction \( h \) such that for every \( 2\text{CSP}_W \) instance \( \varphi, \psi = h(\varphi) \) is a \( q_0 \text{CSP} \) instance satisfying

\[
\text{val}(\varphi) \leq 1 - \epsilon \Rightarrow \text{val}(\psi) \leq 1 - \epsilon / 3.
\]

<table>
<thead>
<tr>
<th>CSP Instance</th>
<th>Arity</th>
<th>Alphabet</th>
<th>Constraint</th>
<th>Gap</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \varphi )</td>
<td>( q_0 )</td>
<td>binary</td>
<td>( m )</td>
<td>( 1 - \epsilon )</td>
</tr>
<tr>
<td>( f(\varphi) )</td>
<td>2</td>
<td>nonbinary</td>
<td>( C'm )</td>
<td>( 1 - 6\epsilon )</td>
</tr>
<tr>
<td>( g(f(\varphi)) )</td>
<td>( q_0 )</td>
<td>binary</td>
<td>( C''C'm )</td>
<td>( 1 - 2\epsilon )</td>
</tr>
</tbody>
</table>
Dinur makes use of expander graphs to construct new constraints.

Let $\varphi$ be a 2CSP$_W$ instance with $n$ variables. The constraint graph $G_{\varphi}$ of $\varphi$ is defined as follows:

1. the vertex set is $[n]$, and
2. $(i, j)$ is an edge if there is a constraint on the variables $u_i, u_j$. Parallel edges and self-loops are allowed.
A $2\text{CSP}_{\mathcal{W}}$ instance $\varphi$ is nice if the following are valid:

1. There is a constant $d$ such that $G_\varphi$ is a $(d, 0.9)$-expander.
2. At every vertex half of the adjacent edges are self loops.

A nice CSP instance looks like an expander. In a nice CSP a $t + 1$ step random walk is very much like a $t$ step random walk.
Lemma. Let $G$ be a $d$-regular $n$-vertex graph, $S$ be a vertex subset and $T = \overline{S}$. Then

$$|E(S, T)| \geq (1 - \lambda_G) \frac{d|S||T|}{|S| + |T|}. \quad (1)$$

The vector $\mathbf{x}$ defined below satisfies $\|\mathbf{x}\|^2 = |S||T|(|S| + |T|)$ and $\mathbf{x} \perp \mathbf{1}$.

$$x_i = \begin{cases} +|T|, & i \in S, \\ -|S|, & i \in T. \end{cases}$$

Let $Z = \sum_{i,j} A_{i,j}(x_i - x_j)^2$. By definition $Z = \frac{2}{d} |E(S, T)|(|S| + |T|)^2$. On the other hand

$$Z = \sum_{i,j} A_{i,j}x_i^2 - 2\sum_{i,j} A_{i,j}x_ix_j + \sum_{i,j} A_{i,j}x_j^2 = 2\|x\|^2 - 2\langle x, Ax \rangle.$$

Since $\mathbf{x} \perp \mathbf{1}$, $\langle x, Ax \rangle \leq \lambda_G \|x\|^2$ (cf. Rayleigh quotient).
Let $G = (V, E)$ be an expander and $S \subseteq V$ with $|S| \leq |V|/2$. The following holds.

$$
\Pr_{(u,v)\in E}[u \in S, v \in S] \leq \frac{|S|}{|V|} \left( \frac{1}{2} + \frac{\lambda_G}{2} \right). \tag{2}
$$

Observe that $|S|/|V| = \Pr_{(u,v)\in E}[u \in S, v \in S] + \Pr_{(u,v)\in E}[u \in S, v \in \overline{S}]$. And by (1), one has

$$
\Pr_{(u,v)\in E}[u \in S, v \in \overline{S}] = \frac{E(S, \overline{S})}{d|V|} \geq \frac{|S|}{|V|} \cdot \frac{1}{2} \cdot (1 - \lambda_G).
$$

We are done by substituting $|S|/|V| - \Pr_{(u,v)\in E}[u \in S, v \in S]$ for $\Pr_{(u,v)\in E}[u \in S, v \in \overline{S}]$.

$$
\Pr_{(u,v)\in E^\ell}[u \in S, v \in S] \leq \frac{|S|}{|V|} \left( \frac{1}{2} + \frac{\lambda_G^\ell}{2} \right). \tag{3}
$$
Step 1: Reduction to Nice Instance

