PCP Theorem

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

Ingo Wegener, 2005

S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof Verification and the Hardness of Approximation Problems. J. ACM, 1998. FOCS 1992.

Irit Dinur. The PCP Theorem by Gap Amplification. J. ACM, 2007. STOC 2006.

The 1^{st} proof is algebraic, the 2^{nd} 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. Efficient Conversion of NP Certificate to PCP Proof
6. Proof of PCP Theorem
7. 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 $\rho : \mathbf{N} \to (0,1)$. A ρ-approximation algorithm $\mathbb{A}$ for a maximum, respectively minimum optimization problem satisfies

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

respectively

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

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 ϵ a $(1-\epsilon)$-approximation algorithm can be designed that runs in $O\left((\frac{1}{\epsilon} - 1) \cdot n^2\right)$ time.
- We say that `SubsetSum` has an FPTAS.

KnapSack

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 ϵ one can design a $(1-\epsilon)$-approximation algorithm of $O\left((\frac{1}{\epsilon} - 1) \cdot n^{\frac{1}{\epsilon}}\right)$ time.
- We say that KnapSack has a PTAS.

Max-3SAT

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

- φ is satisfied if and only if $\mathtt{val}(\varphi) = 1$.

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

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

By definition, FPTAS $\subseteq$ PTAS $\subseteq$ APX $\subseteq$ OPT. [we will see that the containments are strict assuming $\mathbf{P} \neq \mathbf{NP}$.]

Max-IS

$\texttt{Min-VC} + \texttt{Max-IS} = m$.

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 $\texttt{Max-IS}$ unless $\texttt{SAT} \in \textbf{SUBEXP}$.
[1992]. $\texttt{Max-IS}$ is not in APX if $\mathbf{P} \neq \mathbf{NP}$.

---

1. U. Feige, S. Goldwasser, L. Lovász, S. Safra, and M. Szegedy. Interactive Proofs and the Hardness of Approximating Cliques. FOCS 1991. JACM, 1996.

2. S. Arora and S. Safra. Probabilistic Checking of Proofs: A New Characterization of NP. FOCS 1992. JACM, 1998.

Two Views of PCP Theorem

Surprisingly, $\mathbf{IP} = \mathbf{PSPACE}$. More surprisingly, $\mathbf{MIP} = \mathbf{NEXP}$.

The latter can be interpreted as saying that nondeterminism can be traded off for

randomness + interaction.

Interactive Proof Viewpoint

Suppose L is an NP problem and x is an input string.

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

- 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.

Probabilistically Checkable Proofs

Suppose L is a language and $q, r : \mathbf{N} \to \mathbf{N}$.

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

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

$\mathbb{V}^{\pi}(x)$ denotes the random variable with x and π fixed.

Probabilistically Checkable Proofs

1. Proof length $\leq q(n)2^{r(n)}$.
   - At most $q(n)2^{r(n)}$ locations can be queried by verifier.
2. $L \in \mathbf{NTIME}(q(n)2^{O(r(n))})$.
   - 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 $q(n)2^{r(n)}$.

PCP Hierarchy

A language is in $\mathbf{PCP}(r(n), q(n))$ if it has a $(cr(n), dq(n))$-PCP verifier for some c, d.

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

- $\mathbf{PCP}(0, \log) = \mathbf{P}$.
- $\mathbf{PCP}(0, \mathrm{poly}) = \mathbf{NP}$.
- $\mathbf{PCP}(\log, \mathrm{poly}) = \mathbf{NP}$.

PCP Hierarchy

1. $\mathbf{PCP}(\mathrm{poly}, \mathrm{poly}) \subseteq \mathbf{NEXP}$.
2. $\mathbf{PCP}(\log, \log) \subseteq \mathbf{NP}$.
3. $\mathbf{PCP}(\log, 1) \subseteq \mathbf{NP}$.

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. $\mathbf{NP} = \mathbf{PCP}(\log, 1)$.

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

Example

$\texttt{GNI} \in \mathbf{PCP}(\mathrm{poly}, 1)$.

- Suppose both G_0 and G_1 have n vertices.
- Proofs 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 $\mathbf{NP} \subseteq \mathbf{PCP}(o(\log), o(\log))$, then $\mathbf{P} = \mathbf{NP}$.

Scale-Up PCP Theorem

Theorem. $\mathbf{PCP}(\mathrm{poly}, 1) = \mathbf{NEXP}$.

Hardness of Approximation Viewpoint

For many NP-hard optimization problems, computing approximate solutions is no easier than computing the exact solutions.

