CS 880: Advanced Complexity Theory 03/24/06

Lecture 20: Proof of the PCP Theorem

Instructor: Dieter van Melkebeek Scribe: Mark Cowlishaw, Prem Mansukhani

Recall from last lecture that a constraint graph (CG) $G = [(V, E), \Sigma, \mathcal{C}]$ is a graph, along with an alphabet of possible assignments Σ, and a set of constraints defined as a function corresponding to assignments from Σ to each pair of vertices that form an edge in G: $\mathcal{C} = \{\mathcal{C}_e : \Sigma \times \Sigma \to \{0, 1\} | e \in E\}$ and UNSAT(G) is the minimum number of constraints violated under any assignment.

In this lecture we will discuss the satisfiability gap amplification of these constraint graphs using expanders. We will use this technique in our proof of the PCP theorem.

1 Overview of Satisfiability Gap Amplification

Theorem 1 (Satisfiability Gap Amplification). *There exists a polynomial-time transformation T from constraint graph G to constraint graph $T(G)$ and a constant $u > 0$ such that*

$$\begin{aligned} \text{UNSAT}(G) = 0 &\Rightarrow \text{UNSAT}(T(G)) = 0 \\ \text{UNSAT}(G) > 0 &\Rightarrow \text{UNSAT}(T(G)) \geq u \\ |T(G)| &= O(|G|) \end{aligned}$$

In order to prove this theorem, we will use repeated application of the following lemma:

Lemma 1 (Soundness Amplification). *There exists a polynomial-time transformation T from constraint graph G to constraint graph $T(G)$ and a constant $u > 0$ such that*

$$\begin{aligned} \text{UNSAT}(G) = 0 &\Rightarrow \text{UNSAT}(T(G)) = 0 \\ \text{UNSAT}(T(G)) &\geq 2 \cdot \min\{\text{UNSAT}(G), u\} \\ |T(G)| &= O(|G|) \end{aligned}$$

Recall that UNSAT$(G) > 0 \Rightarrow$ UNSAT$(G) \geq \frac{1}{n}$, so that a polynomial number of repetitions of soundness amplification will result in a constant value for UNSAT$(T(G))$.

1.1 Basic Idea - "Powering a Constraint Graph"

The basic idea of soundness amplification is analogous to randomness-efficient error reduction for probabilistic algorithms with one-sided error. The base case is like picking an edge uniformly at random and seeing if the associated constraint is satisfied and to amplify, we pick many edges independently at random and make a new constraint that is the conjunction of all the associated constraints. The problem with this is that the size of the graph grows too large too quickly, so, instead of picking edges uniformly and independently at random, we will instead take a random walk on on a constraint graph G'.

If the graph has constant degree, the size won't grow too much and, if it has good expansion properties, taking the edges from the random walk will be similar to picking edges uniformly at random. This process is called "powering a constraint graph".

1.2 Problems with "Powering"

There are several potential problems we must resolve to make the powering process work:

1. How can we treat constraint graph G as an expander?

   The idea is that we can transform G to a new constraint graph G' from some expander family (that is, with constant degree and spectral gap) with

$$\begin{aligned} |G'| &= O(|G|) \\ \text{UNSAT}(G) = 0 &\Rightarrow \text{UNSAT}(G') = 0 \\ \text{UNSAT}(G') &\geq \beta \cdot \text{UNSAT}(G) \end{aligned}$$

   for some universal constant β. Recall that Reingold's algorithm for $s - t$ connectivity in deterministic log-space used an elaborate method for transforming the original graph G into an expander with desirable properties. In this procedure, the transformation is much simpler.

2. Taking the sth power of a graph G produces edges that consist of all length s paths in G, but the resulting edges have only 2 vertices as endpoints, which are not enough to describe all s constraints. How can we maintain the constraint information in the "powered" graph?

   The idea is to expand the alphabet. If we allow each vertex to include the labels for the vertices reachable in $\frac{s}{2}$ steps, the resulting graph will capture all of the constraints in the original graph. Note that, since the vertices reachable in $\frac{s}{2}$ steps may overlap, it is possible that the resulting assignments to original vertices may conflict. In this case, we always reject (that is, the constraint graph is unsatisfiable). This process is depicted in Figure 1.

