A Full RNS Implementation of Fan and Vercauteren Somewhat Homomorphic Encryption Scheme

Presented by: Vincent Zucca[1]

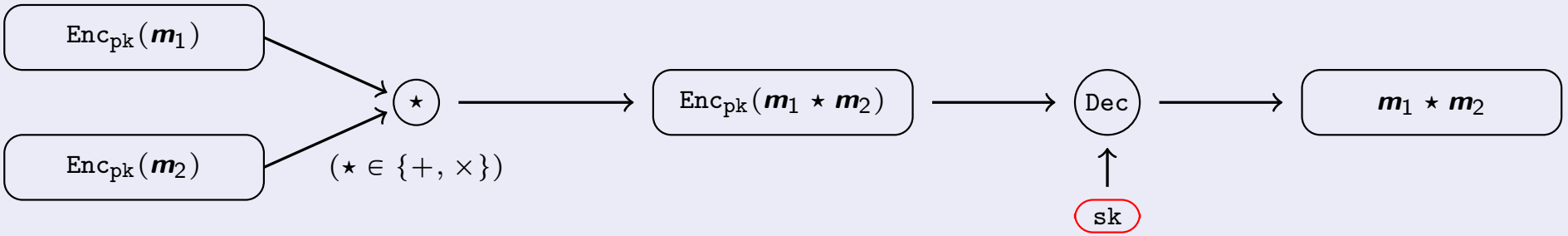
Joint work with:
Jean-Claude Bajard[1], Julien Eynard[2] and Anwar Hasan[2]

[1]Sorbonne Universités, UPMC Univ. Paris 06, CNRS, LIP6 UMR 7606, France
[2]Dept. of Electrical and Computer Engineering, University of Waterloo, Canada

April 24th 2017

Context

Homomorphic Encryption (HE):



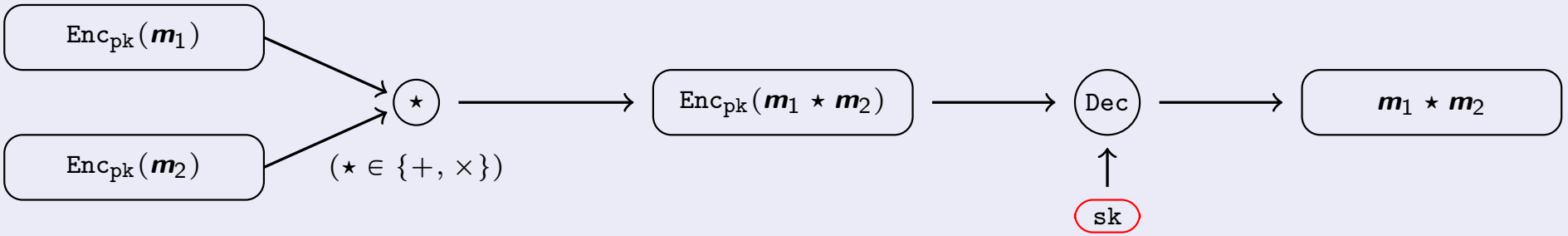
"Noisy encryption"

- Each ciphertext contains a noise.
- After each homomorphic operation the noise grows.
- Decryption remains correct until the noise reaches a certain bound.
  $\Longrightarrow$ Limited number of operations.
  $\Longrightarrow$ "Somewhat" Homomorphic Encryption (SHE).

Problem: not practical enough yet.

Context

Homomorphic Encryption (HE):



"Noisy encryption"

- Each ciphertext contains a noise.
- After each homomorphic operation the noise grows.
- Decryption remains correct until the noise reaches a certain bound.
  $\Longrightarrow$ Limited number of operations.
  $\Longrightarrow$ "Somewhat" Homomorphic Encryption (SHE).

Goal: arithmetical optimization of a certain type of SHE schemes.

Overview of RNS and FV

Chinese Remainder Theorem

Pairwise **coprime** integers $\{q_1, \ldots, q_k\}$: "RNS base" $(q = \prod_{i=1}^{k} q_i)$,

$$\mathbb{Z}/q\mathbb{Z} \cong \mathbb{Z}/q_1\mathbb{Z} \times \ldots \times \mathbb{Z}/q_k\mathbb{Z}$$

Overview of RNS and FV

Chinese Remainder Theorem

Pairwise **coprime** integers $\{q_1, \ldots, q_k\}$: "RNS base" $(q = \prod_{i=1}^{k} q_i)$,

$$\mathbb{Z}/q\mathbb{Z} \cong \mathbb{Z}/q_1\mathbb{Z} \times \ldots \times \mathbb{Z}/q_k\mathbb{Z}$$

Residue Number Systems

- Large $x \in [0, q) \cap \mathbb{Z} \leftrightarrow k$ **small residues** $(x \bmod q_1, \ldots, x \bmod q_k)$.
- **Parallel**, **carry-free** arithmetic $+, -, \times, \div$ on residues.
- **Non positional** number system.