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

$$
\begin{aligned}
x \in L &\quad\Rightarrow\quad \texttt{val}(f(x)) = 1, \\
x \notin L &\quad\Rightarrow\quad \texttt{val}(f(x)) < \rho.
\end{aligned}
$$

- Figure out the significance of the theorem by letting $L = \texttt{3SAT}$.

The PCP Theorem, Hardness of Approximation

PCP Theorem cannot be proved using Cook-Levin reduction.

- $\texttt{val}(f(x))$ 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 ρ-approximation algorithm for `Max-3SAT` then $\mathbf{P} = \mathbf{NP}$.

- The ρ-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 qCSP instance φ with n variables is a collection of constraints $\varphi_1, \ldots, \varphi_m : \{0,1\}^n \to \{0,1\}$ such that for each $i \in [m]$ the function φ_i depends on q of its input locations.

We call q the arity of φ, and m the size of φ.

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

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

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

---

qCSP is a generalization of 3SAT.

1. We assume that $n \leq qm$.
2. Since every φ_i can be described by a formula of size $q2^q$, and every variable can be coded up by $\log n$ bits, a $q\mathtt{CSP}$ instance can be described by $O(mq2^q \log n)$ bits.
3. The greedy algorithm for $\mathtt{MAX\text{-}3SAT}$ can be applied to $\mathtt{MAX}q\mathtt{CSP}$ to produce an assignment satisfying $\geq \frac{\mathtt{val}(\varphi)}{2^q} m$ constraints.

Gap CSP

Suppose $q \in \mathbf{N}$ and $\rho \leq 1$.

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

We say that ρ-$\mathtt{GAP}q\mathtt{CSP}$ is NP-hard if for every NP-problem L some P-time computable function $f : L \to \rho$-$\mathtt{GAP}q\mathtt{CSP}$ exists such that

$$
\begin{aligned}
x \in L &\Rightarrow \mathrm{val}(f(x)) = 1, \\
x \notin L &\Rightarrow \mathrm{val}(f(x)) < \rho.
\end{aligned}
$$

PCP Theorem. There exists some $\rho \in (0,1)$ such that ρ-$\mathtt{GAP3SAT}$ is NP-hard.

PCP Theorem. There exist $q \in \mathbf{N}$ and $\rho \in (0, 1)$ such that ρ-$\mathtt{GAP}q\mathtt{CSP}$ is NP-hard.

Equivalence Proof

PCP Theorem $\Rightarrow$ **PCP Theorem**.

This is essentially the Cook-Levin reduction.

1. Suppose **NP** $\subseteq$ **PCP**$(\log, 1)$. Then $\mathtt{3SAT}$ has a PCP verifier $\mathbb{V}$ that makes q queries using $c \log n$ random bits.
2. Given input x with $|x| = n$ and random string $r \in \{0,1\}^{c \log n}$, $\mathbb{V}(x, r)$ is a Boolean function of type $\{0,1\}^q \to \{0,1\}$.
3. $\varphi = \{\mathbb{V}(x, r)\}_{r \in \{0,1\}^{c \log n}}$ is a P-size $q\mathtt{CSP}$ instance.
    - By completeness, $x \in \mathtt{3SAT} \Rightarrow \mathtt{val}(\varphi) = 1$.
    - By soundness, $x \notin \mathtt{3SAT} \Rightarrow \mathtt{val}(\varphi) \leq 1/2$.
4. The map from $\mathtt{3SAT}$ to $\frac{1}{2}$-$\mathtt{GAP}q\mathtt{CSP}$ is P-time computable.
    - $\mathbb{V}$ runs in P-time.

Equivalence Proof

PCP Theorem $\Leftarrow$ **PCP Theorem**.

Suppose $L \in \mathbf{NP}$ and ρ-$\mathtt{GAP}q\mathtt{CSP}$ is NP-hard for some $q \in \mathbf{N}$, $\rho < 1$. By assumption there is some P-time reduction $f : L \to \rho$-$\mathtt{GAP}q\mathtt{CSP}$.

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

Equivalence Proof

PCP Theorem $\Leftrightarrow$ **PCP Theorem**.

This is very much like the equivalence between $\mathtt{SAT}$ and $\mathtt{3SAT}$.

1. Let $\epsilon > 0$ and $q \in \mathbf{N}$ be such that $(1-\epsilon)$-$\mathtt{GAP}q\mathtt{CSP}$ is NP-hard.
2. Let $\varphi = \{\varphi_i\}_{i=1}^m$ be a $q\mathtt{CSP}$ instance with n variables.
3. Each φ_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 ϵ fragment of the constraints of φ, then all assignments fail at least a $\frac{\epsilon}{q2^q}$ fragment of the clauses of the $\mathtt{3SAT}$ instance.