The reduction consists of three steps.

\[ q_0 \text{CSP instance} \xrightarrow{\text{Step 1.1}} 2\text{CSP}_{2^{q_0}} \text{ instance} \]
\[ \xrightarrow{\text{Step 1.2}} 2\text{CSP}_{2^{q_0}} \text{ instance with regular constraint graph} \]
\[ \xrightarrow{\text{Step 1.3}} \text{nice } 2\text{CSP}_{2^{q_0}} \text{ instance.} \]

In all the three steps the fraction of violated constraints decreases.
Step 1: Reduction to Nice Instance

**Step 1.1.** There exists a CL-reduction that maps a $q_0$CSP instance $\varphi$ to a $2$CSP$_{2q_0}$ instance $\psi$ such that

$$\text{val}(\varphi) \leq 1 - \epsilon \Rightarrow \text{val}(\psi) \leq 1 - \frac{\epsilon}{q_0}.$$ 

Suppose $\varphi$ has variables $x_1, \ldots, x_n$ and $m$ constraints.

- The new instance $\psi$ has variables $x_1, \ldots, x_n, y_1, \ldots, y_m$, where $y_i \in \{0, 1\}^{q_0}$ codes up an assignment to $x_1, \ldots, x_n$.
- For each variable $x_j$ in $\varphi_i$, construct the constraint $\psi_{i,j}$ stating that $y_i$ satisfies $\varphi_i$ and $y_i$ is consistent to $x_j$.

What is the formula $\psi_{ij}(y_i, x_j)$? What is the size of $\psi_{ij}(y_i, x_j)$?
Step 1: Reduction to Nice Instance

Step 1.2. There exist an absolute constant $d$ and a CL-reduction that maps a $2\text{CSP}_W$ instance $\varphi$ to a $2\text{CSP}_W$ instance $\psi$ such that

$$\text{val}(\varphi) \leq 1 - \epsilon \Rightarrow \text{val}(\psi) \leq 1 - \frac{\epsilon}{100Wd}$$

and that $G_\psi$ is $d$-regular.

Let $\{G_k\}_k$ be an explicit $(d-1, 0.9)$-expander. We get $\psi$ by replacing each $k$-degree node of $G_\varphi$ by $G_k$ and adding the identity constraint (of the form $y'_i = y_{j'}$, where $i \in [n]$) to each edge $(j, j')$ of $G_k$. If $\varphi$ has $m$ constraints, $\psi$ has $dm$ constraints.

Suppose $\text{val}(\varphi) \leq 1 - \epsilon$ and $v$ is an unsatisfying assignment to $\psi$.

It suffices to prove that $v$ violates at least $\frac{\epsilon m}{100W}$ constraints of $\psi$. 

Continue on the next slide.

Fact. For every $c \in (0, 1)$ there is a constant $d$ and an algorithm that, given input $n$, runs in $\text{poly}(n)$ time and outputs an $(n, d, c)$-expander.
Step 1: Reduction to Nice Instance

Let \( u \) be the assignment to \( \varphi \) that is defined by the plurality of the assignment \( v \) to \( \psi \).

Let \( t_i \) be the number of \( v^j_i \)'s, where \( j \in [k] \), that disagree with \( u_i \). Clearly \( t_i \leq k(1 - \frac{1}{W}) \).

If \( \sum_{i=1}^{n} t_i \) is large, each \( G_k \) already contains enough violated constraints.

1. \( \sum_{i=1}^{n} t_i \geq \frac{1}{4} \epsilon m \). Let \( S_i = \{ y^j_i \mid v^j_i = u_i \} \) and let \( \overline{S}_i = \{ y^1_i, \ldots, y^k_i \} \setminus S_i \). The number of constraints of \( G_k \) violated by \( v \) is at least

\[
E(S_i, \overline{S}_i) \geq (1 - \lambda_{G_k}) \frac{(d - 1)|S_i||\overline{S}_i|}{|S_i| + |\overline{S}_i|} = \frac{1}{10} \frac{d - 1}{k} |S_i||\overline{S}_i| \geq \frac{1}{10W} t_i,
\]

where \( \geq \) is due to \( |S_i| \geq k/W \). Now \( \sum_{i \in [n]} E(S_i, \overline{S}_i) \geq \frac{\epsilon m}{40W} = \frac{\epsilon}{40Wd} \cdot dm \).

