SHE

FHE → 바로사용 (아직 우회적 방법사용)

linearization ⇒ FHE 아니라, ring element 를 다시 줄여준다
↳ bootstraping 을 해야 FHE 이다.

- Vector 가 아닌 polynomial로 만들어서 (한방에 ?) 할수 있는 것.
- Cyclotomic polynomial $f(x) = x^n+1$ (n은 2^k 형태) → x^2+1, x^4+1, x^8+1, $x^{16}+1$
  ↳ 빠른 치환 위해 $\mathbb{Z}_q / \langle f(x) \rangle$ 이면 $x^n = -1$
  (덧셈, 곱셈 정의 가능해서 Ring 정의 가능

distribution 가우시안 에서 가장가까운 정수 여러개 뽑아서 e의 계수 (다항식) 설정

$(a_1, a_0) \Rightarrow (x' + ax^5 + -3x^2 + \ldots, \sim \sim \sim)$ 다항식 $a_1 \leftarrow R_q$ (uniform)

S가 noise에 영향 줘서 R_q에서 뽑았으나 X (가우시안 에서)
요즘은 binary 에서 뽑아서 noise 영향 ↓ 이고 빠르게함 (그래도 안전, n이 매우 크기 때문)

$f, g, u \leftarrow X$ 해서 $ct = (c_0, c_1) = (a_0u + tg + m, a_1u + tf)$
처럼
↳ 쓰는 이유는, 공개되있어 안전하지 않아 암호문 잘 숨기는 효과 (a_0, a_1 이 공개 돼있으니 $+fu$ 로 숨김)
→ $(c_0, c_1) \cdot (1, s)$ 로 Dec 도 된다.

* 효율성 측면

① $\left[\begin{array}{l} c_0 = (a_0, a_1, \ldots, a_n) \\ c_1 = (a_0', a_1', \ldots, a_n') \end{array} \right) \Rightarrow$ product (tensor 이기때문) $O(n^2)$

② $\left[\begin{array}{l} c_0 = \\ c_1 = \end{array} \right) \Rightarrow$ 또 $O(n^2)$ 이지만 FFT also 쓰면 $O(n)$ 이다. 그래서 RLWE 를 사용
↑ Cyclo 필요 (Famous !) ↳ ∴ 곱셈 빠름. $O(n)$

$\mathbb{Z}/\langle 3 \rangle = \mathbb{Z}/3\mathbb{Z}$ $\mathbb{Z} = \{0, 1, 2, \ldots, \}$, $3\mathbb{Z} = \{0, 3, 6, 9, \ldots\}$

$7 \equiv 1 \in \mathbb{Z}/\langle 3 \rangle$

$\therefore \left(\begin{array}{cccc} 0 & 3 & 6 & \cdots \\ 1 & 4 & 7 & \cdots \\ 2 & 5 & 8 & \cdots \end{array} \right)$ 같다고 생각한다.

ex) $\mathbb{Z}[x] / \langle x^4+1 \rangle = \mathbb{Z}[x] / (x^4+1)\mathbb{Z}[x]$

$= \{ a(x) \mid a(x) = (x^4+1) b(x)$ for some $b \in \mathbb{Z}[x] \}$

↳ $x^5+x+1 \Rightarrow x(x^4+1)+1 = 1$

$\mathbb{Z}[x]/\langle 3 \rangle$ ($7 = 3 \times 2 + 1 = 1$) 인것과 같은원리
↳ 3을 0으로 취급했으니 $x^4+1 = 0$ 으로 생각

RLWE ⇒ 요즘도 이 기반으로 사용 ∴ 이해좋음

시간표 확인

2월 15일 저녁 ! (칸)
이번주 workshop (아침 5~7시) 이후 링크 알려주심.
↳ 교수님은 목요일날 할 6시쯤 ?

