Lecture 23:

Probabilistically Checkable Proof Systems:

Check proof in time sublinear in size of proof
Review:

\[ x \cdot y = \sum x_i y_i \quad \text{"inner product"} \]
\[ x \times y = (x_y - x_i y_j \ldots x_k y_k) \quad \text{"outer product"} \]

Fact: if \( a \neq b \), then \( \Pr_{\tilde{r}} [\tilde{a} \cdot \tilde{r} \neq \tilde{b} \cdot \tilde{r}] = \frac{1}{2} \), also true for \( = \mod 2 \).

if \( A \cdot B = C \), then \( \Pr_{\tilde{r}} [A \cdot B \cdot \tilde{r} \neq C \cdot \tilde{r}] \leq \frac{1}{2} \), proof via pairing argument e.g. lec 20 last page notes.

Self-correcting:

if \( f \) is \( \frac{\epsilon}{2} \)-close to linear:

\[ g(x) = \begin{cases} 0(\log \frac{1}{\epsilon}) \times \text{times} \newline \text{Pick } y \text{ randomly} \newline \text{answer } \leftarrow f(y) + f(x-y) \newline \text{Output } \text{most common answer} \end{cases} \]

then: \( \forall x, \Pr [g(x) = f(x)] \geq 1 - \frac{\epsilon}{2} \)

Self-testing:

Given \( f \):

\[ \begin{cases} 0(\frac{1}{\epsilon}) \times \text{times} \newline \text{Pick } x, y \text{ randomly} \newline \text{if } f(x) + f(y) \neq f(x+y), \text{ output fail and halt} \newline \text{Output pass} \end{cases} \]

If \( f \) linear, test passes
If \( f \) \( \epsilon \)-far from linear, \( \Pr [\text{test fails}] \geq \frac{3}{4} \)
Probabilistically Checkable Proofs

\[ \text{def. } L \in \text{PCP}(\Gamma, \delta) \text{ if } \exists V (\text{p-time TM}) \text{ s.t.} \]
\[ 1) \forall x \in L \exists \Pi \text{ s.t. } \Pr_{\text{random strings}} [V, \Pi \text{ accepts}] = 1 \]
\[ 2) \forall x \in \overline{L} \forall \Pi', \Pr_{\text{random strings}} [V, \Pi' \text{ accepts}] < \frac{1}{4} \]

where \( V \) uses at most \( r(n) \) random bits
\( + \) makes at most \( q(n) \) queries to \( \Pi \)
 1 bit each

\[ \exists q, r \text{ s.t. } \text{PCP}(r, q) \text{ is equivalent to } \text{NP} \text{.} \]

\[ \text{e.g. } \text{SAT} \in \text{PCP}(o(n), n) \]
\( \forall \text{ look at all settings of vars} \)

Today: \( \text{Thm} \quad \text{NP} \leq \text{PCP}(d(n^3), o(1)) \)

Actually: \( \text{Thm} \quad \text{NP} \leq \text{PCP}(o(\log n), o(1)) \)

How can it be?

Verifier doesn't get to see any significant portion of assignment? Why?
SAT: $F = \bigwedge C_i \text{ s.t. } C_i = (y_{i_1} v y_{i_2} v y_{i_3})$ where $y_{i_j} \in \{x_i, \overline{x_i}, x_{i+1}, \overline{x_{i+1}}\}$.

Is $F$ satisfiable? ← if so, how could you prove this?

A first crack:

$\Pi$: setting of sat assignment $a$

<table>
<thead>
<tr>
<th>$a_1$</th>
<th>$a_2$</th>
<th>$a_3$</th>
<th>$\vdots$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T$</td>
<td>$F$</td>
<td>$\vdots$</td>
<td></td>
</tr>
</tbody>
</table>

Protocol for $\Pi$:

Pick random clause $C_i$

check if setting $a$ satisfies $C_i$

Why good?

if $a$ satisfies $C_i$ then $\Pr[\Pi \text{ succeeds}] = 1$

Why bad?

if $a$ doesn't satisfy $C_i$

$\exists$ clause $i$ s.t. $a$ doesn't satisfy $C_i$.

