A Full RNS Variant of Approximate Homomorphic Encryption

Jung Hee Cheon, Kyoohyung Han, Andrey Kim (Seoul National University)

Miran Kim (UTHealth), **Yongsoo Song** (UC San Diego)

SAC 2018

Residue Number System (a.k.a. CRT)

A Full RNS Variant of Approximate Homomorphic Encryption

Jung Hee Cheon, Kyoohyung Han, Andrey Kim (Seoul National University)

Miran Kim (UTHealth), **Yongsoo Song** (UC San Diego)

SAC 2018

Background

Secure Computation

- Differential Privacy
- (Secure) Multi-Party Computation
- (Fully) Homomorphic Encryption
  - Semantic security.
  - Non-interactive.
  - Reusable.
  - Long-term storage, Unlimited sources.

$$\begin{tikzcd}
ct \arrow[r, "Eval(f{,}\cdot)"] & Eval(f, ct) \arrow[d, "Dec"] \\
m \arrow[u, "Enc"] \arrow[r, "f(\cdot)"'] & f(m)
\end{tikzcd}$$

Landscape of HE Schemes

Scheme	Word Encryption	Bit Encryption	Approximate Encryption
Scheme (Library)			
Plaintext Space			
Operation			

Landscape of HE Schemes

Scheme	Word Encryption	Bit Encryption	Approximate Encryption
Scheme (Library)	BGV (HElib) B/FV (SEAL, NFLlib)		
Plaintext Space	Finite field + Packing		
Operation	Addition, Multiplication		

Landscape of HE Schemes

Scheme	Word Encryption	Bit Encryption	Approximate Encryption
Scheme (Library)	BGV (HElib) B/FV (SEAL, NFLlib)	FHEW, TFHE	
Plaintext Space	Finite field + Packing	Single Bit	
Operation	Addition, Multiplication	Binary Gate + Bootstrapping	

Landscape of HE Schemes

Scheme	Word Encryption	Bit Encryption	Approximate Encryption
Scheme (Library)	BGV (HElib) B/FV (SEAL, NFLlib)	FHEW, TFHE	HEAAN
Plaintext Space	Finite field + Packing	Single Bit	Real / Complex + Packing
Operation	Addition, Multiplication	Binary Gate + Bootstrapping	Addition, Multiplication, Rounding

Approximate HE (HEAAN, 慧眼)

- Design
  - Homomorphic Encryption for Arithmetic of Approximate Numbers [CKKS (AC'17)]
  - Bootstrapping [CHKKS (EC'18)]

Approximate HE (HEAAN, 慧眼)

- Design
  - Homomorphic Encryption for Arithmetic of Approximate Numbers [CKKS (AC'17)]
  - Bootstrapping [CHKKS (EC'18)]

- Applications in Machine Learning

Approximate HE (HEAAN, 慧眼)

Design

- Homomorphic Encryption for Arithmetic of Approximate Numbers [CKKS (AC'17)]
- Bootstrapping [CHKKS (EC'18)]

Applications in Machine Learning

- Training of Logistic Regression Model

  [KSW+ (JMI'18), KSK+ (iDASH'17, BMC'18), CKKS (IEEE Access'18)]

Approximate HE (HEAAN, 慧眼)

Design

- Homomorphic Encryption for Arithmetic of Approximate Numbers [CKKS (AC'17)]
- Bootstrapping [CHKKS (EC'18)]

Applications in Machine Learning

- Training of Logistic Regression Model

  [KSW+ (JMI'18), KSK+ (iDASH'17, BMC'18), CKKS (IEEE Access'18)]
- Matrix Computation & Evaluation of Neural Networks [JKLS (CCS'18)]

Approximate Computation

- Numerical Representation
  - $1.234 = 1234 \cdot 10^{-3}$.
  - Scaling factor $p = 10^3$.

Approximate Computation

- Numerical Representation
  - $1.234 = 1234 \cdot 10^{-3}$.
  - Scaling factor $p = 10^3$.

- Fixed-Point Arithmetic
  - 1.234×5.678

Approximate Computation

- Numerical Representation
  - $1.234 = 1234 \ \cdot 10^{-3}$.
  - Scaling factor $p = 10^3$.

