**18.405J/6.841J Advanced Complexity Theory** May 21, 1999

# Lecture 21

*Lecturer: Daniel A. Spielman* *Scribe: Louay Bazzi*

## 1 From Last Lecture

We recall from the last lecture that a language $L$ is PCP$(r(n), q(n))$ if $\exists$ a polynomial time verifier $V$ with access to $O(r(n))$ random bits and random access to a proof $\Pi$, of which it can read $O(q(n))$ random bits such that :

$$w \in L \Rightarrow \exists\ \Pi \text{ s.t. } \Pr[V(w,\Pi) \text{ accepts }] = 1$$

$$w \notin L \Rightarrow \forall\ \Pi,\ \Pr[V(w,\Pi) \text{ accepts }] < 1/2.$$

Our objective was to prove

**Theorem 1** *3SAT* $\in$ *PCP*$(\log^{O(1)} n, \log^{O(1)} n)$.

Say that $\phi$ is a 3CNF over $n$ variables. We argued that we will encode an assignment $a$ of $\phi$ using an error correction code as follows. Think of $a$ as function from $\{0,1\}^m$ to $\{0,1\}$ ($m = \log n$), take the (unique) multi-linear extension $A$ of this function over a field $\mathcal{F}$ whose size is $\Theta(m^2)$, and represent $A$ as a table $T : \mathcal{F}^m \to \mathcal{F}$ whose size is $O(2^{O(\log^{O(1)} n)})$.

Then we proposed a test that given any table $T$ can efficiently check if it is a multi-linear extension in the sense that: 1) If $T$ is multi-linear then always accept 2) If probability of acceptance is $> 1/100$ then $\exists$ a multi-linear $A$ s.t.

$$\Pr_{V_1,\ldots,V_n \in \mathcal{F}}[A(V_1,\ldots,V_n) = T(V_1,\ldots,V_n)] > 99/100$$

3) the test uses $O(\log^{O(1)} n)$ random bits and reads only $O(\log^{O(1)} n)$ bits from the table.

At the end of last class we began to arithmetize $\phi(x_1,\ldots,x_n)$. We were looking for an arithmetization that enables the verifier to check through an IP protocol if the assignment given by the table is a satisfying assignment. The big picture is that the verifier will use a number of random bits polynomial in $m$ and hence poly-logarithmic in $n$. The number of bits that the prover sends to the verifier is also poly-logarithmic in $n$. To turn this into a PCP, will store in the probabilistically checkable proof, in addition to the table $T$, all the responses of the prover indexed by the IP verifier random bits. So, the PCP verifier will first perform the multi-linearity test on the table, and then will engage in an IP protocol with the proof.

To arithmetize $\phi$, we started by arithmetizing each of its clauses. For any clause $c$ of $\phi$, we constructed a formula CC$(A, c)$ that is 0 iff $A$ satisfies $c$. The Clause Check is given by

$$CC(A,c) = \sum_{\substack{V_1^{(1)},\ldots,V_m^{(1)} \in \{0,1\} \\ V_1^{(2)},\ldots,V_m^{(2)} \in \{0,1\} \\ V_1^{(3)},\ldots,V_m^{(3)} \in \{0,1\}}} \Pi_{j=1}^{3} \mathcal{X}_{j,c}(V_1^{(j)},\ldots,V_m^{(j)}) \left(\mathcal{S}_{j,c} - A(V_1^{(j)},\ldots,V_m^{(j)})\right),$$

where $\mathcal{X}_{j,c}$ is the multi-linear extension of the boolean function $X_{j,c} : \{0,1\}^m \to \{0,1\}$ given by

$$X_{j,c}(V_1,\ldots,V_m) = \left\{ \begin{array}{ll} 1 & \text{if the index of } j\text{th variable of clause } c \text{ is } V_1,\ldots,V_m \text{ in binary} \\ 0 & \text{otherwise,} \end{array} \right\}$$

and

$$S_{j,c} = \left\{ \begin{array}{ll} 1 & \text{if the } j\text{th variable of clause } c \text{ is not negated} \\ 0 & \text{otherwise .} \end{array} \right.$$

