NP in PCP(poly,1)

Monday, June 4, 2018 4:07 PM

We will prove a weaker version of the PCP theorem:

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

Verifier will expect the prover to supply an expanded encoding of the usual certificate for the problem in NP.

Walsh-Hadamard Code

Let $u \in \{0,1\}^n$ Define function $f_u : \{0,1\}^n \to \{0,1\}$

For $x \in \{0,1\}^n$ $f(x) = x \cdot u$

(dot product mod 2.)

For example
$$x = 1\ 0\ 1\ 1\ 0\ 1\ 1$$
$$u = 1\ 0\ 0\ 1\ 1\ 0\ 1$$
$$x \cdot u = 1+0+0+1+0+0+1 = 1.$$

One way to specify f_u is the input output table:

x	$f_u(x)$
0 0 ... 0	0
0 0 ... 1	1
⋮	⋮
1 1 ... 1	1

2^n rows: one for every possible input x.

The I/O table is a 2^n-bit string that specifies f_u. Call this string $WH(u)$

NP in PCP(poly,1)

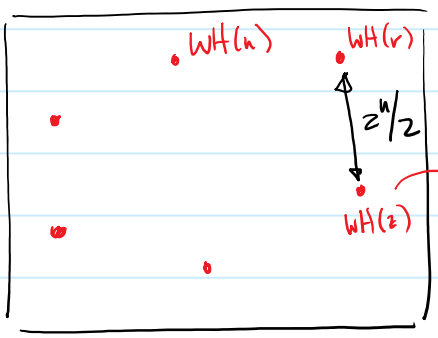
Monday, June 4, 2018 4:17 PM

If $|u|=n$ then $|WH(u)| = 2^n$.

If $f \in \{0,1\}^{2^n}$ is $WH(u)$ for some $u \in \{0,1\}^n$

then f is said to be a Walsh-Hadamard code word.

Note the # of Walsh-Hadamard code words of length 2^n is 2^n. (Each one corresponds to a $u \in \{0,1\}^n$)



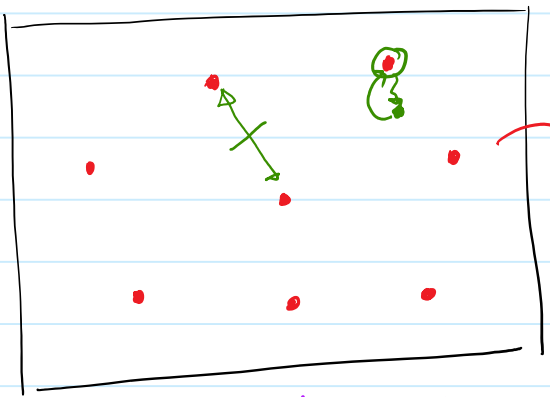
all 2^n-bit strings.
2^{2^n} strings in this set.

WH code-words.
Only 2^n of those.

In general the idea behind an error correcting code is to encode strings of length n as strings of length $d > n$

Code: $\{0,1\}^n \to \{0,1\}^d$

The range of function Code is the set of code words.



all 2^d strings.

only 2^n code words.
goal is to have them spaced far apart.

Can recover original string if a few bits get flipped in transmission.

NP in PCP(poly,1)

Monday, June 4, 2018 4:25 PM

Random Subset Principle:

If $u, v \in \{0,1\}^n$ and $u \neq v$ then for $1/2$ of all the x's in $\{0,1\}^n$ $\quad f_u(x) \neq f_v(x)$

This means that any two Walsh-Hadamard code words differ in $\frac{2^n}{2}$ locations.

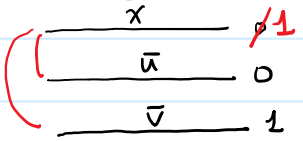
$u \neq v \Rightarrow$ distance between $WH(u)$ & $WH(v)$ is $2^n/2$.

