PCPs and HDX - Lecture 8

Scriber: Renen Perlman

January 3, 2017

In this lecture we begin to prove the PCP theorem. We start with reviewing the formulation of the theorem, and giving a high-level approach of how we should understand it. Then, we give the outline of the proof, and show the first of its two main parts, the *gap amplification*. The second one, *alphabet reduction*, will be described in the following lecture.

1 The High-Level Approach

In previous lectures we've described the PCP theorem as follows:

Theorem 1.1. *There exists some fixed $\epsilon_0 > 0$ and a polynomial time algorithm that on input a 3COL instance $G = (V_G, E_G)$, outputs a graph $H = (V_H, E_H)$ such that:*

- *Completeness: If G is 3-colorable, then H is 3-colorable as well.*
- *Special Soundness: If G is* not *3-colorable, then for every coloring of H, at least ϵ_0 fraction of the edges of H are violated.*

We reformulate the theorem in the spirit of the course: Recall the following definitions - Let $H = (V_H, E_H)$ be some graph, and let $c : V_H \to \{1, 2, 3\}$ be some coloring of its vertices. We define the rejection probability of c in the graph H as

$$\mathrm{rej}_H(c) = \Pr_{(u,v)\in E}[c(u) = c(v)] \ .$$

Similarly, we define the rejection probability of H as

$$\mathrm{rej}(H) = \min_{c:V_H\to\{1,2,3\}} \mathrm{rej}_H(c) \ .$$

Note that if a graph $G = (V_G, E_G)$ is not 3-colorable then we can lower bound its rejection probability by $\mathrm{rej}(G) \geq 1/m$, where $m = |E_G|$. Therefore, we can reformulate the properties of the transformation from G to H as follows:

- If $\mathrm{rej}(G) = 0$, then $\mathrm{rej}(H) = 0$.
- If $\mathrm{rej}(G) \geq 1/m$, then $\mathrm{rej}(H) \geq \epsilon_0$.

Recall that we've defined the expansion of a graph by:

$$\gamma(H) = \min_{\substack{c:V_H\to\{1,2,3\} \\ c\notin \mathrm{SAT}(H)}} \frac{\mathrm{rej}_H(c)}{\mathrm{dist}(c, \mathrm{SAT}(H))} \ .$$

A stronger property that we could have required from H to satisfy is that $\gamma(H) \geq \gamma_0$. This would imply the previous formulation: if G is not 3-colorable, then $\mathrm{SAT}(H) = \phi$, so for every coloring $c : V_H \to \{1, 2, 3\}$ we would have $\mathrm{rej}_H(c) \geq \gamma_0 \cdot \mathrm{dist}(c, \mathrm{SAT}(H)) = \gamma_0 \cdot 1$, and so $\mathrm{rej}(H) \geq \gamma_0$.

It is very likely that with small modifications the current proof techniques give the stronger property, however, this has not been proved. This is mainly from a "historical reason", since the terminology of expansion of a constraint graph was introduced after the PCP theorem was proved, and this stronger property is not necessary for the proof.

Remark 1.1. *The conditions* $\begin{cases} \text{If } rej(G) = 0, \text{ then } rej(H) = 0 \\ \gamma(H) \geq \gamma_0 \end{cases}$ *could be strictly stronger than the original formulation of Theorem 1.1, since we require that* $\gamma(H) \geq \gamma_0$*, even if* G *is 3-colorable. It could be that* $rej(H) = 0$*, though* $\gamma(H) = o(1)$*.*

2 Outline of the Proof

The transformation $G \curvearrowright H$ is done by repeating an iterative step several times:

$$G =: G_0 \curvearrowright G_1 \curvearrowright \ldots \curvearrowright G_k =: H$$

The iterative step is such that $\mathrm{rej}(G_{i+1}) \geq 2 \cdot \mathrm{rej}(G_i)$ as long as $\mathrm{rej}(G_i) < \epsilon_0$, and if $\mathrm{rej}(G_i) = 0$, then $\mathrm{rej}(G_{i+1}) = 0$. Note that if G is not 3-colorable, then $\mathrm{rej}(G) \geq 1/m = \Omega(1/n^2)$, so we need $k = O(\log n)$ number of iterations.

The Iterative Step $G_i \curvearrowright G_{i+1}$

The iterative step consists of the following 3 basic steps:

Step 1 - Preprocessing: We construct $G_i \curvearrowright G'_i$ such that G'_i is d-regular for constant $d(=4)$. This transformation is similar to the one shown in Homework 1, but instead of replacing each vertex $v \in V$ by a cycle of length d_v (d_v is the degree of v), we replace it by a 3-regular expander graph on d_v vertices. This guarantees that to $\mathrm{rej}(G'_i) \geq \Omega(1) \cdot \mathrm{rej}(G_i)$. Note that from this step on, G_i is a general constraint graph, rather than a $3COL$ instance (since some of the constraints are $=$ and some are $\neq$).