Overview of RNS and FV

Chinese Remainder Theorem

Pairwise **coprime** integers $\{q_1, \ldots, q_k\}$: "RNS base" $(q = \prod_{i=1}^{k} q_i)$,

$$\mathbb{Z}/q\mathbb{Z} \cong \mathbb{Z}/q_1\mathbb{Z} \times \ldots \times \mathbb{Z}/q_k\mathbb{Z}$$

Residue Number Systems

- Large $x \in [0, q) \cap \mathbb{Z} \leftrightarrow k$ **small residues** $(x \bmod q_1, \ldots, x \bmod q_k)$.
- **Parallel**, **carry-free** arithmetic $+, -, \times, \div$ on residues.
- **Non positional** number system.

Base extensions $\mathcal{B}_q = \{q_1, \ldots, q_k\} \rightarrow \mathcal{B} = \{m_1, \ldots, m_\ell\}$

- Fast approximate base extension: x in $\mathcal{B}_q \rightarrow |x|_q + \alpha q$ in $\mathcal{B}$
  $\rightsquigarrow$ Fast but q-overflow $\alpha \in [0, k) \cap \mathbb{Z}$
- If needed, add an extra modulus m_{sk} to $\mathcal{B}_q$ to correct α efficiently

Overview of RNS and FV

Ambiant space in *FV scheme (Fan and Vercauteren, 2012)*

$\mathcal{R} = \mathbb{Z}[X]/(X^n + 1)$, $n = 2^h \leftrightarrow$ integer polynomials of degree $< n$

- t: **plaintext** modulus, $\boldsymbol{m} \in \mathcal{R}_t = \mathcal{R}/t\mathcal{R}$ (coeff. modulo t)
- q: **ciphertext** modulus ($\gg t$), $\boldsymbol{c} \in \mathcal{R}_q \times \mathcal{R}_q$ (coeff. modulo q)

Overview of RNS and FV

Ambiant space in *FV scheme (Fan and Vercauteren, 2012)*

$\mathcal{R} = \mathbb{Z}[X]/(X^n + 1)$, $n = 2^h \leftrightarrow$ integer polynomials of degree $< n$

- t: **plaintext** modulus, $\boldsymbol{m} \in \mathcal{R}_t = \mathcal{R}/t\mathcal{R}$ (coeff. modulo t)
- q: **ciphertext** modulus ($\gg t$), $\boldsymbol{c} \in \mathcal{R}_q \times \mathcal{R}_q$ (coeff. modulo q)

Common optimizations *for arithmetic on...*

...coefficients: **Residue Number Systems**

q free of form: choose $q = q_1 \dots q_k$ (small prime moduli q_i)

$$\mathbb{Z}/q\mathbb{Z} \cong \mathbb{Z}/q_1\mathbb{Z} \times \dots \times \mathbb{Z}/q_k\mathbb{Z}$$

Overview of RNS and FV

Ambiant space in *FV scheme (Fan and Vercauteren, 2012)*

$\mathcal{R} = \mathbb{Z}[X]/(X^n + 1)$, $n = 2^h \leftrightarrow$ integer polynomials of degree $< n$

- t: **plaintext** modulus, $\boldsymbol{m} \in \mathcal{R}_t = \mathcal{R}/t\mathcal{R}$ (coeff. modulo t)
- q: **ciphertext** modulus ($\gg t$), $\boldsymbol{c} \in \mathcal{R}_q \times \mathcal{R}_q$ (coeff. modulo q)

Common optimizations *for arithmetic on...*

...coefficients: **Residue Number Systems**

q free of form: choose $q = q_1 \dots q_k$ (small prime moduli q_i)

$$\mathbb{Z}/q\mathbb{Z} \cong \mathbb{Z}/q_1\mathbb{Z} \times \ldots \times \mathbb{Z}/q_k\mathbb{Z}$$