Proof: $\quad u = \bar{u}\,0 \qquad \bar{u}, \bar{v} \in \{0,1\}^{n-1}$

$v = \bar{v}\,1$

Consider $\bar{x} \in \{0,1\}^{n-1}$

if $\quad \bar{u}\bar{x} \neq \bar{v}\bar{x} \quad$ then $\quad u \cdot (\bar{x}0) \neq v \cdot (\bar{x}0)$



if $\quad \bar{u}\bar{x} = \bar{v}\bar{x} \quad$ then $\quad u \cdot (\bar{x}1) \neq v \cdot (\bar{x}1)$

$1/2$ of x's $\quad u \cdot x = v \cdot x$

$u \cdot x \neq v \cdot x.$

NP in PCP(poly,1)

Tuesday, June 5, 2018 7:59 AM

The WH code words are the set of all linear functions:

$f(x) + f(y) = f(x+y)\ \forall x,y \iff \exists u\ \ f(x) = u \cdot x\ \ \forall x$

$\Leftarrow$ Suppose $f(x) = u \cdot x$

$$f(x) + f(y) = u \cdot x + u \cdot y = u \cdot (x+y) = f(x+y)$$

$\Rightarrow$ Suppose $f(x) + f(y) = f(x+y)\ \ \forall x, y.$

Let $e_i = 0 \ldots 0\ 1\ 0 \ldots 0$ (1 in i^{th} location.)

f is completely defined by $f(e_i)$ for $i = 1, \ldots, n$.

for example: $f(10110) = f(e_1) + f(e_3) + f(e_4)$

Construct u: i^{th} bit of u is $f(e_i)$

$\Rightarrow \quad f(e_i) = u \cdot e_i$ //

Verifier needs to:

1. Check that the proof is in the correct format (i.e. check that the proof is a W-H code word.)
2. Check that the proof is a correct witness for the input. (For example check that it encodes a satisfying assignment for ϕ).

NP in PCP(poly,1)

Tuesday, June 5, 2018 8:08 AM

Given f, we want to test that $f = WH(n)$ for some n.

Same as: test if $f(x) + f(y) = f(x+y)$ $\forall x, y$.

We can't do this exhaustively. We want to be able to make this check w/ a constant # of probes to f.

Natural test: Pick $x + y$ at random and test if $f(x) + f(y) = f(x+y)$ (requires 3 probes).

If f is linear, verifier will accept.

However if f is close to linear, we could easily miss that!

With local tests, we can only hope to reject functions that are far from linear.

Definition Let $\rho \in [0,1]$ $f, g : \{0,1\}^n \to \{0,1\}$ are ρ-close if $Pr_x [f(x) = g(x)] \geq \rho$

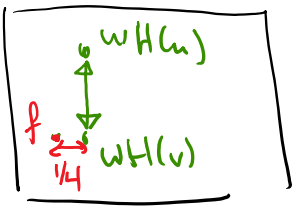
Theorem: (linearity testing) [BLR]
Let $f: \{0,1\}^n \to \{0,1\}$ be such that

$$Pr_{x,y} [f(x) + f(y) = f(x+y)] \geq \rho \text{ for } \rho > 1/2$$

Then f is ρ-close to a linear function.

NP in PCP(poly,1)

Thursday, May 31, 2018 9:10 AM



2^n-bit strings.

For $\delta < 1/2$ If f is not $(1-\delta)$-close to a linear function, the probability one fails to detect this is $(1-\delta)$.

$(1-\delta)^{1/\delta} \approx 1/e$.

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

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

$\bar{f}$ is uniquely determined by f because two distinct linear functions differ in $\geq 1/2$ of their bits.

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

x is not randomly chosen, so we can not assume that $f(x) = \bar{f}(x)$ x could be one of the locations where they differ.

We want a randomized procedure that uses f to produce the correct $\bar{f}(x)$ for all x, with high probability

↑ probability over random choices that the verifier controls.

NP in PCP(poly,1)

Thursday, May 31, 2018 9:10 AM

$\bar{f}(x) = h \cdot x$

The following requires two calls to f:

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

$f(x') + f(x'')$

$\bar{f}(x') + \bar{f}(x'')$

$\bar{f}(x' + x'')$

$\bar{f}(x' + x' + x) = \bar{f}(x)$

x'' and x' are distributed uniformly (but are not independent).