Next, we construct $G'_i \curvearrowright G''_i$ by adding more edges to G''_i in order to make it an edge expander graph, and put trivial constraints on those edges, i.e. constraints that always accept. This reduces the rejection probability of G''_i by a constant factor. Note that in both transformations, if $\mathrm{rej}(G_i) = 0$, then $\mathrm{rej}(G'_i) = \mathrm{rej}(G''_i) = 0$. In all we let

$$\mathsf{Prep}(G_i) := G''_i.$$

Step 2 - Gap Amplification: We construct $G_i \curvearrowright \mathsf{Balls}(G_i) =: G'_i$, where Balls is some polynomial time algorithm, which we describe in the next section. The transformation is such that $\mathrm{rej}(G'_i) \geq \Omega(t) \cdot \mathrm{rej}(G_i)$, as long as $\mathrm{rej}(G_i) < O(1/t)$, for some parameter t. We choose t to be large enough to "cover" for the decrease in the rejection probability of the other steps, so that after the third step, the rejection probability of G_{i+1} would be $\mathrm{rej}(G_{i+1}) \geq 2 \cdot \mathrm{rej}(G_i)$ (as long as $\mathrm{rej}(G_i) < \epsilon_0$), and if $\mathrm{rej}(G_i) = 0$, then $\mathrm{rej}(G_{i+1}) = 0$.

The main drawback of this step is that it increases the alphabet of the constraint system. Specifically, $|\Sigma(G'_i)| = |\Sigma(G_i)|^{O(d^t)} = 3^{O(d^t)}$, so t has to be very small for the alphabet to remain

under control.

Step 3 - Alphabet Reduction: We construct $G_i \curvearrowright \Sigma\text{−}\mathsf{Reduce}(G_i) =: G'_i$ such that $|\Sigma(G'_i)| = O(1)(= 3)$. This step decreases the rejection probability by a constant factor, and preserves the property that $\text{rej}(G_i) = 0$. Finally, we set $G_{i+1} := G'_i$.

We see that we have to set large enough $t = O(1)$ such that $\text{rej}((\Sigma\text{−}\mathsf{Reduce} \circ \mathsf{Balls} \circ \mathsf{Prep})(G_i)) \geq 2\text{rej}(G_i)$. We remark that we have to construct G_{i+1} in such a way that $\left|V_{G_{i+1}}\right| \leq O(1) \cdot |V_{G_i}|$. This ensures that $|V_H| = |V_{G_k}| = 2^{O(k)} \cdot |V_{G_0}| = poly(n)$. Letting the size of the graph to increase by a larger factor in each iteration would lead to super polynomial running time for the whole transformation.

3 Gap Amplification Step

We now turn to describe the gap amplification step, moving from G to $\mathsf{Balls}(G)$. In this step we assume that the graph $G = (V, E)$ is a d-regular expander.

3.1 Intuition from error reduction

To develop some intuition, let us consider the following natural error reduction transformation for a constraint graph verifier.

Recall that a constraint system can be viewed as a verifier that is trying to verify whether the given assignment satisfies the constraints. The verifier chooses a random constraint, reads the the values of the variables from the proof, and accepts if and only if the constraint is satisfied. A satisfying assignment will always be accepted by the verifier. However, and unsatisfying assignment Sometimes causes the verifier to falsely accept. The soundness error is exactly the fraction of constraints that cause the verifier to say yes when it should say no.

What is an easy way to decrease the soundness error? Repeating the verifier multiple times will do the trick. Think of a new verifier chooses t constraints independently at random. A satisfying assignment is still accepted with probability 1. An assignment that caused a soundness error of α, now causes a soundness error of α^t. Indeed, let E be the total set of constraints, and let $F \subset E$ be the set of constraints that are violated by the given assignment. Denote $1 - \alpha = \epsilon = \frac{|F|}{|E|}$. If the original rejection probability was $\epsilon = 1 - \alpha$, then now it is $1 - (1 - \epsilon)^t \approx t\epsilon$, a t-fold increase.

Error reduction can be made "randomness efficient" by selecting the t constraints not completely independently, but rather using an efficient sampler. In particular, it turns out, that the t constraints can be selected as the edges on a length t random walk in expander graph. This is not hard to see, and is exactly like one proves error reduction for randomized algorithms with one-sided error.

The main issue now is moving from a verifier that reads $\approx t$ proof locations back to a coloring problem, in which each constraint reads only a constant number of bits. We can try to recurse: use a PCP to encode the value of each t-tuple of vertices. Since we are now encoding only a small number of bits, we can afford to use very inefficient PCPs such as the quadratic functions PCP. Now we reach the crux of the problem: *agreement*. We must ensure that the encodings of different local views agree with each other. We must not allow different paths that hit the same vertex to assign it different colors.