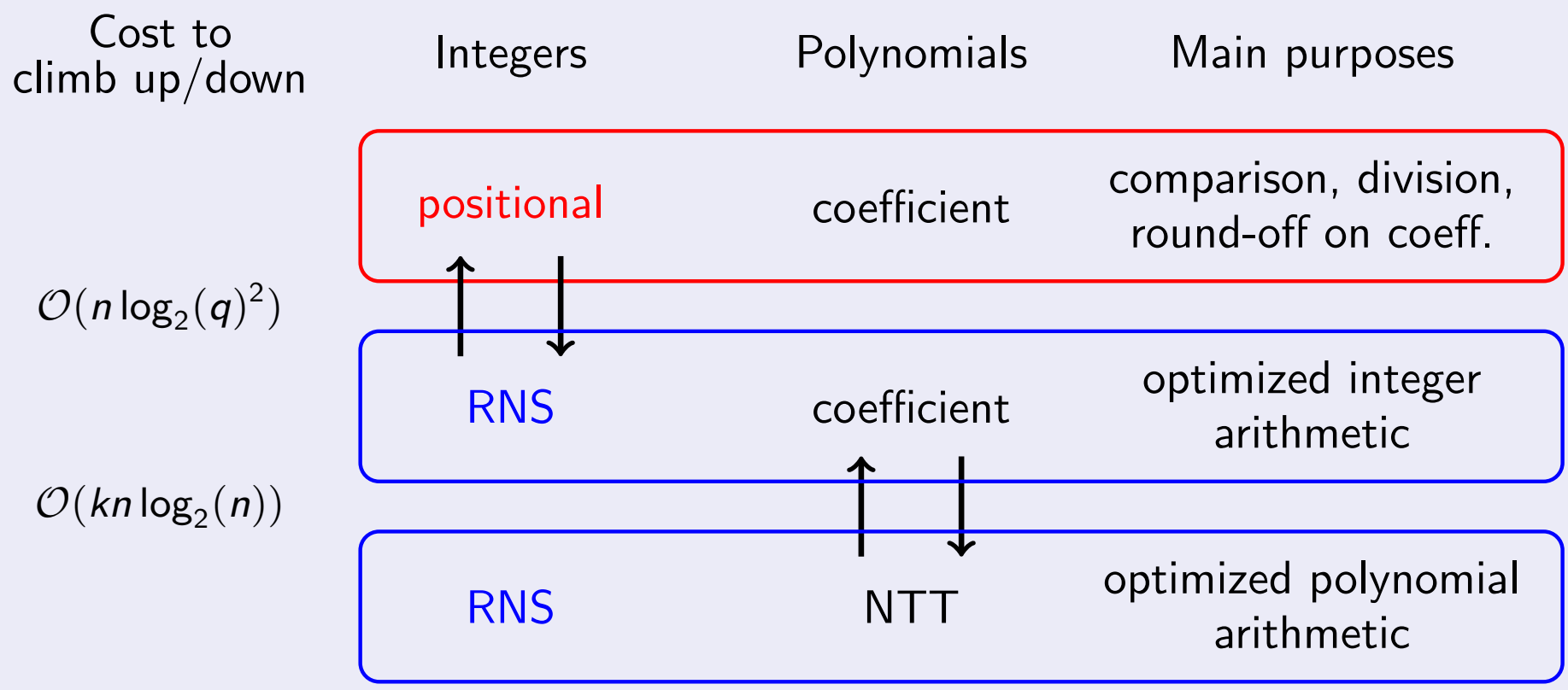
...polynomials: **Number Theoretic Transform**

