CMPUT 675: Computational Complexity Theory **Winter 2019**

Lecture 21 (Mar 26): The PCP Theorem: Part I

Lecturer: Zachary Friggstad *Scribe: Zachary Friggstad*

21.1 Preliminaries

We now have enough background to prove the PCP theorem.

Between Lectures 21 and 22, these notes begin with a slight recap of what we discussed at the end of lecture 20 and contain a bit of the discussion at the start of lecture 23. They also contain more detail than what was discussed in the lectures and they lift the assumption that the expanders are loopless. I wanted a comprehensive writeup you could use as reference if you wanted to verify the details (and it was good for me to be reminded of some details too).

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

The easy direction $\mathbf{PCP}(O(\log n), O(1)) \subseteq \mathbf{NP}$ was established earlier: recall it is simply because the behaviour of a $(O(\log n), O(1))$-**PCP** verifier V can be efficiently simulated on all $2^{O(\log n)} = \text{poly}(n)$ random strings of length $O(\log n)$. In this way, we can get an **NP**-verifier for any $L \in \mathbf{PCP}(O(\log n), O(1))$ that expects the proof π to be accepted by V for all random strings.

So we focus on showing $\mathbf{NP} \subseteq \mathbf{PCP}(O(\log n), O(1))$. As discussed earlier, it suffices to give a "gap introducing" Karp reduction from 3SAT to any constraint satisfaction problem. By gap introducing, we mean that in the *no* case where an instance ϕ of 3SAT is not satisfiable then in the resulting CSP at most a ρ-fraction of constraints can be satisfied by any satisfying assignment where $\rho < 1$ is some constant.

We briefly recall some definitions from earlier lectures:

- For integers q, W, an instance of $q\text{CSP}_W$ is a constraint satisfaction where each clause depends on a subset of at most q variables which take values in $[W] = \{1, \ldots, W\}$. We say q is the **arity** of the CSP and W is the **alphabet size** of the CSP.

  The actual representation of the alphabet is not important, sometimes we use $\{0, \ldots, W-1\}$ or a different set of size W.
- For $q\text{CSP}_W$ instance ϕ, let $\text{sat}(\phi)$ be the maximum fraction of clauses that can be satisfied by some assignment of values $[W]$ to the variables.
- A reduction f that maps instances ϕ of $q\text{CSP}_W$ to instances ϕ' of $q'\text{CSP}_{W'}$ is a **CL**-reduction if:
  - Computing ϕ' takes $\text{poly}(|\phi|)$ time.
  - If $\text{sat}(\phi) = 1$ then $\text{sat}(\phi') = 1$ (i.e. f preserves satisfiability).
  - There is some constant C such that if ϕ has m clauses then ϕ' has $\leq C \cdot m$ clauses.

  Essentially, a CL-reduction if it is a Karp reduction with "linear blowup" in size.

Our main lemma to prove the PCP theorem is the following:

Lemma 1 *There are constants $q_0 \geq 3, \epsilon_0 > 0$ such that there is a CL-reduction f from $q_0\mathrm{CSP}_2$ to $q_0\mathrm{CSP}_2$ with the additional property that for any instance ϕ of $q_0\mathrm{CSP}_2$ with, say, sat$(\phi) = 1 - \epsilon$ where $\epsilon \geq 0$, we have sat$(\phi') \leq 1 - \min\{\epsilon_0, 2\epsilon\}$.*

Recall the PCP theorem would then follow simply, the proof is recalled here so these notes are more self-contained.

Proof of the PCP Theorem
An instance of 3SAT can be viewed as an instance of $q_0\mathrm{CSP}_2$. Thus, for any $L \in \mathbf{NP}$ there is a Karp reduction g from L to an instance of $q_0\mathrm{CSP}_2$.

If $x \notin L$ then $\mathrm{sat}(g(x)) \leq 1 - 1/m$ where m is the number of clauses of $g(x)$. Now consider the reduction f from Lemma 1 and apply it to $g(x)$ $\lceil \log_2 m \rceil$ times, i.e. compute $f^k(g(x))$ where $k = \lceil \log_2 m \rceil$. This reduction still runs in polynomial time the number of clauses of $f^i(g(x))$ is at most $C^i \cdot m$ which is polynomial in m (because $i \leq \lceil \log_2 m \rceil$). But if $\textsc{sat}(g(m)) \leq 1 - 1/m$ then $\textsc{sat}(f^k(g(m))) \leq 1 - \min\{\epsilon_0, 2^k/m\} = 1 - \epsilon_0$, as required.