Proof View	**Inapproximability View**
PCP verifier V	$\mathtt{CSP}$ instance φ
PCP proof π	assignment to variables $\mathbf{u}$
proof length $\|\pi\|$	number of variables n
number of queries q	arity of constraints q
number of random bits r	logarithm of number of constraints $\log m$
soundness parameter ϵ	maximum fraction of violated constraints of no instance ρ
$\mathbf{NP} \subseteq \mathbf{PCP}(\log, 1)$	ρ-$\mathtt{GAP}q\mathtt{CSP}$ is NP-hard

The equivalence of the proof view and the inapproximability view is essentially due to the Cook-Levin Theorem for PTM.

Inapproximability

$\texttt{Min-VC}$ and $\texttt{Max-IS}$ are inherently different from the perspective of approximation.

- $\texttt{Min-VC} + \texttt{Max-IS} = n$.
- One can construct an $\frac{n-IS}{n-\rho IS}$-approximation algorithm for $\texttt{Min-VC}$ from a ρ-approximation algorithm for $\texttt{Max-IS}$.

Lemma. There is a P-time computable function f from 3CNF formulas to graphs that maps a formula φ to an n-vertex graph $f(\varphi)$ whose independent set is of size $\mathtt{val}(\varphi)\frac{n}{7}$.

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 φ consisting of m clauses is translated to a graph with $7m$ nodes, and an assignment satisfying l clauses of φ if and only if the graph has an independent set of size l.

Theorem. The following statements are valid.

1. $\exists\rho' < 1$. ρ'-approximation to $\texttt{Min-VC}$ is NP-hard, and
2. $\forall\rho < 1$. ρ-approximation to $\texttt{Max-IS}$ is NP-hard.

There is some $\rho < 1$ such that ρ-approximation to $\texttt{Max-3SAT}$ is **NP**-hard.
By Lemma the ρ-approximation to $\texttt{Max-IS}$ produces a ρ-approximation to $\texttt{Max-3SAT}$.

Referring to the map of Lemma, the minimum vertex cover has size $n - \texttt{val}(\varphi)\frac{n}{7}$.
Let $\rho' = \frac{6}{7-\rho}$. Suppose $\texttt{Min-VC}$ had a ρ'-approximation algorithm.

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

Continue on the next slide.

Suppose $\texttt{Max-IS}$ was ρ_0-approximate. Let k satisfy $\rho_0\binom{IS}{k} \geq \binom{\rho IS}{k}$. Define algorithm:

1. Input a graph G. 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 ρ_0-approximation algorithm to G^k, and derive an independent set of G from the output of the algorithm.

- The largest independent set of G^k is of size $\binom{IS}{k}$, where IS is the maximum independent set of G.
- The output is an independent set of size $\geq \rho_0\binom{IS}{k} \geq \binom{\rho IS}{k}$.
- An independent set of size ρIS can be derived. Contradiction.

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. $\mathbf{NP} \subseteq \mathbf{PCP}(\mathrm{poly}(n), 1)$.

Walsh-Hadamard Code

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

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

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

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

Random Subsum Principle.

- If $\mathbf{u} \neq \mathbf{v}$ then for exactly half the choices of $\mathbf{x}$, $\mathbf{u} \odot \mathbf{x} \neq \mathbf{v} \odot \mathbf{x}$.

Walsh-Hadamard Codewords

A function f of type $\{0,1\}^n \rightarrow \{0,1\}$ is a linear function if

$$f(\mathbf{x}+\mathbf{y}) = f(\mathbf{x}) + f(\mathbf{y}).$$

Walsh-Hadamard codewords are precisely the linear functions. This is because a linear function f is the same as $\mathtt{WH}(\mathbf{f})$, where $\mathbf{f} = (f(\mathbf{e}_1), \ldots, f(\mathbf{e}_n))^\dagger$.

Let $\rho \in [0,1]$. The functions $f, g : \{0,1\}^n \to \{0,1\}$ are ρ-close if

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

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

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

Then f is ρ-close to a linear function.

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 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

$$f(\mathbf{x}+\mathbf{y}) = f(\mathbf{x}) + f(\mathbf{y})$$

randomly for $O(\frac{1}{\delta})$ times.

Local Decoding

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

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

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

By union bound $\widetilde{f}(\mathbf{x}) = \widetilde{f}(\mathbf{x}') + \widetilde{f}(\mathbf{x}'') = f(\mathbf{x}') + f(\mathbf{x}'')$ holds with probability $\geq 1-2\delta$.

Quadratic Equation in $\mathrm{GF}(2)$

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

$$\mathbf{A}(\mathbf{u} \otimes \mathbf{u}) = \mathbf{b},$$

where $\mathbf{u} \otimes \mathbf{u}$ is the tensor product of $\mathbf{u}$.