2. \( \sum_{i=1}^{n} t_i < \frac{1}{4} \epsilon m \). Since \( \text{val}(\varphi) \leq 1 - \epsilon \), there is at least \( \epsilon m \) constraints violated in \( \varphi \) by \( u \). These \( \epsilon m \) constraints are also in \( \psi \) with variables being \( v \).

Since every constraint has two variables, less than \( \frac{1}{4} \epsilon m + \frac{1}{4} \epsilon m \) constraints have valuations in \( \psi \) different from those in \( \varphi \). So at least \( \frac{1}{2} \epsilon m \) constraints are violated.
Step 1: Reduction to Nice Instance

Step 1.3. There is an absolute constant $d$ and a CL-reduction that maps a $2\text{CSP}_W$ instance $\varphi$ with $G_\varphi$ being $d'$-regular for some $d' \leq d$ to a $2\text{CSP}_W$ instance $\psi$ such that

$$\text{val}(\varphi) \leq 1 - \epsilon \Rightarrow \text{val}(\psi) < 1 - \frac{\epsilon}{10d}$$

and that $G_\psi$ is nice, $4d$-regular, and half of the edges adjacent to each vertex are loops.

There is a constant $d$ and an explicit $(d, 0.1)$-expander $\{G_n\}_{n \in \mathbb{N}}$. We may assume that $\varphi$ is $d$-regular (adding self-loops if $d' < d$) and that $\varphi$ contains $n$ variables.

We get $\psi$ from $\varphi$ by adding a tautological constraint for every edge of $G_n$ and adding $2d$ tautological constraints forming self-loops for each vertex. Then

$$\lambda_{G_\psi} \leq \frac{3}{4} + \frac{1}{4} \lambda_{G_\varphi} < 0.9.$$ 

Notice that “adding” decreases $\epsilon$ by a factor $\leq d$ and “adding” by a further factor $\leq 4$. 

---

Computational Complexity, by Fu Yuxi

PCP Theorem

83 / 119
Step 2: Gap Amplification

To amplify the gap, we apply a path-product like operation on constraint graphs so that a new constraint is a conjunction of some old constraints.

Path construction increases the fraction of violated constraints.

We need to address the issue of how to compose two-variable constraints so that every new constraint contains two variables.
Step 2: Gap Amplification

Construction of the 2\text{CSP}_{\mathcal{W}'} instance $\psi^t$: Variables.

1. Let $x_1, \ldots, x_n$ denote the variables of $\psi$.
2. $\psi^t$ contains $n$ variables $y_1, \ldots, y_n \in \mathcal{W}' < \mathcal{W}^{d5t}$, where $y_i$ is an assignment to those of $x_1, \ldots, x_n$ reachable within $t + \delta(t)$ steps from $x_j$. We will let $\delta(t) = \sqrt{t}$.

For $i, j \in [n]$ we say that an assignment to $y_i$ claims a value for $x_j$. 

---

The belt zone is for consistence checking.
Step 2: Gap Amplification

Construction of the $2\text{CSP}_{W'}$ instance $\psi^t$: Constraints.

1. For each $2t+1$ step path $p = (i_1, \ldots, i_{2t+2})$ in $G_\psi$, introduce $C_p = \bigwedge_{j \in [2t+2]} C^j_p$ as a constraint of $\psi^t$ such that for each $j$ that $C^j_p$ appears in $C_p$, the following hold:
   1.1 The length of $(i_1, \ldots, i_j)$ and the length of $(i_{j+1}, \ldots, i_{2t+2})$ are $\leq t + \sqrt{t}$;
   1.2 $C^j_p$ is obtained from the constraint of $x_{ij}$ and $x_{ij+1}$ by replacing $x_{ij}$, $x_{ij+1}$ respectively by $y_{i_1}$'s claim for $x_{ij}$ and $y_{i_{2t+2}}$'s claim for $x_{ij+1}$.
Step 2: Gap Amplification

**Lemma.** There is an algorithm that given $t > 1$ and a nice $2\text{CSP}_W$ instance $\psi$ with $n$ variables, $m = \frac{dn}{2}$ edges, $d$-degree $G_\psi$, produces a $2\text{CSP}_{W'}$ instance $\psi^t$ satisfying 1-4.