That is, $f^m \circ g$ is a Karp reduction from L to $q_0\mathrm{CSP}_2$ such that if $x \notin L$ then $\mathrm{sat}(f^m(g(x))) \leq 1 - \epsilon_0$. ■

21.2 Strategy

We begin by outlining the strategy for proving Lemma 1. It is somewhat reminiscent of how we constructed expanders. Recall that with the expander construction we interleaved two operations: one increased the spectral gap but also increased the degree. The other operation reduced the degree and only slightly worsened the spectral gap. Interleaving these two operations produced arbitrarily large expanders.

For the PCP theorem, we consider two main operations: one will decrease the satisfiability of the CSP instance but will greatly increase the size of the alphabet in the CSP instance (from 2 to a large constant). The other will reduce the size of the alphabet from this large constant back to 2 while only slightly increasing the satisfiability of the CSP instance. Throughout, the arity of the CSP will remain bounded by an absolute constant q_0.

The first step, decreasing the satisfiability, will come in two major substeps: **preparation** and **gap amplification**. First, the CSP needs to be tweaked (through some simple modifications) so it is a 2CSP instance with bounded alphabet size such that the graph of constraints is an expander. Then gap amplification uses the fact it is an expander to greatly increase the unsatisfiability of the instance. The final step, **alphabet reduction**, is actually quite easy at this point of the course: we already did the hard work in the proof of $\mathbf{NP} \subseteq \mathbf{PCP}(\mathrm{poly}(n), O(1))$ in Lecture 15.

A brief overview of how the parameters vary is summarized in Figure 21.1. For entries with arity 2, we can think of the CSP ϕ as being given by a graph G_ϕ whose variables are vertices and constraints are edges. The last column includes comments on the properties of G_ϕ.

Let (d, λ) be fixed values with $\lambda < 1$ such that there is a family of (d, λ)-expanders $\mathcal{G} = \{G_n\}_{n \geq 1}$ where each G_n can be constructed in poly(n). Note, we do not require *strongly explicit* constructions here. Our specific choice of (d, λ) will be dictated by some properties we require of these expanders in Section 21.3.2.

Let ϕ be an instance of $q_0\mathrm{CSP}_2$ for some constant q_0 we will discover later. Say $\textsc{sat}(\phi) = 1 - \epsilon$ where $\epsilon \leq \epsilon_0$ for some constant $\epsilon_0 > 0$ that we will, again, discover later. In the following table, each row after the first row summarizes the relevant properties of the CSP obtained after applying a reduction we will discuss.

All reductions are CL-reductions. The variable D listed below is yet another constant we will discover when describing the reduction.

Ultimately this proof is due to Dinur [D07], but the original proof of the PCP theorem is from [AS98, ALMSS98].

	arity	alphabet size	sat value	graph comments
original ϕ	q_0	2	$1-\epsilon$	
prep-1	2	2^{q_0}	$\leq 1-\epsilon/q_0$	
prep-2	2	2^{q_0}	$\leq 1-\epsilon/(3dq_0)$	graph is regular
prep-3	2	2^{q_0}	$\leq 1-\epsilon/(9dq_0)$	graph is an expander
gap amplification	2	D	$\leq 1-4\epsilon$	
alphabet reduction	q_0	2	$1-2\epsilon$	

Figure 21.1: The sequence of steps followed to prove Lemma 1.

The proof we follow for the gap amplification step deviates noticeably from the book [AB09]. It uses a simplification in [R06] and the presentation is partially inspired by course notes from [GO05]. Though, it is similar in spirit to the presentation of gap amplification in the book [AB09].

Ultimately, after these notes there should be only two key details that are missing: the analysis of the linearity test and the construction of the "base" $((2d)^{50}, d, 0.01)$-expander mentioned in the expander construction lecture. We will provide the analysis of the linearity test next, so only the expander construction will be missing from the course notes (a source was provided in the expander lecture).

21.3 Preparation

Again, every reduction presented from this point forward will be a CL-reduction.

