PRINCETON UNIVERSITY COS 522: COMPUTATIONAL COMPLEXITY

Lecture 13: PCP (Probabilistically Checkable Proofs)

Lecturer: *Sanjeev Arora* Scribe:*Nir Ailon*

1 Definitions

In the traditional view of NP, we had the following players: an input x, a certificate (a proof) π and a poly-time verifier V, s.t.:

$$x \in L \Rightarrow \exists \pi . V^{\pi}(x) = 1$$
$$x \notin L \Rightarrow \forall \pi . V^{\pi}(x) = 0,$$

where V^{π} denotes "a verifier with access to π".

An $(r(n), q(n))$-restricted verifier is a randomized verifier with access to at most $O(r(n))$ random bits and at most $O(q(n))$ bits of the proof. In other words, the verifier cannot access the entire proof, like it can in the classic NP definition, but only a restricted window. It can query at most $O(q(n))$ positions in the proof, it can choose the positions, and it has $O(r(n))$ coin flips at its disposal. It turns out, that querying even a small snippet of the proof suffices to give the verifier surprising power. This will be formulated below as the PCP theorem. We start with a definition of a family of complexity classes:

DEFINITION 1 *$PCP(r(n), q(n))$ is the class of all languages accepted by an $(r(n), q(n))$-restricted verifier V in the following sense:*

$$x \in L \Rightarrow \exists \pi . \Pr[V^{\pi}(x) = 1] = 1$$
$$x \notin L \Rightarrow \forall \pi . \Pr[V^{\pi}(x) = 0) \geq \frac{1}{2}$$

One can add the notion of adaptive vs. nonadaptive verifiers: an adaptive verifier can decide at each moment which bit to query based on the history of computation before that moment, and a nonadaptive selects its queries in advance. Basically, nonadaptive is general enough.

2 Observations

1. $PCP(r(n), q(n)) \subseteq NTIME(2^{O(r(n))} q(n))$. PROOF: If V is a verifier for $L \in PCP(r(n), q(n))$, then without loss of generality, we can assume that the "proof" π has at most $2^{O(r(n))} q(n)$ bits - $q(n)$ bits accounted for any possible outcome of the coin flips. Instead of performing a PCP verification, the verifier can treat π as a usual witness and simply simulate all possible outcomes of coin flips and calculate the probability of accepting. If the probability is 1 then output "accept" else output "reject". □

2. $NP = PCP(0, poly(n)) = \bigcup_{c>0} PCP(0, n^c)$.

3. $coRP = PCP(poly(n), 0)$.

4. GRAPH-NONISOMORPHISM $\in PCP(poly(n), 1)$. PROOF: Say the input to a "yes" instance of GRAPH-NONISOMORPHISM is $x = \{G_1, G_2\}$, where G_1, G_2 have both n nodes. π contains, for each labelled graph H with n nodes, a bit 0/1 corresponding to whether H is

isomorphic to G_1 or to G_2 (an arbitrary bit for neither case). In other words, π is an array of bits indexed by some encoding of the graph H.

The verifier picks G_1 or G_2 at random, together with a random permutation. She applies the permutation to the vertices of the chosen graph to obtain an isomorphic graph, H, and queries the corresponding bit of π. If the query bit corresponds to the choice of G_1/G_2, then she outputs "success", otherwise "failure".

If G_1 and G_2 are indeed non-isomorphic, then for the π constructed above, every query for which $H \approx G_1$ returns a 0 bit, and every query for which $H \approx G_2$ returns a 1 bit. If G_1 and G_2 are isomorphic, then for any "proof" π the query bit will correspond to the index of the randomly chosen graph with probability $\frac{1}{2}$, as required. □

5. $PERM \in PCP(poly(n), poly(n))$. PROOF: Exercise. □

3 The PCP theorem and its implication to approximation

The PCP theorem [AS '92, ALMSS '92] is

THEOREM 1
$NP = PCP(\log n, 1)$.

Note: Hastaad proved the following stronger result: NP equals PCP with $O(\log n)$ random bits and exactly 3 query bits.

The PCP theorem gives a new surprising characterization for NP, but the main interest in it lies in its implication to hardness results for approximating NP-hard problems. This motivation is explained in the following example. MAX-3SAT is the problem of finding, given a 3-CNF boolean formula as input, an assignment that maximizes the number of satisfied clauses. This problem is of course NP-hard, because the corresponding decision problem, 3SAT, is NP-complete.

DEFINITION 2 *An algorithm computes an α-approximation to MAX-3SAT if it computes (in polynomial time) for every instance ϕ an assignment that satisfies at least $\frac{OPT(\phi)}{\alpha}$ clauses.*

The PCP theorem implies

COROLLARY 2
There exists a $\epsilon > 0$ s.t. $(1+\epsilon)$-approximation to MAX-3SAT is NP-hard.

