Lecture 21&22

In this lecture, we will prove that $\mathbf{NP} = \mathbf{PCP}(O(\log n), poly \log n)$. Since one direction is easy to see, we will prove below that $\mathbf{NP} \subseteq \mathbf{PCP}(O(\log n), poly \log n)$.

1 Overview

It suffices to prove that the **NP**-complete problem 3SAT $\in \mathbf{PCP}(O(\log n), poly \log n)$. There are three major components in proving this result:

- The verifier expects the proof π to consist of a low-degree polynomial $P : \mathbb{F}^l \to \mathbb{F}$ and P is represented by its value at each of the $|\mathbb{F}|^l$ points. The verification procedure picks a random constraint on the polynomial P among $poly(n)$ constraints. If the 3SAT-formula ϕ is satisfiable, then there exists a low-degree polynomial P that satisfies all the constraints; on the other hand, if ϕ is not satisfiable, then less than half of the constraints are satisfied.

- Above we require that the proof consists of a low-degree polynomial. However, there is no guarantee that the prover will oblige by conforming to this restriction. Thus, we need an efficient way to enforce this low-degree restriction on the proof, and this is given by low-degree tests. Ideally, we would like to have a test that, given access to an oracle $f : \mathbb{F}^l \to \mathbb{F}$, outputs yes with probability 1 if f is a degree $\leq d$ polynomial and outputs no with high probability otherwise. This, however, is not possible, since we can have an f which disagrees with a degree d polynomial at one point and agrees with the polynomial everywhere else, and thus passes any test that only queries f at poly(m, d) places with high probability. We thus relax the soundness condition and only require the low-degree test to reject with high probability if f disagrees with any degree d polynomial on more than δ-fraction of the points with an appropriately chosen δ.

- The gap between the completeness and soundness of the low-degree test above still leaves us with a problematic situation: assume the prover provides a function $f : \mathbb{F}^l \to \mathbb{F}$ that agrees with a degree d polynomial p at more than $1 - \delta$-fraction of the points, and it passes the low-degree test. Then, we need to check a random constraint on p, but how can we get access to the polynomial p with oracle access to f? The polynomial reconstruction algorithm computes $p(x)$ for any given $x \in \mathbb{F}^l$ with high probability given oracle access to f.

Armed with the above three major components, we can now describe our **PCP** verifier. The verifier is given a 3SAT instance ϕ. The prover then supplies an oracle for a function $f : \mathbb{F}^l \to \mathbb{F}$. The verification procedure proceeds as follows:

1. First run the low-degree tests on f. Reject if the test rejects.

2. Pick a random constraint to see if it satisfies on f. Use the polynomial reconstruction algorithm to do the checking. Reject if the constraint does not pass.

3. Accept otherwise.

In the verification procedure, we make sure that we use $O(\log n)$ random bits and query *poly* $\log n$ bits from the oracle. In the following, we will provide details on each of the three components.

2 Arithmetization of 3SAT

We begin by describing how to arithmetize 3SAT. An instance ϕ of 3SAT consists of n variables and t clauses where each clause is of the form $\{x_{i_1} = b_1 \vee x_{i_2} = b_2 \vee x_{i_3} = b_3\}$ where each $b_j \in \{0,1\}$. In this way, we can view ϕ as an indicator function $\phi : \{1,2,\dots,n\}^3 \times \{0,1\}^3 \to \{0,1\}$ where $\phi(i_1,i_2,i_3,b_1,b_2,b_3) = 1$ if there is a clause of the form $\{x_{i_1} = b_1 \vee x_{i_2} = b_2 \vee x_{i_3} = b_3\}$ in the instance ϕ.

To arithmetize ϕ, we pick h and m, where $h = \log n$ and $m = \log n / \log\log n$ so that $h^m = n$. Now, set $H = \{1,2,\dots,h\}$, denote h_i the ith element in H and identify $\{1,2,\dots,n\}$ with H^m in some canonical way (e.g. representation in base h). Extending $b_j (\in \{0,1\})$ to H, the instance ϕ can be viewed as an m'-variate function $\phi : H^{m'} \to \{0,1\}$ where $m' = 3m+3$. (Whenever $b_j \notin \{0,1\}$, ϕ is evaluated to 0.) Similarly, an assignment $A : \{1,2,\dots,n\} \to \{0,1\}$ can be viewed as a function $A : H^m \to \{0,1\}$.

