MCS 521 Project: Dinur's Proof of the PCP Theorem

Gregoire Fournier

March 3, 2025

The PCP Theorem provides a characterisation of NP as the set of languages that have a "locally testable" membership proof. This robust way of looking at proofs has an important consequence: it implies that many optimization problems are NP-hard both to solve exactly and to approximate; which makes the P versus NP question central to inapproximability theory.
The PCP's motivation comes from the idea of interactive proof and was first proven using algebra techniques (low-degree extension over finite fields, low-degree test, parallelization through curves, a sum-check protocol, and the Hadamard and quadratic functions encodings). The key part of Dinur's simpler proof, is the gap amplification lemma 7 that allows to iteratively improve the soundness parameter of the PCP from close to 1 to being strictly bounded away from 1. This strategy has been compared to the zig-zag construction of expander graphs and Reingold's deterministic logspace algorithm for undirect connectivity.

1 Introduction to the PCP

The goal of this report is to introduce the PCP theorem and the combinatorial proof of Dinur[1]. We will explain how PCP yields hardness of approximation results with an example. We will use some observations from Arora and Barak's book on complexity [2].

Recall the definitions of some complexity classes:

Definition 1 (Class NP). The language L is in NP iff there is a polynomial time deterministic verifier V(a TM) and a prover P, with the following properties:

- "Completeness": For every $x \in L$, P can write a proof/certificate of length poly($|x|$) that V accepts.
- "Soundness": For every $x \notin L$, no matter what poly(|x|)-length proof P writes, V rejects.

Definition 2 (Class PCP$[\mathbf{r}, \mathbf{q}]$). The class PCP $[\mathbf{r}, \mathbf{q}]$ is defined to contain all languages L for which there is a (poly-time) verifier V that uses $O(\mathbf{r})$ random bits, reads $O(\mathbf{q})$ bits from the proof, and guarantees:

- "Completeness": if $x \in L$ then there is a proof π such that $\Pr[V^\pi(x) \text{ accepts}] = 1$, where $V^\pi(x)$ denotes the output of V on input x and proof π.
- "Soundness": if $x \notin L$ then for any proof π, $\Pr[V^\pi(x) \text{ accepts}] \leq \frac{1}{2}$.

The PCP theorem states that every language in NP has a verifier that uses at most $O(\log n)$ random bits and reads $O(1)$ bits from the proof.

Theorem 3 (PCP theorem, [3][4]). NP $\subseteq$ PCP$[\log n, 1]$.

Note that this is sometimes written as NP = PCP$[\log n, 1]$). Indeed, $\supseteq$ is immediate as PCP$[\log n, 1] \subseteq$ NTIME$(2^{O(\log n)})$ = NP.

2 Gap constraint satisfaction and the PCP theorem

Definition 4 (ρGAP-qCSP, $\rho \in (0,1), q \in \mathbb{N}$). A qCSP instance is a collection $\mathcal{C}$ of m constraints over an alphabet Σ such that each constraint depends on at most q literals ($|\Sigma| = 2$ is the case of boolean variables). Defining UNSAT$(\mathcal{C})$ the minimum fraction of unsatisfied constraints, the ρGAP-qCSP problems consist in :

- Outputting YES UNSAT$(\mathcal{C})$=0;
- Outputting NO if UNSAT$(\mathcal{C}) \geq \rho$.

It turns out we can tie this problem to the PCP theorem:

Theorem 5. The following are equivalent:

1. The PCP theorem;
2. There exists ρ, q such that ρGAP-qCSP is NP-hard.

We prove in appendix A that there is a $\rho > 0$ that makes ρGAP-3CSP NP-Hard.

3 The PCP Theorem by Gap Amplification

To each instance of qCSP, we can associate a constraint graph:

Definition 6 (Constraint (or Gaifman) graph for binary constraints). $G =< (V, E), \Sigma, \mathcal{C} >$ is called a constraint graph, if:

- (V, E) is an undirected graph;
- V is a set of variables taking values in Σ;
- $e = (u, v) \in E$ iff (u, v) forms a constraint, i.e $(u, v) \in \mathcal{C}$ and so UNSAT(G) = UNSAT$(\mathcal{C})$.

