CS 221: Computational Complexity	**Prof. Salil Vadhan**
Lecture Notes 21	
April 14, 2010	**Scribe: Brad Seiler**

Agenda

1. $\mathbf{NP} \subseteq \mathbf{PCP}(\text{poly}(n), O(1))$

Recap

$L \in \mathbf{PCP}_{c,s}(r(n), q(n))$ means that we have a PPT oracle algorithm V that has access to $r(n)$ coins and may read $q(n)$ bits from the proof oracle π, s.t.:

- Completeness: $x \in L \implies \exists \pi, \Pr_r[V^\pi(x;r) = 1] \geq c(n)$
- Soundness: $x \notin L \implies \forall \pi, \Pr_r[V^\pi(x;r) = 1] \leq s(n)$

Today, $c(n) = 1, s(n) = 1/2$

PCP Theorem

Last time we stated without proof:

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

We don't have time to give the full proof of the PCP theorem (it would take a couple of weeks), but instead will prove the following weaker version:

Theorem 2 (Easier PCP Theorem) $\mathbf{NP} \subseteq \cup_c \mathbf{PCP}(n^c, O(1))$ *(with exponential proof length).*

All known proofs of the full PCP theorem use this weaker PCP theorem as one of their building blocks.

Proof Sketch:

1. Work with **NP**-complete problem: Quadratic Equations over $\mathbb{Z}_2 = \text{GF}(2)$
2. **PCP** proof will be all quadratic functions of a satisfying assignment.
3. **PCP** verifier will:
   (a) Check that proof is "close" to a valid encoding of some assignment u.
   (b) Decode to a proper encoding with only $O(1)$ queries.
   (c) Verify that a random linear combination of the original system of equations is satisfied by u.

□

The Problem

Definition 3 QUADRATIC EQUATIONS *over* $\mathbb{Z}_2$. *Given a system of equations, each of the form*

$$\sum_{i<j} a_{ij} x_i x_j + \sum_i b_i x_i = c.$$

where all arithmetic is modulo 2, is there an assignment to the variables $\{x_i\}$ *satisfying all the equations?*

Claim 4 QUADRATIC EQUATIONS *over* $\mathbb{Z}_2$ *is* **NP**-*complete.*

> **Proof of claim:** Reduction from CIRCUIT SATISFIABILITY. For $C(x_1, \dots, x_n)$, introduce variables $x_{n+1}, \dots, x_m$ for the binary gates in C.
>
> $x_i = x_j \wedge x_k \mapsto x_i = x_j \cdot x_k$
>
> $x_i = \neg x_j \mapsto x_i = 1 - x_j$
>
> Add equation $x_m = 1$, where x_m is the output gate. □

Walsh-Hadamard Encoding

Linear Functionsa

Definition 5 *For* $u \in \{0,1\}^n = \mathbb{Z}_2^n$, *the* <u>*Walsh-Hadamard encoding*</u> *of* u, $\mathrm{WH}(u) \in \mathbb{Z}_2^{2^n}$, *consists of all* $\mathbb{Z}_2$*-linear functions of* u.
That is, for each $v \in \mathbb{Z}_2^n$, $\mathrm{WH}(u)_v = u \odot v = \sum_i u_i v_i \pmod 2 = u^T v$.
Equivalently,we can view $\mathrm{WH}(u)$ *as a function* $\mathrm{WH}(u) : \mathbb{Z}_2^n \to \mathbb{Z}_2$, *where* $\mathrm{WH}(u)[v] = u \odot v$. *That is,* $\mathrm{WH}(u)$ *is the linear function whose coefficients are given by* u.

Lemma 6 $\forall u_1 \neq u_2, \Pr_v[u_1 \odot v = u_2 \odot v] = 1/2$.

That is, WH is an error-correcting code with relative distance 1/2. This gives hope that we can distinguish satisfying assignments from non-satisfying ones with O(1) probes.

Quadratic Functions

Look at WH encoding of $u \otimes u \in \mathbb{Z}_2^{n^2}$ where $(u \otimes v)_{ij} = u_i v_j$, also can be considered as matrix uv^T, where the vectors are written as column vectors. Opposite of the inner product $\odot$.
Thus, $\mathrm{WH}(uu^T) \in \mathbb{Z}_2^{2^{n^2}}$ contains all homogenous quadratic functions of u. If $A \in \mathbb{Z}_2^{n \times n}$, then $\mathrm{WH}(uu^T)[A] = \sum_{i,j} A_{ij} u_i u_j$.

The PCP Proof Oracle

Given an instance of QUADRATIC EQUATIONS with n variables, our PCP oracle will consist of two functions $f : \mathbb{Z}_2^n \to \mathbb{Z}_2$ and $g : \mathbb{Z}_2^{n^2} \to \mathbb{Z}_2$ that are supposed to be $f = \mathrm{WH}(u)$ and $g = \mathrm{WH}(uu^T)$ for some satisfying assignment u. (However, we must prove soundness regardless of what functions (f, g), the verifier gets as oracle.)

Checking Closeness

Our goal is to test that (f, g) are "close" to $(\mathrm{WH}(u), \mathrm{WH}(uu^T))$ for some u. Define "close" by: f_1, f_2 are δ-close if $\Pr_x[f_1(x) = f_2(x)] \geq \delta$.