1. $W' < W^{d^{5t}}$ and $\psi^t$ has at most $n \cdot d^t + \sqrt{t} + 1$ constraints.
2. If $\psi$ is satisfiable then $\psi^t$ is satisfiable.
3. For $\epsilon < \frac{1}{d\sqrt{t}}$, if $\text{val}(\psi) \leq 1 - \epsilon$, then $\text{val}(\psi^t) \leq 1 - \ell\epsilon$ for $\ell = \frac{\sqrt{t}}{10^4dW^5}$.
4. The formula $\psi^t$ is produced in $\text{poly}(m, W^{d^{5t}})$ time.

**Gap Amplification** follows immediately from the lemma.

- If $\ell = 6$ and $W = 2^{q_0}$, we get a constant $t$ and a constant $\epsilon_0 = \frac{1}{d\sqrt{t}}$.
- In this case $\psi^t$ has $O(m)$ constraints and is produced in $\text{poly}(m)$ time.
Step 2: Gap Amplification

The conditions 1, 2, 4 of the lemma are satisfied.
Step 2: Gap Amplification

Fix an arbitrary assignment \(v_1, \ldots, v_n\) to \(y_1, \ldots, y_n\).

We would like to define an assignment to \(x_1, \ldots, x_n\) from \(v_1, \ldots, v_n\).

1. Let \(Z_i \in [W]\) be a random variable defined by the following.
   - Starting from the vertex \(i\), take a \(t\) step random walk in \(G_\psi\) to reach some vertex \(k\);
   - output \(v_k\)'s claim for \(x_i\).
   
   Let \(w_i\) denote the most likely value of \(Z_i\).

2. We call \(w_1, \ldots, w_n\) the **plurality assignment** to \(x_1, \ldots, x_n\). Clearly \(\Pr[Z_i = w_i] \geq \frac{1}{W}\).

You are what your friends think you are.
Step 2: Gap Amplification

A random $2t+1$ step path $p = (i_1, \ldots, i_{2t+2})$ is picked up with probability $\frac{2}{n \cdot d^{2t+1}}$. Equivalently such a random path can be chosen in either of the following manners.

1. Pick up a random node, and take a $2t+1$ step random walk from the node.
2. Pick up a random edge, and then take a $j$-step random walk from one node of the edge and a $(2t−j)$-step random walk from the other node of the edge.

Pick up a random constraint of $\psi^t$ is the same as picking up a random $2t+1$ step path.
Step 2: Gap Amplification

For $j \in [2t+1]$ the $j$-th edge $(i_j, i_{j+1})$ in $p$ is truthful if $v_{i_1}$ claims the plurality value for $i_j$ and $v_{i_{2t+2}}$ claims the plurality value for $i_{j+1}$.

Suppose $\text{val}(\psi) \leq 1 - \epsilon$. There is a set $F$ of $\epsilon m = \epsilon \frac{dn}{2}$ constraints in $\psi$ violated by the assignment $x_1 = w_1, \ldots, x_n = w_n$

- If $p$ has an edge that is both truthful and in $F$, the constraint $C_p$ is violated.

Claim. Let $\delta = \frac{1}{100W}$. For each $j \in [t + 1, t + \delta \sqrt{t}]$,

$$\Pr_p \Pr_{e \in E}[^{\text{the } j\text{th edge of } p \text{ is truthful}} \mid ^{\text{the } j\text{th edge of } p \text{ is } e}] \geq \frac{1}{2W^2}.$$  

Consequently an edge in $p$ is violated with probability at least $\frac{\epsilon}{2W^2}$. 

Computational Complexity, by Fu Yuxi PCP Theorem 91 / 119
A $2t+1$ step random walk with $(i_j, i_{j+1}) = e$ can be generated by a $(j-1)$-step random walk from $i_j$ and a $(2t-j+1)$-step random walk from $i_{j+1}$. It boils down to proving

$$\Pr[v_i \text{ claims the plurality value for } i_j] \cdot \Pr[v_{2t+2} \text{ claims the plurality value for } i_{j+1}] \geq \frac{1}{2W^2}. \quad (4)$$

If $j = t + 1$ then the left hand side of (4) is at least $1/W^2$. Otherwise observe that

- a $j$-step random walk can be generated by tossing coin for $j$ times and taking $S_j$-step non-self-loop random walk, where $S_j = \#\text{head's}$, and that
- the statistical distance $\Delta(S_t, S_{t+\delta\sqrt{t}})$ is bounded by $10\delta$.