Observe that since the number of satisfied constraints is an integer, deciding whether $\mathcal{C}$ is satisfiable is the same as deciding UNSAT$(\mathcal{C}) \geq 1/m$. Therefore for $|\Sigma| = 3$, the gap problem $1/m$-GAP qCSP is a generalization of 3COL and is NP-hard.
The issue is that this gap depends on m. To widen the gap, we will iteratively show that ε-GAP qCSP is NP-hard for larger and larger values of ε.

Theorem 7 (Main). There exists Σ_0 such that the following holds: for any finite alphabet Σ there exist $C > 0$ and $0 < \alpha < 1$ such that, given a constraint graph $G =< (V, E), \Sigma, \mathcal{C} >$, one can construct in polynomial time, a constraint graph $G' =< (V', E'), \Sigma_0, \mathcal{C}' >$ such that:

- $|G| \leq C|G'|$;
- If $\text{UNSAT}(G) = 0$ then $\text{UNSAT}(G') = 0$;
- If $\text{UNSAT}(G) = \varepsilon$ then $\text{UNSAT}(G') \geq \min(2\varepsilon, \alpha)$ for $\alpha > 0$.

Repeating this step logarithmically many times yields G_{final} that either verifies $\text{UNSAT}(G_{\text{final}} = 0)$ if $\text{UNSAT}(G) = 0$; and $\text{UNSAT}(G_{\text{final}}) \geq 1/2$ (or α) if $\text{UNSAT}(G) \neq 0$, which proves the PCP.
The proof and construction revolves around the three following steps: graph powering, preprocessing and alphabet reduction by composition.

3.1 Graph Powering for gap amplification

Definition 8 (Graph Powering). Let $G =< (V, E), \Sigma, \mathcal{C} >$ be a d-regular constraint graph, and let $t \in \mathbb{N}$. A sequence $(u_0, \ldots, u_t)$ is called a t-step walk in G if for all $i \in [t-1]$, $(u_i, u_{i+1}) \in E$. We define $G_t :=< (V, \mathbf{E}), \Sigma^{d^{\lceil t/2 \rceil}}, \mathcal{C}^t >$ to be the following constraint graph:

- u and v are connected by k parallel edges in $\mathbf{E}$ if the number of t-step walks from u to v in G is exactly k;
- The alphabet is $\Sigma^{d^{\lceil t/2 \rceil}}$. For any $u \in V$ taking value $a \in \Sigma^{d^{\lceil t/2 \rceil}}$, a can be seen as the assignment $a : \Gamma(u) \to \Sigma$ such that $\Gamma(u)$ is the set of vertices reached during $\lceil t/2 \rceil$-walks starting from u, $|\Gamma(u)| < d^{\lceil t/2 \rceil}$;
- The constraint associated with an edge $\mathbf{e} = (u, v) \in \mathbf{E}$ is satisfied by a pair of values $a, b \in \Sigma^{d^{\lceil t/2 \rceil}}$ iff the following holds: There is $\sigma : \Gamma(u) \cup \Gamma(v) \to \Sigma$ that satisfies every constraint $c(e)$ where $e \in E \cap (\Gamma(u) \times \Gamma(v))$, and such that

  $$\forall u' \in \Gamma(u), v' \in \Gamma(v), \quad \sigma(u') = a_{u'}, \sigma(v') = b_{v'}$$

  Where $a_{u'}$ is the value a assigns $u \in \Gamma(u)$, and $b_{v'}$ the value b assigns $v \in \Gamma(v)$.

Although the constraint satisfaction seem intricate, it looks pretty natural. It might be reminiscent to Weisfeiler Lehman algorithm related to the graph isomorphism problem. It is immediate that $\text{UNSAT}(G) = 0$ implies $\text{UNSAT}(G') = 0$

Lemma 9 (Amplification Lemma). Let $0 < \lambda < d$, and Σ be constants. There exists a constant $\beta_2(\lambda, d, |\Sigma|) > 0$, such that for every $t \in \mathbb{N}$ and for every d-regular constraint graph $G =< (V, E), \Sigma, C >$ with a self-loop on each vertex and $\lambda(G) \leq \lambda$ such that:

$$\text{UNSAT}(G^t) \geq \beta_2 \sqrt{t}\, \text{UNSAT}(G), \frac{1}{t})$$

This powering operation amplifies the gap factor $\sqrt{t}$ at the price of a linear blowup in the size of the graph (the number of edges is multiplied by d^{t-1}). First note that the constraint graph is an expander, and some elements of the proof are developed in the appendix B.