Optimized polynomial product (n a power of 2): $\mathcal{O}(n \log_2(n))$

$\rightarrow$ **matches with RNS** representation

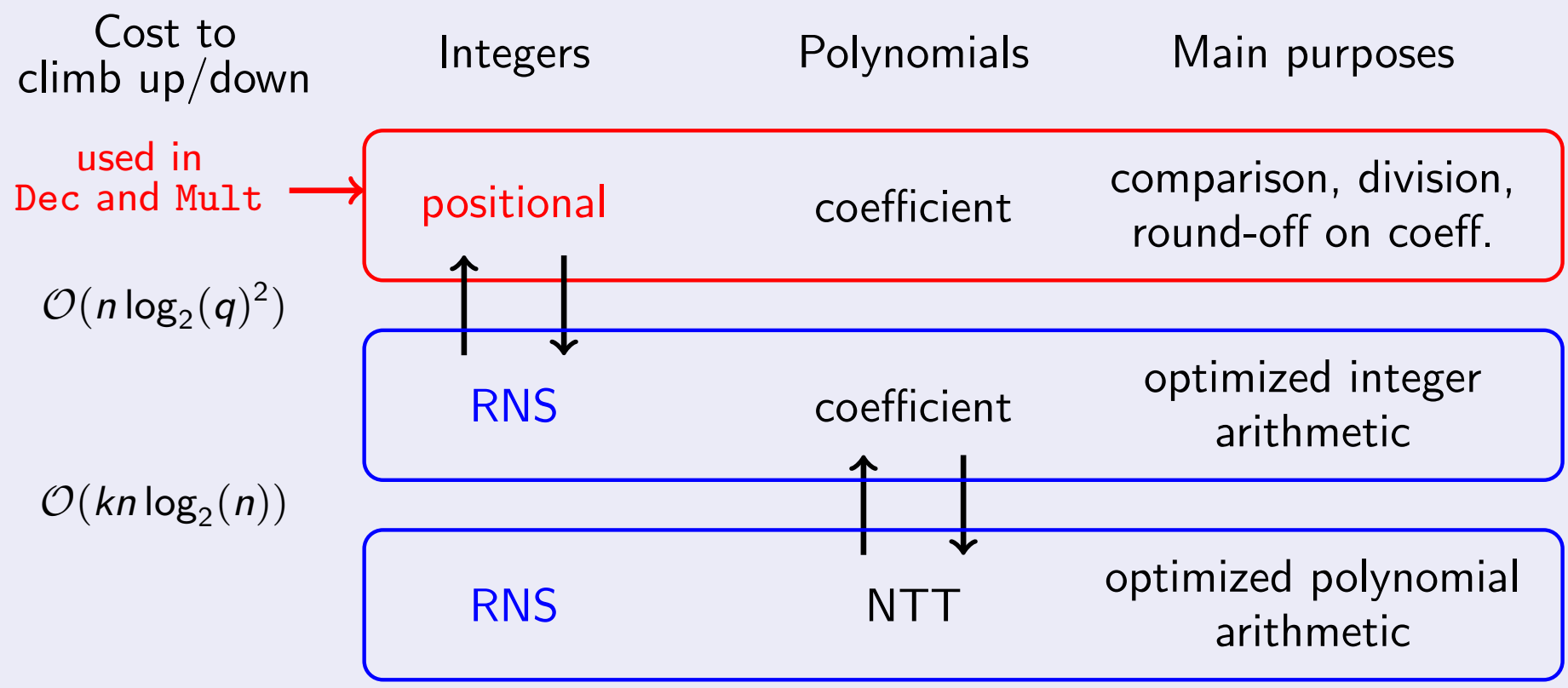
Overview of RNS and FV

"Ladder" of representations



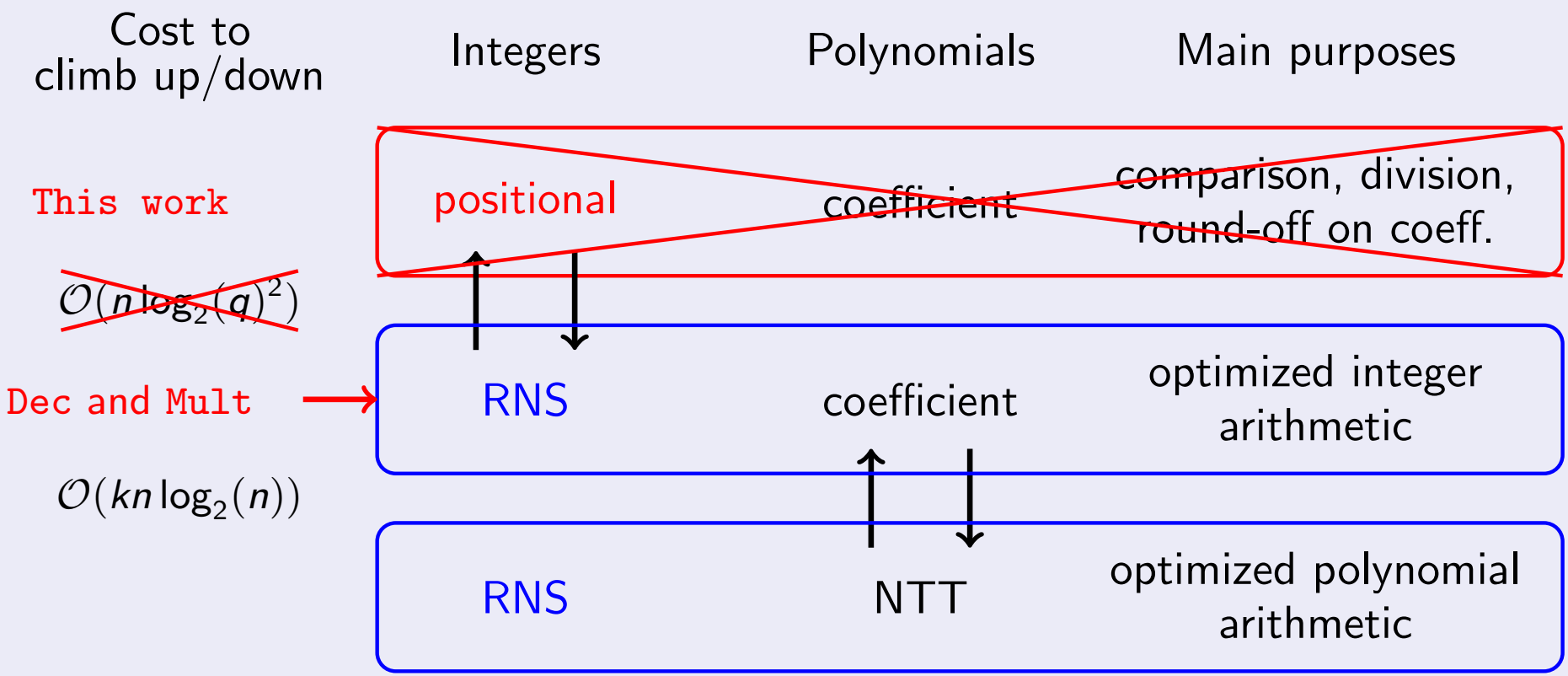
Overview of RNS and FV

"Ladder" of representations



Overview of RNS and FV

"Ladder" of representations



Overview of RNS and FV

- χ_{key} and χ_{err}: "**narrow**" distributions on $\mathcal{R}_q$;
- $\mathcal{U}$: **uniform** distribution on $\mathcal{R}_q$

Overview of RNS and FV

- χ_{key} and χ_{err}: "**narrow**" distributions on $\mathcal{R}_q$;
- $\mathcal{U}$: **uniform** distribution on $\mathcal{R}_q$

Key Generation

1. sample $\boldsymbol{s} \leftarrow \chi_{key}$
2. sample $(\boldsymbol{a}, \boldsymbol{e}) \leftarrow \mathcal{U} \times \chi_{err}$
3. output $\mathbf{pk} = (\boldsymbol{p_0}, \boldsymbol{p_1}) = ([-(\boldsymbol{as} + \boldsymbol{e})]_q, \boldsymbol{a})$ (RLWE sample)
   $\mathbf{sk} = \boldsymbol{s}$

Overview of RNS and FV

- χ_{key} and χ_{err}: "**narrow**" distributions on $\mathcal{R}_q$;
- $\mathcal{U}$: **uniform** distribution on $\mathcal{R}_q$