Prob that $f(x') \neq \bar{f}(x')$ OR $f(x'') \neq \bar{f}(x'') \leq 2\cdot\delta$.

$$\text{Prob}\left[f(x') = \bar{f}(x') \text{ AND } f(x'') = \bar{f}(x'') \right] > 1 - 2\delta.$$

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

Proof that $NP \subseteq PCP(\text{poly}(n), 1)$

We will show an NP-complete language in $PCP(\text{poly}(n), 2)$

The NP-complete language will be QUADEQ:
= language of systems of quadratic equations over GF(2) that are satisfiable ($GF(2) = \{0,1\}$, $+, \cdot \bmod 2$)

NP in PCP(poly,1)

Thursday, May 31, 2018 1:44 PM

Here is an instance of QUADEQ:

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

Satisfiable with the all 1's assignment.

Can show Circuit SAT $\propto$ QUADEQ.

Idea: have a variable for each wire:

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

$\Rightarrow$

$x + y - z - xy = 0$

... details omitted.

$z_n = 1$.

Since $u_i = (u_i)^2$, can assume that there are no linear terms.

NP in PCP(poly,1)

Tuesday, June 5, 2018 8:21 AM

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

$u_1u_3 + u_2u_2 = 0$

$\Rightarrow$ $u_2u_1 + u_1u_3 = 1$

$u_3u_3 + u_1u_1 = 0$

1,1	1,2	1,3	2,1	2,2	2,3	3,1	3,2	3,3
0	0	1	0	1	0	0	0	0
0	0	1	1	0	0	0	0	0
1	0	0	0	0	0	0	0	1

$b = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}$

$A \cdot \begin{pmatrix} u_1u_1 \\ \vdots \\ u_nu_n \end{pmatrix}$

The system is satisfiable iff $\exists$ n^2-vector V which can be expressed as $u \otimes u$ for some n-vector u such that: $AV = b$.

$u \otimes u = \langle u_1u_1, u_1u_2, \ldots, u_nu_n \rangle$

Are there u_1 u_2 u_3 s.t.

$V = \langle u_1u_1, u_1u_2, u_1u_3, u_2u_1, u_2u_2, u_2u_3, u_3u_1, u_3u_2, u_3u_3 \rangle$

$\vec{u} \otimes \vec{u}$

$A \cdot V = b$?

Overview - 1

Friday, June 8, 2018 8:19 AM

Now we give a PCP system for QUADEQ

Let A, b be an instance of QUADEQ.

Suppose (A, b) is satisfiable by $u \in \{0,1\}^n$

Verifier will expect an encoded version of $\vec{u}$ — n-bits

and an encoded version of $u \otimes u$. — n^2-bits.

Encoding scheme is Walsh-Hadamard encoding.

For $u \in \{0,1\}^n$ let $f_u : \{0,1\}^n \to \{0,1\}$

$f_u(x) = x \cdot u$.

$WH(u)$ is the 2^n-bit string indicating the value of f_u for every possible input.

$f : \{0,1\}^n \to \{0,1\} \iff 2^n$-bit strings.

Verifier expects: $WH(u)$ and $WH(u \otimes u)$.

$WH(u)$: 2^n bits.

$WH(u \otimes u)$: 2^{n^2} bits.

Overview - 2

Friday, June 8, 2018 8:30 AM

(1) Prover puts down 2^n-bit string $f: \{0,1\}^n \to \{0,1\}$.
2^{n^2}-bit string $g: \{0,1\}^{n^2} \to \{0,1\}$

Need to verify that f & g are $(1-\epsilon)$ close to some Walsh-Hadamard code words.

Verify $\exists\ u \in \{0,1\}^n$ f differs from $WH(u)$ in a fraction of $\leq \epsilon$ locations.

$\exists\ w \in \{0,1\}^{n^2}$ g differs from $WH(w)$ in a fraction of $\leq \epsilon$ locations.

