$NP \subseteq PCP(\text{poly}(n), 1)$ from the Walsh-Hadamard Code.

Note Title 5/22/2013

(Source: Arora - Barak Section 11.5.)

Theorem: (Exponential-sized PCP system for NP)
$NP \subseteq PCP(\text{poly}(n), 1)$.

Verifier will expect prover to supply an encoded (i.e. expanded) encoding of the usual certificate for the problem in NP.
Here's a description of the encoding scheme.

Walsh-Hadamard Code:

encode binary strings of length n by linear functions in n variables over $GF(2)$ ← binary arithmetic mod 2.

$WH: \{0,1\}^* \longrightarrow \{0,1\}^*$

$u \longrightarrow$ truth table for fcn $f_u: x \to x \odot u = \sum_{i=1}^{n} x_i u_i$

$|u| = n \qquad |WH(u)| = 2^n$

↳ we will also refer to f_u as the string describing the truth table for function f_u.

if $f \in \{0,1\}^{2^n}$ is $= WH(u)$ for some u, then it is a Walsh-Hadamard code word.

Random Subsum Principle: if $u \neq v$ then for ½ of the x's $u \odot x \neq v \odot x$.

WH is an error-correcting code w/ min dist ½.
for every $u \neq v$ $WH(u)$ + $WH(v)$ differ in at least ½ of the bits.

The WH code words are the set of all linear functions

$f(x)+f(y)=f(x+y)\ \forall x,y \Rightarrow \exists u \quad f(x)=u\odot x \ \forall x$

↳ can determine f by $f(e_i)\ \forall i \quad e_i=(0\cdots 010\cdots 0)$ ← i^{th} location.

→ $f(e_i)=u_i$.

Given f, we want to test if it is $WH(u)$ for some u.
(i.e. we want to test if f is linear).

We want to test: $\forall x,y \quad \underbrace{f(x)+f(y)}_{GF(2).}=\underbrace{f(x+y)}_{\text{vector addition over } GF(2)^n}$.

We can't afford to do this test exhaustively. We want to be able to do the test with only a constant # of probes to f. ⇒ natural test: pick $x+y$ at random. and test $f(x)+f(y)=f(x+y)$.

Clearly if f really is linear then we will always accept.
However if f is very close to being linear, we could easily miss that!
With local tests, we can only hope to reject functions f that are far from linear.

Definition: Let $\rho\in[0,1]$ $f,g:\{0,1\}^n\to\{0,1\}$ are ρ-close if $\Pr_x[f(x)=g(x)]\geq\rho$.
f is ρ-close to a linear function if $\exists$ linear g s.t f & g are ρ-close.

Theorem: (linearity testing) [BLR]
Let $f:\{0,1\}^n\to\{0,1\}$ be such that
$\Pr_{x,y}[f(x)+f(y)=f(x+y)]\geq\rho$ for $\rho>1/2$.

Then f is ρ close to a linear function.

For $\delta\leq 1/2$
If f is not $(1-\delta)$ close to a linear fcn then the prob one probe fails to detect this is $(1-\delta)$.

Can repeat $(1/\delta)$ times to get the prob of failure to detect non-linearity $\leq 1/2$

Suppose that for some $\delta < 1/4$ $f: \{0,1\}^n \to \{0,1\}$ is $(1-\delta)$-close to some linear fcn $\bar{f}$.

$\bar{f}$ is uniquely determined by f because two linear fcns differ in at least $1/2$ of their bits.

Given x, we want to determine $\bar{f}(x)$ but only have access to f.

We could assume $\bar{f}(x) = f(x)$ but x is not randomly chosen and could be one of the places where f + $\bar{f}$ differ.

We want a randomized test that works for all x w/ some good probability.

The following test requires two calls to f:

1. pick $x' \in_R \{0,1\}^n$
2. $x'' = x' + x$
3. $y' = f(x')$ $\quad y'' = f(x'')$
4. Output $y' + y''$.

x'' + x' are distributed uniformly (although dependent). The probability that either is "bad" $\leq 2\delta$.

↳ $f \neq \bar{f}$ at that point.

$$\Pr[f(x') = \bar{f}(x') \text{ and } f(x'') = \bar{f}(x'')] > 1 - 2\delta.$$

this is known as the self-correction of the WH code.

Proof that NP $\subseteq$ PCP(poly(n), 1) :

Will show an NP-complete language in PCP(log(n), 1).

The NP-complete language will be QUADEQ:

= language of systems of quadratic equations over GF(2) that are satisfiable. Note GF(2) = {0, 1} with arithmetic mod 2.

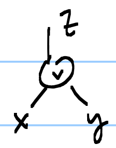
Here is an instance of QUADEQ:

$$u_1u_2 + u_3u_4 + u_1u_5 = 1.$$
$$u_2u_3 + u_1u_4 = 0.$$
$$u_1u_4 + u_3u_5 + u_3u_4 = 1$$

This system is satisfiable by the all 1's assignment.

Can show that Circuit-SAT $\propto$ QUADEQ.

Idea: have a variable for each wire



$\Rightarrow \quad (1-x)(1-y) = (1-z)$ etc.

... details omitted.

Since $u_i = u_i^2$, we can assume there are no linear terms.

Then m linear equations over n unknowns can be described by an $m \times n^2$ matrix A and an m-vector b.

