6.841: Advanced Complexity Theory Fall 2012

Lecture 17 — November 8, 2012

Prof. Dana Moshkovitz *Scribe: Mark Bun*

1 Overview

In the previous lecture, we saw an overview of probabilistically checkable proofs, the PCP theorem, and hardness of approximation. The PCP theorem gives a new characterization of **NP** as problems having proofs which can be checked using $O(\log n)$ randomness and only a constant number of bits of the proof. In this lecture, we embark on a combinatorial proof of the PCP theorem itself. A roadmap and part of the main reduction will be given this time, and we will finish the proof in the following lecture.

2 Two approaches to the PCP theorem

As one of the most significant results in complexity theory, the PCP theorem is also one of the hardest to prove. There are currently several proofs known for the theorem. They follow two main approaches, which are

1. **Algebraic:** This is the traditional approach used in the original papers of Arora and Safra [2] and Arora, Lund, Motwani, Sudan and Szegedy [1]. It requires a fair amount of technical machinery, most notably in the problem of low degree testing. The best known parameters for the PCP theorem come from this approach.
2. **Combinatorial:** In 2005, Dinur [3] discovered a combinatorial proof that exploits the connection between PCPs and hardness of approximation. We will use her strategy in this and the upcoming lecture.

3 Reformulation

To make it more amenable to a combinatorial argument, we reformulate the PCP theorem as a statement about the hardness of a certain gapped CSP. Suppose we are given a graph $G = (V, E)$, an alphabet Σ, and a collection of edge constraints $\Phi = \{\varphi_e \subset \Sigma \times \Sigma\}_{e \in E}$. Let $A : V \to \Sigma$ be an assignment of symbols to the vertices of G. We say that $e = (u, v)$ is a *satisfied edge* under this assignment if $\varphi_e(A(u), A(v)) = 1$.

Theorem 1 (Reformulation of the PCP theorem)**.** *Given a constrained graph (G, Σ, Φ), it is* **NP***-hard to distinguish between the following cases:*

- *(Completeness) There exists an assignment $A : V \to \Sigma$ that satisfies every edge.*

- *(Soundness) For every assignment $A : V \to \Sigma$, at most 99.9% of the edges are satisfied.*

It is easy to see that Theorem 1 implies $\mathbf{NP} \subset \mathbf{PCP}(O(\log n), O(1))$, which is the hard direction of the PCP theorem as stated in the last lecture. Let L be any language in **NP**. To devise a PCP for L, run the reduction to the problem described above. A proof for an instance x will correspond to an assignment A, and locations in the proof correspond to the vertices. To verify this proof, we pick an edge at random and check whether A satisfies it. If $x \in L$, then a satisfying assignment exists and its check will pass with probability 1. On the other hand if $x \notin L$, then for any assignment, this check will fail with probability at least .001, which we can amplify to 1/6 via repetition.

We leave it as an exercise to show the other direction. That is, $\mathbf{NP} \subset \mathbf{PCP}(O(\log n), O(1))$ implies Theorem 1.

4 Proof outline

As a starting point, we consider the following modification to the main theorem.

Proposition 2. *Consider the situation of Theorem 1. It is* **NP***-hard to distinguish between the cases:*

- *There exists an assignment such that all edges are satisfied.*
- *For every assignment, the fraction of satisfied edges is at most $1 - 1/|E|$.*

This version is immediate given the classical theory of **NP**-completeness. The statement that at most a $1 - 1/|E|$ fraction of the edges are satisfied is equivalent to there being any unsatisfied edge. Hence this is just the problem of deciding whether some assignment satisfies all the edges. If we take $\Sigma = \{\text{red, white, blue}\}$ and $\varphi_e(c_1, c_2) \iff c_1 \neq c_2$, then we get a straightforward reduction from the **NP**-complete 3-coloring problem.

The idea of the proof of Theorem 1 is to give a reduction that amplifies the gap between the two cases. That is, we want to cut the $1 - 1/|E|$ soundness parameter down to a constant 0.999. This reduction will actually come as an iteration of reductions of the following form.

Proposition 3. *Given (G, Σ, Φ), there is a polynomial-time algorithm constructing (G', Σ', Φ') such that*

1. *(Completeness) If G is satisfiable, then G' is satisfiable.*
2. *(Soundness) If at most $1 - \delta$ edges of G are satisfiable, then at most $1 - \delta'$ edges of G' are satisfiable, where $\delta' \geq \min\{2\delta, 0.001\}$.*
3. *$|\Sigma'|$ is bounded by a universal constant C.*
4. *$|G'| \leq M|G|$ for a constant M.*

We now argue that Proposition 3 is enough to achieve the soundness required by Theorem 1. Starting with $\delta = 1/|E|$, we repeat this reduction $\log |E|$ times so that we end up with $\delta' \geq 0.001$.

In each repetition, the alphabet size remains bounded. Moreover, the size of the graph increases by a factor of at most $M^{\log |E|} = poly(|E|)$, so the cumulative reduction still runs in polynomial-time.

Our goal now is to prove Proposition 3, which itself we break up into two reductions.