So $\Pr[\Pi \text{ finds unsatisfiable clause}] \geq \frac{1}{m}$

Since $m = \#\text{clauses}$, $\frac{1}{m}$ could be very big.

This isn't so good. Need to repeat $O(m)$ time to find unsat clause.
Notation: \( x = (x_1, \ldots, x_n) \)
\( y = (y_1, \ldots, y_n) \)

3SAT:
\[
F = \bigwedge C_i \quad \text{\(i^{th}\) clause}
\]
\[
C_i = (y_{i_1} \lor y_{i_2} \lor \overline{y}_{i_3}) \quad \text{where} \quad y_{i_j} \in \{x_1, \ldots, x_n, \overline{x}_1, \ldots, \overline{x}_n\}
\]

"Arithmetization" of 3SAT:

boolean formula \( F \) \iff arithmetic formula \( A(F) \) over \( \mathbb{Z}_2 \)

\[
\begin{align*}
T & \iff 1 \\
F & \iff 0 \\
X_i & \iff X_i \\
\overline{X}_i & \iff 1 - X_i \\
\alpha \land \beta & \iff \alpha \cdot \beta \\
\alpha \lor \beta & \iff 1 - (1 - \alpha)(1 - \beta) \\
\alpha \lor \beta \lor \gamma & \iff 1 - (1 - \alpha)(1 - \beta)(1 - \gamma)
\end{align*}
\]

Examples:
\[
(x_1 \lor x_2) \land \overline{x}_3 \iff (1 - (1 - x_1)(1 - x_2))(1 - x_3)
\]
\[
x_1 \lor x_2 \lor x_3 \iff 1 - (1 - x_1)(1 - (1 - x_2))(1 - x_3) = 1 - (1 - x_1)(x_2)(1 - x_3)
\]

For \( a = (a_1, \ldots, a_n) \):
- \( F \) satisfied by \( a \) if \( A(a) = 1 \)
- \( F \) satisfiable iff \( A(F) = 1 \)

Consider \( C_0(x) = (\hat{\xi}_1(x), \hat{\xi}_2(x), \ldots) \)

- Won't arithmetize the whole formula, just each clause separately \( \Rightarrow \) low degree
- (oh by the way, take the complement)

\[ \text{(NOTE complements of each clause } \xi_i \text{ evaluate to 0 iff } x \text{ satisfies the clause)} \]

\[ \text{Note: each } \xi_i \text{ is degree 3 poly. in } x \text{ and verifier knows its coefficients!} \]
High level idea: special encoding of assignment

- proof "writes out" all linear funs of assignment
  - deg 2
  - deg 3

- possible "confusion": "symmetric" for linear case

\[ f_x(a) = x \cdot a = A_a(x) \]

inner product

- for deg 2, deg 3

\[ B_a(y) = (a \cdot a)^T \cdot y \]

\[ C_a(z) = (a \cdot a \cdot a)^T \cdot z \]

\[ A_a, B_a, C_a \] are all linear
  \[ \Rightarrow \] can test & self-correct

- are \( A_a, B_a, C_a \) consistent? (e.g. from same \( a \)?)

- Is "\( a \)" a sat assignment?
  (this is where we "win" over obvious encoding)

Idea of switching role of \( x \cdot a \) is important here!
Example

\[ G = (x_1 \lor x_2) \land (\bar{x}_1 \lor x_2) \]

\[ \bar{A}(C_1) = x_1 + x_2 - x_1 x_2 \quad \Rightarrow \quad A(G)(a) = 1 - a_1 - a_2 + a_1 a_2 \]

\[ \bar{A}(C_2) = 1 - x_1 + x_1 x_2 \quad \Rightarrow \quad A(G)(a) = -a_1 - a_2 a_1 \]

Evaluate at \( x = a \):

\[ \Sigma r_x C_x(a) = r_1 (1-a_1 = a_2 + a_1 a_2) + r_2 (a_1 = a_2) \]

\[ = (r_1 - r_2) \cdot 1 + (-r_1 + r_2) \cdot a_1 + (-r_1 - a_2) + (r_1 - r_2) a_1 a_2 \]