21.3.1 Prep-1

Input: A $q_0\text{CSP}_2$ instance ϕ. Say $\text{sat}(\phi) = 1-\epsilon$ (could be $\epsilon = 0$, *eg.* ϕ is satisfiable).
Output: A 2CSP_W instance ϕ' where $W = 2^{q_0}$ with $\text{sat}(\phi') \leq 1-\epsilon/q_0$.

Say ϕ has n variables $X = \{x_1, \ldots, x_n\}$ and m clauses $\mathcal{C} = \{C_1, \ldots, C_m\}$. Consider the following CL-reduction that converts ϕ to an instance ϕ' of $2\text{CSP}_{2^{q_0}}$ as follows:

- The variables of ϕ' are $X \cup \mathcal{C}$.
- The alphabet size is 2^{q_0}. For variables of ϕ' coming from a clause C_i of ϕ, view the entries in the alphabet as a tuple $\{0,1\}^{q_0}$ of bits providing an assignment to variables in C_i. If C_i depends on fewer than q_0 variables, just use the first few in the tuple.
- For variables of ϕ' coming from a variable x_j of ϕ, again view the alphabet as a tuple of bits $\{0,1\}^{q_0}$ and think of the first bit in the tuple as the "value" for x_j.
- For each clause C_i and each variable x_j appearing in C_i, add a constraint to ϕ' depending on x_j and C_i that is satisfied by an assignment $\sigma : X \cup \mathcal{C} \to \{0,1\}^{q_0}$ if $\sigma(C_i)$ satisfies C_i itself and the value it assigns x_j equals the value of x_j given by $\sigma(x_j)$.

The number of clauses is $\leq q_0 \cdot m$ and it is clear that satisfiable instances remain satisfiable, so this is a CL-reduction.

Lemma 2 *If sat(ϕ) $= 1 - \epsilon$ then sat(ϕ') $\leq 1 - \epsilon/q_0$.*

Proof. Consider some assignment $\sigma : X \cup \mathcal{C} \to \{0,1\}^{q_0}$ to the variables of ϕ'. By restricting this to X, we get an assignment $\sigma' : X \to \{0,1\}$ to the variables of ϕ. The fraction of clauses that are *not* satisfied by σ' is at least ϵ. For each clause C_i that is not satisfied by ϕ', the corresponding q_0 clauses in ϕ' that involve C_i cannot all be satisfied.

That is, if $\sigma(C_i)$ does not satisfy C_i then all q_0 clauses involving C_i in ϕ' are not satisfied. Otherwise, if $\sigma(C_i)$ does satisfy C_i then its assignment to some variable x_j of C_i disagrees with $\sigma(x_j)$. ■

Observe: The constraint graph of ϕ' has no loops.

21.3.2 Prep-2

Input: A 2CSP$_W$ instance ϕ'. Say sat(ϕ') $\leq 1 - \epsilon'$.
Output: A 2CSP$_W$ instance ϕ'' with sat(ϕ'') $\leq 1 - \epsilon'/3d$. Also, the constraint graph for ϕ'' is $(d+1)$-regular for some constant d.

Let ϕ' be an instance of 2CSP$_W$ on m clauses and let $X', \mathcal{C}$ be the variables and clauses of ϕ', respectively. Consider a family $\mathcal{G} = \{G_n\}_{n \geq 1}$ of (d, λ)-expanders with $\lambda < 1$ such that G_n can be constructed in poly(n) time and such that for any n and any $S \subseteq V(G_n)$ with $|S| \leq n/2$, we have $|\delta(S)| \geq 2 \cdot |S|$. The existence of such a family for some constants (d, λ) is an exercise question.

Consider the following reduction that maps ϕ' to an instance ϕ'' of 2CSP$_W$. The point is that the graph of constraints is $(d+1)$-regular after the reduction.

For each $x'_j \in X'$, say x'_j lies in m_j constraints of ϕ'. Replace x'_j with a copy of G_{m_j} where each vertex of G_{m_j} is thought of as a copy of x'_j. The m_j original constraints that involved x'_j have x'_j replaced by a vertex of G_{m_j} (a copy of x'_j) in a 1-to-1 fashion. Finally, each edge of G_{m_j} is regarded as a constraint that is satisfied only if both endpoints have the same value.