In this language, 3SAT can be restated as follows: ϕ is satisfiable if and only if there exists an assignment $A : H^m \to \{0,1\}$ such that for all $i_1, i_2, i_3 \in H^m$ and for all $b_1, b_2, b_3 \in H$,

$$\phi(i_1,i_2,i_3,b_1,b_2,b_3)(A(i_1)-b_1)(A(i_2)-b_2)(A(i_3)-b_3) = 0 \tag{1}$$

Now we introduce the notion of polynomial codes: we fix a field $\mathbb{F}$of size $\log^3 n$ that contains H and extend the function $\phi : H^{m'} \to \{0,1\}$ to a low-degree polynomial $\hat{\phi} : \mathbb{F}^{m'} \to \mathbb{F}$ such that $\phi(x) = \hat{\phi}(x)$ for all $x \in H^{m'}$. The following Lemma says that such a low-degree polynomial exists:

Lemma 1
For every function $f : H^m \to \mathbb{F}$ there is a polynomial $\hat{f} : F^m \to \mathbb{F}$ such that $f(x) = \hat{f}(x)$ for all $x \in H^m$. Every variable in $\hat{f}$ has degree at most h, and $\hat{f}$ is a degree hm polynomial. $\hat{f}$ is called a low-degree extension of f. (The extension might not be unique.)

Similarly, we extend the assignment function $A : H^m \to \{0,1\}$ to a low-degree polynomial $\hat{A} : \mathbb{F}^m \to \mathbb{F}$ such that $A(x) = \hat{A}(x)$ for all $x \in H^m$. Now ϕ is satisfiable if and only if there exists an assignment polynomial $\hat{A} : F^m \to \mathbb{F}$ such that for all $i_1, i_2, i_3 \in H^m$ and for all $b_1, b_2, b_3 \in H$,

$$\hat{\phi}(i_1,i_2,i_3,b_1,b_2,b_3)(\hat{A}(i_1)-b_1)(\hat{A}(i_2)-b_2)(\hat{A}(i_3)-b_3) = 0 \tag{2}$$

Denote the polynomial in the above equation to be $\hat{P}_1$. Since $\hat{\phi}$ and $\hat{A}$ have degree at most h in each variable, $\hat{P}_1$ is an m'-variate polynomial of degree at most $2h$ in each variable and thus has (total) degree at most $2hm'$.

2.1 First Try

Now let us assume the proof π contains not only the assignment polynomial P_0 claimed to be $\hat{A}$, but also a polynomial P_1 that is supposed to be $\hat{P}_1$. The verifier needs to check the following constraints:

- (C0). $P_1(x) = \hat{P}_1(x)$ for all $x \in \mathbb{F}^{m'}$
  (The verifier can efficiently check constraints C0 since he can efficiently compute $\hat{\phi}(x)$ and thus $\hat{P}_1(x)$ once given the assignment polynomial P_0.)
- (C00). $P_1(x) = 0$ for all $x \in H^{m'}$

Constraints C0 satisfy the soundness requirement of **PCP**, that is, if $P_1 \neq \hat{P}_1$, then $\mathbf{Pr}_x[P_1(x) = \hat{P}_1(x)] \leq 1/2$. This follows from the following Schwartz-Zippel Lemma:

Lemma 2 (Schwartz-Zippel)
Two distinct polynomials $\mathbb{F}^m \to \mathbb{F}$ of (total) degree d disagree on at least $1 - \frac{d}{|\mathbb{F}|}$ fraction of points in $\mathbb{F}^m$.

However, constraints C00 do not satisfy the soundness requirement of **PCP**, because it's possible for P_1 to be zero on all but one point of $H^{m'}$ and thus to pass the test with high probability if we evaluate P_1 at a random point in $H^{m'}$.