It turns out that looking at paths does not work[1], but looking at balls does. In the proof we essentially show that if a typical pair of balls agree on their intersection then they must be globally consistent with some underlying coloring.

3.2 The balls construction

For every vertex $v \in V$, we define the ball of radius t centered at v by:

$$B(v,t) = \{u \in V \mid \mathrm{dist}_G(v,u) \leq t\} \ .$$

Note that since G is d regular, then:

$$|B(v,t)| \leq 1 + d + d \cdot (d-1) + d \cdot (d-1)^2 + \ldots + d \cdot (d-1)^{t-1} =: D_t = O(d^t) \ .$$

We define $\mathsf{Balls}(G) := G' = (V, E', C')$. The vertex set is the same as in G. We set $\Sigma' = \Sigma^{D_t}$, and we think of an assignment for the vertices of G' as a collection of local ball-colorings $a_v : B_v \to \Sigma$. Even before specifying which pairs of vertices have a constraint between them in G', we can already describe the constraints:

Let $v, v' \in V$ be two vertices, and let $a_v, a_{v'}$ be their two assignments. $a_v, a_{v'}$ satisfy the constraint if and only if:

1. Agreement: For every common vertex $u \in B(v,t) \cap B(v',t)$, we require that the two assignments agree on u, namely $a_v(u) = a_{v'}(u)$. And,
2. Old constraints: For any pair $(u,u') \in E$ such that $u, u' \in B(v,t) \cup B(v',t)$, we require the assignments for u and u' to satisfy the constraints on (u,u') (those assignments are the same, both under a_v and $a_{v'}$, by the previous item).

We still have to define the distribution over pairs v, v' for our new constraint graph. Already we can see that this definition of the constraints guarantees completeness: a satisfying assignment for G easily translates to a satisfying assignment for G', no matter what distribution on pairs of v, v' we choose.

For soundness however, the choice of distribution over v, v' is important. We need to ensure that the intersection of two balls wouldn't be too small, nor too large. Roughly speaking, if B_v and $B_{v'}$ have small intersection, then an assignment could easily satisfy only constraints in the intersection, while violating many edges outside of it. If the intersection is too large, then intuitively the vertices v, v' are too correlated so their assignments may be tailored to the constraints in them without caring about other constraints involving the same vertices.

The constraints involving a vertex $v \in V'$, are described through a lazy-random-walk distribution $LRW(v, p = 1/t)$ defined below. We connect v to $v' \leftarrow LRW(v, p = 1/t)$ the endpoint of this random walk with weight proportional to its probability of occuring.
$LRW(v,p)$:

1. Set $v' \leftarrow v$.
2. Toss a p-biased coin, having probability p to output 1.
   (a) If we get 0, output the vertex v'.
   (b) Otherwise, choose uniformly at random v'' a neighbor of v'. Set $v' \leftarrow v''$, and go to step 2.

[1] To be precise: I do not know that it works nor that it does not work. It would be very nice to resolve this question.

Two (unimportant) technicalities arise. First, our definition gives a weighted constraint graph and not an unweighted one. This can be easily corrected by duplicating constraints, so we can safely ignore this point. Second, the random walk may, in theory, never halt. We can change the distribution to halt after $100 \cdot t$ steps. This has a negligible effect on the weights so we can safely ignore this as well.

We sketch an analysis of the soundness of this construction. In what follows, we refer to an assignment for the original constraint graph, G, as a coloring to distinguish it from an assignment to the balls. Let us first consider a simplified case, where the assignment to the balls, i.e. all of the local functions $\{a_v : B_v \to \Sigma\}_v$, come from restricting a single global coloring $c : V \to \{1, 2, 3\}$, i.e. $a_v = c|_{B_v}$. Namely, set $a_v(u) := c(u)$ for every $u \in B(v, t)$. In this case it is easy to show that the gap gets amplified. Indeed, a uniformly random ball in this construction, contains a uniformly random path of length t in the graph G. We have seen in the earlier section about error reduction that approximately $t \cdot \epsilon$ fraction of paths must see violated constraints. This implies that the new rejection probability is at least t times what it was before.

The general case in the collection of local assignments is not guaranteed to come from one global coloring. here, implicitly, we need to prove agreement expansion: namely, that the fact that many of the agreement constraints are satisfied imply that many of the local assignments agree with some global coloring. We define this global coloring by a plurality of vote, and then proceed in two steps

- Prove agreement: many of the local assignments must agree with the global coloring that we defined through majority
- Imitate the error reduction: show that the gap increased t-fold using similar arguments to the case of the error reduction. This is more tricky because it needs to be coupled with the previous item.