- Fixed-Point Arithmetic
  - $1.234 \times \ 5.678 = (1234 \times 5678) \cdot 10^{-6}$

Approximate Computation

- Numerical Representation
  - $1.234 = 1234 \cdot 10^{-3}$.
  - Scaling factor $p = 10^3$.

- Fixed-Point Arithmetic
  - $$\begin{aligned} 1.234 \times 5.678 &= (1234 \times 5678) \cdot 10^{-6} \\ &= 7006652 \cdot 10^{-6} \end{aligned}$$

Approximate Computation

Numerical Representation

- $1.234 = 1234 \cdot 10^{-3}$.
- Scaling factor $p = 10^3$.

Fixed-Point Arithmetic

- $1.234 \times 5.678 = (1234 \times 5678) \cdot 10^{-6}$

$$= 7006652 \cdot 10^{-6} \mapsto 7007 \cdot 10^{-3} = 7.007.$$

Approximate Computation

Numerical Representation

- $1.234 = 1234 \cdot 10^{-3}$.
- Scaling factor $p = 10^3$.

Fixed-Point Arithmetic

- $$\begin{aligned} 1.234 \times 5.678 &= (1234 \times 5678) \cdot 10^{-6} \\ &= 7006652 \cdot 10^{-6} \mapsto 7007 \cdot 10^{-3} = 7.007. \end{aligned}$$
- Division by scaling factor p (a.k.a. Rounding operation).

(Leveled) Approximate HE

- Approximate Encoding / Encryption
  - (Ring) LWE-based.

(Leveled) Approximate HE

- Approximate Encoding / Encryption
  - (Ring) LWE-based.
  - $x \mapsto m = \lfloor p \cdot x \rceil$. p : scaling factor. m : significant digits of x.

$$ct = Enc_{sk}(m) \quad \Longrightarrow \quad \langle ct, sk \rangle \ (mod\ q_l) = m + e$$

(Leveled) Approximate HE

- Approximate Encoding / Encryption
  - (Ring) LWE-based.
  - $x \mapsto m = \lfloor p \cdot x \rceil$. p : scaling factor. m : significant digits of x.

$$ct = Enc_{sk}(m) \quad \Longrightarrow \quad \langle ct, sk \rangle \ (mod\ q_l) = m + e \ \approx p \cdot x.$$

(Leveled) Approximate HE

- Approximate Encoding / Encryption
  - (Ring) LWE-based.
  - $x \mapsto m = \lfloor p \cdot x \rceil$. p : scaling factor. m : significant digits of x.

$$ct = Enc_{sk}(m) \quad \Longrightarrow \quad \langle ct, sk \rangle \ (mod\ q_l) = m + e \ \approx p \cdot x.$$

- Approximate Homomorphic Operations
  - $Mult$: $\big(Enc(m_1), Enc(m_2)\big) \mapsto Enc(m \approx m_1 m_2 \approx p^2 \cdot x_1 x_2)$.

(Leveled) Approximate HE

Approximate Encoding / Encryption

- (Ring) LWE-based.
- $x \mapsto m = \lfloor p \cdot x \rceil$. p : scaling factor. m : significant digits of x.

$$ct = Enc_{sk}(m) \quad \Longrightarrow \quad \langle ct, sk \rangle \pmod{q_l} = m + e \approx p \cdot x.$$

Approximate Homomorphic Operations

- $Mult$: $\big(Enc(m_1), Enc(m_2)\big) \mapsto Enc(m \approx m_1 m_2 \approx p^2 \cdot x_1 x_2)$.
- $Round$: $Enc(m) \pmod{q_l} \mapsto Enc(m' \approx p^{-1} \cdot m) \pmod{q_{l-1}}$ for $p = q_l / q_{l-1}$.