Symbolic variable y는 무엇?

quasi

ex) $ct_0 = (ct_{00}, ct_{01}) \rightarrow \langle ct_0, sk \rangle \Rightarrow ct_{00} + ct_{01}s = (m_0(x) + te(x)) \bmod q$

$ct_1 = (ct_{10}, ct_{11}) \rightarrow \langle ct_1, sk \rangle \Rightarrow ct_{10} + ct_{11}s = (m_1(x) + te_1(x)) \bmod q$

↳ 곱하면 $(ct_{00} + ct_{01}s)(ct_{10} + ct_{11}s) = m_0(x)m_1(x) + t(e_{\sim})$

* 확장되면 $\left(\sum_{i=0}^{s} ct_{0,i}s^i\right)(\Sigma \sim) \Rightarrow$ 우리는 s를 모르니 x로 표현

$f_{Dec}(x) = \left(\sum_{i=0}^{s} ct_{0,i}x^i\right)(\sim)$ x에 s를 넣어줄때 Dec 가능

About Ring LWE Assumption

$q \geq 4(2t\sigma^2\sqrt{n})^{D+1} \cdot (2n)^{\frac{D}{2}}\sqrt{A} \Rightarrow$ mod q 에서 q가 만족하면 안전하다. (완화식)

↳ X에서 뽑고, e를 추출하고, 이것저것 잡동사니 해서 나온식.

* $\log_q t$ 쓰는 이유. (2 제곱 커서 error 너무 커서 곤란다 t-base로

• 합성곱

=> 두 함수를 곱해서 합함 $(f*g)(t) = \int_{-\infty}^{\infty} f(\tau)\, g(t-\tau)\, d\tau$ => g를 반전시키고, t만큼 이동 그후 f와 하나씩 곱함

- 어디 사용?

  └> 행렬의 합성곱 (DL), 신호필터, 라플라스 변환, 푸리에 변환

  → f, g 가 있는데 f는 행렬, 원래신호, 이미지 / g는 가중치, 필터 로 표현

  이것 f·g를 합성곱하면 우리의 목적 값 얻을수 있다.

- 행렬의 합성곱

$$\begin{pmatrix} 0&3&2&1\\ 2&1&0&2\\ 1&3&2&0\\ 2&0&3&3 \end{pmatrix} \times \begin{pmatrix} 1&0&1\\ 0&1&0\\ 1&0&1 \end{pmatrix} = \begin{pmatrix} 6&7\\ 10&8 \end{pmatrix}$$

입력 (4×4) × 필터 (3×3) ← 가중치 = 결과 (피처맵)

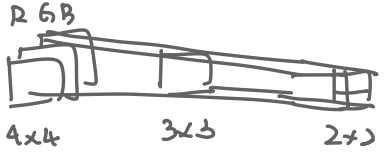
$0\times1 + 3\times0 + 2\times1$
$2\times0 + 1\times1 + 0\times0$ => 6
$1\times1 + 3\times0 + 2\times1$

$$\begin{pmatrix} 6&7\\ 10&8 \end{pmatrix} + 2 = \begin{pmatrix} 8&9\\ 12&10 \end{pmatrix}$$

편향 (bias)

이미지는 RGB 입력해서 (3차원 data) $\times \begin{pmatrix} 1&0&1\\ 0&1&0\\ 1&0&1 \end{pmatrix}$ => 각각더해서 $\begin{pmatrix} 12&15\\ 16&13 \end{pmatrix}$ 이됨.

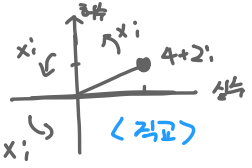
즉,

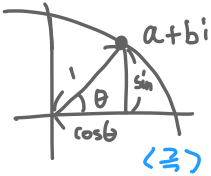


특징을 유지하며 이미지 사이즈 줄임

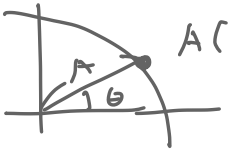
FFT (고속푸리에 변환)

- 복소 평면 (실수 - x / 허수 - y)





$\sqrt{a^2+b^2} = \sqrt{r^2(\cos^2\theta + \sin^2\theta)} = r$ =>



$A(\cos\theta + i\sin\theta)$

⇓

θ와 같이 A로 표현

- 오일러 공식 (벡터로 생각)

극좌표 $z = \cos\theta + i\sin\theta$ 를 미분 -> $\left(\frac{dz}{d\theta} = -\sin\theta + i\cos\theta\right) \times (-i)$ => $-\frac{dz}{d\theta}i = \cos\theta + i\sin\theta = z$