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

Quadratic Equation in $\mathrm{GF}(2)$

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

$$\begin{aligned} u_1 u_2 + u_3 u_4 + u_1 u_5 &= 1 \\ u_1 u_1 + u_2 u_3 + u_1 u_4 &= 0 \end{aligned}$$

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

QUADEQ is NP-Complete

$\texttt{CKT-SAT} \leq_K \texttt{QUADEQ}$.

- The inputs and the outputs are turned into variables.
- Boolean equality $x \vee y = z$ relating the inputs to the output is turned into algebraic equality $(1-x)(1-y) = 1-z$ in $\mathrm{GF}(2)$, which is equivalent to $xx + yy + xy + zz = 0$.
- $\neg x = z$ is turned into $xx + zz = 1$.
- $x \wedge y = z$ is turned into $xy + zz = 0$.

From NP Certificate to PCP Proof

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

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

- The proof is a string of length $2^n + 2^{n^2}$.
- Using the proof it is straightforward to verify probabilistically if $(\mathbf{A}, \mathbf{b}) \in \mathtt{QUADEQ}$.

Verifier for 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 $f(\mathbf{r}) = \mathbf{u} \odot \mathbf{r}$ and $g(\mathbf{z}) = \mathbf{w} \odot \mathbf{z}$.

Verifier for QUADEQ

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

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

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

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

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 $\mathbf{A}(\mathbf{u}\otimes\mathbf{u}) = \mathbf{b}$ 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

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

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

$q\mathtt{CSP}_W$ is analogous to $q\mathtt{CSP}$ except that the alphabet is $[W]$ instead of $\{0,1\}$.

The constraints are functions of type $[W]^q \to \{0,1\}$.

For $\rho \in (0,1)$, we define the promise problem $\rho\text{-}\mathtt{GAP}q\mathtt{CSP}_W$ analogous to $\rho\text{-}\mathtt{GAP}q\mathtt{CSP}$.

$\mathtt{3COL}$ is a case of $\mathtt{2CSP}_3$.

PCP Theorem states that ρ-$\mathtt{GAP}q\mathtt{CSP}$ is NP-hard for some q, ρ.

The proof we shall describe is based on the following:

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

The idea is to start with an NP-problem, then apply Step 2 repeatedly.

Let f be a function mapping CSP instances to CSP instances.

It is a CL-reduction (complete linear-blowup reduction) if it is P-time computable and the following are valid for every CSP instance φ.

- Completeness. If φ is satisfiable then $f(\varphi)$ is satisfiable.
- Linear Blowup. If φ has m constraints, $f(\varphi)$ has no more than Cm constraints over a new alphabet W.
  - C and W depend on the arity and the alphabet size of φ, but neither on the number of constraints nor on the number of variables of φ.

Main Lemma. There exist constants $q_0 \geq 3$, $\epsilon_0 > 0$ and CL-reduction f such that for every q_0CSP instance φ and every $\epsilon < \epsilon_0$, $f(\varphi)$ is a q_0CSP instance satisfying

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

---

q_0CSP Instance	Arity	Alphabet	Constraint	Gap
φ	q_0	binary	m	$1-\epsilon$
$\Downarrow$	$\Downarrow$	$\Downarrow$	$\Downarrow$	$\Downarrow$
$f(\varphi)$	q_0	binary	Cm	$1-2\epsilon$

Proof of PCP Theorem

Let $q_0 \geq 3$ and $\epsilon_0 > 0$ be given by the Main Lemma. A CL-reduction from q_0CSP to $(1-2\epsilon_0)$-GAPq_0CSP is obtained as follows:

1. q_0CSP is NP-hard.
2. For a q_0CSP instance φ with m constraints, apply Main Lemma for $\log m$ times to amplify the gap. We get an instance ψ.
3. If φ is satisfiable, then ψ is satisfiable. Otherwise according to Main Lemma

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

4. $|\psi| \leq C^{\log m} m = \mathtt{poly}(|\varphi|)$. Conclude that $(1-2\epsilon_0)$-$\mathrm{GAP}q_0$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 $q\mathtt{CSP}$ instance can be turned into a "nice" $q\mathtt{CSP}_W$ instance.
2. Gap Amplification. Construct a CL-reduction f that increases both the gap and the alphabet size of a "nice" $q\mathtt{CSP}$ 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 ℓ, q, there are number W, $\epsilon_0 \in (0,1)$ and a CL-reduction $g_{\ell,q}$ such that for every $q\mathrm{CSP}$ instance φ, $\psi = g_{\ell,q}(\varphi)$ is a $2\mathrm{CSP}_W$ instance that satisfies the following for all $\epsilon < \epsilon_0$.