(Leveled) Approximate HE

❑ Approximate Encoding / Encryption

- (Ring) LWE-based.
- $x \mapsto m = \lfloor p \cdot x \rceil$. p : scaling factor. m : significant digits of x.

$$ct = Enc_{sk}(m) \quad \Longrightarrow \quad \langle ct, sk \rangle \pmod{q_l} = m + e \approx p \cdot x.$$

❑ Approximate Homomorphic Operations

- $Mult$: $\big(Enc(m_1), Enc(m_2)\big) \mapsto Enc(m \approx m_1 m_2 \approx p^2 \cdot x_1 x_2)$.
- $Round$: $Enc(m) \pmod{q_l} \mapsto Enc(m' \approx p^{-1} \cdot m) \pmod{q_{l-1}}$ for $p = q_l / q_{l-1}$.
- Leveled Structure : $(q_L = p^L) > (q_{L-1} = p^{L-1}) > \cdots > (q_1 = p)$.

Main Result

Motivation

Ring structure $R_q = \mathbb{Z}_q[x]/(x^n + 1)$.

Expensive operation & High-precision library $(\log q = 250{\sim}800)$.

Motivation

Ring structure $R_q = \mathbb{Z}_q[x]/(x^n + 1)$.

Expensive operation & High-precision library $(\log q = 250{\sim}800)$.

Residue Number System (RNS) : $\mathbb{Z}_q \ \cong\ \mathbb{Z}_{p_1} \times \mathbb{Z}_{p_2} \times \cdots \times \mathbb{Z}_{p_L}$.

Motivation

Ring structure $R_q = \mathbb{Z}_q[x]/(x^n + 1)$.

Expensive operation & High-precision library ($\log q = 250{\sim}800$).

Residue Number System (RNS) : $\mathbb{Z}_q \cong \mathbb{Z}_{p_1} \times \mathbb{Z}_{p_2} \times \cdots \times \mathbb{Z}_{p_L}$.

Scheme	**Word Encryption**	**Approximate Encryption**
Representation	HElib (Double-CRT) [GHS12b]	
Homo. Operations	Full RNS B/FV Variants [BEHZ17, HPS18]	
Library	SEAL (v2.3)	

Motivation

Ring structure $R_q = \mathbb{Z}_q[x]/(x^n + 1)$.

Expensive operation & High-precision library ($\log q = 250{\sim}800$).

Residue Number System (RNS) : $\mathbb{Z}_q \cong \mathbb{Z}_{p_1} \times \mathbb{Z}_{p_2} \times \cdots \times \mathbb{Z}_{p_L}$.

Scheme	Word Encryption	Approximate Encryption
Representation	HElib (Double-CRT) [GHS12b]	This Work
Homo. Operations	Full RNS B/FV Variants [BEHZ17, HPS18]	
Library	SEAL (v2.3)	RNS HEAAN

Idea1 : Approx RNS Basis

- Rounding Operation
  - $Enc(m)\ (mod\ q_l) \longmapsto Enc\left(p_l^{-1} \cdot m\right)\ (mod\ q_{l-1})$ for $p_l = q_l/q_{l-1}$.

Idea1 : Approx RNS Basis

- Rounding Operation
  - $Enc(m)\ (mod\ q_l) \longmapsto Enc\left(p_l^{-1} \cdot m\right)\ (mod\ q_{l-1})$ for $p_l = q_l / q_{l-1}$.

What if we don't use the same $p = p_l$ for all l?

Idea1 : Approx RNS Basis

- Rounding Operation
  - $Enc(m)\ (mod\ q_l) \longmapsto Enc\left(p_l^{-1} \cdot m\right)\ (mod\ q_{l-1})$ for $p_l = q_l/q_{l-1}$.

What if we don't use the same $p = p_l$ for all l?

$q_L = p_1 p_2 \ldots p_L$ for approximate basis $p_l \approx p$.

$Enc\left(p_l^{-1} \cdot m\right) \approx Enc(p^{-1} \cdot m)$ (w/ approximation error)

Idea1 : Approx RNS Basis

- Rounding Operation
  - $Enc(m)\ (mod\ q_l) \mapsto Enc\left(p_l^{-1} \cdot m\right)\ (mod\ q_{l-1})$ for $p_l = q_l / q_{l-1}$.

What if we don't use the same $p = p_l$ for all l?

