Today

- NP ⊆ PCP[O(\log n), \text{poly log } n].

Last time

- Defined PCP.
- Verifier is probabilistic. Tosses r(n) coins.
- Verifier interacts with an oracle (i.e., has random access to a proof string). Makes q(n) queries.
- Accepts valid proofs with probability ≥ c(n). (i.e., if x ∈ L, there exists π s.t. ...)
- Accepts invalid theorems with probability ≤ s(n). (i.e., if x ∉ L, for all π ...)
- PCP_{c,s}[r,q] class of such languages L.

- One subscript implies c = 1 suppressed.
- Zero subscripts implies c = 1, s = 1/2.

Last time (contd.)

- Mentioned best known result: NP ⊆ PCP_{1,\frac{1}{2}+\epsilon}[O(\log n), 3]. [Hastad].
- Consequence: Approximating MAX SAT to within 15/16 + \delta, for any \delta > 0 is NP-hard.
- Today: A simpler PCP theorem.
Main ingredients

- NP hardness of an algebraic problem.
- PCP verifier for the algebraic problem.

Algebraic problem: Polynomial constraint satisfaction

- Constraint satisfaction problems: Generic class of problems. $x_1, \ldots, x_n$ variables. $C_1, \ldots, C_t$ constraints (clauses). Goal: Find assignment $x_i \rightarrow a_i$ that satisfies as many constraints as possible.
- Typically, no restriction on assignment.

PCS

- $n$ associated with $m$-dimensional space over some field $\mathbb{F}$. I.e., $n = |\mathbb{F}|^m$.
- Assignment is a function $f : \mathbb{F}^m \rightarrow \mathbb{F}$.
- Constraints are arbitrary functions on $f$, given by “truth table” or circuit evaluating them.
- Each constraint will apply to $\log n$ variables.
- Only interested in assignments that are low-degree polynomials.

PCS

- Instance: $(m, \mathbb{F}, d, w; C_1, \ldots, C_t)$, where $C_j$ given by $x_1^{(j)}, \ldots, x_n^{(j)} \in \mathbb{F}^m$ and $A^{(j)} : \mathbb{F}^w \rightarrow \{0, 1\}$, given by arithmetic circuit.
- Yes instances: There exists a degree $d$ polynomial $f : \mathbb{F}^m \rightarrow \mathbb{F}$ such that all constraints satisfied.
- No instances: Every degree $d$ polynomial $f : \mathbb{F}^m \rightarrow \mathbb{F}$, fails to satisfy almost all (90%) constraints.
PCS claims

Lemma 1: PCS has a PCP verifier that tosses $O(\log t + m \log |F|)$ coins, queries the proof $O(wd \log |F|)$ times, and has $c = 1$ and $s = \frac{1}{2}$.

Lemma 2: SAT on $n$ variables reduces to PCS in time $|F|^m$, for any $F, m, d, w$ such that $F \geq 100wd$ and $(d/m)^m \geq n^c$ and $w \geq d$.

Comments: Lemma 2 is just an NP hardness result?

- Weaker soundness since it only applies to some assignments.
- Stronger since it gives a gap.

Missing ingredients in PCP proof

- Hardness of PCS.
- Low-degree testing
- Self-correction of polynomials.

Proof of Lemma 1

PCP Verifier:

- Expects proof oracle to be a degree $d$ polynomial $f : \mathbb{F}^m \rightarrow \mathbb{F}$.
- Step 1: Test function $f$ is close to some degree $d$ polynomial $p$. (“Low-degree testing”).
- Build oracle for $p$ (“Polynomial self-correction”).
- Pick random constraint $C_j$ and verify if $p$ satisfies $C_j$.

Self-correction problem

Given oracle $f : \mathbb{F}^m \rightarrow \mathbb{F}$ s.t. there exists a polynomial $p : \mathbb{F}^m \rightarrow \mathbb{F}$ s.t. $\Pr_{x \in \mathbb{F}^m}[f(x) \neq p(x)] \leq \delta$.

Given also $a \in \mathbb{F}^m$.

Compute $p(a)$. 
Basic idea: Lines in $\mathbb{F}^m$

Pick random $r \in \mathbb{F}^m$.

Look at line $\ell(t) = (1-t)a + tr$.

$p|_\ell$ is degree $d$ polynomial.

We want $p|_\ell(0)$.

$\ell(t)$ is random point of $\mathbb{F}^m$, except if $t = 0$.
So $p_\ell(t) = f(\ell(t))$ w.p. $1-\delta$.

Self-correction algorithm

• Pick $r \in \mathbb{F}^m$ at random.
• Let $\tau_1, \ldots, \tau_{d+1}$ distinct $\in \mathbb{F}$.
• Compute $h$ of degree $d$ s.t. $h(\tau_i) = f((1-\tau_i)a + \tau_ir)$.
• Output $h(0)$.

Analysis

• $\Pr_r[\exists i \text{ s.t. } p|_\ell(\tau_i) \neq f(\ell(\tau_i))] \leq (d + 1)\delta$.
• W.p. $1 - (d + 1)\delta$, $h = p|_\ell$ and so $h(0) = p(\ell(0)) = p(a)$.

Above due to [BeaverFeigenbaum, Lipton].

Low-degree testing

How to test if arbitrary function $f$ is close to some polynomial of degree $d$?

Run time $\text{poly}(m, d)$.

Can’t examine whole function.

Can’t even write coefficients!
Idea

If function is close to a polynomial, then its self-correction equals itself at most points. Test this.