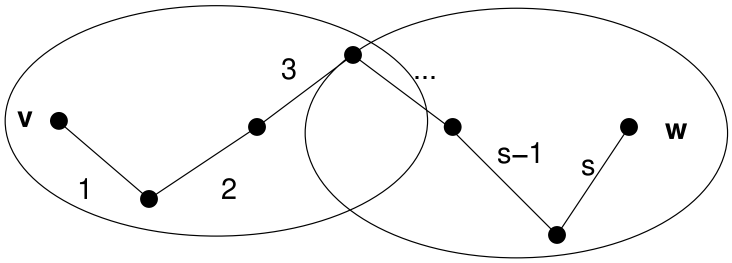

Figure 1: An edge in the sth Power of a Graph

Note that, for an s step random walk on an expander of degree d, this will expand the alphabet $|\Sigma'| = |\Sigma|^{d^{\frac{s}{2}+1}}$, however, for constant d and s, this is still a constant size. We would like this powered graph to amplify the soundness of G while not growing too much larger:

Lemma 2. *There exists a polynomial time transformation T and positive constant γ that transforms a constraint graph G to a constraint graph $T(G)$ with the following properties (for any $s \geq 1$):*

$$\begin{aligned} |T(G)| &= O(|G|) \\ \text{UNSAT}(G) = 0 &\Rightarrow \text{UNSAT}(T(G)) = 0 \\ \text{UNSAT}(T(G)) &\geq \gamma\sqrt{s} \cdot \min\left\{\frac{1}{\sqrt{s}}, \text{UNSAT}(G)\right\} \end{aligned}$$

How does this compare with taking s edges uniformly and independently at random from E? If we pick s edges uniformly and independently at random, we have:

$$(1 - \text{UNSAT}(T(G)) = (1 - \text{UNSAT}(G))^s$$

Where $(1 - \text{UNSAT}(G))$ is the fraction of satisfiable constraints in constraint graph G. A simple bound is given by

$$\text{UNSAT}(T(G)) \geq \gamma \cdot s \cdot \min\{\text{UNSAT}(G), \frac{1}{s}\}$$

As shown in Figure 2

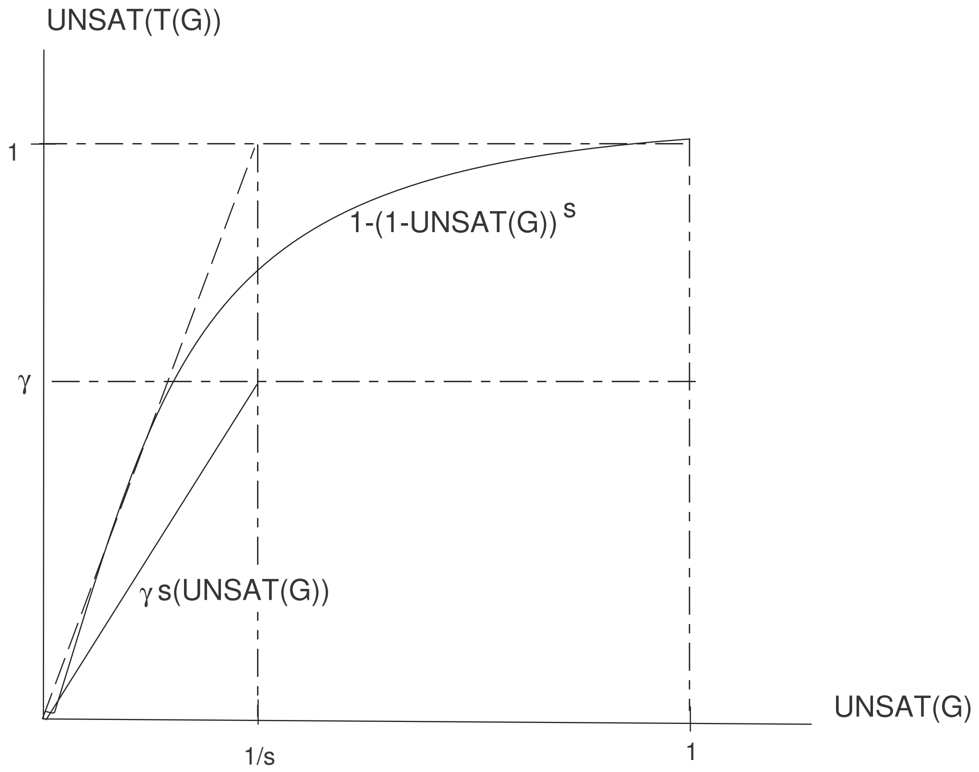

Figure 2: The Relationship Between $\text{UNSAT}(T(G))$ and $\text{UNSAT}(G)$

3. How can we reduce the size of the alphabet again?

   We can use proof composition (which we will discuss in a later lecture) to reduce the alphabet of $T(G)$ to some universal alphabet with constant size.