$q_L = p_1 p_2 \ldots p_L$ for approximate basis $p_l \approx p$.

$Enc\left(p_l^{-1} \cdot m\right) \approx Enc(p^{-1} \cdot m)$ (w/ approximation error)

$$R_{q_L} \cong R_{p_1} \times R_{p_2} \times \cdots \times R_{p_L} \text{ for } q_L = p_1 p_2 \ldots p_L.$$

Idea1 : Approx RNS Basis

- Polynomial Arithmetic
  - Number Theoretic Transformation (NTT): $R_{p_l} \to \mathbb{Z}_{p_l}^n$

Idea1 : Approx RNS Basis

- Polynomial Arithmetic
  - Number Theoretic Transformation (NTT): $R_{p_l} \rightarrow \mathbb{Z}_{p_l}^n$
  - Should be a prime number with $p_l \equiv 1 \ (mod\ 2n)$.

Idea1 : Approx RNS Basis

- Polynomial Arithmetic
  - Number Theoretic Transformation (NTT): $R_{p_l} \rightarrow \mathbb{Z}_{p_l}^n$
  - Should be a prime number with $p_l \equiv 1\ (mod\ 2n)$.

- Example ($p = 2^{55},\ n = 2^{15}$)

$$p_1 = 80000000080001, p_2 = 80000000130001, p_3 = 7FFFFFFE90001, \ldots$$

$$R_{p_1} \times R_{p_2} \times \cdots \times R_{p_L} \cong \mathbb{Z}_{p_1}^n \times \mathbb{Z}_{p_2}^n \times \ldots \times \mathbb{Z}_{p_L}^n.$$

Idea2 : Approx Modulus Switching

- Non-Polynomial Algorithms
  - Key-switching process (e.g. Homomorphic multiplication)

Idea2 : Approx Modulus Switching

- Non-Polynomial Algorithms
  - Key-switching process (e.g. Homomorphic multiplication)
  - Mod Raising : $R_{q_l} \rightarrow R_{\Delta \cdot q_l}, \quad a \mapsto a.$
  - Mod Reduction : $R_{\Delta \cdot q_l} \rightarrow R_{q_l}, \quad b \mapsto \lfloor b \,/\, \Delta \rceil = (b - [b]_\Delta)/\Delta.$

Idea2 : Approx Modulus Switching

- Non-Polynomial Algorithms
  - Key-switching process (e.g. Homomorphic multiplication)
  - Mod Raising : $R_{q_l} \to R_{\Delta \cdot q_l}$, $a \mapsto a.$
  - Mod Reduction : $R_{\Delta \cdot q_l} \to R_{q_l}$, $b \mapsto \lfloor b \,/\, \Delta \rfloor = (b - [b]_\Delta)/\Delta.$
  - $\mathrm{RNS}_{(p_1, p_2, \ldots, p_l)}(a) = (a_i)_{i \in [l]}.$

Idea2 : Approx Modulus Switching

- Non-Polynomial Algorithms
  - Key-switching process (e.g. Homomorphic multiplication)
  - Mod Raising : $R_{q_l} \rightarrow R_{\Delta \cdot q_l}, \quad a \mapsto a.$
  - Mod Reduction : $R_{\Delta \cdot q_l} \rightarrow R_{q_l}, \quad b \mapsto \lfloor b / \Delta \rceil = (b - [b]_\Delta)/\Delta.$
  - $\mathrm{RNS}_{(p_1, p_2, \ldots, p_l)}(a) = (a_i)_{i \in [l]}.$

Alternative algorithms without RNS conversions?