3.2 Preprocessing

The aim of this step is to turn a constraint graph into one compatible with the amplification step.

Lemma 10 (Preprocessing Lemma). There exist constants $0 < \lambda < d$ and $\beta_1 > 0$ such that any constraint graph G can be transformed into a constraint graph G' such that:

- G is d-regular with self-loops, and $\lambda(G) \leq \lambda < d$;
- G' has the same alphabet as G, and $\text{size}(G') = O(\text{size}(G))$;
- $\beta_1 \cdot \text{UNSAT}(G) \leq \text{UNSAT}(G') \leq \text{UNSAT}(G)$

3.3 Alphabet Reduction by Composition

The graph powering operation increases the alphabet size, which is an issue to repeat the process.

Lemma 11 (Composition Lemma). Assume the existence of an assignment tester $\mathcal{P}$, with constant rejection probability $\varepsilon > 0$, and alphabet Σ_0 of size $O(1)$. There exists $\beta_3 > 0$ that depends only on $\mathcal{P}$, such that given any constraint graph $G =< (V, E), \Sigma, \mathcal{C} >$, one can compute, in linear time, the constraint graph $G' = G \circ \mathcal{P}$, such that:

- $\text{size}(G') = c(\mathcal{P}, |\Sigma|) \cdot$ size(G);
- $\beta_3 \cdot \text{UNSAT}(G) \leq \text{UNSAT}(G') \leq \text{UNSAT}(G)$.

4 Some hardness of approximation results using PCP

Constructing, for any $\delta > 0$, a probabilistically checkable proof for NP which uses logarithmic randomness and δ amortized free bits, Hastad [5] proved that the size of the largest clique in a graph with n nodes is hard to approximate in polynomial time within a factor $n^{1-\varepsilon}$.

Using a 3-query PCP, Hastad [6] also showed that for every $\varepsilon > 0$, there is no polynomial-time $(7/8 + \varepsilon)$-approximation for MAX3SAT unless P = NP.

Recall that the soundness parameter of a PCP system is the probability that the verifier may accept a false statement. The soundness parameter can be made arbitrary small by increasing the number of queries. Yet for some applications we need a system with, say, three queries, but an arbitrarily small constant soundness parameter
Raz [7] has shown that this can be achieved if we consider systems with non binary alphabet, using parallel repetition (of independent copies of a verifier). For any $\varepsilon > 0$, there exists Σ (of size poly$(1/\varepsilon)$), such that Gap-Label-Cover$(\Sigma)_{1,\varepsilon}$ is NP-hard.

References

[1] Irit Dinur. The pcp theorem by gap amplification. 2007.

[2] Sanjeev Arora and Boaz Barak. Computational Complexity: A Modern Approach. 2009.

[3] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. 1998.

[4] Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: A new characterization of np. 1998.

[5] Johan Håstad. Clique is hard to approximate within $n^{1-\epsilon}$. 1999.

[6] Johan Håstad. Some optimal inapproximability results. 2001.

[7] Ran Raz. A parallel repetition theorem. 1995.

[8] N. Linial and A. Wigderson. Expander graphs and their applications. lecture notes of a course: http://www.math.ias.edu/ boaz/expandercourse/, 2003.

[9] Omer Reingold, Salil Vadhan, and Avi Wigderson. Entropy waves, the zig-zag graph product, and new constant-degree. 2004.

A Equivalence of the PCP and gap amplification (theorem 5)

For $\Leftarrow$, the proof revolves around V running a reduction from an NP-complete language L to the gap constraint satisfaction problem. Then for a proof π, V select a clause at random and check its 3 variables values, V accepts if the clause is satisfied.
Then if $x \in L$, $\Pr[V^\pi(x) \text{ accepts}] = 1$ and if $x \notin L$, $\Pr[V^\pi(x) \text{ accepts}] = 1 - s$, we can repeat it $O(1)$ times independently to get $\frac{1}{2}$.

