Walsh-Hadamard spectrum for Boolean Functions.

to compute efficiently Best affine approximations to $F = F(x_1, \dots, x_n)$.

$a = (a_1, \dots, a_n)$ $\quad a_i \in \{0,1\}$

$x = (x_1, \dots, x_n)$ $\quad$ variables $x_i \in \{0,1\}$

$\Rightarrow \quad a \cdot x = a_1 x_1 + \dots + a_n x_n$ mod 2.

---

two numbers

$N_{0,a} = \#\, x = (x_1 \dots x_n)$ s.t. $F(x_1 \dots x_n) = a \cdot x$

$N_{1,a} = \#\, x = (x_1 \dots x_n)$ s.t. $F(x_1 \dots x_n) \neq a \cdot x$ $\quad = ax + 1$

$\Rightarrow \quad N_{0,a} + N_{1,a} = 2^n$

---

probabilities

$p_a = Pr(F(x) = a \cdot x) = \frac{N_{0,a}}{2^n}$

$q_a = Pr(F(x) = a \cdot x + 1) = \frac{N_{1,a}}{2^n}$

$\Rightarrow \quad p_a + q_a = 1.$

Walsh-Hadamard spectrum for F

$W = (W_0, W_1, \dots, W_{2^n-1})$ ← rationals

$W_a = \frac{1}{2^n} \sum_{x \text{ over all } n\text{-bit strings}} (-1)^{F(x) + a \cdot x} = \frac{1}{2^n}(N_{0,a} - N_{1,a}) =$

$(-1)^{F(x)+a\cdot x} = \begin{cases} 1 & F(x) = ax \\ -1 & F(x) = ax + 1 \end{cases}$

$= \frac{2^n}{2^n} = 1$

$N_{0,a} + N_{1,a} = 2^n$

$$= \frac{2 \cdot N_{0,a} - 2^n}{2^n} = 2P_a - 1 \quad \text{By def. of } P_a$$

$$\Rightarrow \underbrace{P_a = \frac{1 + W_a}{2}}_{\text{Compute } W} \quad \text{for every } a = 0, 1, \ldots, 2^n - 1$$

in matrix form

$$(W_0 \ldots W_a \ldots W_{2^n-1}) = \frac{1}{2^n} \left((-1)^{F(0)} \ldots (-1)^{F(x)} \ldots (-1)^{F(2^n-1)} \right) \times$$

$$\times \underbrace{\begin{pmatrix} & a & \\ & \vdots & \\ \cdots & (-1)^{a \cdot x} & \cdots \end{pmatrix}}_{H_n \quad 2^n \times 2^n \text{ matrix}}$$

e.g. $n = 1$

$$H_1 = \begin{matrix} \\ 0 \\ 1 \end{matrix} \begin{matrix} \begin{matrix} 0 & \quad & 1 \end{matrix} \\ \begin{pmatrix} (-1)^{0 \cdot 0} & (-1)^{0 \cdot 1} \\ (-1)^{1 \cdot 0} & (-1)^{1 \cdot 1} \end{pmatrix} \end{matrix} = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

TR. $H_n = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}^{[n]} = \left\{ \begin{pmatrix} H_{n-1} & H_{n-1} \\ H_{n-1} & -H_{n-1} \end{pmatrix} \right\} 2^n$

(2^{n-1} | 2^{n-1} columns; 2^n rows)

computing W from its definition is slow

arith. op. is 2^{2n}.

faster algorithm in $n \cdot 2^n$ operations.

Fast Algorithm to compute W

input. Truth table of Boolean function

$F(x_1 \dots x_n)$ $\quad F(0), F(1), \dots, F(2^n-1)$

output W.-H. spectrum for F.

notation $W_{k,a}$ 2^k-string with integer entries

$0 \le k \le n$ and $0 \le a \le 2^{n-k}-1$.

$$W_k = \Big(\underbrace{W_{k,0}}_{2^k}, \underbrace{W_{k,1}}_{2^k}, \dots, \underbrace{W_{k,2^{n-k}-1}}_{2^k} \Big)$$

2^n

1. Initialize $W_{0,a} = (-1)^{F(a)}$, $0 \le a \le 2^n - 1$.

2. loop $0 \le k \le n-1$ $\quad W_k \to W_{k+1}$

$$W_{k+1,a} = \Big(\overbrace{W_{k,2a} + W_{k,2a+1}}^{2^{k+1}}, \underbrace{W_{k,2a} - W_{k,2a+1}}_{2^k} \Big)$$

$0 \le a \le 2^{n-k-1} - 1$

3. return $W = \dfrac{W_{n,0}}{2^n}$.

complexity. at each stage of the loop

2^k add. + 2^k subt. $= 2^{k+1}$

$0 \le a \le 2^{n-k-1} - 1$

$\Rightarrow \quad 2^{k+1} \cdot 2^{n-k-1} = 2^n$ arith. op.