Idea2 : Approx Modulus Switching

$$\mathrm{RNS}_{p_i}^{-1}(a_i) \equiv \sum_i \left[a_i \cdot \widehat{p_i}^{\,-1}\right]_{p_i} \cdot \widehat{p_i} \pmod{q_l} \quad \text{for} \quad \widehat{p_i} = q_l/p_i.$$

$$\sum_i \left[a_i \cdot \widehat{p_i}^{\,-1}\right]_{p_i} \cdot \widehat{p_i} = q_l \cdot t + a \quad \text{for a small } t.$$

Idea2 : Approx Modulus Switching

$$\mathrm{RNS}_{p_i}^{-1}(a_i) \equiv \sum_i \left[a_i \cdot \widehat{p_i}^{-1}\right]_{p_i} \cdot \widehat{p_i} \ (mod\ q_l) \quad \text{for} \quad \widehat{p_i} = q_l / p_i.$$

$$\sum_i \left[a_i \cdot \widehat{p_i}^{-1}\right]_{p_i} \cdot \widehat{p_i} = q_l \cdot t + a \quad \text{for a small } t.$$

- Our Approx Mod Raising Algorithm (from q_l to $\Delta \cdot q_l$)

$$R_{p_1} \times \cdots \times R_{p_l} \to R_{p_1} \times \cdots \times R_{p_l} \times R_{\Delta_1} \times \cdots \times R_{\Delta_k},$$

Idea2 : Approx Modulus Switching

$$\mathrm{RNS}_{p_i}^{-1}(a_i) \equiv \sum_i \left[a_i \cdot \widehat{p_i}^{-1}\right]_{p_i} \cdot \widehat{p_i} \ (mod\ q_l) \quad \text{for} \quad \widehat{p_i} = q_l/p_i.$$

$$\sum_i \left[a_i \cdot \widehat{p_i}^{-1}\right]_{p_i} \cdot \widehat{p_i} = q_l \cdot t + a \quad \text{for a small } t.$$

- Our Approx Mod Raising Algorithm (from q_l to $\Delta \cdot q_l$)

$$R_{p_1} \times \cdots \times R_{p_l} \to R_{p_1} \times \cdots \times R_{p_l} \times R_{\Delta_1} \times \cdots \times R_{\Delta_k},$$

$$(a_1, \ \ldots \ , \ a_l) \mapsto \left((a_1, \ldots, a_l), \ (b_1, \ldots, b_k)\right)$$

$$b_j = \sum_i \left[a_i \cdot \widehat{p_i}^{-1}\right]_{p_i} \cdot \widehat{p_i} \ (mod\ \Delta_j).$$

Idea2 : Approx Modulus Switching

$$\mathrm{RNS}_{p_i}^{-1}(a_i) \equiv \sum_i \left[a_i \cdot \widehat{p_i}^{-1}\right]_{p_i} \cdot \widehat{p_i} \ (mod\ q_l) \quad \text{for} \quad \widehat{p_i} = q_l/p_i.$$

$$\sum_i \left[a_i \cdot \widehat{p_i}^{-1}\right]_{p_i} \cdot \widehat{p_i} = q_l \cdot t + a \quad \text{for a small } t.$$

- Our Approx Mod Raising Algorithm (from q_l to $\Delta \cdot q_l$)

$$R_{p_1} \times \cdots \times R_{p_l} \to R_{p_1} \times \cdots \times R_{p_l} \times R_{\Delta_1} \times \cdots \times R_{\Delta_k},$$

$$(a_1, \ \ldots \ , \ a_l) \mapsto \big((a_1, \ldots, a_l), \ (b_1, \ldots, b_k)\big) = \mathrm{RNS}_{p_i, \Delta_j}(q_l \cdot t + a).$$

$$b_j = \sum_i \left[a_i \cdot \widehat{p_i}^{-1}\right]_{p_i} \cdot \widehat{p_i} \ \ (mod\ \Delta_j).$$

Idea2 : Approx Modulus Switching

$$\mathrm{RNS}_{p_i}^{-1}(a_i) \equiv \textstyle\sum_i \left[a_i \cdot \widehat{p_i}^{-1}\right]_{p_i} \cdot \widehat{p_i} \ (mod\ q_l) \quad \text{for} \quad \widehat{p_i} = q_l / p_i.$$

$$\textstyle\sum_i \left[a_i \cdot \widehat{p_i}^{-1}\right]_{p_i} \cdot \widehat{p_i} = q_l \cdot t + a \quad \text{for a small } t.$$