Key Generation

1. sample $\boldsymbol{s} \leftarrow \chi_{key}$
2. sample $(\boldsymbol{a}, \boldsymbol{e}) \leftarrow \mathcal{U} \times \chi_{err}$
3. output $\mathbf{pk} = (\boldsymbol{p_0}, \boldsymbol{p_1}) = ([-(\boldsymbol{as} + \boldsymbol{e})]_q, \boldsymbol{a})$ (RLWE sample)
   $\mathbf{sk} = \boldsymbol{s}$

Encryption

$[\boldsymbol{m}]_t \in \mathcal{R}_t$ to be encrypted, public key $\mathbf{pk}$,

1. sample $(\boldsymbol{e_1}, \boldsymbol{e_2}, \boldsymbol{u}) \leftarrow (\chi_{err})^2 \times \mathcal{U}$
2. output $(\boldsymbol{c}_0, \boldsymbol{c}_1) = ([\Delta[\boldsymbol{m}]_t + \boldsymbol{p_0 u} + \boldsymbol{e_1}]_q, [\boldsymbol{p_1 u} + \boldsymbol{e_2}]_q)$ $(\Delta = \lfloor \frac{q}{t} \rfloor)$

Overview of RNS and FV

- χ_{key} and χ_{err}: "**narrow**" distributions on $\mathcal{R}_q$;
- $\mathcal{U}$: **uniform** distribution on $\mathcal{R}_q$

Key Generation

1. sample $\boldsymbol{s} \leftarrow \chi_{key}$
2. sample $(\boldsymbol{a}, \boldsymbol{e}) \leftarrow \mathcal{U} \times \chi_{err}$
3. output $\mathbf{pk} = (\boldsymbol{p_0}, \boldsymbol{p_1}) = ([-(\boldsymbol{as} + \boldsymbol{e})]_q, \boldsymbol{a})$ (RLWE sample)
   $\mathbf{sk} = \boldsymbol{s}$

Encryption

$[\boldsymbol{m}]_t \in \mathcal{R}_t$ to be encrypted, public key $\mathbf{pk}$,

1. sample $(\boldsymbol{e_1}, \boldsymbol{e_2}, \boldsymbol{u}) \leftarrow (\chi_{err})^2 \times \mathcal{U}$
2. output $(\boldsymbol{c}_0, \boldsymbol{c}_1) = ([\Delta[\boldsymbol{m}]_t + \boldsymbol{p_0 u} + \boldsymbol{e_1}]_q, [\boldsymbol{p_1 u} + \boldsymbol{e_2}]_q)$ $(\Delta = \lfloor \frac{q}{t} \rfloor)$

$$[\boldsymbol{c}_0 + \boldsymbol{c}_1 \boldsymbol{s}]_q = \Delta[\boldsymbol{m}]_t + \boldsymbol{v} \pmod{q} \text{ with } \boldsymbol{v}\text{: "fresh noise"}$$

Full RNS variant of FV decryption

The original decryption process

$(\boldsymbol{c_0}, \boldsymbol{c_1})$ encrypting $[\boldsymbol{m}]_t$ with noise $\boldsymbol{v}$: $[\boldsymbol{c_0} + \boldsymbol{c_1 s}]_q = \Delta[\boldsymbol{m}]_t + \boldsymbol{v} + q\boldsymbol{r}$

1. scale-down: $\frac{t}{q} \cdot [\boldsymbol{c_0} + \boldsymbol{c_1 s}]_q = [\boldsymbol{m}]_t + \frac{\boldsymbol{v'}}{q} + t\boldsymbol{r}$
2. round-off: $\lfloor \frac{t}{q} \cdot [\boldsymbol{c_0} + \boldsymbol{c_1 s}]_q \rceil = [\boldsymbol{m}]_t + \lfloor \frac{\boldsymbol{v'}}{q} \rceil + t\boldsymbol{r}$

$$\text{If } \|\boldsymbol{v}\|_\infty = \max(|\boldsymbol{v}_i|) < B_{dec} \Rightarrow \lfloor \frac{\boldsymbol{v'}}{q} \rceil = 0$$

$$\mathtt{Dec}(\boldsymbol{c}, \boldsymbol{sk}) = [\lfloor \frac{t}{q} \cdot [\boldsymbol{c_0} + \boldsymbol{c_1 s}]_q \rceil]_t = [[\boldsymbol{m}]_t + t\boldsymbol{r}]_t = [\boldsymbol{m}]_t.$$

Full RNS variant of FV decryption

The original decryption process

$(\boldsymbol{c_0}, \boldsymbol{c_1})$ encrypting $[\boldsymbol{m}]_t$ with noise $\boldsymbol{v}$: $[\boldsymbol{c_0} + \boldsymbol{c_1 s}]_q = \Delta[\boldsymbol{m}]_t + \boldsymbol{v} + q\boldsymbol{r}$