Linearity Testing

We need to test that $f : \mathbb{Z}_2^n \to \mathbb{Z}_2$ is δ-close to some $\mathrm{WH}(u)$, or some linear function on $\mathbb{Z}_2^n$.

Definition 7 (Blum–Luby–Rubinfeld Linearity Test) *Pick* $x, y \xleftarrow{R} \mathbb{Z}_2^n$*, check if* $f(x)+f(y) = f(x+y)$*. Repeat* $O(1)$ *times.*

Theorem 8 *The BLR Linearity Test satifisfies:*

- *Completeness: If* f *is linear, then* $\Pr_{x,y}[f(x) + f(y) = f(x + y)] = 1$.
- *Soundness: If* $\Pr_{x,y}[f(x) + f(y) = f(x + y)] \geq 1 - \delta$ *then* f *is* $(1 - O(\delta))$*-close to some linear function* $\tilde{f}$ *(i.e.* $\tilde{f} = \mathrm{WH}(u)$ *for some* u*).*

Another perspective is that the linearity test is a sublinear-time algorithm for the promise problem: This gives a sublinear algorithm for the promise problem:

$$\begin{aligned} \textsc{Test}_\varepsilon\textsc{Linearity}_Y &= \{f : f \text{ is linear}\} \\ \textsc{Test}_\varepsilon\textsc{Linearity}_N &= \{f : f \text{ is far from linear}\}. \end{aligned}$$

Note that the input length here is 2^n if f is a function from $\mathbb{Z}_2^n \to \mathbb{Z}_2$. However, the BLR linearity test just reads a constant number of bits from this input and runs in time $O(n)$.
"Property Testing" studies general algorithm problems of this type. On PS6, you will see an example of a property testing algorithm for a graph property.
For a proof of Linearity Testing, see next lecture.

PCP Verifier

Checking that f, g are close to linear

For small constant δ, PCP Verifier will run linearity test on f, g $O(1/\delta)$ times, to ensure that f, g are $(1 - \delta)$ close to some pair of linear functions.

Decoding them to a valid encoding

Claim 9 *Assuming* f, g *are* $(1 - \delta)$*-close to two linear functions* $(\tilde{f}, \tilde{g})$*, we can compute* $\tilde{f}, \tilde{g}$ *on any desired input with* $O(1)$ *probes to* f, g*, using random-self-reducibility of linear functions.*

Proof of claim: To compute $\tilde{f}(x)$, pick $y \leftarrow \mathbb{Z}_2^n$, output $f(x + y) - f(y)$. If $f = \tilde{f}$, a linear function, then this always works. But if f is $(1 - \delta)$-close to linear $\tilde{f}$, then $\forall x, \Pr_y[f(x + y) - f(y) \neq f(x)] \leq 2\delta$. (This works for g too.)
This allows a query to $\tilde{f}(x)$ on a arbitrary input x,even if $f(x) \neq \tilde{f}(x)$. □

From now on assume access to $\tilde{f}, \tilde{g}$.

Testing consistency of $\tilde{f}, \tilde{g}$

Claim 10 *Given oracle access to $\tilde{f}, \tilde{g}$, we can test that $\tilde{f} = \mathrm{WH}(u)$ and $\tilde{g} = \mathrm{WH}(uu^T)$ for some u. Since $\tilde{f}$ is linear, we are only checking that $\tilde{f}$ and $\tilde{g}$ use the same u.*

> **Proof of claim:** Choose a random $r, s \to \mathbb{Z}_2^n$ and check that $\tilde{f}(r)\tilde{f}(s) = \tilde{g}(rs^T)$.
>
> Completeness: If $\tilde{f} = \mathrm{WH}(u), \tilde{g} = \mathrm{WH}(uu^T)$, then $\tilde{g}(rs^T) = \sum_{i,j}(rs^T)_{ij}(uu^T)_{ij} = \sum_{i,j} r_i s_j u_i u_j = \sum_{i,j} r_i u_i s_j u_j = (r \odot u)(s \odot u) = \tilde{f}(r)\tilde{f}(s)$.
>
> Soundness: Suppose that $\tilde{f} = \mathrm{WH}(u)$ but $\tilde{g} = \mathrm{WH}(B), B \neq uu^T$. Applying Lemma 6 to a row on which B and uu^T differ, we have: $\Pr_s[Bs \neq (uu^T)s] \geq 1/2$. Furthermore, $\Pr_{r,s}[r^T Bs \neq r^T uu^T s] \geq 1/4$. Since $\tilde{g}(r,s) = r^T Bs$ and $\tilde{f}(r)\tilde{f}(s) = (r^T u)(u^T s)$, this proves soundness. □

Now assume that $(\tilde{f}, \tilde{g}) = (\mathrm{WH}(u), \mathrm{WH}(uu^T))$.

Testing that u satisfies the system

Claim 11 *We can test whether u satisfies a random linear combination of the quadratic equations to see if it satisfies the system.*

> **Proof of claim:** If u satisfies the system then it will satisfy any linear combination.
>
> If u does not satisfy the system, then it will fail to satisfy a random linear combination with probability $1/2$. □