$$\mathrm{val}(\varphi) \leq 1 - \epsilon \Rightarrow \mathrm{val}(\psi) \leq 1 - \ell\epsilon.$$

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

$$\mathrm{val}(\varphi) \leq 1 - \epsilon \Rightarrow \mathrm{val}(\psi) \leq 1 - \epsilon/3.$$

CSP Instance	Arity	Alphabet	Constraint	Gap
φ	q_0	binary	m	$1-\epsilon$
$\Downarrow$	$\Downarrow$	$\Downarrow$	$\Downarrow$	$\Downarrow$
$f(\varphi)$	2	nonbinary	$C'm$	$1-6\epsilon$
$\Downarrow$	$\Downarrow$	$\Downarrow$	$\Downarrow$	$\Downarrow$
$g(f(\varphi))$	q_0	binary	$C''C'm$	$1-2\epsilon$

Dinur makes use of expander graphs to construct new constraints.

Let φ be a $2\mathtt{CSP}_W$ instance with n variables. The constraint graph G_φ of φ 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 $2\mathtt{CSP}_W$ instance φ is nice if the following are valid:

1. There is a constant d such that G_φ 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.

Step 1: Reduction to Nice Instance

The reduction consists of three steps.

$$
\begin{aligned}
q_0\texttt{CSP} \text{ instance} &\stackrel{\text{Step1.1}}{\Rightarrow} 2\texttt{CSP}_{2^{q_0}} \text{ instance} \\
&\stackrel{\text{Step1.2}}{\Rightarrow} 2\texttt{CSP}_{2^{q_0}} \text{ instance with regular constraint graph} \\
&\stackrel{\text{Step1.3}}{\Rightarrow} \text{nice } 2\texttt{CSP}_{2^{q_0}} \text{ instance.}
\end{aligned}
$$

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_0CSP instance φ to a 2CSP$_{2^{q_0}}$ instance ψ such that

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

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

- The new instance ψ 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 φ_i, construct the constraint $\psi_{i,j}$ stating that y_i satisfies φ_i and y_i is consistent to x_j.

Step 1: Reduction to Nice Instance

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

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

and that G_ψ is d-regular.

Let $\{G_k\}_k$ be an explicit family of $(d-1)$-regular expander. We get ψ by replacing each k-degree node of G_φ by G_k and adding the identity constraint (of the form $y_i^j = y_i^{j'}$, where $i \in [n]$) to each edge (j, j') of G_k. If φ has m constraints, ψ has dm constraints.

Suppose $\mathtt{val}(\varphi) \leq 1-\epsilon$ and **v** is an assignment to the variables of ψ.

It suffices to prove that **v** violates at least $\frac{\epsilon m}{100W}$ constraints of ψ. [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 $\mathtt{poly}(n)$ time and outputs an (n, d, c)-expander.

Step 1: Reduction to Nice Instance

Let $\mathbf{u}$ be the assignment to φ that is defined by the plurality of $\mathbf{v}$.

Let t_i be the number of v_i^j'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 $S_i = \{y_i^j \mid v_i^j = u_i\}$ and let $\overline{S_i} = \{y_i^1, \ldots, y_i^k\} \setminus S_i$. The number of constraints of G_k violated by $\mathbf{v}$ is

$$E(S_i, \overline{S_i}) \stackrel{(1)}{\geq} (1 - \lambda_{G_k}) \frac{(d-1)|S_i||\overline{S_i}|}{|S_i| + |\overline{S_i}|} = \frac{1}{10} \frac{d-1}{k} |S_i||\overline{S_i}| \geq \frac{1}{10W} t_i,$$

   where the last inequality is due to $|S_i| \geq k/W$. Now $\sum_{i \in [n]} E(S_i, \overline{S_i}) \geq \frac{\epsilon m}{40W}$.

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

   Since every constraint has two variables, less than $\frac{1}{4}\epsilon m + \frac{1}{4}\epsilon m$ constraints have valuations in ψ different from those in φ. 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 $2\mathtt{CSP}_W$ instance φ with G_φ being d'-regular for some $d' \leq d$ to a $2\mathtt{CSP}_W$ instance ψ such that

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

and that G_ψ 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 $\{G_n\}_{n\in\mathbf{N}}$. We may assume that φ is d-regular (adding self-loops if $d' < d$) and that φ contains n variables.

We get ψ from φ 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 ϵ by a factor $\leq d$ and "adding" by a further factor ≤ 4.

Lemma. Let G be a d-regular n-vertex graph, S be a vertex subset and $T = \overline{S}$. Then

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

The vector $\mathbf{x}$ defined below satisfies $\|\mathbf{x}\|_2^2 = |S||T|(|S|+|T|)$ and $\mathbf{x}\perp\mathbf{1}$.

