COM S 6810 Theory of Computing April 14th, 2009

Lecture 23: PCP: Probabilistic Checkable Proofs

Instructor: Rafael Pass *Scribe: Jean-Baptiste Jeannin*

The idea behind a probabilistic checkable proof is:

- the prover writes a proof
- the verifier is lazy and/or does not have time, and therefore he only checks part of the proof. With only that part of the proof he must decide (with some error probability) whether the proof is correct or not.

There is no interaction involved here, since the prover writes the proof once and for all, then does not answer any request from the verifier.

Definition 1 (PCP) *A language* $L \in \mathrm{PCP}(r, q)$ *if there exists a PPT verifier* V *that on an input* (X, Π) *uses at most* r *coins and reads at most* q *bits of* Π*, and:*

- *if* $x \in L$, $\exists \Pi$ *such that* $\Pr(\mathrm{V}(\mathrm{X}, \Pi) = 1) = 1$
- *if* $x \in L$, $\exists \Pi$ *such that* $\Pr(\mathrm{V}(\mathrm{X}, \Pi) = 1) \leq \frac{1}{2}$

In this definition X *represents the theorem to prove, and* Π *the proof submitted by the prover.*

Definition 2 ($\mathrm{PCP}_{C,S}$) *A language* $L \in \mathrm{PCP}_{C,S}(r, q)$ *if there exists a PPT verifier* V *that on an input* (X, Π) *uses at most* r *coins and reads at most* q *bits of* Π*, and:*

- *if* $x \in L$, $\exists \Pi$ *such that* $\Pr(\mathrm{V}(\mathrm{X}, \Pi) = 1) \geq \mathrm{C}$
- *if* $x \in L$, $\exists \Pi$ *such that* $\Pr(\mathrm{V}(\mathrm{X}, \Pi) = 1) \leq \mathrm{S}$

These definitions, in particular the completeness and soundness parts, are very close to the definitions of MA we have seen. This is because the protocol here, with the prover first committing to a proof then the verifier verifying it, is also very close to the protocol used in MA.

The soundness factor ($\frac{1}{2}$ in the first definition, C in the second) can be arbitrarily reduced by repeating the verifying step several times; this however has a cost, since if we repeat it i times, we have to use $i \times r$ random bits instead of r.

A verifier is said to be *non-adaptive* if it just looks at X, flips r coins to decide which positions to look in the proof Π, looks at these positions and decides whether to accept or reject the proof. A verifier is said to be *adaptive* if a bit looked at in the proof Π will decide which subsequent bits will be looked at in Π. In this class we will be mostly interested in non-adaptive verifiers.

Claim 1 (trivial) $\mathrm{PCP}(0, \mathrm{poly}(n)) = \mathrm{NP}$

Claim 2 $\mathrm{PCP}(\log n, \mathrm{poly(n)}) = \mathrm{NP}$

Proof. $(\supseteq)$ is trvial using the easy claim above. For $(\subseteq)$, enumerating all possible random strings, we still get a polynomial time algorithm.

Theorem 1 (PCP) $\mathrm{PCP}(O(\log n), O(1)) = \mathrm{NP}$

Theorem 2 (Håstad PCP) *For every* $\delta > 0$, $\mathrm{PCP}_{1-\delta,\frac{1}{2}+\delta}(O(\log n), 3) = \mathrm{NP}$.
Furthermore the verifier is non-adaptive and the test is linear: the verifier V *picks 3 positions* i, j *and* k *and a bit* b, *and accepts if and only if* $\Pi_i + \Pi_j + \Pi_k = b \bmod 2$, *where* Π_i *is the* i*-th bit of the proof* Π.

We will not prove these theorems, but a lighter version that we will see later in this lecture. But first let us look at the problem MAX E3LIN, which is the problem of finding a solution maximizing the number of satisfied equations among a given system of linear equations modulo 2, where each equation involves 3 variables.