   Lemma 3. *There exists a polynomial-time transformation T, a function f and universal constant $\delta > 0$ such that T transforms a constraint graph G on alphabet Σ into a constraint*

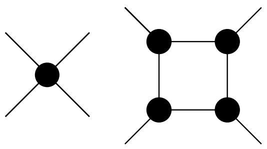

Figure 3: Ordinary Vertex Replacement

graph $T(G)$ on a universal alphabet Σ' where $|\Sigma'| = O(1)$ and:

$$|T(G)| = O(f(|\Sigma|) \cdot |G|)$$
$$\text{UNSAT}(G) = 0 \Rightarrow \text{UNSAT}(T(G)) = 0$$
$$\text{UNSAT}(T(G)) = \delta \cdot \text{UNSAT}(G)$$

Now we gave the details of step (1), the transformation from the original constraint graph to an expander.

2 Transforming a Constraint Graph into an Expander

The process of transforming the original constraint graph into a graph with good expansion properties uses three steps:

1. Make the degree constant
2. Make the spectral gap constant
3. Add self-loops

We perform the third step for technical reasons we will discuss in the next lecture. Through all of these steps, we must ensure that the satisfiability gap does not deteriorate too much (by at most a constant factor), and that the size of the graph does not expand by too much (again, by at most a constant factor).

2.1 Making the Degree Constant

To transform the constraint graph into a graph of constant degree, we will use a process similar to the replacement product. First, we replace every vertex v of degree d_v in $G = (V, E)$ with a graph H_v with d_v vertices and constant degree d as shown in Figure 3. When this process is complete, the graph will have constant degree $d + 1$. In order to maintain the satisfiability gap, we will add equality constraints to each edge in H_v which force the assignments to the vertices in the cloud of v to be identical. However, instead of using a simple cycle for H_v, we will take H_v from a family of expanders. Note that the size of the resulting graph $G' = (V', E')$ will not be too much larger, with $|V'| = 2|E|$ (since each cloud has d_v vertices) and $|E'| = \frac{(d+1)}{2} \cdot 2 \cdot |E| = (d + 1)|E|$.

In order to use G' in place of G, We must show that this process does not diminish the satisfiability gap by too much. To see this, consider an assignment to the vertices of G', $\sigma' : V' \to \Sigma'$, and a corresponding assignment $\sigma : V \to \Sigma$ where $\sigma(v)$ for $v \in V$ is the value σ' assigns to a plurality of vertices in the cloud of v.

Let B be the set of "bad" edges in G: $B = \{e = (u, v) \in E | C_e(\sigma(u), \sigma(v)) = 0\}$, let B' be the set of "bad" edges in G': $B' = \{e' = (u', v') \in E' | C'_{e'}(\sigma'(u'), \sigma'(v')) = 0\}$, and let D be the set of "dissident" vertices in G': $D = \{v' \in V' | v' \in \text{cloud}(v) \wedge \sigma'(v') \neq \sigma(v)\}$. Then $|B| \leq |B'| + |D|$.

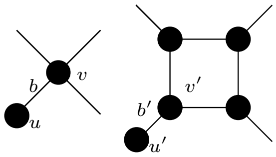

Figure 4: Violated Edge B

To see this, consider an edge $b \in B$ with endpoints (u, v), and the associated edge $b' \in V'$ with endpoints (u', v') as shown in Figure 4. Clearly, if $\sigma'(u') = \sigma(u)$ and $\sigma'(v') = \sigma(v)$ (that is, if both endpoints of b' have the plurality assignment from their cloud), then $b' \in B'$. If not, then either u' or v' are in D. Clearly, even if u' or v' are dissidents, it is still possible that $b' \in B'$, so this estimate of the size of B is an overestimate, but it is good enough for our purposes. For any $v \in D$, let $D_{v,a}$ with $a \in \Sigma'$ denote the set of vertices in the cloud of v with assignment a: $D_{v,a} = \{v' \in V' | v' \in \text{cloud}(v) \wedge \sigma'(v') = a\}$. Then, for any $a \neq \sigma(v)$:

$$|D_{v,a}| \leq \frac{d_v}{2} \Rightarrow |E(D_{v,a}, \overline{D_{v,a}})| \geq \phi \cdot |D_{v,a}|$$