1. Achieve the completeness, soundness requirements while maintaining a linear blowup in the graph size. However, the alphabet size can increase dramatically.
2. Reduce $|\Sigma'|$ to maintain a constant bound.

We prove half of the first step today, and leave the rest to the subsequent lecture.

5 First reduction

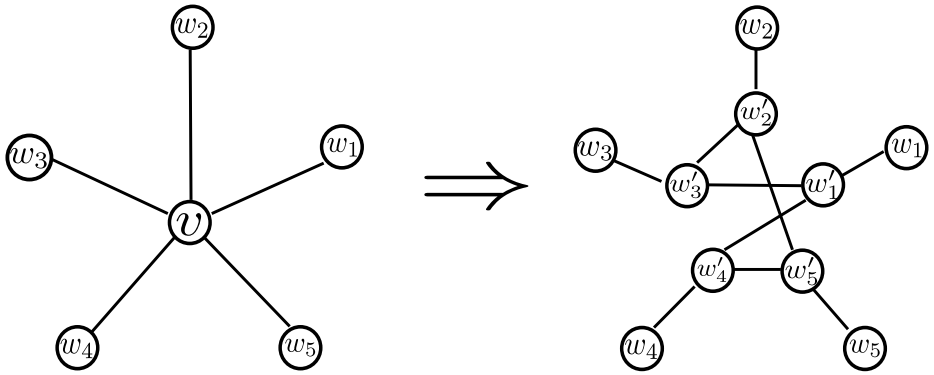
The first reduction itself comes in two phases: a structure-building step, and a graph powering step.

1. **Make G into an expander.** Specifically, we want to turn G into a degree-d expander with self-loops having $d = \Theta(1)$ and normalized second eigenvalue $\lambda < 1$. We preserve completeness with this reduction, but the gap may increase to, say, $1 - \delta/1000$. However, Σ is fixed and $|G|$ grows by at most a constant factor.
2. **Amplify the gap with graph powering.** If at most a $1 - \delta$ fraction of the edges in G are satisfiable, then at most $1 - \min\{t\delta, 0.001\}$ of the edges in G' are satisfiable, for some parameter t. Completeness is preserved and $|G|$ increases linearly. This is the step that blows up $|\Sigma'|$.

5.1 Getting an expander

We will convert G into an expander in the simplest way possible – by adding in the edges of an explicitly constructed expander on the same vertex set. But first, we have to replace G with a d-regular graph for $d = \Theta(1)$.

Figure 1: Local transformation to a d-regular graph (this is a toy example with $d = 3$)



Making G d-regular We want the maximum fraction of satisfiable edges in the new graph to be close to the maximum fraction for the original graph. Consider a vertex $v \in V$ with degree D. We replace v with a collection of vertices w' for each w incident to v, as shown in the figure. The idea is to add constraints that force the new vertices to get the same symbol in a maximally satisfying assignment. We do this by constructing a $(d-1)$-regular explicit expander on top of these D new vertices, and adding these edges to the graph. The constraints $\varphi_{(v,w)}$ now become the constraints $\varphi_{(w,w')}$, and we place an equality constraint on each edge of the expander. Note that the size blowup after repeating this for every vertex is still linear in $|G| = |V| + |E|$.

The intuition for why we want to use an expander (instead of a cycle graph, for instance) comes from the expander mixing lemma on the problem set.

Lemma 4. *Suppose $G = (V, E)$ is a d-regular graph with second eigenvalue λ. Then for all $A, B \subset V$,*

$$\left| E(A,B) - \frac{d|A||B|}{|E|} \right| \leq \lambda d \sqrt{|A||B|},$$

where $E(A, B)$ is the number of edges with one endpoint in A and the other in B.

Informally, the lemma says that the number of edges between A and B is close to what we would expect in a random graph. Thus, any non-constant assignment for the new vertices w' will lead to a large fraction of edges being unsatisfied.

Overlaying an expander Let H be a $(|V|, d, \lambda)$-expander graph on vertex set V taken from an explicit family of expanders. To turn G into an expander, we simply take the union of its edges with the edges of H. It is an exercise in linear algebra to show that the result also has good expansion. We take the constraints on the new edges to always be satisfied, which will shrink the gap δ by a constant factor. But this is permissible, because we will use the second phase of the reduction to amplify the gap even further.

5.2 Preview of graph powering

We ran out of time to give the details of the graph powering step today, but here is a preview for the next lecture. Recall that we want to transform an expander G into a graph G' with a larger gap. The idea is to make the edges of G' correspond to endpoints of random walks in G. Thanks to the mixing properties of expanders, the unsatisfied edges of G will show up in a lot of random walks, and hence in the the edges of G'. We'll make this precise next time, including details for how assignments and constraints work in G'.

References

[1] 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.

[2] S. Arora and S. Safra, *Probabilistic checking of proofs: A new characterization of NP*, Journal of the ACM, 45(1):70-122, 1998.

[3] I. Dinur, *The PCP theorem by gap amplification*, Journal of the ACM, 51(3):12, 2007.

[4] D. Moshkovitz, *The tale of the PCP theorem*, XRDS: Crossroads, The ACM Magazine for Students, 18(3):23-26, 2012.