- Our Approx Mod Raising Algorithm (from q_l to $\Delta \cdot q_l$)

$$R_{p_1} \times \cdots \times R_{p_l} \rightarrow R_{p_1} \times \cdots \times R_{p_l} \times R_{\Delta_1} \times \cdots \times R_{\Delta_k},$$

$$(a_1, \ \ldots \ , a_l) \mapsto \left((a_1, \ldots, a_l), \ (b_1, \ldots, b_k)\right) = \mathrm{RNS}_{p_i, \Delta_j}(q_l \cdot t + a).$$

$$b_j = \textstyle\sum_i \left[a_i \cdot \widehat{p_i}^{-1}\right]_{p_i} \cdot \widehat{p_i} \ \ (mod\ \Delta_j).$$

RNS Friendly Computation & Correctness of Homo Operations

(w/ additional noise)

Summary

- Idea 1: Approximate Basis
  - $q_L = p_1 p_2 \dots p_L$ with $p_L \approx p$ for RNS decomposition.
  - Approximate error ($p_l^{-1}m \approx p^{-1}m$) of the Rounding algorithm.

Summary

- Idea 1: Approximate Basis
  - $q_L = p_1 p_2 \dots p_L$ with $p_L \approx p$ for RNS decomposition.
  - Approximate error ($p_l^{-1} m \approx p^{-1} m$) of the Rounding algorithm.

- Idea 2: Full-RNS Variant
  - Approximate modulus-switching algorithms $R_{q_l} \leftrightarrow R_{\Delta \cdot q_l}$.
  - Additional noise.

Summary

- Idea 1: Approximate Basis
  - $q_L = p_1 p_2 \dots p_L$ with $p_L \approx p$ for RNS decomposition.
  - Approximate error ($p_l^{-1} m \approx p^{-1} m$) of the Rounding algorithm.

- Idea 2: Full-RNS Variant
  - Approximate modulus-switching algorithms $R_{q_l} \leftrightarrow R_{\Delta \cdot q_l}$.
  - Additional noise.

Efficiency & Convenience of Implementation (GMP, NTL free)

Summary

- Idea 1: Approximate Basis
  - $q_L = p_1 p_2 \dots p_L$ with $p_L \approx p$ for RNS decomposition.
  - Approximate error ($p_l^{-1} m \approx p^{-1} m$) of the Rounding algorithm.

- Idea 2: Full-RNS Variant
  - Approximate modulus-switching algorithms $R_{q_l} \leftrightarrow R_{\Delta \cdot q_l}$.
  - Additional noise.

Efficiency & Convenience of Implementation (GMP, NTL free)

vs Precision loss of computation

HEAAN vs RNS HEAAN

Variant	L	N	$\log q$	$\lceil \log Q_L \rceil$	Enc (ms)	Dec (ms)	Add (ms)	Cmult (ms)	Mult&RS (ms)
HEAAN	5	2^{15}	55	336	332	106	30	204	740
	10	2^{15}		611	530	135	32	281	1,355
	15	2^{16}		886	1,465	344	70	762	4,169
HEAAN-RNS	5	2^{15}	55	336	31	4.6	2.9	25	85
	10	2^{15}		611	58	7.8	4.3	44	164
	15	2^{16}		886	177	10.0	15.5	125	563

- 8x ~ 12x speed up

HEAAN vs RNS HEAAN

Function	N	$\log q$	$\log p$	Consumed levels	Input precision	Total time	Amortized time
x^{16}	2^{13}	155	30	4	14 bits	0.31s	0.07ms
x^{-1}						0.45s	0.11ms
$\exp(x)$						0.65s	0.16ms
x^{1024}	2^{15}	620	56	10	36 bits	7.46s	0.43ms

HEAAN
- 14 bits precision

Function	Degree	N	$\log q$	$\lceil \log Q_L \rceil$	Total time	Amortized time
x^{-1}	15	2^{14}	55	281	167ms	21μs
$\exp x$	7	2^{14}	55	281	164ms	20μs
Sigmoid	7	2^{14}	55	281	161ms	19μs

RNS HEAAN
- 32 bits precision

https://github.com/HanKyoohyung/HEAAN-dev

Questions?