Intuitively it is very likely that starting from a same vertex a $(t+\delta\sqrt{t})$-step random walk and a $t$-step walk would end up in the same vertex.

Thus the left hand side of (4) is at least $(\frac{1}{W} - 10\delta) \left( \frac{1}{W} - 10\delta \right) \geq \frac{1}{2W^2}$.
Step 2: Gap Amplification

Let \( V \) be the random variable for the number of edges among the middle \( \delta \sqrt{t} \) edges that are truthful and in \( F \).

\[ \Pr[V > 0] \] is the probability of a \( \psi^t \)'s constraint being violated. If \( \Pr[V > 0] \geq \epsilon' \), then at least \( \epsilon' \) fraction of \( \psi^t \)'s constraints are violated.

**Lemma.** For every non-negative random variable \( V \), \( \Pr[V > 0] \geq \frac{E[V]^2}{E[V^2]} \).

**Proof.**

\[ E[V|V>0]^2 \leq E[V^2|V>0] \] by convex property. The lemma follows from the following.

1. \( E[V^2|V>0] = \sum_i i^2 \cdot \Pr[V=i|V>0] = \sum_i i^2 \cdot \frac{\Pr[V=i]}{\Pr[V>0]} = \frac{E[V^2]}{\Pr[V>0]} \).
2. \( E[V|V>0]^2 = (\sum_i i \cdot \Pr[V=i|V>0])^2 = \left( \sum_i i \cdot \frac{\Pr[V=i]}{\Pr[V>0]} \right)^2 = \left( \frac{E[V]}{\Pr[V>0]} \right)^2 \). 

\[ \square \]
Step 2: Gap Amplification

Claim. $E[V] \geq \frac{\delta \sqrt{t} \epsilon}{2W^2}$.

Proof. By the previous claim, the probability of an edge in the middle interval of size $\delta \sqrt{t}$ that is truthful and in $F$ is at least $\frac{\epsilon}{2W^2}$. Then $E[V] \geq \frac{\delta \sqrt{t} \epsilon}{2W^2}$ by linearity.

Claim. $E[V^2] \leq 11\delta \sqrt{t} d \epsilon$.

Proof.

- Let $V'$ be the number of edges in the middle interval that are in $F$. Now $V \leq V'$. It suffices to show that $E[V'^2] \leq 11\delta \sqrt{t} d \epsilon$.
- For $j \in \{t + 1, \ldots, t + \delta \sqrt{t}\}$, let $l_j$ be an indicator random variable that is 1 if the $j$th edge is in $F$ and 0 otherwise. Then $V' = \sum_{j \in \{t+1,\ldots,t+\delta\sqrt{t}\}} l_j$.
- Let $S$ be the set of end points of the edges in $F$. Then $\frac{|S|}{dn} \leq \epsilon$. continue on next slide.
Step 2: Gap Amplification

\[
E[V'^2] = E \left[ \sum_j l_j^2 \right] + E \left[ \sum_{j \neq j'} l_j l_{j'} \right]
\]

\[
= \epsilon \delta \sqrt{t} + 2 \sum_{j < j'} \Pr[j\text{th edge is in } F \land j'\text{th edge is in } F]
\]

\[
\leq \epsilon \delta \sqrt{t} + 2 \sum_{j < j'} \Pr[j\text{th vertex of walk lies in } S \land j'\text{th vertex of walk lies in } S]
\]

\[
\leq \epsilon \delta \sqrt{t} + 2 \sum_{j} \Pr[j\text{th vertex of walk lies in } S] \cdot \sum_{j' > j} \Pr_{(a, b) \in E' - j} [a \in S, b \in S]
\]

\[
\leq (3) \epsilon \delta \sqrt{t} + 2 \sum_{j} d \epsilon \cdot \sum_{j' > j} d \epsilon \left( \frac{1}{2} + \frac{(\lambda_G)^{j' - j}}{2} \right)
\]

\[
\leq \epsilon \delta \sqrt{t} + (\delta \sqrt{t})^2 (d \epsilon)^2 \left( 1 + \sum_{k \geq 1} (\lambda_G)^k \right) \quad \lambda_G < 0.9 \text{ and } \delta < 1 \text{ and } \sqrt{td} \epsilon < 1
\]