- if there exists an assignment satisfying all equations, then it is easy and efficient to find using a Gaussian elimination.
- if there is no such assignment, the first idea is to pick a random assignment. Then the expectation for the number of satisfied equations is given by:

$$E(\#\text{satisfied equations}) = \sum_{i \in [m]} E(\text{equation i is satisfied}) = \frac{m}{2}$$

- if there is no such assignment, a second idea would be to choose an assignment x_1' for x_1 such as maximizing $E(\#\text{sat equations}|x_1 = x_1')$. Since

$$E(\#\text{sat equations}) = \frac{1}{2}E(\#\text{sat equations}|x_1 = 0) + \frac{1}{2}E(\#\text{sat equations}|x_1 = 1)$$

  we know that either $E(\#\text{sat equations}|x_1 = 0)$ or $E(\#\text{sat equations}|x_1 = 1)$ has to be greater than $E(\#\text{sat equations})$, where each of the $E(\#\text{sat equations}|x_1 = x_1')$ can be computed as half of the formulas remaining (see formula above). Then we can choose an assignment for x_2 in the same way, etc.

Corollary 1 *For $\delta > 0$, approximating MAX E3LIN better than $\frac{1}{2} + \delta$ is NP-complete.*

This means that we cannot do better than a random assignment if P≠NP.

Proof. We show than even if it is possible to satisfy a $1 - \delta$ fraction of the equations, there is no polytime algorithm that finds an assignment satisfying more than a $\frac{1}{2} + \delta$ fraction of them. Assume for contradiction that there exists such a polytime algorithm A, we show how to decide SAT using A.

Let f be a formula in SAT. Then each clause in f can be easily transformed in an equation modulo 2 Then we run Håstad's PCP verifier on this set of equations and with all possible random coins. If f is satisfiable, then for a fraction $\geq 1 - \delta$ of the possible random coins Håstad's PCP verifier will say so; if f is not satisfiable, then for a fraction at most $\leq \frac{1}{2} + \delta$ of the possible random coins Håstad's PCP verifier will say it is satisfiable. Therefore, as soon as $\delta < \frac{1}{4}$ this allows us to decide whether f is satisfiable. ■

One might wonder whether it is possible to remove the δ in Håstad's verifier. In fact, it is not possible because of the construction above.

Theorem 3 (Williamson) *It is possible to approximate MAX E2LIN within 0.878.*

This means that we cannot get down to 2 in Håstad's unless we constrain the soundness to be less than $1 - 0.878$. This is because the probability of cheating has to be greater than 0.878.

Theorem 4 (PCP Light) $\text{NP} \subseteq \text{PCP}(\text{poly}(n), O(1))$

Proof. Here we only give an idea of the proof. The complete proof will be seen in next class.

First remember the interactive proof for GRAPHNONISO: the verifier gives a permutation of one of the graphs and the prover is supposed to say from which graph it came from. A PCP proof for this problem would be to enumerate from which graph it came from for all possible permutations. This proves that GRAPHNONISO$\subseteq$ PCP$(\text{poly}(n), O(1))$.

Then let us consider the problem the satisfiability of a system of quadratic equations modulo 2. We consider m equations of the type

$$\Sigma_{i,j \in [N]} C_{i,j}^k X_i X_j = C_k \bmod 2$$

In homework 1 we proved that deciding whether such a system is satisfiable is NP-complete. This is in fact exactly of the form

$$< \overrightarrow{C}^k, \overrightarrow{X}^{(2)} >= C^k \bmod 2$$

with $\overrightarrow{X}^{(2)} = (X_1X_1, X_1X_2, X_1X_3, ..., X_2X_1, ..., X_nX_n)$.

Let $a_1, ..., a_n$ be a satisfying assignment. Consider all linear functions $\overrightarrow{V}$ on $\overrightarrow{a}^{(2)}$ and

$$\Pi[\overrightarrow{V}] =< \overrightarrow{V}, \overrightarrow{a}^{(2)} >= \sum_{i,j} V_{ij}a_ia_j$$

Idea of the verification step: Given such a PCP

1. Test whether it is a linear function, if it is call it λ,
2. Test $\lambda = \overrightarrow{a}^{(2)}$,
3. For check whether all equations agree with C^k, use a subset sum.