<table>
<thead>
<tr>
<th>( r_1 )</th>
<th>( r_2 )</th>
<th>( \Sigma r_x C_x(a) )</th>
<th>( \text{sat case} )</th>
<th>( \text{unsat case} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>( a_1 = a_2 )</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>( 1 - a_1 + -a_2 + a_1 a_2 )</td>
<td>( 1 \cdot 0 + 1 = 0 )</td>
<td>( 1 - 0 - 0 = 1 )</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>( 1 - a_2 )</td>
<td>( 1 - 1 = 0 )</td>
<td>( 1 - 0 = 1 )</td>
</tr>
</tbody>
</table>
Need to convince Verifier that $C(\bar{a}) = (0, 0, \ldots, 0)$ w/o sending a
How do you test if a vector is all 0?

"Weird idea:" assume $\exists$ little birdie who tells $V$ dot products of $C$ with random vectors (mod 2)

Fix a

$$
(\hat{c}_1(a), \ldots, \hat{c}_m(a)).(r_1 \ldots r_n) \equiv \sum r_x \hat{c}_x(a) \mod 2
$$

$$
\Pr[\sum r_x \hat{c}_x(a) = 0] = \begin{cases} 
1 & \text{if } \forall x. \hat{c}_x(a) = 0 \\
\frac{1}{2} & \text{otherwise} \text{ (exists } x \text{ s.t. } \hat{c}_x(a) \neq 0) 
\end{cases}
$$

$\Rightarrow$ different behavior when $C(a)$ is satisfied

Why? remember the pairing argument we did in lecture 20? (see last pg of notes)

But: Why believe the birdie? (e.g. birdie can just answer "0" all the time?)

does it help?

1) We know $r_x$'s

2) we know coeffs of polys of $\hat{c}_x$'s

3) $\hat{c}_x$'s have deg $\leq 3$ in $a_x$'s

$$
\sum r_x \hat{c}_x(a) = \prod \sum \hat{c}_x(a) x_i^x + \sum \hat{c}_x(a) B_{ij} + \sum \hat{c}_x(a) q_{i,j,k} (mod 2)
$$

from here on:

$a_x$ $\Rightarrow X_i$: no relation to variables of 3SAT

$p_{ij}$ $\Rightarrow q_{ij}$

$V_{ijk}$ $\Rightarrow Z_{ijk}$

$V$ does know these

- depend on $r_x$'s + coeffs of polys
- do not depend on $a_x$'s
- computed by $V$
- since working mod 2, all values are $\in \{0, 1, 3\}$
def A : \mathbb{F}_2^n \to \mathbb{F}_2 \\
A(x) = \sum_{i=1}^n a_i x_i = a^T x

V knows this

\[ B(y) = \sum_{i,j} a_{ij} y_i y_j = (a \circ a)^T y \]

outer product: if \( Z = b \odot c \)

\[ Z_{ij} = b_i \odot c_j \]

\[ C(z) = \sum_{i,j,k} a_{ijk} z_{ijk} = (a \circ a \circ a)^T z \]

\[ \text{supposed to be } A \odot B \odot C \]

\[ \text{but we need to check this} \]

Proof \( \Pi \) contains:

- Complete description of truth table of \( \bar{A}, \bar{B}, \bar{C} \) for all inputs \( x, y, z \)
  - we only need the values at one input!!
  - but this makes the checks a lot easier to do
  
  \[ \text{namely } x = \alpha, \ y = \beta, \ z = \gamma \]

What does verifier need to check in \( \Pi \)?