\[
\leq 11 \delta \sqrt{td} \epsilon.
\]
Finally we conclude that

\[
\Pr[V > 0] \geq \frac{\mathbb{E}[V]^2}{\mathbb{E}[V^2]}
\]

\[
\geq \left( \frac{\delta \sqrt{t} \epsilon}{2W^2} \right)^2 \cdot \frac{1}{11\delta \sqrt{td}\epsilon}
\]

\[
= \delta \cdot \frac{\sqrt{t}}{44dW^4} \cdot \epsilon
\]

\[
= \frac{1}{100W} \cdot \frac{\sqrt{t}}{44dW^4} \cdot \epsilon
\]

\[
> \frac{\sqrt{t}}{10^4dW^5} \epsilon
\]

\[
= \ell \epsilon.
\]
Step 3: Alphabet Reduction

We look for an algorithm that transforms a $2\text{CSP}_W$ to a $q_0\text{CSP}$ instance.

A simple idea is to turn a variable over $[W]$ to $\log W$ boolean variables.

- A constraint can be turned into a circuit $C$ of size bounded by $2^{2\log W} < W^4$.
- This would produce a CSP instance of arity $2\log W$.

The problem with this idea is that $2\log W$ is greater than $q_0$ (in fact $W \geq 2^{q_0}$).

- If we apply **Gap Amplification** and **Alphabet Reduction** for $\log m$ times, we would get a CSP instance whose arity depends on input size.
Step 3: Alphabet Reduction

A more sophisticated idea is to design a PCP checker for constraint checking!

1. We turn the $2\text{CSP}_W$ problem to evaluation checking problem for $\text{CKT-SAT}$.
2. We further turn it to solution checking problem for QUADEQ.
3. We then apply the construction of the PCP verifier (with $q_0$ queries!) for QUADEQ.
4. Finally we turn the PCP verifier to a $q_0 \text{CSP}$ instance.

PCP of Proximity, Verifier Composition, Proof Composition.

- In some occasions a verifier is allowed to make only a small or constant number of queries. In other words it cannot read any complete assignment to variables.
- A solution is to see an assignment as part of a proof. Consequently a verifier can only get to see a fragment of the proof.

This is the PCP verifier for QUADEQ!
Step 3: Alphabet Reduction

Suppose a constraint has been converted to a QUADEQ instance.

- Let $u_1$ and $u_2$ be assignments to $\log W$ variables.
- Let $c$ be bit string of size $\ell = \text{poly}(W)$ that represents the quadratic equations derived from the circuit $C$. We assume that the first $2 \log W$ bits of $c$ are $u_1u_2$.

Let $\pi_1\pi_2\pi_3$ be a PCP proof for the QUADEQ instance, where

- $\pi_1$ is supposedly $\text{WH}(u_1)$, $\pi_2$ is supposedly $\text{WH}(u_2)$ and $\pi_3$ is supposedly $\text{WH}(c)$. 
Step 3: Alphabet Reduction

The PCP verifier does the following:

1. Check that $\pi_1$, $\pi_2$ and $\pi_3$ are 0.99-close to $\text{WH}(u_1)$, $\text{WH}(u_2)$ and $\text{WH}(c)$ respectively.
2. Check that the first $2 \log W$ bits of $c$ are $u_1u_2$. This is done by concatenation test:
   2.1 Choose randomly $x, y \in \{0, 1\}^{\log W}$.
   2.2 Check that $\pi_3(xy0|c|^{-2\log W}) = \pi_1(x) + \pi_2(y)$.

Confer the reduction from CKT-SAT to QUADEQ.
Step 3: Alphabet Reduction

PCP of Proximity.

There is a verifier $\mathcal{V}$ that, given any circuit $C$ with $2k$ input variables, runs in $\text{poly}(|C|)$ time, uses $\text{poly}(|C|)$ random bits, and enjoys the following property.

1. If $u_1, u_2 \in \{0, 1\}^k$ and $u_1 u_2$ is a satisfying assignment for $C$, then there is some $\pi_3 \in \{0, 1\}^{2\text{poly}(|C|)}$ such that $\mathcal{V}$ accepts $\text{WH}(u_1) \text{WH}(u_2) \pi_3$ with probability 1.

