Lecture 8: Composition

Lecturer: *Dana Moshkovitz* Scribe: *Andrey Grinshpun and Dana Moshkovitz*

In this lecture we finally prove the PCP Theorem:

$$NP \subseteq PCP_{1,0.9999}[O(\log(n)), O(1)].$$

We do that using the PCP Theorem with poly-logarithmic number of queries we already established, and applying it in a recursive way.

1 Recursive Structure

Where do we see a recursive structure in PCP Theorems? Using a PCP we verify that a predicate ϕ is satisfied by some $x \in \{0,1\}^n$. This is done by adding a proof $\pi \in \Sigma^m$, looking at a few locations in π, and checking that the locations satisfy some other predicate φ_w (the locations and the predicate depend on the randomness w of the verifier). In other words, we reduced the problem of verifying a predicate on n bits to the problem of verifying other predicates on much less than n bits. This is a recursive structure!

Given this observation, one comes up with the following plan: if we have a PCP that makes relatively many queries (e.g., $poly \log n$ queries), and we want a PCP that makes much less queries (e.g., $poly \log(poly \log n)$ queries), we can verify the φ_w's using a PCP, rather than directly. The purpose of this lecture is to make this plan work.

The obstacle is that a PCP, as we defined so far, would verify that φ_w is satisfiable (which it always is), rather than that the answers that the prover would have given satisfy φ_w. How to define PCPs that support the latter is not at all clear. It took almost two decades of research to come up with the definitions we present next.

2 Definitions

The PCP definitions that we give are stronger than our usual definition of PCP. Luckily, the techniques we developed can be adapted to the stronger objects.

2.1 Locally Decode or Reject

The definition of PCP asks the proof π to certify the *existence* of an $x \in \{0,1\}^n$ that satisfies a predicate ϕ. A natural approach, which is indeed the approach we have been taking, is to make π an *encoding* of x, and argue that any π that is accepted with good probability *corresponds* to an x that satisfies ϕ.

In the next definition we are interested in an encoding of x from which the individual bits of x (satisfying ϕ) can be recovered. The verifier is given an $i \in [n]$ and should output x_i. The definition is adapted to the PCP setting, and makes the following points:

- On input a correct encoding π of x that satisfies ϕ, the verifier should decode x_i for every $i \in [n]$ it gets as input.

- It is possible that there is no x satisfying ϕ that corresponds to π, or that there is, but x_i^2 cannot be recovered by making few queries. The verifier may reject in such cases.
- For most $i \in [n]$ the verifier should work as intended (i.e., either decode x_i or reject), but there can be a small fraction of i's on which it does not.

Formally, the definition is as follows:

Definition 1 (Locally Decode or Reject Codes)**.** *Let $\Phi \subseteq \Sigma^n$. We say that $E : \Sigma^n \to \Sigma^m$ is a locally decode or reject code (LDRC), if there is a verifier/decoder that given access to $y \in \Sigma^m$ and an $i \in [n]$ makes q queries to y and either outputs $b \in \Sigma$ or rejects:*

- *If $y = E(x)$ for $x \in \Phi$, then the verifier/decoder always outputs x_i.*
- *If the verifier/decoder rejects with probability at most ε over the choice of $i \in [n]$ and the internal randomness, then there is $x \in \Phi$, such that, with probability at least $1 - \varepsilon'$, the verifier either returns x_i or rejects.*

The algebraic construction that we presented can be adapted to yield an LDRC. We leave it as a homework exercise.

2.2 Robust PCP

Suppose that the verifier picks randomness and queries the proof. We say that the answers to the queries are α-close to satisfying the verifier, if there are answers to the queries that satisfy the verifier and are α-close (as a string) to the answers seen.

The next definition asks for PCPs where in the soundness case, on many of the choices of queries, not only the answers do not satisfy the verifier – they are not even close to satisfying the verifier:

Definition 2 (Robust PCP)**.** *We say that a PCP is* α-robust*, if in the soundness condition, for every proof π, with probability at most s, the answers to the verifier's queries are α-close to satisfying.*

Every robust PCP is also a PCP, but a PCP is not necessarily (the stronger) robust PCP. Luckily, the algebraic construction can be adapted to give a robust PCP for a small $\alpha = poly(m)(d/|\mathbb{F}|)^{\Omega(1)}$. Moreover, any PCP can be converted to a robust PCP with large α, bounded away from 1. These facts are left as homework.

