NTT

https://youtu.be/nUghnBg6ijc

정보컴퓨터공학과 송경주

한성대학교
HANSUNG UNIVERSITY

CryptoCraft LAB

NTT?

- NTT: Number theoretic transform
  - 링 상에서 요소들의 곱셈을 효율적으로 수행
  - Dilithium: $\mathbb{Z}_q[X]/(X^n+1)$
  - finite field 상의 정수 곱셈을 수행할 때 효율적

Normal multiplication

- 길이 n의 두 정수 다항식 $\mathbf{a} = a_0 + a_1X^1 + a_2X^2 + \cdots,$ 에 대해 곱셈 수행 시
$\mathbf{b} = b_0 + b_1X^1 + b_2X^2 + \cdots,$

  $n \times n$ 만큼의 곱셈을 수행해야 함. → 계산 복잡도 $O(n^2)$: n 이 커질수록 복잡도↑

$$\boxed{\mathbf{c}_k = (\mathbf{a} \cdot \mathbf{b})_k = \sum_{i+j=k} a_i b_j}$$

Normal multiplication

- 길이 n의 두 정수 다항식 $\mathbf{a} = a_0 + a_1X^1 + a_2X^2 + \cdots,$ 에 대해 곱셈 수행 시
  $\mathbf{b} = b_0 + b_1X^1 + b_2X^2 + \cdots,$

  $n \times n$ 만큼의 곱셈을 수행해야 함. → 계산 복잡도 $O(n^2)$: n 이 커질수록 복잡도↑

$$\boxed{\mathbf{c}_k = (\mathbf{a} \cdot \mathbf{b})_k = \sum_{i+j=k} a_i b_j}$$

- NTT 연산 사용 시 위와 같은 조건의 곱셈에 대한 효율을 높여 줌.
  - NTT + point-wise multiplication + Inverse NTT → $O(\mathrm{n}\log n)$

Chinese remainder theorem (CRT)

- CRT: Chinese remainder theorem (중국 나머지 정리)

  ex) moduli m1=11, m2=23, N=11 x 23= 253

$$x \equiv 3 \bmod 11$$

$$x \equiv 21 \bmod 23$$

Chinese remainder theorem (CRT)

CRT: Chinese remainder theorem (중국 나머지 정리)

ex) moduli $m_1 = 11$, $m_2 = 23$, $\mathrm{N} = 11 \times 23 = 253$

- 다음 두 조건을 만족하는 x 찾기

$$x \equiv 3 \bmod 11$$
$$x \equiv 21 \bmod 23$$

- 두 조건을 만족하는 수: x = 113 mod 253

- 113 mod 256으로부터 기존 3, 21을 찾을 수 있음
  - 113 mod 11 = 3
  - 113 mod 23 = 21

We are going to take the second equation and iterate over the multiples of 23 to see if the first equation holds. Let's go:

$$0 \cdot 23 + 21 = 21 \stackrel{?}{\equiv} 3 \bmod 11 \qquad \text{Nope.}$$
$$1 \cdot 23 + 21 = 44 \stackrel{?}{\equiv} 3 \bmod 11 \qquad \text{Nope.}$$
$$2 \cdot 23 + 21 = 67 \stackrel{?}{\equiv} 3 \bmod 11 \qquad \text{Nope.}$$
$$3 \cdot 23 + 21 = 90 \stackrel{?}{\equiv} 3 \bmod 11 \qquad \text{Nope.}$$
$$\boxed{4 \cdot 23 + 21 = 113 \stackrel{?}{\equiv} 3 \bmod 11 \qquad \textbf{Yes!}}$$

Now we know $x = 113 \bmod 253$!