1. scale-down: $\frac{t}{q} \cdot [\boldsymbol{c_0} + \boldsymbol{c_1 s}]_q = [\boldsymbol{m}]_t + \frac{\boldsymbol{v'}}{q} + t\boldsymbol{r}$
2. round-off: $\lfloor \frac{t}{q} \cdot [\boldsymbol{c_0} + \boldsymbol{c_1 s}]_q \rceil = [\boldsymbol{m}]_t + \lfloor \frac{\boldsymbol{v'}}{q} \rceil + t\boldsymbol{r}$

$$\text{If } \|\boldsymbol{v}\|_\infty = \max(|\boldsymbol{v}_i|) < B_{dec} \Rightarrow \lfloor \tfrac{\boldsymbol{v'}}{q} \rceil = 0$$

$$\mathtt{Dec}(\boldsymbol{c}, \boldsymbol{sk}) = [\lfloor \tfrac{t}{q} \cdot [\boldsymbol{c_0} + \boldsymbol{c_1 s}]_q \rceil]_t = [[\boldsymbol{m}]_t + t\boldsymbol{r}]_t = [\boldsymbol{m}]_t.$$

Issue for RNS (**non positional**) representation

How to compute $(\lfloor \frac{t}{q} \cdot x \rceil \bmod t)$ in RNS?

Full RNS variant of FV decryption

Our contribution: computing a round-off in RNS

In RNS, exact division can be done efficiently, so we use:

$$\left\lfloor \frac{t}{q} \cdot x \right\rceil = \frac{tx - |tx|_q}{q} + b, \ (b \in \{0,1\})$$

1. **fast approximate extension** of $|tx|_q$ (in RNS) to RNS base $\{t\}$:

$$\texttt{FastBconv}(tx, q, \{t\}) = |tx|_q + \alpha q \bmod t \quad (\alpha \in [0,k) \cap \mathbb{Z})$$

2. $\frac{tx-(|tx|_q+\alpha q)}{q} = \lfloor \frac{t}{q} \cdot x \rceil - \alpha = \lfloor \frac{t}{q} \cdot x \rceil - E \bmod t$ (with $E = \alpha + b \leqslant k$)

Full RNS variant of FV decryption

Our contribution: computing a round-off in RNS

In RNS, exact division can be done efficiently, so we use:

$$\left\lfloor \frac{t}{q} \cdot x \right\rceil = \frac{tx - |tx|_q}{q} + b, \ (b \in \{0,1\})$$

1. **fast approximate extension** of $|tx|_q$ (in RNS) to RNS base $\{t\}$:

$$\texttt{FastBconv}(tx, q, \{t\}) = |tx|_q + \alpha q \bmod t \quad (\alpha \in [0,k) \cap \mathbb{Z})$$

2. $\frac{tx-(|tx|_q+\alpha q)}{q} = \lfloor \frac{t}{q} \cdot x \rceil - \alpha = \lfloor \frac{t}{q} \cdot x \rceil - E \bmod t$ (with $E = \alpha + b \leqslant k$)

Remark: since tx cancels modulo t, only compute $\frac{-(|tx|_q+\alpha q)}{q} \bmod t$.

Full RNS variant of FV decryption

Our contribution: computing a round-off in RNS

In RNS, exact division can be done efficiently, so we use:

$$\left\lfloor \frac{t}{q} \cdot x \right\rceil = \frac{tx - |tx|_q}{q} + b, \ (b \in \{0,1\})$$

1. **fast approximate extension** of $|tx|_q$ (in RNS) to RNS base $\{t\}$:

$$\texttt{FastBconv}(tx, q, \{t\}) = |tx|_q + \alpha q \bmod t \quad (\alpha \in [0,k) \cap \mathbb{Z})$$

2. $\frac{tx-(|tx|_q+\alpha q)}{q} = \lfloor \frac{t}{q} \cdot x \rceil - \alpha = \lfloor \frac{t}{q} \cdot x \rceil - E \bmod t$ (with $E = \alpha + b \leqslant k$)

Remark: since tx cancels modulo t, only compute $\frac{-(|tx|_q+\alpha q)}{q} \bmod t$.

An error occurs

Need to correct the error E for a correct decryption.

Full RNS variant of FV decryption

Our contribution: correcting the error

Rewrite in $\mathbb{Z}$: $$tx = \lfloor \frac{t}{q} \cdot x \rceil q + [tx]_q$$

scale by $\gamma \in \mathbb{N}$: $$\left\lfloor \gamma \frac{t}{q} \cdot x \right\rceil - E = \gamma \left\lfloor \frac{t}{q} \cdot x \right\rceil + \left\lfloor \gamma \frac{[tx]_q}{q} \right\rceil - E$$

Full RNS variant of FV decryption

Our contribution: correcting the error

Rewrite in $\mathbb{Z}$: $$tx = \left\lfloor \frac{t}{q} \cdot x \right\rceil q + [tx]_q$$

scale by $\gamma \in \mathbb{N}$: $$\left\lfloor \gamma \frac{t}{q} \cdot x \right\rceil - E = \gamma \left\lfloor \frac{t}{q} \cdot x \right\rceil + \left\lfloor \gamma \frac{[tx]_q}{q} \right\rceil - E$$

Now comes the **trick**

If gap $\varepsilon > 0$ $\left(-\frac{q}{2} + \varepsilon \leqslant [tx]_q \leqslant \frac{q}{2} - \varepsilon\right)$ then if $\gamma\varepsilon \geqslant k + \frac{1}{2}$ we have:

$$\left| \left\lfloor \gamma \frac{[tx]_q}{q} \right\rceil - E \right| < \frac{\gamma}{2}$$