## 2 Completing the Arithmetization

We have a clause check $CC(A, c)$ that is zero iff $A$ satisfies $c$. We want something like $CC$, but equal to zero iff all the clauses of $\phi$ are satisfied. There are many ways to do this, but the following will prove helpful later in the proof. The idea is use some randomness. To get started, consider $\sum_c r_c CC(A, c)$, where the sum is over the clauses of $\phi$, and the $r_c$'s are chosen at random from $\mathcal{F}$. Since $\mathcal{F}$ is large enough, with a high probability this sum is zero iff $A$ satisfies all the clauses. But there is a problem with this approach, it requires too many random bits. To fix the problem we use pseudo randomness. Say that $\phi$ has $t$ clauses $c_1, \ldots, c_t$ , and assume that $t$ is a power of 2, say $2^s$ (if this not the case we can always duplicate some clauses). Also if $a \in \{0,1\}^s$, let $c_a$ mean $c_k$, where $a$ is the binary representation of $k$. Now, consider the Formula Check

$$FC(\phi, A) = \sum_{a \in \{0,1\}^s} r_1^{a_1} \ldots r_s^{a_s} CC(A, c_a),$$

where $r_1, \ldots, r_s$ are chosen at random $\mathcal{F}$. This requires only $O(s \log |\mathcal{F}|) = O(m \log m)$ random bits only ( recall that $|\mathcal{F}| = \Theta(m^2)$). On the other hand, if we look at $FC(\phi, A)$ as a multi-linear polynomial in $r_1, \ldots, r_s$, it follows immediately from Shwarz Lemma that

$$\begin{aligned} &\Pr_{r_1,\ldots,r_s \in \mathcal{F}} [FC(\phi, A) = 0 \text{ iff } A \text{ satisfies all the clauses of } \phi] \\ &\geq 1 - \Pr_{r_1,\ldots,r_s \in \mathcal{F}} [(r_1,\ldots,r_n) \text{ is a zero of } FC(\phi, A)] \\ &\geq 1 - \frac{s}{|F|} = 1 - O\left(\frac{1}{m}\right). \end{aligned}$$

The verifier needs to be convinced that $FC(\phi, A) = 0$. To make this suitable for the IP machinery, we use the definition of $CC(A, c_a)$, to express $FC(\phi, A)$ as

$$FC(\phi, A) = \sum_{\substack{V_1^{(1)},\ldots,V_m^{(1)} \in \{0,1\} \\ V_1^{(2)},\ldots,V_m^{(2)} \in \{0,1\} \\ V_1^{(3)},\ldots,V_m^{(3)} \in \{0,1\}}} Q(V_1^{(1)},\ldots,V_m^{(3)}),$$

where

$$Q(V_1^{(1)},\ldots,V_m^{(3)}) = \sum_{a\in\{0,1\}^s} r_1^{a_1}\ldots r_s^{a_s}\Pi_{j=1}^3 \mathcal{X}_{j,c}(V_1^{(j)},\ldots,V_m^{(j)})\left(\mathcal{S}_{j,c} - A(V_1^{(j)},\ldots,V_m^{(j)})\right).$$

Observe that $Q$ is of degree at most 6 in each $V_i^{(j)}$ and of degree at most $6m$ in all the $V_i^{(j)}$'s. This is because $\mathcal{X}_{i,c}(V_1^{(j)},\ldots,V_m^{(j)})$ and $\mathcal{S}_{j,c} - A(V_1^{(j)},\ldots,V_m^{(j)})$ are both multi-linear.

# 3 The IP protocol

The interactive proof of $FC(\phi, A) = 0$ is very similar to the one we used for #SAT.