We can easily go back to the a_i values by just doing a modulo operation:

$$113 \bmod 11 = 3$$
$$113 \bmod 23 = 21$$

Chinese remainder theorem (CRT)

CRT: Chinese remainder theorem (중국 나머지 정리)

ex) moduli $m_1 = 11$, $m_2 = 23$, $\mathrm{N} = 11 \times 23 = 253$

- 다음 두 조건을 만족하는 x 찾기

$$x \equiv 3 \bmod 11$$
$$x \equiv 21 \bmod 23$$

We are going to take the second equation and iterate over the multiples of 23 to see if the first equation holds. Let's go:

$$0 \cdot 23 + 21 = 21 \overset{?}{\equiv} 3 \bmod 11 \quad \text{Nope.}$$
$$1 \cdot 23 + 21 = 44 \overset{?}{\equiv} 3 \bmod 11 \quad \text{Nope.}$$
$$2 \cdot 23 + 21 = 67 \overset{?}{\equiv} 3 \bmod 11 \quad \text{Nope.}$$
$$3 \cdot 23 + 21 = 90 \overset{?}{\equiv} 3 \bmod 11 \quad \text{Nope.}$$
$$\boxed{4 \cdot 23 + 21 = 113 \overset{?}{\equiv} 3 \bmod 11 \quad \textbf{Yes!}}$$

Now we know $x = 113 \bmod 253$!

We can easily go back to the a_i values by just doing a modulo operation:

$$113 \bmod 11 = 3$$
$$113 \bmod 23 = 21$$

- 두 조건을 만족하는 수: x = 113 mod 253

- 113 mod 256으로부터 기존 3, 21을 찾을 수 있음
  - 113 mod 11 = 3
  - 113 mod 23 = 21

- 위의 두 조건을 만족하는 x를 빠르게 찾기 위한 방법 → CRT

Chinese remainder theorem (CRT)

- <u>두 조건을 만족하는 x를 빠르게 찾기 위한 방법 → CRT</u>
- 큰 다항식을 CTR기반으로 나누어 계산

1. We have to be working in a ring.
2. m_i values may not be equal to 1.
3. All m_i values have to be coprime to one another.

- How?
  - 작은 수에 적용한 결과를 큰 수로 확장 가능

  Ex) 기존 두 모듈러 연산 값에 $\times \boxed{3}$

아까 찾은 두 조건을 만족하는 113에 대해 × 3 연산을 수행 시
같은 조건을 만족하는 수가 됨

$$x \equiv \boxed{3} \cdot 3 \equiv 9 \bmod 11$$
$$x \equiv \boxed{3} \cdot 21 \equiv 17 \bmod 23$$

$$\longrightarrow \boxed{3} \cdot 113 = 339 \equiv 86 \bmod 253$$

- RSA에 적용 → RSA의 경우와 마찬가지로 CRT를 사용하여 계수 곱셈의 수를 줄일 수 있다.
- How?
  - RSA 에서 사용되는 4096bit modulus $N = p \times q$ 에 대해 x를 mod N이 아닌 mod p, mod q 상에서의 상태로 만들어 연산 진행 : $a_1 \equiv x \bmod p$, $a_2 \equiv x \bmod q$. : 연산 대상이 x에서 더 작은 수인 a_1, a_2 로 변경됨
  - N은 4096-bit, p, q는 2048-bit 이므로 2048-bit 상에서의 연산으로 동작 → 훨씬 빠른 속도로 연산 가능

Number Theoretic Transform (NTT)

- RSA vs Dilithium in multiplication
  - 두 알고리즘 모두 많은 multiplication 수행
  - 차이점: RSA는 큰 정수의 곱, Dilithium은 ring을 사용한 곱셈

- Dilithium, Kyber에 대해서도 적용 가능 (= ring 곱셈에서도 사용 가능)

- 해당 링 $R = \mathbb{Z}_q[X]/(X^n + 1) = \mathbb{Z}_{17}[X]/(X^4 + 1)$ 상에서의 다항식 A, B 의 곱셈

$$\boxed{\mathbf{c}_k = (\mathbf{a} \cdot \mathbf{b})_k = \sum_{i+j=k} a_i b_j}$$

: 4×4 = 16 번의 곱셈 연산 수행

Number Theoretic Transform (NTT)

$$a = 2 + 7X^3 \longrightarrow \boxed{\mathbb{Z}_{17}[X]/(X^4+1)} \longrightarrow a = 2 + 7X^3$$

$$\mathbb{Z}_{17}[X]/(X^2-4) \qquad\qquad \mathbb{Z}_{17}[X]/(X^2+4)$$

$$\begin{aligned}
\mathbf{a} \bmod (X^2-4) &\equiv 2 + 7X^3 \\
&\equiv 2 + 7X^3 - 7X(X^2-4) \\
&\equiv 2 + (7\cdot 4)X \\
&\equiv 2 + 28X \\
&\equiv \boxed{2 + 11X}
\end{aligned}$$

$$\begin{aligned}
\mathbf{a} \bmod (X^2+4) &\equiv 2 + 7X^3 \\
&\equiv 2 + 7X^3 - 7X(X^2+4) \\
&\equiv 2 - (7\cdot 4)X \\
&\equiv 2 - 28X \\
&\equiv \boxed{2 + 6X}
\end{aligned}$$