$\rightsquigarrow$ compute $\left[\left\lfloor \gamma \frac{t}{q} \cdot x \right\rceil - E \right]_\gamma = \left\lfloor \gamma \frac{[tx]_q}{q} \right\rceil - E$ gives exactly the error

Full RNS variant of FV decryption

Our contribution: correcting the error

Rewrite in $\mathbb{Z}$: $$tx = \left\lfloor \frac{t}{q} \cdot x \right\rceil q + [tx]_q$$

scale by $\gamma \in \mathbb{N}$: $$\left\lfloor \gamma \frac{t}{q} \cdot x \right\rceil - E = \gamma \left\lfloor \frac{t}{q} \cdot x \right\rceil + \left\lfloor \gamma \frac{[tx]_q}{q} \right\rceil - E$$

Now comes the **trick**

If gap $\varepsilon > 0$ $\left(-\frac{q}{2} + \varepsilon \leqslant [tx]_q \leqslant \frac{q}{2} - \varepsilon\right)$ then if $\gamma\varepsilon \geqslant k + \frac{1}{2}$ we have:

$$\left| \left\lfloor \gamma \frac{[tx]_q}{q} \right\rceil - E \right| < \frac{\gamma}{2}$$

$\rightsquigarrow$ compute $\left[\left\lfloor \gamma \frac{t}{q} \cdot x \right\rceil - E \right]_\gamma = \left\lfloor \gamma \frac{[tx]_q}{q} \right\rceil - E$ gives exactly the error

Strategy

1. Compute $\lfloor \gamma \frac{t}{q} \cdot x \rceil$ modulo t **and** γ
2. Use **centered remainder** modulo γ **to correct** the error

Full RNS variant of FV decryption

$\mathtt{Dec}_{\mathtt{RNS}}((\boldsymbol{c}_0, \boldsymbol{c}_1), \boldsymbol{s}, \gamma)$

Require: $(\boldsymbol{c}_0, \boldsymbol{c}_1)$ an encryption of $[\boldsymbol{m}]_t$, and $\boldsymbol{s}$ the secret key, both in base q; an integer γ coprime to t and q

Ensure: $[\boldsymbol{m}]_t$

1: **for** $m \in \{t, \gamma\}$ **do**
2: $\quad \boldsymbol{s}^{(m)} \leftarrow \mathtt{FastBconv}(-\gamma t(\boldsymbol{c_0} + \boldsymbol{c_1 s}), q, \{m\}) \cdot |q^{-1}|_m$
3: **end for**
4: $\tilde{\boldsymbol{s}}^{(\gamma)} \leftarrow \left[\boldsymbol{s}^{(\gamma)}\right]_\gamma$
5: $\boldsymbol{m}^{(t)} \leftarrow \left[(\boldsymbol{s}^{(t)} - \tilde{\boldsymbol{s}}^{(\gamma)}) \times |\gamma^{-1}|_t\right]_t$
6: **return** $\boldsymbol{m}^{(t)}$

Full RNS variant of FV decryption

$\mathtt{Dec}_{\mathtt{RNS}}((\boldsymbol{c}_0, \boldsymbol{c}_1), \boldsymbol{s}, \gamma)$

Require: $(\boldsymbol{c}_0, \boldsymbol{c}_1)$ an encryption of $[\boldsymbol{m}]_t$, and $\boldsymbol{s}$ the secret key, both in base q; an integer γ coprime to t and q

Ensure: $[\boldsymbol{m}]_t$

1: **for** $m \in \{t, \gamma\}$ **do**
2: $\quad \boldsymbol{s}^{(m)} \leftarrow \mathtt{FastBconv}(-\gamma t(\boldsymbol{c_0} + \boldsymbol{c_1 s}), q, \{m\}) \cdot |q^{-1}|_m$
3: **end for**
4: $\tilde{\boldsymbol{s}}^{(\gamma)} \leftarrow \left[\boldsymbol{s}^{(\gamma)}\right]_\gamma$
5: $\boldsymbol{m}^{(t)} \leftarrow \left[(\boldsymbol{s}^{(t)} - \tilde{\boldsymbol{s}}^{(\gamma)}) \times |\gamma^{-1}|_t\right]_t$
6: **return** $\boldsymbol{m}^{(t)}$

Results

- Better asymptotic complexity: $\mathcal{O}(n^3) \rightarrow \mathcal{O}(n^2 \log_2(n))$.
- Very flexible in terms of parallelization.
- Modified bound for noise: $\|\boldsymbol{v}\|_\infty < \frac{\Delta - |q|_t}{2} - \frac{k\Delta}{\gamma}$.
  (although no significant consequence in practice)

Full RNS variant of FV multiplication

Original homomorphic multiplication of $(\boldsymbol{c}_0, \boldsymbol{c}_1)$ by $(\boldsymbol{c}'_0, \boldsymbol{c}'_1)$

Issues in original process for an RNS variant

1. Product $(\tilde{\boldsymbol{c}}_0, \tilde{\boldsymbol{c}}_1, \tilde{\boldsymbol{c}}_2) = (\boldsymbol{c}_0\boldsymbol{c}'_0, \boldsymbol{c}_0\boldsymbol{c}'_1 + \boldsymbol{c}'_0\boldsymbol{c}_1, \boldsymbol{c}_1\boldsymbol{c}'_1)$ over $\mathbb{Z}$