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

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

$$Z = \sum_{i,j} A_{i,j}\mathbf{x}_i^2 - 2\sum_{i,j} A_{i,j}\mathbf{x}_i\mathbf{x}_j + \sum_{i,j} A_{i,j}\mathbf{x}_j^2 = 2\|\mathbf{x}\|_2^2 - 2\langle \mathbf{x}, A\mathbf{x}\rangle.$$

Since $\mathbf{x}\perp\mathbf{1}$, $\langle \mathbf{x}, A\mathbf{x}\rangle \leq \lambda_G\|\mathbf{x}\|_2^2$ (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

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

We are done by substituting $|S|/|V|-\Pr_{(u,v)\in E}[u\in S, v\in\overline{S}]$ 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 2: Gap Amplification

To amplify the gap, we apply a path-product like operation on constraint graphs.

Step 2: Gap Amplification

Lemma. There is an algorithm that given $t > 1$ and a nice $2\mathtt{CSP}_W$ instance ψ with n variables, m edges and d-degree G_ψ, produces a $2\mathtt{CSP}_{W'}$ instance ψ^t satisfying 1-4.

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

Gap Amplification follows immediately from the lemma.

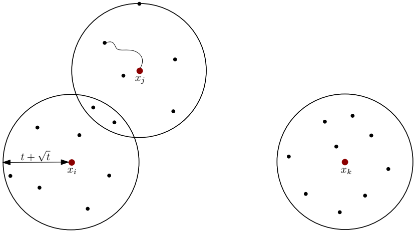
- If $\ell = 6$ and $W = 2^{q_0}$, we get a constant t and a constant $\epsilon_0 \approx \frac{1}{d\sqrt{t}}$.
- In this case ψ^t has $O(n)$ constraints and is produced in $\mathrm{poly}(m)$ time.

Step 2: Gap Amplification

Construction of the $2\mathtt{CSP}_{W'}$ instance ψ^t: Variables.

1. Let $x_1, \ldots, x_n$ denote the variables of ψ.
2. Introduce n new variables $\mathbf{y}_1, \ldots, \mathbf{y}_n \in W' = W^{d^{5t}}$, where $\mathbf{y}_i$ is understood as an assignment to those of $x_1, \ldots, x_n$ that appear in paths within $t + \sqrt{t}$ steps from x_i.

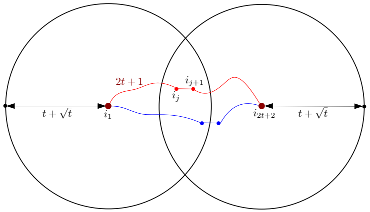
For $i, j \in [n]$ we say that an assignment to $\mathbf{y}_i$ claims a value for x_j.



Step 2: Gap Amplification

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

1. For each $2t+1$ step path $p = (i_1, \ldots, i_{2t+2})$ in G_ψ, introduce $C_p = \bigwedge_{j\in[2t+2]} C_p^j$ as a constraint of ψ^t such that for each j that C_p^j 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_p^j is obtained from the constraint of x_{i_j} and $x_{i_{j+1}}$ by replacing $x_{i_j}, x_{i_{j+1}}$ by $\mathbf{y}_{i_1}$'s claim for x_{i_j} and $\mathbf{y}_{i_{2t+2}}$'s claim for $x_{i_{j+1}}$ respectively.



Step 2: Gap Amplification

The conditions 1, 2, 4 of the lemma are satisfied.

Step 2: Gap Amplification

Fix an arbitrary assignment $\mathbf{v}_1, \ldots, \mathbf{v}_n$ to $\mathbf{y}_1, \ldots, \mathbf{y}_n$.

We would like to define an assignment to $x_1, \ldots, x_n$ from $\mathbf{v}_1, \ldots, \mathbf{v}_n$.

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_ψ to reach some vertex k; output $\mathbf{v}_k$'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 $\Pr[Z_i = w_i] \geq \frac{1}{W}$.

Step 2: Gap Amplification

Denote by p a random $2t+1$ step path $(i_1, \ldots, i_{2t+2})$.

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

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

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

Step 2: Gap Amplification

Claim. Let $E = G_\psi$'s edge set. $\Pr_p \Pr_{e \in_R E}[\text{the } j\text{th edge of } p \text{ is } e] = \frac{1}{|E|} = \frac{2}{nd}$.

Proof.

In a random walk from a random vertex of a regular graph, every edge in the walk is a random edge. □

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

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

Proof.