$\vdots \qquad\qquad \vdots$

Number Theoretic Transform (NTT)

- NTT 동작
  1. 다항식 a, b를 degree-0 으로 나눈다.
  2. 나눠진 a_i, b_i에 대해 point-wise multiplication $c_i = a_i, b_i$ 을 수행
  3. CRT을 통해 c_i을 결합하여 c 생성

Number Theoretic Transform (NTT)

$R_q = \mathbb{Z}_q[X]/(X^{256}+1)$: Dilithium ring

- 두 다항식의 곱으로 나눔: $(X^{128}-\alpha)(X^{128}+\alpha) = (X^{256}+1)$
- 임의의 수 α 의 값은?

Number Theoretic Transform (NTT)

$R_q = \mathbb{Z}_q[X]/(X^{256}+1)$: Dilithium ring

- 두 다항식의 곱으로 나눔: $(X^{128}-\alpha)(X^{128}+\alpha) = (X^{256}+1)$
- 임의의 수 α 의 값은?

$$\boxed{(X^{256}+1) = (X^{128}-\alpha)(X^{128}+\alpha)}$$

$$(X^{256}+1) = X^{256} + (-\cancel{\alpha}+\cancel{\alpha})X^{128} - \alpha^2$$

$$1 = -\alpha^2$$

$$\boxed{\begin{aligned} \alpha^2 &= -1 \\ \alpha^4 &= 1 \end{aligned}}$$

$\alpha^4 = 1$
$\alpha = \sqrt[4]{1}$
→ $\alpha^2 = -1$ 을 만족 해야 하므로 $\alpha \neq 1$
→ 따라서 $\alpha^2, \alpha^3 \neq 1$

Number Theoretic Transform (NTT)

$$\boxed{(X^{128}+\alpha) = (X^{64}-\gamma)(X^{64}+\gamma)}$$
$$(X^{128}+\alpha) = X^{128} + (-\cancel{\gamma}+\cancel{\gamma})X^{64} - \gamma^2$$
$$\alpha = -\gamma^2$$

$$\boxed{\begin{aligned}\gamma &= \sqrt{-\alpha} = \sqrt{(-1)\cdot\alpha} \\ &= \sqrt{\alpha^2\cdot\alpha} = \left(\sqrt{\alpha}\right)^3\end{aligned}}$$

$$\boxed{(X^{128}-\alpha) = (X^{64}+\beta)(X^{64}-\beta)}$$
$$(X^{128}-\alpha) = X^{128} + (-\beta+\beta)X^{64} - \beta^2$$

$$\beta^2 = \alpha$$
$$\boxed{\beta = \sqrt{\alpha}}$$

- 위처럼 하위 레이어의 primitive root들을 모두 α로 표현 가능!
- 또한, $y_1 = \sqrt{x},\ \ y_2 = (\sqrt{x})^3$ 의 규칙을 가짐
- kth 1의 primitive root 를 $\zeta_k = \sqrt[k]{1}$ 로 표시
- 따라서 첫번째 layer: $\alpha = \zeta_4$
- 두번째 layer: $\beta = \sqrt{\alpha} \quad = \sqrt{\zeta_4} \quad = \zeta_8$

$$\gamma = \left(\sqrt{\alpha}\right)^3 = \left(\sqrt{\zeta_4}\right)^3 = \zeta_8^3$$

$$(X^{64}-\zeta_8)(X^{64}+\zeta_8)(X^{64}-\zeta_8^3)(X^{64}+\zeta_8^3)$$

$$(X^{32}-\zeta_{16})(X^{32}+\zeta_{16})(X^{32}-\zeta_{16}^5)(X^{32}+\zeta_{16}^5)(X^{32}-\zeta_{16}^3)(X^{32}+\zeta_{16}^3)(X^{32}-\zeta_{16}^7)(X^{32}+\zeta_{16}^7)$$

$$(X^{256}+1)$$
$$\vdots$$
$$(X-\zeta)(X+\zeta)(X-\zeta^{129})(X+\zeta^{129})\cdots(X-\zeta^{127})(X+\zeta^{127})(X-\zeta^{255})(X+\zeta^{255})$$