(2) Verify that $w = u \otimes u$.

(3) Verify that u satisfies the instance of QUADEQ

$$A \cdot w = b.$$

NP in PCP(poly,1)

Thursday, May 31, 2018 1:44 PM

Now we give a PCP system for QUADEQ

Let A, b be an instance of QUADEQ.

Suppose (A,b) is satisfiable by $u \in \{0,1\}^n$

Verifier V gets access to 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 $u \otimes u$)

For u corresponding to a satisfying assignment the Verifier will accept the corresponding π w/ prob **1**.

Step 1: Check that f & g are linear functions.

Do a $(1-\epsilon)$-linearity test on $f+g$.

→ ϵ arb. small constant.

If either f or g is not $(1-\epsilon)$ close to a linear function then test fails w/ high probability.

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

$\tilde{g}: \{0,1\}^{n^2} \to \{0,1\}^n$

f is $(1-\epsilon)$ close to $\tilde{f}$ g is $(1-\epsilon)$-close to $\tilde{g}$.

(In a correct proof $f = \tilde{f}$ and $g = \tilde{g}$). $\hat{f}(x)$.

NP in PCP(poly,1)

Thursday, May 31, 2018 1:44 PM

We will assume that the verifier can query $\tilde{f}$ and $\tilde{g}$ directly. This is because of self-correction:
Can recover any $\tilde{f}(x)$ w.p. $\geq (1-2\epsilon)$.

The number of queries to $\tilde{f}$ or $\tilde{g}$ in later steps is ~ 30
So the prob that any query fails $\leq 60\epsilon$.
Assume ϵ is small enough that all queries succeed w.p. $\geq .9$.

Rename $\tilde{f}$ + $\tilde{g}$ to be f + g. 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 $w = u \otimes u$. $(u_1u_1, u_1u_2, \ldots, u_nu_n)$

Do the following 10 times:
Pick r + r' at random from $\{0,1\}^n$
Verify $f(r) \cdot f(r') = g(r \otimes r')$ If not → REJECT
$(r_1r_1', r_1r_2', \ldots, r_nr_n')$

In a correct proof:

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

$$= (u \otimes u) \cdot (r \otimes r') = g(r \otimes r')$$

NP in PCP(poly,1)

Thursday, May 31, 2018 1:44 PM

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 $U_{ij} = u_i u_j$

Let W be a $n \times n$ matrix $W_{s \text{ div } n,\ s \bmod n} = w_s$

$$g(r \otimes r') = W \cdot (r \otimes r') = \sum_{i,j} W_{in+j} \cdot r_i r'_j = r W r'$$

$$(r_1 \ldots r_i \ldots r_n) \begin{bmatrix} w_1 & w_2 & \cdots & w_n \\ w_{n+1} & w_{n+2} & & w_{2n} \\ \vdots & & & \\ w_{n^2-n+1} & & & w_{n^2} \end{bmatrix} \begin{bmatrix} r'_1 \\ r'_2 \\ \vdots \\ r'_j \\ \vdots \\ r'_n \end{bmatrix} = g(r \otimes r')$$

(w_{in+j} sits in row i, column j)

$$f(r) f(r') = (u \cdot r)(u \cdot r') = \sum_i u_i r_i \sum_j u_j r'_j = \sum_{i,j} u_i u_j r_i r'_j = r U r'$$

$$(r_1\ r_2 \ldots r_i \ldots r_n) \begin{bmatrix} u_1u_1 & u_1u_2 & \cdots & u_1u_n \\ u_2u_1 & u_2u_2 & \cdots & u_2u_n \\ \vdots & & & \\ u_nu_1 & \cdots & & u_nu_n \end{bmatrix} \begin{bmatrix} r'_1 \\ r'_2 \\ \vdots \\ r'_j \\ \vdots \\ r'_n \end{bmatrix} = f(r) f(r')$$

($u_i u_j$ sits in row i, column j)

V rejects if $rWr' \neq rUr'$

NP in PCP(poly,1)