PROOF: Let L be any NP-complete language, let V be a verifier for L, and let π a proof queried by V (in the $PCP(\log n, 1)$ sense). For every possible outcome r of the $O(\log n)$ random bits, V queries 3 bits from π. If we consider the bits of π as variables, then V queries the variables x_1^r, x_2^r, x_3^r. Then it does some computation on these variables, whose outcome is expressible by a 3-CNF formula with a bounded number of clauses (any boolean function in 3 variables can be expressed using $\leq 2^3 = 8$ clauses). Denote this 3-CNF formula by ϕ_r. Consider the boolean formula

$$\Phi = \bigwedge_r \phi_r$$

in the variables of π. The PCP theorem tells us that if $x \in L$, then Φ is satisfiable, i.e., all its clauses can be satisfied. On the other hand, for $x \notin L$, any "proof" π is such that at least $\frac{1}{2}$ of the possible queries result in rejection. In other words, for any assignment for Φ, at least $\frac{1}{2}$ of the expressions $\{\phi_r\}$ are not satisfied. Since each ϕ_r has at most 8 clauses, we get that at least $\frac{1}{16}$ of the clauses of Φ are unsatisfied.

To make the proof complete, we have to make sure that computing Φ from x can be done in polynomial time. This proves that it is NP-hard to distinguish between satisfiable 3-CNF formulas and 3-CNF formulas with at most fraction $\frac{15}{16}$ of the clauses satisfiable. In particular, it is NP-hard to compute a $\frac{16}{15}$-approximation to MAX-3-SAT. □

Remark: The PCP theorem gives us a new reduction for languages in NP which has the property of robustness. In the classic Cook-Levin reduction, computations of a Turing machine are encoded as CNF formulas. If a verifier does not actually check all the clauses of this formula, then a prover can fool it to believe that for a $x \notin L$ there is an accepting computation. This is true because it is the same as saying that by slightly changing a Turing machine computation, a non accepting computation can be diverted into an accepting one. So a verifier that checks only a few "windows of computation" will be fooled with very high probability. This does not happen in the reduction described in the above proof.

4 Towards proving the PCP theorem

The nontrivial part is, of course, $NP \subseteq PCP(\log n, 1)$. It suffices to show that 3SAT $\in PCP(\log n, 1)$. Instead of using the usual certificate for satisfiablity of a boolean 3CNF formula ϕ, i.e., the satisfying assignment, we will apply any algebraic transformation to the instance of the problem such that a certificate will be a multivariate polynomial.

We start with some definitions.

DEFINITION 3 *A* monomial *is an expression of the form* $ax_1^{d_1}x_2^{d_2}...x_m^{d_m}$*, where* a *is from some ground field* F*,* $x_1..x_m$ *are formal variables and* $d_1..d_m$ *are positive integers. The* degree *of the monomial is* $\sum_{i=1}^{m} d_i$*. A* polynomial *is a sum of monomials. The* degree *of a polynomial is the degree of its highest-degree monomial.*

In what follows we consider only finite fields, i.e. $F = GF(q)$ for q a power of some prime.

LEMMA 3
(Schwartz-Zippel) If a degree d *polynomial* p *over* $F = GF(q)$ *is nonzero, then*

$$\Pr[p(a_1..a_m) \neq 0] \geq 1 - \frac{d}{q},$$

where the probability is over all choices of $a_1..a_m \in F$*.*

PROOF: By induction on the number of variables in p. If p is of degree 0 (p is a constant $\neq 0$) then the above probability is trivially 1. If there is one variable, then it is a well known fact that p has at most d zeroes. Therefore,

$$\Pr_{a \in F}[p(a) \neq 0] \geq 1 - \frac{d}{q}.$$

Assume the statement is true for polynomials with up to $m-1$ variables, and let p be a polynomial in m variables $x_1..x_m$. For each possible assignment $a_m \in F$ for the single variable x_m, Let $p_{|x_m=a_m}$ be the polynomial that is obtained by the substitution. The polynomial $p_{|x_m=a_m}$ is in $m-1$ variables $x_1, .., x_{m-1}$ and has degree at most d. By the induction hypothesis,

$$\Pr_{a_1,..,a_{m-1} \in F}[p_{|x_m=a_m}(a_1, .., a_{m-1}) \neq 0] \geq 1 - \frac{d}{q}.$$

Therefore,

$$\Pr_{a_1..a_m \in F}[p(a_1, ..., a_m) \neq 0] = E_{a_m \in F}[\Pr_{a_1..a_{m-1} \in F}[p_{|x_m=a_m}(a_1, .., a_{m-1}) \neq 0]]$$
$$\geq 1 - \frac{d}{q}$$

as required. □

The Schwartz-Zippel lemma is the "source" of robustness for the reduction that will be constructed.