Where ϕ is the edge expansion of the expander, and the first condition holds if a is not the plurality assignment to cloud(v). Since each edge in the expander H_v is associated with an equality constraint, any edge between dissident and non-dissident vertices will correspond to a violated constraint and thus the edge will be in $|B'|$:

$$\begin{aligned}|B'| \geq \frac{1}{2} \sum_{v, a \neq \sigma(v)} |E(D_{v,a}, \overline{D_{v,a}})| \geq \frac{\phi}{2} \sum_{v, a \neq \sigma(v)} |D_{v,a}| \\ = \frac{\phi}{2} \cdot |D|\end{aligned}$$

Therefore, rewriting the expression for $|B|$, we obtain:

$$\begin{aligned}|B| \leq |B'| + |D| \leq (1 + \frac{2}{\phi})|B'| \\ \text{UNSAT}(G) \leq \frac{|B|}{|E|} \leq (1 + \frac{2}{\phi})\frac{|B'|}{|E|} \\ = (d+1)(1 + \frac{2}{\phi})\frac{|B'|}{|E'|}\end{aligned}$$

Since $\frac{|B'|}{|E'|}$ is UNSAT(G′), this shows that the satisfiability gap of G',

$$\text{UNSAT}(G') = k \cdot \text{UNSAT}(G)$$

for some constant k.

2.2 Making the Spectral Gap Constant

To make the spectral gap constant, we superimpose an expander of degree $2d+1$, resulting in a spectral gap that is the average of the spectral gaps of the two graphs, reducing the overall spectral gap to a constant. If we let A, B, C be the normalized adjacency matrices of the new graph, the original graph, and the expander respectively, with $A = \frac{B+C}{2}$, then:

$$\begin{aligned} x^T A x &= \frac{1}{2}(x^T B x + x^T C x) \\ &\leq \frac{1}{2}(x^T x + \lambda_2(C) x^T x) \\ &\leq \frac{\lambda_2(C)+1}{2} \end{aligned}$$

resulting in a constant spectral gap. It is important that this step not decrease the size of the satisfiability gap by too much, nor increase the size of the graph by too much. Since we are adding $\frac{d \cdot |V|}{2}$ edges to a graph with $\frac{(d+1)\cdot|V|}{2}$ edges, the size increases by a constant factor. Additionally, since we are effectively adding no new constraints (i.e. the constraints for the expander edges will be trivially satisfiable), and since the total number of edges goes up by a constant factor, the satisfiability will be decreased by only a constant factor as well.

2.3 Adding Self-loops

Finally, for technical reasons we add self-loops at each vertex. Clearly, this will only increase the graph size by $|V|$ edges (a factor < 2), and will preserve the satisfiability of satisfiable constraint graphs, since the constraints associated with self-loops will be trivially satisfiable. Since we are only adding $|V|$ satisfiable constraints, the satisfiability gap will only decrease by a constant factor < 2.

3 Realizing the Soundness Amplification Lemma

In order to produce constraint graphs with a constant satisfiability gap, each iteration of this process must increase the satisfiability gap by a factor of 2, in accordance with Lemma 1. The first step (Turning the constraint graph into an expander) decreases the gap by a factor of a constant β, the second step (powering) increases the gap by $\gamma\sqrt{s}$ for constant γ, and the third step (decreasing the alphabet size) decreases the gap by a constant δ. Thus, after the process, the soundness of the resulting graph is $\text{UNSAT}(T(G)) = \beta \cdot \gamma \cdot \sqrt{s} \cdot \delta \cdot \text{UNSAT}(G)$. We must choose a high enough power s so that $\beta \cdot \gamma \cdot \sqrt{s} \cdot \delta \geq 2$, yielding $s \geq \left(\frac{2}{\beta \cdot \gamma \cdot \cdot \delta}\right)^2$.