Algorithm:

- Repeat many times:
  - Pick $a, r \in \mathbb{F}^m$ at random.
  - Let $\tau_1, \ldots, \tau_{d+1}$ distinct $\in \mathbb{F}$.
  - Compute $h$ of degree $d$ s.t. $h(\tau_i) = f((1 - \tau_i)a + \tau_ir)$.
  - Verify $h(0) = f(a)$.

Analysis

Non-trivial. Beyond scope of interesting lectures!

Theorem [Rubinfeld-Sudan, ALMSS]: Every iteration gives $\min\{\delta/c, \gamma\}$ probability of detecting cheating, if $f$ is $\delta$ far from every degree $d$ poly.

R-S result $\gamma = \Theta(1/d)$, $c = 2$.

ALMSS: $\gamma > 0$, but $\gamma \sim 0$, $c = 2$.

$\delta$-the-art, $c = 1 + o(1)$, $\gamma = 1 - o(1)$, where $o(1)$ depends on $d/|\mathbb{F}|$.

PCS hardness

- Skip problem statement for now.
- Will play with proof of $\#P$ in IP and define some polynomial straight line programs.
- Will shrinkwrap into hardness of PCS later.

Idea

- Arithmetize SAT, and “count” number of clauses unsatisfied. (Not number of satisfying assignments).
- For intuition, think of $n = 2^m$ and $|n| = \{0, 1\}^m$.
- Given SAT formula $\phi$, think of assignment as a function $A: \{0, 1\}^m \rightarrow \{0, 1\}$.
- Extend assignment into function $\hat{A}: \mathbb{F}^m \rightarrow \mathbb{F}$ for some appropriate field $\mathbb{F}$.
- Prop: Every function $A: \{0, 1\}^m \rightarrow \{0, 1\}$ can be extended into polynomial $\hat{A}: \mathbb{F}^m \rightarrow \mathbb{F}$ of degree one in each variable.
Prop: Every function $A : H^m \to \mathbb{F}$ can be extended into polynomial $\hat{A} : \mathbb{F}^m \to \mathbb{F}$ of degree $|H| - 1$ in each variable

**Idea (contd.)**

- Think of $\phi : \{0, 1\}^{3m+3} \to \{0, 1\}$.
  - Typical clause $A(i_1) = b_1$ or $A(i_2) = b_2$ or $A(i_3) = b_3$.
  - Specified by $i_1, i_2, i_3 \in \{0, 1\}^m, b_1, b_2, b_3 \in \{0, 1\}$.
  - $\phi(i_1, i_2, i_3, b_1, b_2, b_3) = 1$ if clause in $\phi$ and 0 o.w.
- Extend $\phi$ into $\hat{\phi}$.

**Contrast with \#P scenario**

- $m$ now is $\log n$ ...
- Have an existential quantifier on $A$.
- Wanted to prove a sum condition on $\{0, 1\}^m$, now we have a “for all” condition
- Previously used sum on integers to convert “for all” to sum condition and then used CRT to reduce to finite field question. But this mizes badly with existential quantifier.
- Will redo proof ... that works.
Polynomial straightline program

- \( p_0 = \text{SAT} \) on \( m' \) variables.
- Will define \( p_1, \ldots, p_{m'} \) \( p_i \) defined by simple rule from \( p_{i-1} \). (i.e. can compute \( p_i \) with oracle access to \( p_{i-1} \).)
- Goal: If evolved correctly \( p_{m'} \equiv 0 \) in complete case, and \( \neq 0 \) in unsound case.
- \( p_i(y_1, \ldots, y_i, x_{i+1}, \ldots, x_{m'}) = p_{i-1}(y_1, \ldots, 0, x_{i+1}, \ldots, x_{m'}) + y_i p_{i-1}(y_1, \ldots, 1, x_{i+1}, \ldots, x_{m'}) \).
- Claim: \( p_{i-1} \) zero on \( \mathbb{F}^{i-1} \times \{0, 1\}^{n-i+1} \) iff \( p_i \) zero on \( \mathbb{F}^i \times \{0, 1\}^{n-i} \).

PCS problem instances

- New assignment \( p : \mathbb{F}^{m'+1} \rightarrow \mathbb{F} \) polynomial of degree \( 2m' + 1 \).
- Supposedly \( p(i, x) = p_i(x) \) and \( p(-1, y, z) = A(y) \). (Assume \( -1, 0, 1, \ldots, m' \) are distinct elements of field.)
- Constraints \( C_{i,x} : p_i(x_1, \ldots, x_m) = p_{i-1}(x_1, \ldots, x_{i-1}, 0, \ldots, x_m) + x_i p_{i-1}(x_1, \ldots, x_{i-1}, 0, \ldots, x_m) \) if \( i \in \{1, \ldots, m' \} \); \( p_i(x) = 0 \) if \( i = m'+1 \) and \( p_i(i_1, i_2, i_3, b_1, b_2, b_3) = \phi(...) \phi(-1(i_1) - b_1)...(p_{i-1}(i_3) - b_3) \) if \( i = 0 \).
- Constraint \( C_x = \bigwedge_i C_{i,x} \).

PCS problem instances

Analysis

Completeness: Following the rules leads to all constraints being satisfied.

Soundness:

- Take polynomial \( p : \mathbb{F}^{m'+1} \rightarrow \mathbb{F} \) and let \( A : \{0, 1\}^m \rightarrow \mathbb{F} \) be restriction of \( p \) to first variable \( = -1 \) and variables \( m+1, \ldots, m'+1 \) being set to 0.
- This assignment fails to satisfy some clause. So application of rules will lead to \( p_{m'} \) being mostly non-zero.
- Prover may cheat on some rule \( i \), but then \( C_{i,x} \) will be violated for most \( x \).
• No matter what $C_x$ is mostly unsatisfied.