정리하면 $\frac{dz}{z} = \frac{1}{-i}d\theta = id\theta$ => 양변 적분 $\ln|z| = i\theta + C$

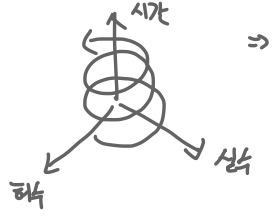
(오일러 공식에 i 넣고도 계산)

θ가 0이면 $z=1$ 임으로 $C=0$ 이다. 즉 $\ln|z| = i\theta$ 이다

즉 $z = e^{i\theta} = \cos\theta + i\sin\theta$

↳ $e^{\pi i} = -1$ -> $e^{\pi i} + 1 = 0$

- 복소평면 극좌표계 3D 표현



=> 시간 - 실수 평면에 투영 => $\cos\theta$

시간 - 허수 평면에 투영 => $\sin\theta$

* $\cos(2i+5)$, $\sin(2+2i)$ 이런것도 가능하다!

$e^{ix} = \cos x + i\sin x$

$e^{-ix} = \cos x - i\sin x$

$e^{ix} - e^{-ix} = 2i\sin x \Rightarrow \sin x = \dfrac{e^{ix} - e^{-ix}}{2i}$

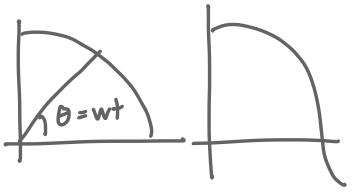
$\sin ix = \dfrac{e^{-x} - e^{x}}{2i}$

Same way $\Rightarrow \cos ix = \dfrac{e^{-x} + e^{x}}{2}$

• 푸리에 급수 (수열의 합)

$$\Rightarrow \text{①}\quad f(t) = \sum_{n=-\infty}^{\infty} C_n e^{j\frac{2\pi n}{T}t} = \sum_{n=-\infty}^{\infty} C_n e^{jn\omega t} = \sum_{n=-\infty}^{\infty} C_n (\cos(n\omega t) + j\sin(n\omega t))$$

(오일러 공식, 가중치)



$\theta = \omega t \quad \omega = \frac{2\pi}{T}$

$$\text{②}\quad C_n = \frac{1}{T}\int_{T/2}^{T/2} f(t) e^{-j\frac{2\pi n}{T}t}\, dt$$

↳ 이것으로 어떠한 주기함수든 표현할수 있다!

↳ 왜? 함수의 직교성을 이용 ⇒ 모든 점을 표현 하려면 축(dimension)의 개수를 무한히 늘린다.

↳ ex) $\vec{a}\cdot\vec{b} = 0$ 이고 2차원의 점임

$\vec{a}\cdot\vec{b}\cdot\vec{c} = 0$ 이고 3차원의 점임

푸리에 급수 = (계수 C_n) × (단순 주기함수 $e^{jn\omega t}$)

↳ 계수는 축, 주기함수와 직교성을 가지는 것을 증명한다? ⇒ 모든 주기함수 표현가능

↳ 생각할게.... 나 뇌야랑 거리 멀어

① 에 ② 대입 하고 극한을 취해준다. (T가 ∞이면 비주기함수도 주기함수로 볼수 있다)

$$\lim_{T\to\infty} f(t) = \lim_{T\to\infty} \sum_{n=-\infty}^{\infty} \frac{1}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} f(t)\, e^{-j\frac{2\pi n}{T}t}\, dt\, e^{j\frac{2\pi n}{T}t} \quad \text{치환하고 정리하면}$$

$$f(t) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(t)\, e^{-j2\pi\nu t}\, dt\, e^{j2\pi\nu t}\, d\nu$$

↳ 계수의 역할 이자 푸리에 변환 식이다.

- 푸리에 변환 vs 이산(고속) 푸리에 변환

=> 푸리에급수, 변환은 시간에 따라 연속이여야 함. 그러나 디지털 신호는 이산적이라 이산 푸리에 변환 사용
이산 푸리에 변환 (DFT)은 $O(N^2)$ 걸려서 고속푸리에 변환 (FFT) 사용 $O(N\log N)$ 걸림.
↳ 합성곱을 빨리함.