2. For $\pi_1, \pi_2 \in \{0, 1\}^{2^k}$ and $\pi_3 \in \{0, 1\}^{2\text{poly}(|C|)}$, if $\mathcal{V}$ accepts $\pi_1 \pi_2 \pi_3$ with probability $> 1/2$, then $\pi_1$ and $\pi_2$ are 0.99-close to $\text{WH}(u_1)$ and $\text{WH}(u_2)$ respectively for some $u_1, u_2 \in \{0, 1\}^k$, where $u_1 u_2$ is a satisfying assignment to $C$.
Step 3: Alphabet Reduction

The PCP verifier can be turned into a $q_0$CSP instance. This is the proof of the equivalence between PCP Theorem and PCP Theorem.
Step 3: Alphabet Reduction

**Fact.** If the value of the old CSP is $\leq 1 - \epsilon$, then the value of the new CSP is $\leq 1 - \frac{1}{3} \epsilon$.

Suppose an assignment to the new variables satisfied $> 1 - \frac{1}{3} \epsilon$ fraction of the new constraints. By decoding an assignment to the old variables satisfied a $1 - \delta$ fraction of the old constraints. For each violated old constraint $C_s$, at least half of the set $C_s$ of the new constraints is violated. Thus $\frac{1}{2} \delta \leq \frac{1}{3} \epsilon$. So at least $1 - \delta \geq 1 - \frac{2}{3} \epsilon > 1 - \epsilon$ fraction of the old constraints were violated, contradicting to the assumption.

This finishes the proof of **Alphabet Reduction**.
Threshold Result by Håstad’s 3-Bit PCP Theorem
Håstad Theorem. For every $\delta \in (0, 1)$ and every $L \in \text{NP}$, there is a PCP verifier $V$ for $L$ that makes three binary queries and satisfies completeness with parameter $1 - \delta$ and soundness with a parameter at most $\frac{1}{2} + \delta$. Moreover, given a proof $\pi \in \{0, 1\}^m$, $V$ chooses $i_1, i_2, i_3 \in [m]$ and $b \in \{0, 1\}$ according to some distribution and accepts iff $\pi_{i_1} + \pi_{i_2} + \pi_{i_3} = b \pmod{2}$.

Threshold Result

An instance of MAX-E3LIN consists of finitely many equations of the form

$$x_{i_1} + x_{i_2} + x_{i_3} = b$$

(5)

taking values in $GF(2)$. One looks for the size of the largest set of satisfiable equations.

According to Håstad Theorem an NP-complete problem has a PCP verifier that checks if a proof validates equations of the form (5). It is NP-hard to compute a $\frac{1/2+\delta}{1-\delta}$-approximation to MAX-E3LIN. In other words $(\frac{1}{2}+\epsilon)$-approximation to MAX-E3LIN is NP-hard for all $\epsilon \in (0, 1/2)$.

It is straightforward to give a $\frac{1}{2}$-approximation algorithm for MAX-E3LIN.
Threshold Result for MAX-3SAT

**Fact.** For all $\epsilon \in (0, 1/8)$ computing $(\frac{7}{8} + \epsilon)$-approximation to MAX-3SAT is NP-hard.

Convert $x + y + z = 0$ to four clauses $\overline{x} \lor y \lor z$, $x \lor \overline{y} \lor z$, $x \lor y \lor \overline{z}$, $\overline{x} \lor \overline{y} \lor \overline{z}$ and $x + y + z = 1$ to four clauses $x \lor \overline{y} \lor \overline{z}$, $\overline{x} \lor y \lor \overline{z}$, $\overline{x} \lor \overline{y} \lor z$, $x \lor y \lor z$.

If an assignment to $x, y, z$ satisfies the equation, it satisfies all the four clauses. Otherwise it satisfies three of the four clauses.

With the help of the previous reduction, we see that in the yes case at least a $1 - \delta$ fraction of the clauses are satisfied, and in the no case at most a $1 - \left(\frac{1}{2} - \epsilon\right) \times \frac{1}{4}$ fraction of the clauses are satisfied.

There is a $\frac{7}{8}$-approximation algorithm to MAX-3SAT.
Proof of Håstad’s 3-Bit PCP Theorem
Let $W \in \mathbb{N}$.

A function $f : \{\pm 1\}^W \to \{\pm 1\}$ is a coordinate function if $f(x_1, \ldots, x_W) = x_w$ for some $w \in W$. In other words $f = \chi_{\{w\}}$.