The system is satisfiable if $\exists$ n^2-vector U which can be expressed as $u \otimes u$ for some n-vector u such that

$$AU = b.$$

We now describe a PCP system for QUADEQ

Let A, b be an instance of QUADEQ

Suppose (A, b) satisfiable by $u \in \{0,1\}^n$.
Verifier V gets access to a proof $\pi \in \{0,1\}^{2^n + 2^{n^2}}$
π is interpreted as a pair of functions:

$f: \{0,1\}^n \to \{0,1\}$ → WH encoding of u
$g: \{0,1\}^{n^2} \to \{0,1\}$. → WH encoding of $u \otimes u$.

So for u satisfying assignment, the verifier will accept the corresponding π w.p. 1.

Step 1: Check that f & g are linear functions.
Do a $(1-\epsilon)$ - linearity test on f & g.
If either f or g is not $(1-\epsilon)$-close to a linear function, the test fails w/ high probability.

$\Rightarrow$ assume $\exists$ linear $\tilde{f}: \{0,1\}^n \to \{0,1\}$
$\tilde{g}: \{0,1\}^{n^2} \to \{0,1\}$

ϵ : arbitrarily small constant.

f is $(1-\epsilon)$ close to $\tilde{f}$, g is $(1-\epsilon)$ close to $\tilde{g}$.
(In a correct proof, test passes and $f = \tilde{f}$, $g = \tilde{g}$).

We will assume in fact that we can query $\tilde{f}$ and $\tilde{g}$ directly at any point. This because local decoding allows us to recover any $\tilde{f}(x)$ with prob at least $(1-2\epsilon)$.

The number of queries to $\tilde{f}$ or $\tilde{g}$ in later steps is ~ 20.
So the prob that any of these queries fails $\leq 40\epsilon$.
We will assume that ϵ is small enough that all these queries succeed with prob $\geq .9$.

From now on: rename $\tilde{f}$ & $\tilde{g}$ to be f & g.
Furthermore, assume f & g are linear.

$f(x) = x \cdot u$ for some u. $\quad x, u \in \{0,1\}^n$
$g(y) = y \cdot w$ for some w. $\quad y, w \in \{0,1\}^{n^2}$

Step 2 Verify that $w = u \otimes u$.
Do the following test ten times:
pick $r + r'$ at random from $\{0,1\}^n$.
Verify $f(r) f(r') = g(r \otimes r')$. if not $\Rightarrow$ reject.

In a correct proof:

a correct proof is not rejected here.

$$f(r) f(r') = \left(\sum_{i \in [n]} u_i r_i\right)\left(\sum_{i \in [n]} u_i r_i'\right) = \sum_{i,j \in [n]} u_i u_j r_i r_j'$$
$$= (u \otimes u) \cdot (r \otimes r')$$
$$= g(r \otimes r').$$

Now suppose $w \neq u \otimes u$.
We claim that one test fails w.p. $\geq 1/4$.
Prob of rejecting at least one trial is $1 - (3/4)^{10} > .9$.

Let U be an $n \times n$ matrix: $U_{i,j} = u_i u_j$.

Let W be an $n \times n$ matrix $\quad W_{s \operatorname{div} n,\, s \bmod n} = w_s$.
representing w in the same way

$$g(r \otimes r') = w \cdot (r \otimes r') = \sum_{i,j \in [n]} \overset{W_{i,j}}{w_{i \cdot n + j}} r_i r_j' = r W r'$$

$$f(r) f(r') = (u \cdot r)(u \cdot r') = \left(\sum_{i=1}^{n} u_i r_i\right)\left(\sum_{j=1}^{n} u_j r_j'\right) = \sum_{i,j} u_i u_j r_i r_j'$$
$$= r U r'$$

V rejects if $rWr' \neq rUr'$.

Random Subsum principle: if $W \neq U$

Then at least $1/2$ of all r satisfy $rW \neq rU$

(let e be the column where W & U differ.

prob $\underbrace{r \cdot (j^{th} \text{ col of } U)}_{j^{th} \text{ bit of } rU} \neq \underbrace{r \cdot (j^{th} \text{ col of } W)}_{j^{th} \text{ bit of } rW.}$ w.p. $\geq 1/2$.

Now conditioning on $rW \neq rU$, the prob that a random r' has $rWr' \neq rUr'$ is $\geq 1/2$.

Trial rejects w.p. $\geq 1/4$.

Step 3: Verify that g encodes a satisfying assignment.

First show how to verify that the k^{th} equation is verified.

Check:

$$\sum_{ij} A_{k,(i,j)} u_i u_j = b_k$$

Let $z_k \in \{0,1\}^{n^2} = \underline{\underline{A_{k,(i,j)}}}$.

$\llcorner$ known to verifier.

$$\sum_{ij} A_{k,(i,j)} u_i u_j = g(z_k). \Rightarrow \text{test } g(z_k) = b_k.$$

But we can't check all $k \in \{1,\dots,m\}$.

We can check a random subset of the k's and use random subsum principle.

Pick random $r \in \{0,1\}^m$.

Compute $rA = z_r = \sum_{i=1}^{m} r_i z_i$ — i^{th} row of matrix A.

test if $g(z_r) = rb$.

Suppose $\exists k.\ g(z_k) \neq b_k$.

$$g(z_r) = g\left(\sum r_i z_i\right) = \sum_i r_i \cdot g(z_i)$$ this is $\neq$ $r \cdot b$

w.p. $\geq 1/2$.