For each $x'_j \in X$, let X''_j be the nodes of the copy of G_{m_j} that replaced variable x'_j and let $X'' = \cup_{x'_j \in X'} X''_j$ be the variables of ϕ''.

This is a CL-reduction. Again, it is clear that if ϕ' is satisfiable then so is ϕ'': have all copies of x'_j in X''_j take the original value of x'_j. Each G_n has at most $d \cdot n$ edges which is certainly an overestimate but does account for the possibility of loops. So the number of new constraints is at most $\sum_j d \cdot m_j = d \sum_j m_j = 2dm$, recalling the constraint graph for ϕ' has no loops. Thus, the total number of constraints of ϕ'' is at most $2dm + m \leq 3dm$.

Lemma 3 *There is an assignment $\sigma : X'' \to [W]$ for ϕ'' satisfying the maximum number of clauses possible such that for each original variable x'_j, σ assigns each variable in X''_j the same value.*

The proof of this lemma is also an assignment question.

Corollary 1 *If sat(ϕ') $= 1 - \epsilon'$ then sat(ϕ'') $\leq 1 - \epsilon'/(2d+1)$.*

Proof. By the previous Lemma, there is an optimum assignment $\sigma : X'' \to [W]$ for ϕ'' assigning the same value to all copies of a variable x'_j of ϕ'. Interpret this as an assignment $\sigma' : X' \to [W]$ in ϕ where $\sigma'(x'_j)$ is the common value σ assigns to all copies of x'_j in X''_j.

Then σ' fails to satisfy at least $\epsilon' \cdot m$ clauses of ϕ'. The corresponding clauses in ϕ'' are also unsatisfied, so at least an $\epsilon/(2d+1)$-fraction of constraints of ϕ'' are not satisfied by σ. ∎

Observe that the graph of constraints for ϕ'' is a $(d+1)$-regular graph and may contain loops.

21.3.3 Prep-3

Input: A 2CSP$_W$ instance ϕ'' whose constraint graph $G_{\phi''}$ is $(d+1)$-regular. Say $\text{sat}(\phi'') = 1 - \epsilon''$.
Output: A 2CSP$_W$ instance ϕ''' whose constraint graph is a $(2d+1, \lambda')$-expander for some constant $\lambda' < 1$ where $\text{sat}(\phi''') \leq 1 - \epsilon''/3$.

Consider the same family of expanders $\mathcal{G}$ from the last section. Let ϕ'' be an instance of 2CSP$_W$ such that its constraint graph $G_{\phi''}$ is $(d+1)$-regular. The point of this section is to turn the constraint graph into an expander graph.

Consider the reduction that takes ϕ'' to another instance of 2CSP$_W$ as follows. Let n be the number of variables/vertices of ϕ''. Map the nodes of the expander G_n in a one-to-one fashion to the nodes of $G_{\phi''}$ and add all edges of G_n to $G_{\phi''}$. These correspond to new constraints that are satisfied by any variable assignment.

Again, this is clearly a CL reduction.

Lemma 4 *If sat$(\phi'') = 1 - \epsilon''$ then sat$(\phi''') \leq 1 - \epsilon''/4$*

Proof. Any assignment to ϕ''' leaves at least an ϵ-fraction of the clauses of ϕ'' unsatisfied. An d'-regular graph on n nodes that may have loops contains between $\frac{d}{2} \cdot n$ and $d \cdot n$ edges. As $G_{\phi''}$ is $(d+1)$-regular and since we obtained ϕ''' by adding a d-regular graph, at least $1/3$ of the constraints in ϕ'' are from ϕ'. So at least an $\epsilon/3$-fraction of edges of ϕ''' are not satisfied. ∎

Lemma 5 *The graph $G_{\phi'''}$ of constraints of ϕ''' is a $\left(2d+1, \frac{2}{3} + \frac{\lambda}{3}\right)$-expander.*

Proof. The random walk matrix $\overline{A}$ of $G_{\phi''')}$ is

$$\overline{A} = \frac{d+1}{2d+1} \cdot A_{\phi''} + \frac{d}{2d+1} \cdot A_n$$

where $A_{\phi''}$ is the random walk matrix of $G_{\phi''}$ and A_n is the random walk matrix of G_n. The second largest eigenvalue of A_n is at most λ and the second largest eigenvalue of $A_{\phi''}$ is, trivially, at most 1. Let $x \in \mathbb{R}^n$ be a unit vector that is an eigenvector for the second largest eigenvalue λ' of A.