For $\Rightarrow$, fix L in NP, there is a verifier V that reads $c \log n$ random bits, accesses $q = O(1)$ bits from the proof and decides whether to accept or reject. For each fixed random bit pattern $r \in \{0,1\}^{c\log n}$, V deterministically reads a fixed set of q bits from the proof: $i_1^{(r)}, \dots, i_q^{(r)}$.
Denote by $C(r) \subseteq \{0,1\}^q$ the possible contents of the accessed proof bits that would cause V to accept. Let $N = 2^{O(\log n)}$ be the number of deterministic verifiers associated to the $O(\log n)$ random bits. The reduction converts for each deterministic verifier the q-tuples from $C(r)$ into an equivalent E3CNF formula, adding auxiliary variables if needed. We may assume that each equivalent E3CNF formula has $K = q2^q$ clauses. We takes the conjunctions of the $K \times N$ clauses.
From the PCP this reduction shows that ρGAP-E3CSP with $\rho = 1/2K$ is NP-hard.

B Elements of proof of the amplification lemma (lemma 10)

We start by introducing expander graphs [8] [9]:

Definition 12 (Edge expansion). The edge expansion of a graph $G = (V, E)$, denoted by $h(G)$, is defined as

$$h(G) = \min_{S \subseteq V, |S| \leq |V|/2} \frac{E(S, \bar{S})}{|S|}$$

Lemma 13 (Expanders). There exist $d \in \mathbb{N}$ and $h_0 > 0$, such that there is a polynomial-time constructible family $\{X_n\}_{n\in\mathbb{N}}$ of d-regular graphs X_n on n vertices with $h(X_n) \geq h_0$. Such graphs are called expanders.

An alternate way of looking at expanders is the following:

Definition 14. A d-regular graph G is an (n, d, λ)-expander if $\lambda = \max_{i\neq 0}|\lambda_i(G)|$ and $\lambda < d$. λ is the second largest eigenvalue in absolute value.

There is a relation between the edge expansion and the second eigenvalue:

Theorem 15. Let G be a (n, d, λ)-expander, then $2h(G) \geq (d - \lambda) \geq \frac{h(G)^2}{2d}$.

The following corollary is straightforward adding d_0 self loops to each vertex:

Corollary 16. In other words, large expansion is equivalent to large spectral gap. There exist $d_0' \in \mathbb{N}$ and $0 < \lambda_0 < d_0'$, such that there is a polynomial-time constructible family $\{X_n\}_{n\in\mathbb{N}}$ of d_0'-regular graphs X_n on n vertices with $\lambda(X_n) < \lambda_0$.

Now we estimate the random-like behaviour of a random-walk on an expanders:

Proposition 17. Let $G = (V, E)$ be a d-regular graph with $\lambda(G) = \lambda$. Let $F \subset E$ be a set of edges without self loops, and let K be the distribution on vertices induced by selecting a random edge in F and then a random endpoint.
The probability p that a random walk that starts with distribution K takes the $i + 1^{\text{st}}$ step in F is upper bounded by $\frac{|F|}{|E|} + (\frac{|\lambda|}{d})^i$.

Finally we can turn to the proof of the amplification lemma. The idea is to do the following:
Let us refer to the edges of G^t as walks, since they come from t-step walks in G, and let us refer to the edges of G as edges. Given an assignment for G^t, $\vec{\sigma} : V \to \Sigma^{dt/2}$, we extract from it a new assignment $\sigma : V \to \Sigma$ by assigning each vertex v the most popular value among the "opinions" (under $\vec{\sigma}$) of v's neighbours. We then relate the fraction of edges falsified by this "popular-opinion" assignment σ to the fraction of walks falsified by $\vec{\sigma}$.
The probability that a random edge rejects this new assignment is, by definition, at least UNSAT(G). The idea is that a random walk passes through at least one rejecting edge with even higher probability. Moreover, we claim that if a walk does pass through a rejecting edge, it itself rejects with constant probability.

C Remarks

Recall that NP = PCP[log n, 1]. It is not too difficult to see that:

- PCP[0,0] = P = PCP[0,log n];
- NP = PCP[0,poly(n)];
- If SAT $\in$ PCP($r(n)$, 1) for $r(n)$ =o(log n) then P = NP.