1. $P$ sends to $V$ the coefficients of $\hat{Q}_1(z)$, a polynomial claimed to be equal to

$$Q_1(z) = \sum_{V_2^{(1)},\ldots,V_m^{(3)}\in\mathcal{F}} Q(z, V_2^{(1)},\ldots,V_m^{(3)}).$$

$V$ checks if $\hat{Q}_1(0)+\hat{Q}_1(1) = 0$. If this is not true then $V$ rejects, else $V$ selects $a_1$ randomly from $\mathcal{F}$, and send it to $P$.
($P$ convinces $V$ that $FC(\phi, A) = 0$ with a high probability if $\hat{Q}_1(a_1) = Q_1(a_1)$)

.

.

.

i. $P$ sends to $V$ the coefficients of $\hat{Q}_i(z)$, a polynomial claimed to be equal to

$$Q_i(z) = \sum_{V_?^{(?)},\ldots,V_m^{(3)}\in\mathcal{F}} Q(a_1,\ldots,a_{i-1}, z, V_?^{(?)},\ldots,V_m^{(3)}).$$

$V$ check if $\hat{Q}_i(0)+\hat{Q}_i(1) = \hat{Q}_{i-1}(a_{i-1})$. If this is not true then $V$ rejects, else $V$ selects $a_i$ randomly from $\mathcal{F}$, and send it to $P$.
($P$ convinces $V$ that $\hat{Q}_{i-1}(a_{i-1}) = Q_{i-1}(a_{i-1})$ with a high probability if $\hat{Q}_i(a_i) = Q_i(a_i)$ )

.

.

.

3m. $P$ sends to $V$ the coefficients of $\hat{Q}_{3m}(z)$, a polynomial claimed to be equal to

$$Q_{3m}(z) = Q(a_1,\ldots,a_{3m-1}, z).$$

$V$ checks if $\hat{Q}_{3m}(0)+\hat{Q}_{3m}(1) = \hat{Q}_{3m-1}(a_{3m-1})$ If this is not true then $V$ rejects, else $V$ selects $a_{3m}$ randomly from $\mathcal{F}$, and checks if $\hat{Q}_{3m}(a_{3m}) = Q(a_1,\ldots,a_{3m})$, by computing the later from the table. If this is not true then $V$ rejects.
($V$ checks if the value of $\hat{Q}_{3m}(a_{3m})$ is correct according to the table)

If $FC(\phi, A) = 0$ then, there exists an "honest" prover and "perfect" table such that the verifier accepts with a probability 1. Here by a "perfect" table, we mean a one that contains the correct values of $A$ in all its entries.

If on the other-hand $FC(\phi, A) \neq 0$, then , for any prover $P$, $V$ may accepts due two possible reasons:

- Either $P$ remained consistent by keeping on sending polynomials $\hat{Q}_i(z) \neq Q_i(z)$ but satisfying $\hat{Q}_1(0) + \hat{Q}_1(1) = 0$ and $\hat{Q}_i(0) + \hat{Q}_i(1) = \hat{Q}_{i-1}(a_{i-1})$ , for $i = 2, \ldots j$, where $j$ is a round at which $P$ was lucky when $V$ selected an $a_j$ from $\mathcal{F}$ for which $\hat{Q}_j(a_j) = Q_j(a_j)$.
- Or $V$ computed at the last round the value of $Q(a_1, \ldots, a_{3m})$ incorrectly in such a way that it turned out to be equal to $\hat{Q}_{3m}(a_{3m})$.

For a fixed $j$, the probability that $\hat{Q}_j(a_j) = Q_j(a_j)$, given that $\hat{Q}_j(z)$ and $Q_j(z)$ are not equal, is at most

$$\frac{\text{degree}(\hat{Q}_i - Q_i)}{|\mathcal{F}|} \leq \frac{6}{\Theta(m^2)}$$

since the degree of each $Q_i$ is at most 6 (Here we are assuming that $V$ rejects when $P$ sends a polynomial whose degree is more than 6). The degree bound on the $Q_i$'s follows from the fact that $Q$ is a multi-linear function of degree 6 in each variable independently. So the probability that $P$ was lucky at some $j$, is at most $3m(6/\Theta(m^2)) = O(1/m)$.