So by a Rayleigh quotient and the fact x is orthogonal to the all-1 vector to justify the second inequality,

$$|\lambda'| = |\langle \overline{A}x, x\rangle| \leq \frac{d+1}{2d+1} \cdot |\langle A_{\phi''}x, x\rangle| + \frac{d}{2d+1} \cdot |\langle A_n x, x\rangle| \leq \frac{d+1}{2d+1} \cdot 1 + \frac{d}{2d+1} \cdot \lambda \leq \frac{2}{3} + \frac{\lambda}{3}.$$

∎

The bound $\frac{2}{3} + \frac{\lambda}{3}$ from the proof is obviously crude since d is likely modestly large. But any constant bound < 1 suffices.

21.4 Gap Amplification

Input: A 2CSP$_W$ instance ϕ whose constraint graph G is a (d', λ')-expander for some constants d', λ'. Say $\text{sat}(\phi) = 1 - \epsilon$ where ϵ could be 0 (if it is satisfiable).
Output: A 2CSP$_D$ instance ϕ' for some constant D where $\text{sat}(\phi') \leq 1 - \min\{4\epsilon, \epsilon'\}$ where $\epsilon' > 0$ is some constant. Both D and ϵ' will be determined in the proof.

Before describing the gap amplification step, we should discuss a different form of random walk on a graph and analyze its properties. We will begin by assuming the graph has no loops. Of course, the expanders we constructed earlier have loops. The necessary adjustments to the proof to account for loops will be discussed at the end.

21.4.1 Lazy Random Walks

The random walk we consider in the reduction is the same as a standard random walk from a uniformly chosen vertex, except the stopping point is determined by flipping a biased coin after each step rather than just some bounded length.

Let $G = (V; E)$ be a d-regular graph. Consider the following process where $t > 1$ (which will be a fixed constant).

Lazy Random Walk

- $v_0 \sim V$
- $i \leftarrow 0$
- loop
  - v_{i+1} is a random neighbour of v_i chosen by sampling a random edge touching v_i and crossing that edge (it could be $v_{i+1} = v_i$ if a loop at v_i is sampled)
  - $i \leftarrow i + 1$
  - with probability $1/t$, stop the random walk (break the loop)

This walk eventually stops with probability 1 since the probability it takes at least k steps is $(1 - 1/t)^k$.

We also consider the following alternative process. The main difference is that we are given a start vertex and we flip the coin to see if we should stop also before the first step. Let $t > 1$ and let $u \in V$.

Lazy Random Walk 2

- $v_0 \leftarrow u$
- $i \leftarrow 0$
- loop
  - with probability $1/t$, stop the random walk (break the loop)
  - v_{i+1} is a random neighbour of v_i chosen by sampling a random edge touching v_i and crossing that edge (it could be $v_{i+1} = v_i$ if a loop at v_i is sampled)
  - $i \leftarrow i + 1$

We summarize some important properties of these walks.

Let uw be an edge and $k \geq 1$ be an integer. Let $\mathcal{D}(u, v, k)$ be the distribution over walks if we conditioned **Lazy Random Walk 1** to use *exactly* k steps of the form $u \to v$ (in that direction).

Lemma 6 *The distribution over the start vertex a of a path from $\mathcal{D}(u, v, k)$ is the same as the distribution over endpoints obtained from starting* **Lazy Random Walk 2** *from u. Similarly, the distribution over the end vertex b of a path from $\mathcal{D}(u, v, k)$ is the same as the distribution over endpoints obtained from starting* **Lazy Random Walk 2** *from v. The random variables a, b are independent.*

Proof. This is slightly informal, but conveys the ideas.

We first discuss how b is distributed as if it was sampled from **Lazy Random Walk 2** starting from v. Observe that if this lazy random walk ever uses a $u \to v$ step, then it is like it starts over again from v. That is, whether we declare it "failed" and restart it or if we continue it, we get the same distribution over endpoints. Therefore, conditioning the **Lazy Random Walk 2** starting from v on the event that a $u \to v$ step is never taken gives us the same distribution as the original **Lazy Random Walk 2** starting from v.