Remark 2.1 (Projection games)**.** *Robust PCPs with low α, s are ubiquitous in hardness of approximation, and we will revisit them later in the course. They are equivalent to PCPs with low error and two queries that have the projection property, i.e., the answer to the first query determines at most one answer for the second query that would make the verifier accept.*

- *To convert from a two query PCP with projection to a robust PCP: view the first query as choosing the queries, and the queries as all of the possible second queries.*
- *To convert from a robust PCP to a two query PCP with projection: add a proof symbol per randomness string. The symbol is supposed to be the concatenation of the answers to all the queries made for this randomness string.*

3 Composition

Now that we have the definitions we need, we turn to describe composition. We start by describing a composition operation that does not preserve robustness, and then proceed to present a composition that does preserve robustness. As we discussed in Remark 2.1, the latter yields a two query PCP with projection that is important for hardness of approximation.

The latter composition, however, works with a PCP with small degree (the degree of a PCP is the maximal number of randomness strings that may prompt the verifier to query a certain proof location). We follow this section by describing how to reduce the degree.

3.1 Composition that does not preserve robustness

The analogy of code concatenation is a good one to keep in mind: we have an outer code with a large alphabet, and an inner code with a smaller alphabet. (The inner code is applied only on small messages, and so even a construction with the same message-length to alphabet size relation as the outer code can give a smaller alphabet size). We replace the alphabet symbols of the outer code with encodings of them via the inner code to achieve a construction with the same alphabet size as the inner code.

In the context of PCP, there is an outer proof with a large number of queries and an inner proof with a small number of queries. *We ask that the outer PCP be robust, and that the inner PCP be LDRC.* For each block of outer symbols the outer verifier might query, the composed proof has an encoding of the block via the inner LDRC. This allows the composed verifier to inherit its number of queries from the inner LDRC.

More precisely, let us denote the outer and inner verifiers by V_1, V_2, respectively. We denote the composed verifier by V. The proof for V is the proof for V_1 along with, for each randomness w, a proof for V_2 that the q_1 queries V_1 would have made on randomness w satisfy φ_w, the test V_1 makes on randomness w. Note that the inner PCP construction is used on an input with $q_1 \cdot \log(|\Sigma_1|)$ bits, where q_1 is the number of queries of the outer PCP and Σ_1 is the alphabet there.

The verifier V is as follows:

1. Pick randomness w_1 for V_1 and randomness w_2 for V_2.
2. Apply V_2 to verify the q_1 queries V_1 would have made indeed satisfy φ_w. Use V_2 to decode a uniformly random symbol of the q_1 outer symbols. Reject if V_2 rejects.
3. Check that V_2's decoding matches the copy of the symbol in the proof for V_1. Accept if it does; reject otherwise.

We now consider how well this verifier performs: note that previously input sizes were always implicit and so we just wrote, for example, q, but now they no longer are so we write $q(n)$.

Define $n' := q_1(n) \cdot \log(|\Sigma_1(n)|)$.

$$q(n) = 1 + q_2(n')$$

$$r(n) = r_1(n) + r_2(n')$$

$$c(n) \geq 1 - ((1 - c_1(n)) + (1 - c_2(n'))) = c_1(n) + c_2(n') - 1$$

$$s \leq s_1(n) + s_2(n') + \alpha_1(n) \quad (\alpha_1 \text{ is the robustness of } V_1)$$

$$|\Sigma(n)| = \max(|\Sigma_1(n)|, |\Sigma_2(n')|)$$

The analysis of the soundness is left as a homework exercise. The overall idea of the argument is to show that the copy of the proof to V_1 provided in the composed version should cause V_1 to accept reasonably often.

3.2 Composition that preserves robustness

The construction of the previous section does not preserve both soundness and robustness (i.e., low s and low α), even when both the outer verifier and the inner verifier are sound and robust. This is because of the two different tests the verifier makes: one is V_2's test, and the other is the consistency test (steps 2 and 3 above). It suffice to change at most half of the symbols that V queries to make V accept.

In this section we adapt the composition to preserve both soundness and robustness, paving the way to a two-query PCP with projection and low error. The idea is to get rid of the outer proof altogether, and keep only the $2^{r_1(n)}$ inner proofs. To check consistency between the inner proofs, we compare decodings of outer symbols from many inner proofs.

For the construction we make the additional assumption that V_1 is *uniform*, i.e., queries each of its eproof locations exactly the same number of times, and, moreover, its *degree*, i.e., the number of times each proof location is queried, is small. In the next section we show how to adapt any verifier to a uniform verifier with degree $\approx 1/s$.