2.2 Second Try

We will construct a sequence of polynomials which are zero on a larger and larger fraction of points and eventually the last polynomial would be zero on all points and thus can be efficiently tested. Specifically we will define a sequence of low-degree polynomials $P_2, P_3, \ldots, P_{m'+1}$ such that $P_1 = 0$ over $H^{m'}$ if and only if $P_2 = 0$ over $\mathbb{F} \times H^{m'-1}$, and in general for $1 \leq i \leq m'+1$, $P_i = 0$ over $\mathbb{F}^{i-1} \times H^{m'-i+1}$ if and only if $P_{i+1} = 0$ over $\mathbb{F}^i \times H^{m'-i}$. Hence, $P_{m'+1}$ will be identically zero on $F^{m'}$ if and only if P_1 is identically zero on $H^{m'}$. Each of these constraints will satisfy the soundness requirement of **PCP**.

To start with, let us first see how to construct P_2 from P_1. Given $P_1 : \mathbb{F}^{m'} \to \mathbb{F}$, define $P_2 : \mathbb{F}^{m'} \to \mathbb{F}$ to be:

$$P_2(y_1, x_2, x_3, \ldots, x_{m'}) = \sum_{j=1}^{|H|} P_1(h_j, x_2, x_3, \ldots, x_{m'}) y_1^j. \tag{3}$$

Clearly, if P_1 is identically zero on $H^{m'}$, then P_2 is identically zero on $\mathbb{F} \times H^{m'-1}$. Conversely, if $P_1(z_1, \ldots, z_{m'}) \neq 0$ for some $z \in H^{m'}$, then $P_2(y_1, z_2, \ldots, z_{m'})$ is some non-zero univariate polynomial of degree at most $|H|$ in y_1 and so is non-zero on at least $|\mathbb{F}| - |H|$ points. Thus, the converse also holds.

In constructing the sequence of polynomials, we apply the above transformation, once in each variable. In general, in transforming P_{i-1} to P_i, we change variable x_{i-1} to y_{i-1} as described above for the case $i = 2$, namely, for $2 \leq i \leq m'+1$, define

$$P_i(y_1, \ldots, y_{i-1}, x_i, \ldots, x_{m'}) = \sum_{j=1}^{|H|} P_{i-1}(y_1, \ldots, y_{i-2}, h_j, x_i, \ldots, x_{m'}) y_{i-1}^j. \tag{4}$$

Note that $P_1, P_2, \ldots, P_{m'+1}$ all have degree at most $2hm'$. By the same reasoning as above, we can see that $P_i = 0$ over $\mathbb{F}^{i-1} \times H^{m'-i+1}$ if and only if $P_{i+1} = 0$ over $\mathbb{F}^i \times H^{m'-i}$.

Now let us assume that the proof π contains a sequence of polynomials $P_0, P_1, \ldots, P_{m'+1}$ where P_0 is an m-variate assignment polynomial of degree at most mh and $P_1, \ldots, P_{m'+1}$ are m'-variate polynomials of degree at most $2hm'$. The verifier needs to check the following constraints: for all $x \in \mathbb{F}^{m'}$,

- (C1). $P_1(x) = \hat{P}_1(x)$
  ($\hat{P}_1$ is as defined in equation 2 above)
- (Ci). For all $2 \le i \le m'+1$
  $P_i(x_1, \ldots, x_{m'}) = \sum_{j=1}^{|H|} P_{i-1}(x_1, \ldots, x_{i-2}, h_j, x_i, \ldots, x_{m'}) x_{i-1}^j$
  (condition from equation 4 above at point x)
- (C(m'+2)). $P_{m'+1}(x) = 0$.