To compute $Q(a_1, \ldots, a_{3m})$, the verifier uses the table $T$, which might not agree with $A$ in all its entries. All what we know from the multi-linearity test is that $T$ agrees with $A$ in 99/100 of its entries. So, the probability that $V$ computes $Q(a_1, \ldots, a_{3m})$ incorrectly, is at most

$$\Pr_{a_1, \ldots, a_{3m} \in \mathcal{F}}[T(a_{im+1}, \ldots, a_{im+m}) \neq A(a_{im+1}, \ldots, a_{im+m}) \text{ for } i = 0, 1, \text{ or } 2] < \frac{3}{100}.$$

Therefore for any prover $P$, and any table $T$ that passed the multi-linearity test, the probability that $V$ accepts when $FC(\phi, A) \neq 0$ is at most $3/100 + O(1/m) < 1/25$, for $m$ large enough.

Finally note that $V$ can compute $Q(a_1, \ldots, a_m)$ by accessing the table 3 times, while reading each time $O(\log m)$ bit. Note also that the verifier needs $O(m \log m)$ random bits to select the $a_i$'s, and that the prover have to send, in each of the the $3m$ rounds, $O(\log m)$ bit to the verifier since each of the polynomials being sent is of degree $\leq 6$.

# 4 Turning the IP into a PCP

To turn the IP into a PCP, we store in the probabilistically checkable proof $\Pi$, in addition to the table $T$, a huge table $H$ containing all the responses of the prover indexed by the $O(m \log m)$ IP verifier random bits and the $O(m \log m)$ random bits used in the construction of $FC(\phi, A)$. In other words, if $r_1, \ldots, r_s$ are the elements of $\mathcal{F}$ used in the construction of $FC(\phi, A)$ and if the IP verifier is at round $i$, then the IP verifier computes, from $(r_1, \ldots, r_s, a_1, \ldots, a_i)$, the entry of $H$ that contains the $O(\log m)$ bits corresponding the prover's response. So the total size of the probabilistically checkable proof $\Pi$ is $|T| + |H| = 2^{O(\log^{O(1)} n)} + 2^{O(m \log m)} = 2^{O(\log^{O(1)} n)}$.

The number bits to be read by the PCP verifier from the proof is $O(m \log m + \log^{O(1)} n) = O(\log^{O(1)} n)$, where the first terms corresponds to the IP protocol and the second term corresponds to the multi-linearity test. Similarly, the number of random bits needed by the PCP verifier is is $O(m \log m + \log^{O(1)} n) = O(\log^{O(1)} n)$.

Finally,

- If $\phi \notin 3SAT$, then for any probabilistically checkable proof $\Pi$, the probability that the PCP verifier accepts $(\Pi, \phi)$ is bounded by $O(1/m) + 1/25 < 1/2$, where the first term is a bound on the probability that $FC(\phi, A) = 0$ is not equivalent to the fact that $A$ satisfies all the clauses of $\phi$, and the second term is a bound on the probability that IP protocol accepts when $FC(\phi, A) \neq 0$.
- If $\phi \in 3SAT$, then there exits a proof $\Pi$, containing a table that is multi-linear everywhere, and the responses of an "honest" prover, such that the PCP verifier accepts $(\Pi, \phi)$ w.p. 1.

This proves that 3SAT $\in$ PCP$(\log^{O(1)} n, \log^{O(1)} n)$.

# 5 A stronger PCP theorem

The best *PCP* theorem know so far is

**Theorem 2** *$NP \subset PCP(\log n, 1)$.*

The size of the corresponding probabilistically checkable proof is $n^{1+\epsilon}$, for any $\epsilon > 0$. The proof of this theorem is very complicated and we don't know of any simple one.

This result is considered a landmark in complexity theory. We will see in the next lecture how it can be applied to establish the hardness of approximating various optimization problems.