The verifier V is as follows (below m denotes the length of V_1's proof):

1. Pick uniformly at random a location $i \in m$. Let $w_1, \ldots, w_D \in \{0,1\}^{r_1(n)}$ be all the randomness strings that make V_1 query i.
2. For each $1 \leq j \leq D$, apply V_2 to verify the $q_1(n)$ queries V_1 would have made on randomness w_j indeed satisfy φ_{w_j}. Use V_2 to decode the value of the i'th location. Reject if V_2 rejects.
3. Check that the D decodings of the i'th outer location are all consistent. Reject otherwise.

Here is how the composition operation effects the parameters:

$$q(n) = D \cdot q_2(n')$$

$$r(n) = \log m + r_2(n')$$

$$\Sigma(n) = \Sigma_2(n')$$

Composition preserves perfect completeness, but may increase the completeness error if it existed in the first place. The soundness error may slightly increase. The analysis is left as an exercise.

4 Degree Reduction

Recall that the degree of a verifier is the maximal number of randomness strings on which the verifier queries a single location of the proof. In general, the degree can be very large. In the algebraic construction, $D = n^{\Omega(1)}$. In this section we show how to reduce the degree of any verifier to $(1/s)^{\Omega(1)}$, with a nominal damage to the other parameters (randomness, soundness, robustness, completeness, alphabet size). Recall that from the homework we know that $s \geq 1/D$, and so the transformation is essentially optimal.

Given a verifier V of degree D, we will construct a new verifier V^* of degree $d = poly\frac{1}{s}$. The intended proof for V^* is similar to the proof for V, except that each location $i \in [m]$ in V's proof has D copies in V^*'s proof, which we index by $\langle i, j\rangle$ for $j \in [D]$. The idea is to distribute the queries to i among the different copies $\langle i, \cdot\rangle$, so no copy is queried on more than d randomness strings. By carefully choosing how to distribute the queries, we can ensure consistency between the values given to different copies. The careful choice is done using a Ramanujan expander $H = ([D], E_H)$ of degree d.

The verifier V^* is as follows:

1. Pick randomness $w \in \{0,1\}^r$ for V. Let $i_1, \ldots, i_q \in [m]$ be the locations that V would have queried on randomness w. For every $l \in [q]$, assume that w is the t_l'th, $t_l \in [D]$, randomness string (in lexicographic order) that results in i_l being queried.

2. For every $k \in [d]$ do the following: For every $l \in [q]$, let $j_l \in [D]$ be the k'th neighbor of t_l in H. Query locations $\langle i_1, j_1\rangle, \ldots, \langle i_q, j_q\rangle$ in the proof, and perform V's test on them. Reject if V rejects.

3. Reject if there is $l \in [q]$ such that the d queried copies $\langle i_l, \cdot\rangle$ not all have the same symbol in Σ. Otherwise, accept.

The degree of V^* is d. The parameters of V^* in terms of V's parameters are as follows:

$$c^* = c$$
$$s^* \leq \sqrt{2\max\{\alpha, s\}} + \frac{1}{\sqrt[4]{d}}$$
$$\alpha^* \leq \sqrt{2\max\{\alpha, s\}} + \frac{1}{\sqrt[4]{d}}$$
$$r^* = r$$
$$q^* = q \cdot d$$
$$\Sigma^* = \Sigma$$

4.1 Proof of soundness and robustness

We end this section by proving the soundness and robustness of the degree reduction operation. The claim is proven contrapositively. Assume that there is a prover P^* that makes V^* be at least α^*-close to accept with probability at least s^*. Hence, the average closeness of V^* to being satisfied is at least $\alpha^* s^*$. We show how to construct a prover P so that the average closeness of V to being satisfied is at least $\alpha^* s^* - \frac{1}{\sqrt{d}}$. By Markov inequality, with probability at least $\alpha^* s^*/2 - \frac{1}{2\sqrt{d}}$, the verifier V is at least $(\alpha^* s^*/2 - \frac{1}{2\sqrt{d}})$-close to being satisfied.

The prover P that has to decide what symbol to put in location i, picks at random $j \in [D]$ and decides on the value that P^* has in location $\langle i, j\rangle$.