There are basically $|\mathbb{F}|^{m'}$ constraints $C(x)$, one for every point $x \in \mathbb{F}^{m'}$. The constraint for point x is defined in terms of the above $m'+2$ constraints as follows:

$$C(x) = \bigwedge_{j=1}^{m'+2} (Cj)(x). \tag{5}$$

In the verification procedure, the verifier picks a random constraint $z \in \mathbb{F}^{m'}$ and accepts if $C(z)$ passes. Clearly, if ϕ is satisfiable, there exists a proof consisting of a sequence of polynomials that satisfies every constraint. On the other hand, if ϕ is not satisfiable, for all proofs consisting of a sequence of polynomials of the above specified form, less than half of the constraints satisfy. In other words, the above $|\mathbb{F}|^{m'}$ constraints satisfy the soundness requirement of **PCP**. This follows from the following Lemma:

Lemma 3
If there exist polynomials $P_0, P_1, \ldots, P_{m'+1}$ of degree specified above such that

$$\forall 1 \le j \le m'+1 \quad \mathbf{Pr}_{x \in \mathbb{F}^{m'}}[\textit{Constraint Cj above satisfies}] \ge \frac{2hm'+h}{|\mathbb{F}|} \tag{6}$$

$$\bigwedge \quad \mathbf{Pr}_{x \in \mathbb{F}^{m'}}[\textit{Constraint C(m'+2) above satisfies}] \ge \frac{2hm'}{|\mathbb{F}|}, \tag{7}$$

then ϕ is satisfiable.

Proof: The contrapositive of the statement follows from the Schwartz-Zippel Lemma (Lemma 2 above) since all the polynomials involved in the constraints Cj ($1 \le j \le m'+1$) have degree at most $2hm'+h$. □

It follows that if ϕ is not satisfiable, no proof consisting of a sequence of polynomials of the above specified form is accepted with probability greater than $\frac{2hm'+h}{|\mathbb{F}|}$, and this probability is less than a given ϵ for sufficiently large n.

2.3 Bundling Polynomials into a Single Polynomial

In the above, we have a sequence of $O(\log n)$ polynomials involved in the proof and constraints. It turns out that in order to reduce the randomness complexity of the verifier down to $O(\log n)$, we need to bundle the sequence of polynomials $P_0, P_1, \ldots, P_{m'+1}$ into a constant number of polynomials. Below we show how to bundle the $m'+2$ polynomials into a single polynomial P.

We want to define a low-degree polynomial $P : \mathbb{F}^{m'+1} \to \mathbb{F}$ such that for all $x \in \mathbb{F}^m$, $P(i,x) = P_i(x)$. The existence of such a polynomial is guaranteed by the following Lemma:

Lemma 4
Given polynomials $q_0, q_1, \ldots, q_m : \mathbb{F}^l \to \mathbb{F}$ with $|\mathbb{F}| > m$ and each of (total) degree at most d, there exists a degree $d+m$ polynomial $Q : \mathbb{F}^{l+1} \to \mathbb{F}$ such that for $i = 0, 1, \ldots, m$ and for all $x \in \mathbb{F}^m$, $Q(i,x) = q_i(x)$.

Proof: For each $i \in \{0, 1, \ldots, m\}$, there is a unique univariate polynomial δ_i of degree m such that

$$\delta_i(a) = \begin{cases} 1 & if\ a = i \\ 0 & if\ 0 \le a \le m\ but\ a \ne i. \end{cases}$$

Now define polynomial Q as $Q(a,x) = \sum_{i=0}^{m} \delta_i(a) q_i(x)$. Then, we have $Q(i,x) = q_i(x)$ for all $x \in \mathbb{F}^m$. $\Box$

Now instead of containing a sequence of polynomials, the proof consists of one single polynomial P of degree $2hm' + m' + 1$. The verifier picks a random constraint $z \in \mathbb{F}^{m'}$ and checks that $C(z)$ passes as described in section 2, except that he queries at $P(i,x)$ whenever he needs the value $P_i(x)$. Thus we can now characterize the constraint C as follows: C has the form $(x_1, x_2, \ldots, x_{w(n)}, C)$, where each $x_i \in \mathbb{F}^{m'+1}$ and depends on z, C is a $w(n)$-variate function $\mathbb{F}^{w(n)} \to \{0,1\}$ and $w(n)$ is $poly \log n$. With this characterization and by Lemma 3, we have the following key Lemma:

Lemma 5
If the given 3SAT instance ϕ is satisfiable, then there exists a degree $2hm' + m' + 1$ polynomial $p : \mathbb{F}^{m'+1} \to \mathbb{F}$ such that for all constraints $z \in \mathbb{F}^{m'}$, $C(p(x_1^{(z)}), p(x_2^{(z)}), \ldots, p(x_{w(n)}^{(z)})) = 1$; if ϕ is not satisfiable, for every degree $2hm' + m' + 1$ polynomial $p : \mathbb{F}^{m'+1} \to \mathbb{F}$, $C(p(x_1^{(z)}), p(x_2^{(z)}), \ldots, p(x_{w(n)}^{(z)})) = 0$ for at least ϵ-fraction of the constraints $z \in \mathbb{F}^{m'}$.

3 Low-degree Testing

So far, we have simply assumed that the proof given contains a low-degree polynomial. The soundness result shown above holds only when this happens. But in our setup, the proof has no such restrictions so the verifier must check that the proof consists of a low-degree polynomial.

However, there is no way for the verifier to check that a function is a low-degree polynomial with just polynomially many queries because a function could agree with the polynomial in all but a few points. However, we can check if the function is "close" to a low-degree polynomial. By close we mean the following:

DEFINITION 1 *Function $f : \mathbb{F}^l \to \mathbb{F}$ is said to be δ-close to a polynomial $p : \mathbb{F}^l \to \mathbb{F}$ if $\mathbf{Pr}_x[f(x) \neq g(x)] \leq \delta$ when x is drawn uniformly at random from $\mathbb{F}^l$.*

A low-degree test is given $\delta > 0$, $d \in \mathbb{Z}^+$, and oracle access to $f : \mathbb{F}^l \to \mathbb{F}$ and its task is to verify that f is δ-close to a degree $\leq d$ polynomial. The test works as follows:

1. Fix $t_0, t_1, \ldots, t_{d+1} \in \mathbb{F}$.
2. Pick at random $x, y \in \mathbb{F}^l$ and find the unique degree d polynomial q such that $q(t_i) = f(x + t_i y)$ for $i = 1, \ldots, d+1$.
3. Verify that $q(t_0) = f(x + t_0 y)$.

The test works by choosing a random line in $\mathbb{F}^l$. The multivariate polynomial restricted to this line becomes a univariate degree d polynomial. We then evaluate at $d+1$ random points on the line, interpolate to find the polynomial, and then test the interpolated polynomial at another random point.

We have the following theorem regarding the above test:

THEOREM 6 (RUBINFELD-SUDAN)
There exists a constant $\delta_0 > 0$ such that for every $\delta < \delta_0$, if the above low-degree test accepts with probability at least $1 - \delta$, then f is 2δ-close to some degree d polynomial.

4 Polynomial Reconstruction Algorithm

We are still left with a problematic situation: what to do if the proof contains a function that is δ-close to a degree $2hm' + m' + 1$ polynomial $P : \mathbb{F}^{m'+1} \to \mathbb{F}$ and it passes the low-degree tests? In this case, we would like to pick a random constraint and test on the polynomial P. However, we don't have oracle access to P but only to f that is δ-close to P. Recall that we ran into the same situation in lecture 16 on the average-case complexity of the permanent and there we gave a polynomial reconstruction algorithm to evaluate P at any given point with high probability with oracle access to f. For completeness we restate the algorithm here. Given $x \in \mathbb{F}^l$ and oracle for f which is δ-close to a polynomial p, compute $p(x)$ as follows:

1. Pick $y \in \mathbb{F}^l$ at random.
2. Query $f(x + y), f(x + 2y), \ldots, f(x + (d+1)y)$ and let $b_1, \ldots, b_{d+1}$ be the responses.
3. Find, by interpolation, the unique degree d univariate polynomial h such that $h(i) = b_i$ for $1 \leq i \leq d+1$.
4. Output $h(0)$ as the value of $p(x)$.