n stages in the loop

overall $n \cdot 2^n$ operation. $<< 2^{2n}$

Example. $F(x_1 x_2 x_3) = (0\,0\,1\,1\,1\,0\,0\,1)$

$000 \; 001 \; \dots \; 111$

	0	1	2	3	4	5	6	7
W_0	1	1	−1	−1	−1	1	1	−1
W_1	(2	0)	(−2	0)	(0	−2)	(0	2)
W_2	(0	0	4	0)	(0	0	0	−4)
W_3	(0	0	4	−4	0	0	4	4)
W	0	0	$\frac{1}{2}$	$-\frac{1}{2}$	0	0	$\frac{1}{2}$	$\frac{1}{2}$

vector of probabilities $(P_0 P_1 \dots P_7)$

000	001	010	011	100	101	110	111
$\frac{1}{2}$	$\frac{1}{2}$	$\frac{3}{4}$	$\frac{1}{4}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{3}{4}$	$\frac{3}{4}$

$P_a = \frac{1+W_a}{2}$

$X_2,\ X_1+X_2,\ X_1+X_2+X_3$ have app. prob. $\frac{3}{4}$

have —11— $\frac{1}{4}$

X_2+X_3

but X_2+X_3+1 has prob. $1-\frac{1}{4}=\frac{3}{4}$.

best affine approximations to $F(X_1 X_2 X_3)$.

Rule how to compute best affine approximations.

1. compute W.-H. spectrum for F.
2. take W_a max. in absolute value.

set $g = \begin{cases} a\cdot X & \text{if } W_a > 0 \\ a\cdot X + 1 & W_a < 0 \end{cases}$

anyway, $Pr(F=g) = \frac{1+|W_a|}{2}$.

Complexity and Success probability of Affine Approximation Attack.

Bernoulli trials.

(*) $v_1, v_2, v_3, \ldots$ seq. of random variables.

1. $Pr(v_i = 0) = p$ ← failure
   $Pr(v_i = 1) = 1-p = q$ ← success
   for every $i = 1, 2, \ldots$

2. trials are independent

$$Pr(v_{i_1} = a_1, \ldots, v_{i_s} = a_s) = Pr(v_{i_1} = a_1) \cdot \ldots \cdot Pr(v_{i_s} = a_s)$$

any $i_1 \ldots i_s$
any values $a_1, \ldots a_s$

(*) called seq. of Bernoulli trials.

S_n = # successes in n trials.

$$= \sum_{i=1}^{n} v_i \quad \text{over integers}$$

$$= \text{weight}(v_1 \ldots v_n).$$

Fact. 1. Expected (mean) value of S_n is $n \cdot q$

$$E S_n = E\left(\sum_{i=1}^{n} v_i\right) = \sum_{i=1}^{n} E(v_i) = n \cdot q.$$

$$E(v_i) = 1 \cdot q + 0 \cdot p = q$$

Fact. 2. (de Moivre-Laplace limit theorem)

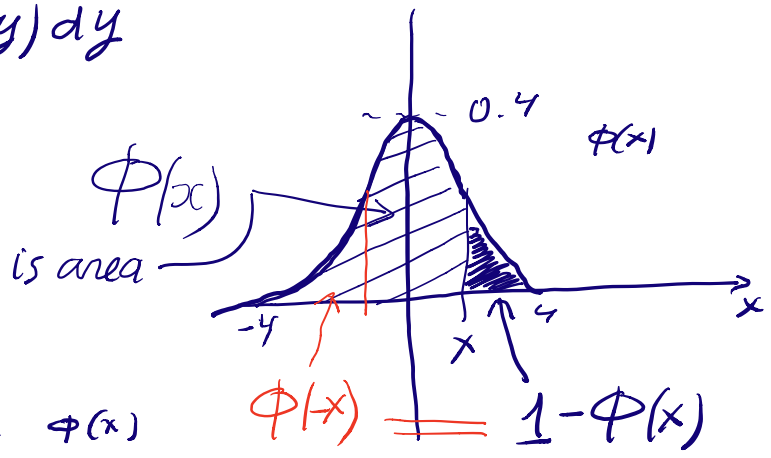
For every fixed real $a \le b$ (includes $\pm\infty$)

$$Pr\left(a \le \frac{S_n - n\cdot q}{\sqrt{n\cdot p\cdot q}} < b \right) \xrightarrow[n\to\infty]{} \Phi(b) - \Phi(a)$$

where $\Phi(x)$ standard normal distribution function.