For every randomness $w \in \{0,1\}^r$ on which V^* is α^*-close to being satisfied, let $P^*_w \in \Sigma^{q \cdot d}$ denote the satisfying assignment. By the definition of V^*, for each one of q symbols corresponding to V's test on randomness w, the symbols in P^*_w corresponding to the d copies of the symbol are all the same. For $w \in \{0,1\}^r$ on which V^* is not α^*-close to being satisfied, let P^*_w be an arbitrary string in $\Sigma^{q \cdot d}$, where the locations corresponding to different copies of the same location receive the same symbol from Σ.

We make the following definitions for $i \in [m]$ and $\sigma \in \Sigma$:

- $X_{i,\sigma}$ consists of the $k \in [D]$ such that the k'th randomness $w \in \{0,1\}^r$ that results in a query to i has P^*_w assign location i the symbol σ.
- $Y_{i,\sigma}$ consists of the $j \in [D]$ for which the symbol in location $\langle i, j \rangle$ is σ.

By regularity, we can express the average over all $w \in \{0,1\}^r$ of the closeness of V^* to being satisfied on randomness w by:

$$\frac{1}{mDd} \sum_{i \in [m]} \sum_{\sigma \in \Sigma} E(X_{i,\sigma}, Y_{i,\sigma})$$

By regularity, we can express the average over all $w \in \{0,1\}^r$ of the expected closeness of V to being satisfied on randomness w by:

$$\frac{1}{mDd} \sum_{i \in [m]} \sum_{\sigma \in \Sigma} |X_{i,\sigma}| \, |Y_{i,\sigma}|$$

Using the expander mixing lemma,

$$\begin{aligned}
\sum_{i \in [m]} \sum_{\sigma \in \Sigma} E(X_{i,\sigma}, Y_{i,\sigma}) &\leq \sum_{i \in [m]} \sum_{\sigma \in \Sigma} \left(|X_{i,\sigma}| \, |Y_{i,\sigma}| + \sqrt{d} \sqrt{|X_{i,\sigma}|} \sqrt{|Y_{i,\sigma}|} \right) \\
&= \sum_{i \in [m]} \sum_{\sigma \in \Sigma} |X_{i,\sigma}| \, |Y_{i,\sigma}| + \sqrt{d} \sum_{i \in [m]} \sum_{\sigma \in \Sigma} \sqrt{|X_{i,\sigma}||Y_{i,\sigma}|} \\
\text{(Cauchy-Schwarz)} \quad &\leq \sum_{i \in [m]} \sum_{\sigma \in \Sigma} |X_{i,\sigma}| \, |Y_{i,\sigma}| + \sqrt{d} \sum_{i \in [m]} \sqrt{\sum_{\sigma \in \Sigma} |X_{i,\sigma}|} \sqrt{\sum_{\sigma \in \Sigma} |Y_{i,\sigma}|} \\
&= \sum_{i \in [m]} \sum_{\sigma \in \Sigma} |X_{i,\sigma}| \, |Y_{i,\sigma}| + mD\sqrt{d}
\end{aligned}$$

5 The PCP Theorem

We can now prove the PCP theorem using composition. We start with a robust PCP verifier V_1 and a locally decode-or-reject verifier V_2. The verifiers can be constructed using the algebraic tools we designed in previous lectures. Together with degree reduction, we achieve arbitrarily low constant soundness and robustness error parameters, and other parameters as follows:

$$q(n) = poly \log n$$

$$r(n) = O(\log n)$$

$$c(n) = 1$$

$$|\Sigma(n)| = O(1)$$

$$D(n) = O(1)$$

Using composition and degree reduction, we preserve low constant soundness and robustness parameters, with:

$$q'(n) = poly(\log \log n)$$

$$r'(n) = O(\log n)$$

$$c'(n) = 1$$
$$\left|\Sigma'(n)\right| = O(1)$$
$$D'(n) = O(1)$$

We can now perform a final composition of this robust PCP verifier V'. The inner construction is a locally decode-or-reject verifier V_{Had} based on linearity testing (see homework). V_{Had} has arbitrary low constant soundness and robustness error parameters. Its other parameters are:

$$q_{Had}(n) = O(1)$$
$$r_{Had}(n) = polyn$$
$$c_{Had}(n) = 1$$
$$|\Sigma_{Had}(n)| = O(1)$$

The final parameters we get from composition are low constant soundness and robustness error with:

$$q_{fin}(n) = O(1)$$
$$r_{fin}(n) = O(\log n)$$
$$c_{fin}(n) = 1$$
$$|\Sigma_{fin}(n)| = O(1)$$