For the first part regarding a, we could reason similarly and get that the distribution over endpoints taking **Lazy Random Walk 2** from u is the same as if we had conditioned this distribution to not take a $v \to u$ step. One just has to observe everything is reversible because the graph is undirected and d'-regular: the probability a **Lazy Random Walk 2** from u that avoids $v \to u$ ends at a is the same as the probability that a **Lazy Random Walk 1** from a that avoids $u \to v$ ends at u.

Independence of the endpoints a and b follows as these two lazy random walks can be done independently (i.e. once the walk from a to the last occurrence of $u \to v$ is known, the endpoint b is still distributed according to **Lazy Random Walk 2** from v). ■

21.4.2 The Variables and Alphabet

The variables of ϕ' are the same as the variables in ϕ, namely we think of them as the vertices V of the constraint graph G of ϕ.

We will fix some constant t soon. For this constant, let the new alphabet size be $D := W^{d'^{t+1}}$ and think of the alphabet as tuples of the form $[W]^{d^{t+1}}$. For vertices u, v, say that u **has an opinion for** v if the shortest path distance $d(u, v)$ in G is at most t. Note that any vertex u has an opinion for at most d^{t+1} vertices.

Think: For $u \in V$ and an assignment $\overline{\sigma}(u) \in [W]^{d'^{t+1}}$, $\overline{\sigma(u)}$ is giving u's opinion for the W-value every nearby node should take. That is, each v that is within distance t of u has a corresponding coordinate in the tuple $[W]d'^{t+1}$ for u. Note, $\overline{\sigma(u)}$ may contain "unused" entries that do not correspond to vertices u has an opinion for, this is ok.

21.4.3 The Constraints

We temporarily consider weighted CSPs. That is, each constraint has a nonnegative weight and all weights sum to 1. The value of an assignment is the total weight of all constraints that are satisfied. So an unweighted CSP with m constraints can be thought of as a weighted CSP where each constraint has weight $1/m$.

We will describe a randomized procedure that generates a constraint randomly. The weight of the constraint will be the probability it is generated. We first describe a procedure that, in fact, samples infinitely many constraints but with decreasing probabilities (akin to the idea that we can toss coins until we see tails). It is more convenient to analyze this procedure. Later, we will fix this issue in a simple way to only generate $O(n)$ constraints.

Constraints - Version 1

Perform **Lazy Random Walk 1** to get a path $a = v_0, v_1, \ldots, v_k = b$. Add an a-b edge/constraint to ϕ' that is satisfied if and only if for *every* step $v_i v_{i+1}$ for which a has an opinion for v_i and b has an opinion for v_{i+1}, the $v_i v_{i+1}$ constraint in ϕ is satisfied by this opinion.

More precisely, if $\overline{\sigma} : V \to [W]^{d'^{t+1}}$ is an assignment for ϕ' then for any $v_i v_{i+1}$ step for which $d(a, v_i) \le t$ and $d(b, v_{i+1}) \le t$, the opinion of a for v_i given by $\overline{\sigma}(a)$ and the opinion of b for v_{i+1} given by $\overline{\sigma}(b)$ satisfy the constraint $v_i v_{i+1}$ in ϕ.

The proof continues in Lecture 22...

References

AB09 S.Arora and B.Barak, Computational Complexity: A Modern Approach, *Cambridge University Press,New York, NY, USA*, 2009, pp. 126–151.

ALMSS98 S.Arora, C.Lund, R.Motwani, M.Sudan, and M. Szegedy, Proof verification and the hardness of approximation problems, Journal of the ACM 45 (3): 501–555, 1998.

AS98 S.Arora and S.Safra, Probabilistic checking of proofs: A new characterization of NP, Journal of the ACM, 45 (1): 70–122, 1998.

D07 I.Dinur, The PCP theorem by gap amplification, Journal of the ACM, 54 (3): 12, 2007.

GO05 V.Guruswami and R.O'Donnell, CSE 533: The PCP Theorem and Hardness of Approximation, `https://courses.cs.washington.edu/courses/cse533/05au/`, 2005.

R06 J.Radhakrishnan, Gap amplification in PCPs using lazy random walks, In Proceedings of ICALP, 96–107, 2006.