The error probability of the above reconstruction is given in the following theorem:

THEOREM 7
For any given $x \in \mathbb{F}^l$, the above algorithm computes $p(x)$ with probability at least $1 - d\delta$ with $d+1$ queries.

5 Summary

Armed with Lemma 5, Theorem 6 and 7, we can now describe our final **PCP** verifier. The verifier is given a 3SAT instance ϕ. The proof then contains a function $f : \mathbb{F}^{m'+1} \to \mathbb{F}$. The verification procedure proceeds as follows:

1. Run the low-degree test on f with $d = 2m'h + m' + 1$ and an appropriate δ (to be specified below). Reject if the low-degree test rejects.

2. Pick a random constraint $z \in \mathbb{F}^{m'}$ and derive $x_1, x_2, \ldots, x_{w(n)} \in \mathbb{F}^{m'+1}$ based on z. ($w(n)$ is *poly* $\log n$.)

3. Use the polynomial reconstruction algorithm to reconstruct $p(x_1^{(z)}), p(x_2^{(z)}), \ldots, p(x_{w(n)}^{(z)})$. Let the reconstructed values to be $b_1, b_2, \ldots, b_{w(n)}$.

   - In order to save the random bits, the verifier uses the same random $y \in \mathbb{F}^{m'+1}$ in reconstructing each $p(x_i^{(z)})$.

4. Reject if $C(b_1, b_2, \ldots, b_{w(n)}) = 0$.

We set $\delta = O(\frac{1}{d \cdot w(n)})$. First we make sure that the above procedure rejects with probability at least 1/2 when the 3SAT instance ϕ is not satisfiable: If the function f is not δ-close to a degree d polynomial p, then by Theorem 6 the first step rejects with probability at least 1/2. Now we only need to consider the case when f is δ-close to some degree d polynomial p. In this case, we have

$$\begin{aligned}\text{Probability of error} \le\ & \mathbf{Pr}[\text{a random constraint passes the check in step 4}] \\ & + \mathbf{Pr}[\text{there exists an } i \text{ such that } b_i \neq p(x_i^{(z)}) \text{ in step 3}]\end{aligned}$$

For sufficiently large n, the first term on the right-hand side, by Lemma 5, can be made less than 1/6. The second term, by Theorem 7, is upper bounded by $d\delta w(n)$ and can be made less than 1/6 by our choice of δ. Thus, the total error probability can be made less than 1/2.

Now we need to make sure that the query complexity of the verifier is *poly* $\log n$ and the randomness complexity of the verifier is $O(\log n)$.

- Query complexity: the verifier makes queries to the proof f in step 1 and 3. The low-degree test in step 1 makes *poly* $\log n$ queries; the polynomial reconstruction algorithm in step 3 makes $(d+1) \cdot w(n) =$ *poly* $\log n$ queries. Thus, the test makes *poly* $\log n$ queries in total.

- Randomness complexity: the verifier flips coins in step 1, 2 and 3. In the low-degree tests in step 1, in order to reduce the error probability down to 1/2, we need to repeat the test described in Section 3 $O(1/\delta)$ times, which is *poly* $\log n$ times by our choice of δ. Since each time the test uses $2\mathbb{F}^{m'+1} = O(\log n)$ random bits, the tests would use *poly* $\log n \cdot O(\log n)$ random bits in total. Yet by using pair-wise independence between the *poly* $\log n$ trials, the verifier can reduce the number of random bits down to $O(\log\log n) + O(\log n)$, which is $O(\log n)$. In step 2, picking a random constraint takes $m' \log \mathbb{F} = O(\log n)$ bits. In step 3, the verifier picks one random $y \in \mathbb{F}^{m'+1}$ for reconstructing all the values and thus also takes $O(\log n)$ bits. Hence, the test uses $O(\log n)$ random bits in total.

This completes the proof of the **PCP** characterization $\mathbf{NP} = \mathbf{PCP}[O(\log n), \mathit{poly} \log n]$.