- 고속 푸리에 변환 원리 ($O(N\log N)$)

=> 전부 변환이 아니라 트징 신호만 골라서 변환시킴
=> N개를 홀, 짝 나누어 계산, ex) 100개 중 10개 골라내, 10개를 단순 연결하여 제외된 신호들 예상

- 이산 푸리에 변환 (DFT)

$x[k] = \sum_{n=0}^{N-1} x[n]\, e^{-j2\pi nk/T}$ (n은 순번, N: 총 표본 갯수) ($A_n = \sum_{j=0}^{N-1} e^{-2\pi ijn/N} a_j$ 이렇게도 가능)

↳ x[n]은 실수 x[k]는 복소수. 진폭과 위상이 다른 N개 성분을 모두 더해 원래 신호 얻음
(x[k]는 N개인데 중심값을 대칭으로 켤레이다. 그래서 모든합은 실수가 된다.

- 역변환 (IDFT)

$x[n] = \frac{1}{N}\sum_{k=0}^{N-1} x[k]\, e^{j2\pi nk/T}$ or $a_n = \sum_{j=0}^{N-1} \frac{1}{N} e^{2\pi ijn/N} A_j$

- 푸리에 변환은 Convolution 을 곱으로 바꿔줄 $(F(a*b) = F(a)\,F(b))$

  주기 N인 수열 $\{a_j\}_{j=0}^{N-1}$, $\{b_j\}_{j=0}^{N-1}$ 의 convolution $\{c_n\}_{n=0}^{N-1}$ $\Rightarrow$ $c_n = \sum_{j=0}^{N-1} a_j b_{n-j}$

  다른 관점으로 convolution에 DFT를 취하면 곱이됨 (증명 생략)

  $\left(b_{N-k} = b_{-k}\text{ 라 가정}\right.$
  $\left.\text{그래서 회식 성립한다}\right)$

* ($\{c_n\}_{n=0}^{N-1}$ 를 직접하면 $O(N^2)$ 이 걸리지만, 두수열에 DFT 취한후 곱하고 IDFT를 취하면 convolution 결과 얻을수 있다 $\Rightarrow$ FFT (주로 Cooley-Tukey Algo)
↳ $2^n = N$ 일때 가능하지만 만들면 됨

$$A_n = \Sigma\sim = \sum_{j=0}^{N/2-1} e^{-2\pi i(2j)n/N} a_{2j} + \sum_{j=0}^{N/2-1} e^{-2\pi i(2j+1)n/N} a_{2j+1}$$

$$= \sum_{j=0}^{N/2-1} e^{-2\pi ijn/(N/2)} a_{2j} + e^{-2\pi in/N} \sum_{j=0}^{N/2-1} e^{-2\pi ijn/(N/2)} a_{2j+1}$$

n 대신 $n+N/2$ 도 넣어도 마찬가지 $e^{-2\pi in/N}$ 만 부호 바뀐다 (2주는 1바퀴 간격)

$$\therefore A_{n+\frac{N}{2}} = \sum_{j=0}^{N/2-1} e^{-2\pi ijn/(N/2)} a_{2j} - e^{-2\pi in/N} \sum_{j=0}^{N/2-1} e^{-2\pi ijn/(N/2)} a_{2j+1}$$

↳ 짝수번째 DFT와 홀수번째 DFT를 알고있다면 전체 DFT 계산하는데 $O(N)$ 이 걸린다

$O(N\log N)$ 에 DFT 계산 가능

함수 표현 ⇒ (계수 / 점들) 3차함수 (ax^3+bx^2+cx+d / 점 4개 필요)

계수 ⟺ 점 (바꾸는 것 (시간) ⇒ FFT)

$A(x)$ = 2차함수

$B(x)$ = 2차함수

$C(x) = A(x) \cdot B(x)$ = 4차함수 (점 5개 필요)

⇒ $A(x)$ 에서 점 5개 $B(x)$ 에서 점 5개 뽑고 그것끼리 곱하서 $C(x)$ 만든다.