1. \( \bar{A}, \bar{B}, \bar{C} \) are of right form
   - all are linear functions
   - Correspond to same assignment \( a \)
     is, \( \bar{A}(x) = a^T x \Rightarrow \bar{B}(y) = (a \circ a)^T y \Rightarrow \bar{C}(z) = (a \circ a \circ a)^T z \)

2. \( \bar{a} \) is a set assignment
   - all \( \bar{C}_i \)'s evaluate to 0 on \( a \)
How to do (4):

- Test $\tilde{A}$, $\tilde{B}$, $\tilde{C}$ are all $\frac{1}{8}$-close to linear fits
  (i.e. Pass if linear, Fail if $\geq \frac{1}{8}$-far from linear)
  in $O(1)$ queries

  - random bits = $O(n^3)$
  - queries = $O(1)$
  - runtime = $O(n^3)$

- From now on, use self-corrector to get
  $sc^{-}A \cdot sc^{-}B \cdot sc^{-}C$
  for all inputs

  - use error parameter that is small enough to do union bound over all
  queries to $\tilde{A}$, $\tilde{B}$, $\tilde{C}$ (but will only be constant)

- Consistency test:
  Goal: Pass iff $sc^{-}A \cdot sc^{-}B \cdot sc^{-}C$

  $sc^{-}C = sc^{-}\tilde{A} \cdot sc^{-}\tilde{B}$

Outer Product Tester:

Pick random $x_1$, $x_2$, $y_1$

Test that $sc^{-}\tilde{A}(x_1) \cdot sc^{-}\tilde{A}(x_2) = \sum_{i} a_i x_i \cdot \sum_{j} a_j x_j$

$= \sum_{i,j} a_i a_j x_i x_j$

$= sc^{-}\tilde{B}(x_1 x_2)$

$sc^{-}\tilde{A}(x) \cdot sc^{-}\tilde{A}(y) = (\sum_{i} a_i x_i \cdot \sum_{j} a_j y_j) = \sum_{i,j} a_i a_j x_i y_j$

$= sc^{-}\tilde{C}(x \cdot y)$
Does it work?

Given, \( \text{sc-} A \uparrow \text{sc-} B \uparrow \text{sc-} C \) are linear maps

If \( b = a \circ a \)

\[ + c = a \circ a \circ a = a \circ b \]  
(by greeen argument on previous page)

Else, if \( b \neq a \circ a \)

\[ A(x) \cdot A(x) = B (x \circ x) \]

Fact: For all test if vectors are independent, then \( \Pr[A \cdot r + b \cdot r \neq C r] \geq \frac{1}{2} \).

If matrices \( A \cdot B \neq C \) then \( \Pr[A \cdot B \cdot r + C r] \leq \frac{1}{2} \).

Some proof as for "weird idea".

\[ \Rightarrow \Pr[\text{sc-} \circ \text{sc-} \circ \text{sc-}] \geq \frac{1}{2} \]

\[ \Rightarrow \Pr[x_1 \cdot (a \circ a) \cdot x_2 + b \cdot x_2 \neq \text{even}] \]

So test fails with prob \( \geq \frac{1}{4} \) !!
How to do (a):

- Recall we are making calls to self corerctor, so we are recovering linear forms $a, a^2, a^3, \ldots$

- We don't actually know $a$, but it represents the assignment $a$

- Is a satisfying? i.e. are all $\hat{c}_a(a) = 0$?

Satisfiability Test:

Pick $r \in K^*$

Compute $\Gamma, \alpha_i^j, \beta_i^j, \gamma_i, z_i, y_i, x_i^j$ ← terms of $r$ + coeff of poly from constraints

query proof to get

$SC - \hat{A}(\alpha_1, \ldots, \alpha_n) = w_0$

$SC - \hat{B}(\beta_1, \ldots, \beta_m) = w_1$

$SC - \hat{C}(\gamma_1, \ldots, \gamma_m) = w_2$

Verify $0 = \Gamma + w_0 + w_1 + w_2 \pmod{2}$

Hopefully means $\sum r_i \hat{c}_a(a) = 0$

Why does it work?

- If $\forall i, \hat{c}_a(a) = 0$ then pass with prob $1$

- If $\exists i$ s.t. $\hat{c}_a(a) \neq 0$ then $(0, \ldots, 0) \neq (\hat{c}_1(a), \ldots, \hat{c}_m(a))$

so $Pr[\sum r_i \hat{c}_a(a) = 0 \pmod{2} = 0 \cdot r_i] = \frac{1}{2}$

after $k$ time, pass all $k$ time with prob $\geq \frac{1}{2^k}$