$$\phi(x) = \frac{e^{-x^2/2}}{\sqrt{2\pi}}$$ standard normal density function

$$\Phi(x) = \int_{-\infty}^{x} \phi(y)\,dy$$



By symmetry of $\Phi(x)$

$$\Phi(-x) = 1 - \Phi(x)$$

Fact 3 Chernoff's bounds

$$Pr\left(|S_n - nq| > a \right) \le 2\cdot e^{-\frac{2a^2}{n}}$$

$$Pr\left(S_n - nq > a \right) \le e^{-2a^2/n}.$$

Back to Affine Approximation Attack.

system of lin. equations:

$$(*)\begin{cases} v_1 = u_1 + x_1 & n_1 \\ v_2 = u_2 + x_2 \\ \dots \\ v_n = u_n + x_n \end{cases}$$

v_i: new var. u_i: lin. functions in unknown initial state bits. x_i: key-stream bits known

$P_2(v_i = 1) = q$ rel. low as $P_2(v_i = 0) = p$ rel. high

By trying $v = (v_1 \dots v_n)$ of weight $\approx q \cdot n$

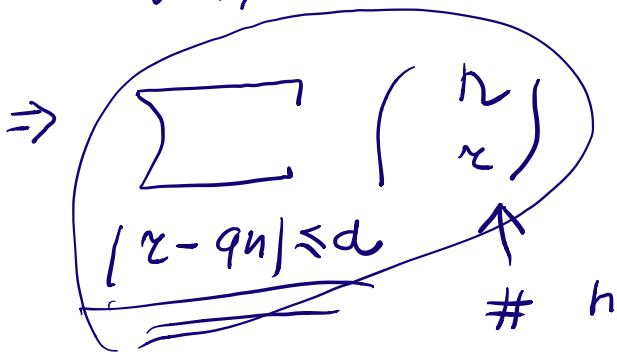
1) success: hit correct vector v $\Rightarrow$ solve $(*)$ and find initial state.

$$P_2(\text{success}) = P_2\left(\left|\text{weight}(\text{correct } v) - q \cdot n\right| \leq d\right)$$

d: parameter of the attack.

2) number of trials.

all binary n-strings for v of weight r s.t. $|r - q \cdot n| \leq d$

$\Rightarrow$ 

n-strings of weight r

$$\binom{n}{r} = \frac{n!}{r!\,(n-r)!}$$

Role of d.

take d small (close to 0)

$\Rightarrow P_2(\text{success})$ should be small

But # trials is small too.

take d large

$\Rightarrow$ $P_2(\text{success})$ is large ($\to 1$)

trials is large too.

trade-off between $P_2(\text{success})$ and # trials.

$$P_2(\text{failure}) = P_2(\,|\text{weight(correct } v) - q\cdot n| > d\,) = (*)$$

given ε find d s.t. $P_2(\text{failure}) = \varepsilon$.

$$\text{weight}(v) = \sum_{i=1}^{n} v_i = S_n$$

$(*) =$

$$P_2(\,|S_n - q\cdot n| > d\,) =$$

$d = t\cdot\sqrt{npq}$ we want to find t to find d.

$$= P_2\left(\left|\frac{S_n - qn}{\sqrt{npq}}\right| > t\right) \longrightarrow 2(1-\Phi(t))$$

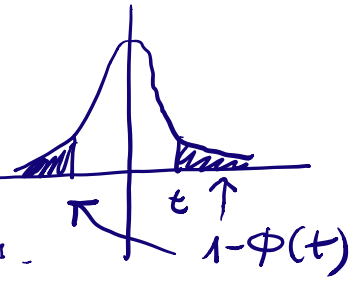
for large

$\Uparrow$

approximation

not very accurate

$q \to 0$ as $n \to \infty$.



solve equation

$$(**)\quad \varepsilon = 2(1-\Phi(t))$$

$\uparrow$

from text-books

computer packages

$t \Rightarrow d \Rightarrow$ complexity of the attack

trials.

Example. $\varepsilon = 0.01$ (failure 1%, success 99%)

solve equation $(**)$ and get $t = 2.61$

$\Rightarrow d = 2.61 \cdot \sqrt{n \cdot p \cdot q}$

try vectors $v = (v_1 \cdots v_n)$ of weight w s.t.

$$|w - qn| \leq 2.61\sqrt{npq}.$$

(correct $n = 200$ and $q > \frac{1}{20}$)

let $n = 200$, $q = \frac{1}{20}$, $\varepsilon = 0.01$

$\Rightarrow d \leq 9. \Rightarrow$

trials $\sum_{|w-10| \leq 9} \binom{200}{w} \approx \underline{1.9 \cdot 10^{26}}$

↑ complexity of the attack with success prob. 99%.

compare with brute force

trials $2^{200} \approx \underline{1.6 \cdot 10^{60}}$

conclude Affine Approx. is much faster.

reason $p = p_q = 1 - \frac{1}{20}$ very close to 1.