Number Theoretic Transform (NTT)

$$\mathbf{a} \in \mathbb{Z}_q[X]/(X^{256}+1)$$

$$\mathbf{a}_L^{(1)} \in \mathbb{Z}_q[X]/(X^{128}-\zeta^{128})$$

$$\mathbf{a}_R^{(1)} \in \mathbb{Z}_q[X]/(X^{128}+\zeta^{128})$$

$X^{128} = \zeta^{128}$, $a_i X^i = a_i \zeta^{128} X^{i-128}$

$X^{128} = -\zeta^{128}$, $a_i X^i = -a_i \zeta^{128} X^{i-128}$

$\mathbf{a}_L^{(1)} = (a_0 + \zeta^{128} a_{128}) + (a_1 + \zeta^{128} a_{129})X + (a_2 + \zeta^{128} a_{130})X^2 + \ldots$: a mod $(X^{128} - \zeta^{128})$

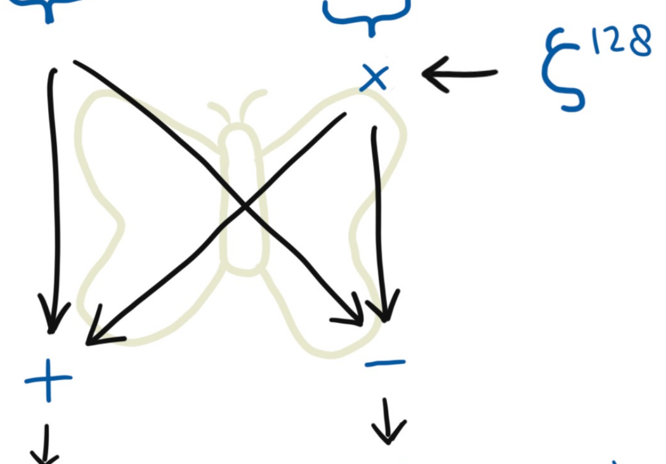
$\mathbf{a}_R^{(1)} = (a_0 - \zeta^{128} a_{128}) + (a_1 - \zeta^{128} a_{129})X + (a_2 - \zeta^{128} a_{130})X^2 + \ldots$: a mod $(X^{128} + \zeta^{128})$

Number Theoretic Transform (NTT)

$$\mathbf{a}_L^{(1)} = (a_0 + \zeta^{128} a_{128}) + (a_1 + \zeta^{128} a_{129})X + (a_2 + \zeta^{128} a_{130})X^2 + \ldots$$: a mod $(X^{128} - \zeta^{128})$

$$\mathbf{a}_R^{(1)} = (a_0 - \zeta^{128} a_{128}) + (a_1 - \zeta^{128} a_{129})X + (a_2 - \zeta^{128} a_{130})X^2 + \ldots$$: a mod $(X^{128} + \zeta^{128})$

기존 다항식 a mod $(X^{256}+1)$: $(a_0 + a_1 X + \cdots + a_{128} X^{128} + \cdots)$



a mod $(X^{128} - \zeta^{128})$ $(a^{(1)}_{L,0} + \cdots)$ $(a^{(1)}_{R,0} + \cdots)$ a mod $(X^{128} + \zeta^{128})$

⋮

$$(X-\zeta)(X+\zeta)(X-\zeta^{129})(X+\zeta^{129})\cdots(X-\zeta^{127})(X+\zeta^{127})(X-\zeta^{255})(X+\zeta^{255})$$

- n차 다항식에 대하여 각 layer tree로 변형하는데 $\log_2 n$ layer에 대해 ζ와 곱셈: $O(\log_2 n)$
- 최종 분해된 다항식의 n 개의 상수에 대해 n-point multiplication: $O(n)$

Q & A