See the next slide. □

Step 2: Gap Amplification

By the previous claim, 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

$$\Pr[\mathbf{v}_{i_1}\text{claims the plurality value for } i_j]\cdot\Pr[\mathbf{v}_{i_{2t+2}}\text{claims the plurality value for } i_{j+1}] \geq \frac{1}{2W^2}. \quad (4)$$

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 $S_j = \sharp$head's, and that
- the statistical distance $\Delta(S_t, S_{t+\delta\sqrt{t}})$ is bounded by 10δ.

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 $\left(\frac{1}{W} - 10\delta\right)\left(\frac{1}{W} - 10\delta\right) \geq \frac{1}{2W^2}$.

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.

- If $\Pr[V > 0] \geq \epsilon'$, then at least ϵ' fraction of ψ^t's constraints are violated.

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

Proof.

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

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

$\square$

Step 2: Gap Amplification

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

Proof.
By the two claims, the probability of an edge in the middle interval of size $\delta\sqrt{t}$ 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 $j \in \{t, \ldots, t + \delta\sqrt{t}\}$, let I_j be an indicator random variable that is 1 if the jth edge is in F and 0 otherwise. Then $V' = \sum_{j\in\{t,\ldots,t+\delta\sqrt{t}\}} I_j$.
- Let S be the set of end points of the edges in F. Then $\frac{|S|}{n} \leq d\epsilon$.

[continue on next slide.]

Step 2: Gap Amplification

$$\begin{aligned}
\mathrm{E}[V'^2] &= \mathrm{E}[\sum_j I_j^2] + \mathrm{E}[\sum_{j\neq j'} I_j I_{j'}] \\
&= \epsilon\delta\sqrt{t} + 2\sum_{j<j'} \Pr[j\text{th edge is in } F \wedge j'\text{th edge is in } F] \\
&\le \epsilon\delta\sqrt{t} + 2\sum_{j<j'} \Pr[j\text{th vertex of walk lies in } S \wedge j'\text{th vertex of walk lies in } S] \\
&\le \epsilon\delta\sqrt{t} + 2\sum_{j} \Pr[j\text{th vertex of walk lies in } S] \cdot \sum_{j'>j} \Pr_{(a,b)\in E^{j'-j}}[a \in S, b \in S] \\
&\overset{(3)}{\le} \epsilon\delta\sqrt{t} + 2\sum_{j} d\epsilon \cdot \sum_{j'>j} d\epsilon \left(\frac{1}{2} + \frac{(\lambda_G)^{j'-j}}{2}\right) \\
&\le \epsilon\delta\sqrt{t} + \delta\sqrt{t}(d\epsilon)^2 \cdot \delta\sqrt{t} + \delta\sqrt{t}(d\epsilon)^2 \cdot \sum_{k\ge 1}(\lambda_G)^k \\
&\le \epsilon\delta\sqrt{t} + \delta^2\sqrt{t}d\epsilon + 9\delta d\epsilon \qquad \delta < 1 \text{ and } \sqrt{t}d\epsilon < 1 \text{ by assumption} \\
&\le 11\delta\sqrt{t}d\epsilon.
\end{aligned}$$

Step 2: Gap Amplification

Finally we conclude that

$$
\begin{aligned}
\Pr[V>0] &\geq \frac{\mathrm{E}[V]^2}{\mathrm{E}[V^2]} \\
&\geq \left(\frac{\delta\sqrt{t}\epsilon}{2W^2}\right)^2 \cdot \frac{1}{11\delta\sqrt{t}d\epsilon} \\
&= \delta \cdot \frac{\sqrt{t}}{44dW^4} \cdot \epsilon \\
&= \frac{1}{100W} \cdot \frac{\sqrt{t}}{44dW^4} \cdot \epsilon \\
&> \frac{\sqrt{t}}{10^4 dW^5}\epsilon \\
&= \ell\epsilon.
\end{aligned}
$$

Step 3: Alphabet Reduction

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

- 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 $2\mathrm{CSP}_W$ problem to evaluation checking problem for `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_0CSP 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 assignment to variables.
- A solution is to see an assignment as part of a proof. Consequently a verifier can only get to see part of a proof.

Step 3: Alphabet Reduction

Suppose a constraint has been converted to a QUADEQ instance.

- Let $\mathbf{u}_1$ and $\mathbf{u}_2$ be assignments to $\log W$ variables.
- Let $\mathbf{c}$ be bit string of size $\ell = \mathtt{poly}(W)$ that represents the quadratic equations derived from the circuit C. We assume that the first $2\log W$ bits of $\mathbf{c}$ are $\mathbf{u}_1\mathbf{u}_2$.