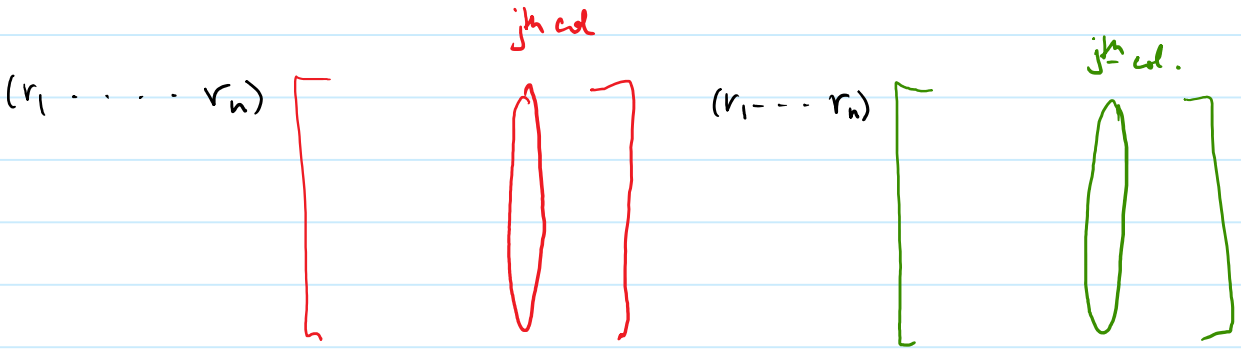
Thursday, May 31, 2018 1:44 PM

if $u \neq v$ then for at least $1/2$ of all possible x's.

$u \cdot x \neq v \cdot x$

Random Subsum Principle: If $W \neq U$ then $\geq 1/2$ of all r satisfy $rW \neq rU$

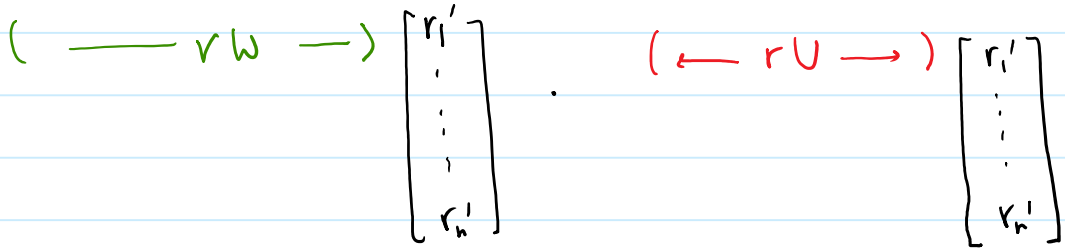
Let j be a column where W & U differ.



$r \cdot (\text{col } j \text{ of } W) \neq r \cdot (\text{col } j \text{ of } U)$ w.p. $\geq 1/2$

j^{th} bit of rW j^{th} bit of rU.

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



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

$\text{Prob}[rWr' \neq rUr'] \geq \text{Prob}[rW \neq rU] \cdot \text{Prob}[rWr' \neq rUr' \mid rW \neq rU]$

NP in PCP(poly,1)

Thursday, May 31, 2018 3:17 PM

Step 3: Verify that g encodes a satisfying assignment.

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

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

$g(z) = 2.\ (h \& u)$

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

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

But we can't check all $k \in \{1,\ldots,m\}$
We can check a random subset of the k's and use the random subset principle

Pick random $r \in \{0,1\}^m$
Compute $rA = z_r = \sum_{i=1}^{n} r_i A_{i,*}$ (i^{th} row of A)

$2^n + 2^{n^2}$

Test if $g(z_r) = rb$.

$\begin{pmatrix} g(z_1) \\ g(z_2) \\ \vdots \\ g(z_m) \end{pmatrix}$ $\begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{pmatrix}$

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

$$g(z_r) = g\left(\sum r_i A_{i*}\right) = \sum_i r_i \cdot g(z_i) \quad \text{this is } \neq r \cdot b$$

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