계수 $A(x), B(x)$ = $C(x)$ (coeff)

FFT $O(n\log n)$ ↓

점 $A(x) \cdot B(x)$ = $C(x)$ (value) ↑ $O(n\log n)$

$A(x) \cdot B(x)$ 는 $O(n)$

naive 하면 $O(n^2)$ 걸리는데

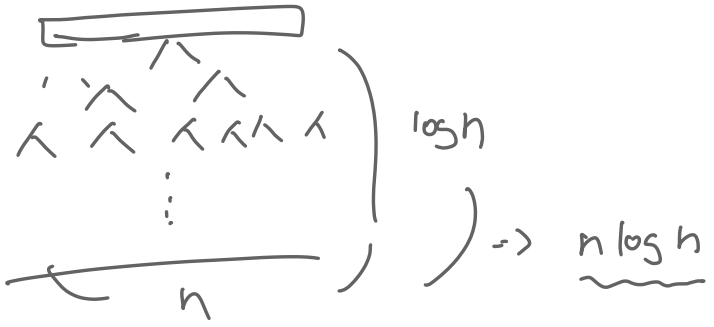
⇒ FFT 쓰면 $O(n\log n + n + n\log n) = O(n\log n)$

x 내림해

even, odd 나눠 계서 $\begin{pmatrix} p(x) = -p(-x) \\ p(x) = p(-x) \cdots \end{pmatrix}$ 같이해서!

복소평면 극좌표 이용? 하면 $x^n = 1$ 무조건 칼로 잘라 있어

짝 홀 - 문제는 해결 된다.



곡황이 이해

↳ $e^{i\pi} = -1$ 이기 때문에
$(e^{\frac{i\pi}{n}})^n = 1$ 이 된다. 즉 nth root of unity 이다.

nth root of unity $(w^j) \Rightarrow w = e^{\frac{2i\pi}{n}}$ / 곡황이 중요!

↳ 1의 n승 제곱근

FFT $P(x) : [p_0, p_1, \ldots, p_{n-1}]$, $\omega = e^{\frac{2\pi i}{n}} : [\omega^0, \omega^1, \ldots, \omega^{n-1}]$

$n = 1 \Rightarrow P(1)$

FFT $P_e(x^2) : [p_0, p_2, \ldots, p_{n-2}]$, $[\omega^0, \omega^2, \ldots, \omega^{n-2}]$

$y_e = [P_e(\omega^0), P_e(\omega^2), \ldots, P_e(\omega^{n-2})]$

FFT $P_o(x^2) : [p_1, p_3, \ldots, p_{n-1}]$, $[\omega^0, \omega^2, \ldots, \omega^{n-2}]$

$y_o = [P_o(\omega^0), P_o(\omega^2), \ldots, P_o(\omega^{n-2})]$

$P(\omega^j) = y_e[j] + \omega^j y_o[j])$

$P(\omega^{j+n/2}) = y_e[j] - \omega^j y_o[j])$

$j \in \{0, 1, \ldots (n/2 - 1)\}$

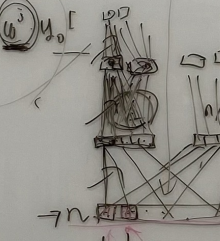
$y = [P(\omega^0), P(\omega^1), \ldots, P(\omega^{n-1})]$

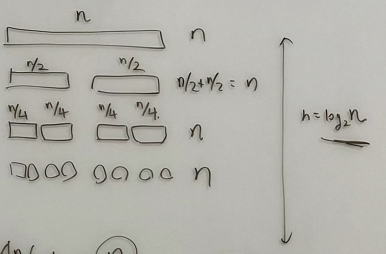
$y_e = [P_e(\omega^0), P_e(\omega^2), \cdots, P_e(\omega^{n-2})]$

$y_o = [P_o(\omega^0), P_o(\omega^2), \cdots, P_e(\omega^{n-2})]$

$P(\omega^j) = y_e[j] + \omega^j y_o[$

$P(\omega^{j+n/2}) =$





$(n/4)\cdot 4 = n$

$n/2 + n/2 = n$

$n \log n$

$n \Rightarrow n$

$f(x)$

$f(\omega^0)\ f(\omega^1)\ f(\omega^2) \cdots f(\omega^{n-1})$

$\{\omega^j\}$: roots of unity

$\omega^1 = e^{\frac{2\pi i}{n}}$: pm