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

- π_1 is supposedly $\mathtt{WH}(\mathbf{u}_1)$, π_2 is supposedly $\mathtt{WH}(\mathbf{u}_2)$ and π_3 is supposedly $\mathtt{WH}(\mathbf{c})$.

Step 3: Alphabet Reduction

The PCP verifier does the following:

1. Check that π_1, π_2 and π_3 are 0.99-close to $\mathtt{WH}(\mathbf{u}_1)$, $\mathtt{WH}(\mathbf{u}_2)$ and $\mathtt{WH}(\mathbf{c})$ respectively.
2. Check that the first $2\log W$ bits of $\mathbf{c}$ are $\mathbf{u}_1\mathbf{u}_2$. This is done by concatenation test:
   2.1 Choose randomly $\mathbf{x}, \mathbf{y} \in \{0,1\}^{\log W}$.
   2.2 Check that $\pi_3(\mathbf{xy}0^{|\mathbf{c}|-2\log W}) = \pi_1(\mathbf{x}) + \pi_2(\mathbf{y})$. [confer the reduction from CKT-SAT to QUADEQ.]

Step 3: Alphabet Reduction

The PCP verifier can be turned into a $q_0\mathrm{CSP}$ instance. This is the proof of the equivalence between PCP Theorem and PCP Theorem.

Step 3: Alphabet Reduction

PCP of Proximity. [there is a PCP verifier for every old constraint.]

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

1. If $\mathbf{u}_1, \mathbf{u}_2 \in \{0,1\}^k$ and $\mathbf{u}_1\mathbf{u}_2$ is a satisfying assignment for C, then there is some $\pi_3 \in \{0,1\}^{2^{\mathrm{poly}(\ell)}}$ such that V accepts $\mathtt{WH}(\mathbf{u}_1)\mathtt{WH}(\mathbf{u}_2)\pi_3$ with probability 1.
2. For $\pi_1, \pi_2 \in \{0,1\}^{2^k}$ and $\pi_3 \in \{0,1\}^{2^{\mathrm{poly}(\ell)}}$, if V accepts $\pi_1\pi_2\pi_3$ with probability $\geq 1/2$, then π_1 and π_2 are 0.99-close to $\mathtt{WH}(\mathbf{u}_1)$ and $\mathtt{WH}(\mathbf{u}_2)$ respectively for some $\mathbf{u}_1, \mathbf{u}_2 \in \{0,1\}^k$ such that $\mathbf{u}_1\mathbf{u}_2$ is a satisfying assignment to C.

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 $\mathcal{C}_s$ of the new constraints is violated. Thus $\frac{1}{2}\delta \leq \frac{1}{3}\epsilon$ by assumption. So $1-\frac{2}{3}\epsilon > 1-\epsilon$ fraction of the old constraints was violated.

Historical Remark

Interactive proof, Zero knowledge, **IP**.

It all started with the introduction of interactive proof systems.

1. Shafi Goldwasser, Silvio Micali, Charles Rackoff. The Knowledge Complexity of Interactive Proofs. STOC'85, SIAM, 1989.
2. László Babai and Shlomo Moran. Arthur-Merlin Games: A Randomized Proof System, and a Hierarchy of Complexity Classes. STOC'85, JCSS, 1988.

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

1. N. Nisan. Co-SAT has multi-prover interactive proofs, e-mail announcement. Nov. 27, 1989.
2. C. Lund, L. Fortnow, H. Karloff, N. Nisan. The polynomial time hierarchy has interactive proofs, e-mail announcement, Dec. 13, 1989.
3. A. Shamir. IP=PSPACE, e-mail announcement, Dec. 26, 1989.

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, $\mathbf{MIP} = \mathbf{NEXP}$, inspired almost all future development of PCP theory and a lot of future development in derandomization theory. It can be interpreted as

$$\mathbf{NEXP} = \mathbf{PCP}(\mathrm{poly}, \mathrm{poly}).$$

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 **NP** $\subseteq$ **PCP**$(\mathrm{polylog}, \mathrm{polylog})$.

---

1. L. Babai, L. Fortnow, L. Levin, and M. Szegedy. Checking Computation in Polylogarithmic Time. STOC, 1991.

2001 Gödel Prize

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

1. U. Feige, S. Goldwasser, L. Lovász, S. Safra, and M. Szegedy. Interactive Proofs and the Hardness of Approximating Cliques. FOCS'91, JACM, 1996.
2. S. Arora and S. Safra. Probabilistic Checking of Proofs: A New Characterization of NP. FOCS'92, JACM, 1998.
3. S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof Verification and the Hardness of Approximation Problems. FOCS'92, JACM, 1998.

2019 Gödel Prize

Irit Dinur. The PCP Theorem by Gap Amplification. J. ACM, 2007. STOC 2006.