The long code for $[W]$ encodes each $w \in [W]$ by all the values of $\chi_{\{w\}}$. 
Local Test for Long Code

Let $\delta \in (0, 1)$.

1. Choose $x, y \in \mathbb{R} \{\pm 1\}^W$.
2. Choose a noise vector $z \in \mathbb{R} \{\pm 1\}^W$ using following distribution: For $i \in [W]$, choose $z_i = +1$ with probability $1 - \rho$ and $z_i = -1$ with probability $\rho$.
3. Accept if $f(xy) = f(xyz)$, reject otherwise.

If $f = \chi_{\{w\}}$, then $f(x)f(y)f(xyz) = x_w y_w (x_w y_w z_w) = z_w$. The test accepts iff $z_w = 1$, which happens with probability $1 - \rho$. 
Lemma. If the test accepts with probability \( \frac{1}{2} + \delta \), then \( \sum_{\alpha} \hat{f}_{\alpha}^3 \left( 1 - 2\rho \right)^{|\alpha|} \geq 2\delta \).

If the test accepts with probability \( \frac{1}{2} + \delta \), then \( E_{x,y,z}[f(x)f(y)f(xyz)] = 2\delta \). By Fourier expansion,

\[
2\delta \leq E_{x,y,z} \left[ \left( \sum_{\alpha} \hat{f}_{\alpha} \chi_{\alpha}(x) \right) \left( \sum_{\beta} \hat{f}_{\beta} \chi_{\beta}(y) \right) \left( \sum_{\gamma} \hat{f}_{\gamma} \chi_{\gamma}(xyz) \right) \right]
\]

\[
= E_{x,y,z} \left[ \sum_{\alpha,\beta,\gamma} \hat{f}_{\alpha} \hat{f}_{\beta} \hat{f}_{\gamma} \chi_{\alpha}(x) \chi_{\beta}(y) \chi_{\gamma}(x) \chi_{\gamma}(y) \chi_{\gamma}(z) \right] = \sum_{\alpha} \hat{f}_{\alpha}^3 E_x[\chi_{\alpha}(z)]
\]

\[
= \sum_{\alpha} \hat{f}_{\alpha}^3 \left( \prod_{w \in \alpha} z_w \right) = \sum_{\alpha} \hat{f}_{\alpha}^3 \prod_{w \in \alpha} E_x[z_w] = \sum_{\alpha} \hat{f}_{\alpha}^3 (1 - 2\rho)^{|\alpha|}.
\]

The smaller \( \alpha \) is, the more significant \( \hat{f}_{\alpha} \) is. See the next corollary.
**Corollary.** If $f$ passes the long code test with probability $\frac{1}{2} + \delta$, then for $k = \frac{1}{2\rho} \log \frac{1}{\epsilon}$, there is $\alpha$ with $|\alpha| \leq k$ such that $\hat{f}_\alpha \geq 2\delta - \epsilon$. 

---

*Computational Complexity, by Fu Yuxi*
Historical Remark
Interactive proof, Zero knowledge, \textbf{IP}.

It all started with the introduction of interactive proof systems.


The authors of the papers shared the first Gödel Prize (1993).

Goldwasser and Sipser. Private Coins versus Public Coins in Interactive Proof Systems. STOC’86.
“1989 was an extraordinary year.”  

László Babai, 1990


On Jan. 17, 1990 another email was sent out by L. Babai, L. Fortnow, and L. Lund.

▶ Non-Deterministic Exponential Time has Two Prover Interactive Protocols. FOCS 1990.
    CC 1991.

The main theorem of the paper, $\text{MIP} = \text{NEXP}$, inspired almost all future development of PCP theory and a lot of future development in derandomization theory. It can be interpreted as

$$\text{NEXP} = \text{PCP}(\text{poly}, \text{poly}).$$
A profitable shift of emphasis was made that, instead of scaling down the time or space complexity of verifier, scales down the randomness and query complexity.

Babai, Fortnow, Levin, and Szegedy showed $\text{NP} \subseteq \text{PCP}(\text{polylog, polylog})$.  

2001 Gödel Prize

1. $\text{NP} \subseteq \text{PCP}(\log \cdot \log \log, \log \cdot \log \log)$.
2. $\text{NP} = \text{PCP}(\log, \log)$.
3. $\text{NP} = \text{PCP}(\log, 1)$.

2019 Gödel Prize