Full RNS variant of FV multiplication

Original homomorphic multiplication of (c_0, c_1) by (c'_0, c'_1)

Issues in original process for an RNS variant

1. Product $(\tilde{c}_0, \tilde{c}_1, \tilde{c}_2) = (c_0 c'_0, c_0 c'_1 + c'_0 c_1, c_1 c'_1)$ over $\mathbb{Z}$
2. Division + Round-off: $\hat{c}_i = \lfloor \frac{t}{q} \cdot \tilde{c}_i \rceil$

$$\rightsquigarrow [\hat{c}_0 + \hat{c}_1 s + \hat{c}_2 s^2]_q = \Delta [m_1 m_2]_t + v'' \bmod q$$

Full RNS variant of FV multiplication

Original homomorphic multiplication of (c_0, c_1) by (c'_0, c'_1)

Issues in original process for an RNS variant

1. Product $(\tilde{c}_0, \tilde{c}_1, \tilde{c}_2) = (c_0 c'_0, c_0 c'_1 + c'_0 c_1, c_1 c'_1)$ over $\mathbb{Z}$
2. Division + Round-off: $\hat{c}_i = \lfloor \frac{t}{q} \cdot \tilde{c}_i \rceil$
   $$\rightsquigarrow [\hat{c}_0 + \hat{c}_1 s + \hat{c}_2 s^2]_q = \Delta [m_1 m_2]_t + v'' \bmod q$$
3. Relinearization: $(\hat{c}_0 + \hat{c}_2 s^2, \hat{c}_1) \xrightarrow{s \text{ private}} (\hat{c}_0 + \hat{c}_2(s^2 + e + as), \hat{c}_1 - a\hat{c}_2)$

Full RNS variant of FV multiplication

Original homomorphic multiplication of (c_0, c_1) by (c'_0, c'_1)

Issues in original process for an RNS variant

1. Product $(\tilde{c}_0, \tilde{c}_1, \tilde{c}_2) = (c_0 c'_0, c_0 c'_1 + c'_0 c_1, c_1 c'_1)$ over $\mathbb{Z}$
2. Division + Round-off: $\hat{c}_i = \lfloor \frac{t}{q} \cdot \tilde{c}_i \rceil$
   $$\rightsquigarrow [\hat{c}_0 + \hat{c}_1 s + \hat{c}_2 s^2]_q = \Delta [m_1 m_2]_t + v'' \bmod q$$
3. Relinearization: $(\hat{c}_0 + \hat{c}_2 s^2, \hat{c}_1) \xrightarrow{s \text{ private}} (\hat{c}_0 + \hat{c}_2(s^2 + e + as), \hat{c}_1 - a\hat{c}_2)$
   - Large noise growth $\|\hat{c}_2 \times e\|_\infty < q \times nB_{\mathtt{err}} \rightarrow$ Original solution is to...

Full RNS variant of FV multiplication

Original homomorphic multiplication of (c_0, c_1) by (c'_0, c'_1)

Issues in original process for an RNS variant

1. Product $(\tilde{c}_0, \tilde{c}_1, \tilde{c}_2) = (c_0 c'_0, c_0 c'_1 + c'_0 c_1, c_1 c'_1)$ over $\mathbb{Z}$
2. Division + Round-off: $\hat{c}_i = \lfloor \frac{t}{q} \cdot \tilde{c}_i \rceil$
   $$\rightsquigarrow [\hat{c}_0 + \hat{c}_1 s + \hat{c}_2 s^2]_q = \Delta [m_1 m_2]_t + v'' \bmod q$$
3. Relinearization: $(\hat{c}_0 + \hat{c}_2 s^2, \hat{c}_1) \xrightarrow{s \text{ private}} (\hat{c}_0 + \hat{c}_2(s^2 + e + as), \hat{c}_1 - a\hat{c}_2)$
   - Large noise growth $\|\hat{c}_2 \times e\|_\infty < q \times nB_{\mathtt{err}} \rightarrow$ Original solution is to...
   - Decompose $\hat{c}_2 = b_0 + b_1 \omega + \ldots + b_\ell \omega^{\ell - 1}$ in radix ω

Full RNS variant of FV multiplication

Original homomorphic multiplication of $(\boldsymbol{c}_0, \boldsymbol{c}_1)$ by $(\boldsymbol{c}'_0, \boldsymbol{c}'_1)$

Issues in original process for an RNS variant

1. Product $(\tilde{\boldsymbol{c}}_0, \tilde{\boldsymbol{c}}_1, \tilde{\boldsymbol{c}}_2) = (\boldsymbol{c}_0\boldsymbol{c}'_0, \boldsymbol{c}_0\boldsymbol{c}'_1 + \boldsymbol{c}'_0\boldsymbol{c}_1, \boldsymbol{c}_1\boldsymbol{c}'_1)$ over $\mathbb{Z}$
2. Division + Round-off: $\hat{\boldsymbol{c}}_i = \lfloor \frac{t}{q} \cdot \tilde{\boldsymbol{c}}_i \rceil$
   $$\rightsquigarrow [\hat{\boldsymbol{c}}_0 + \hat{\boldsymbol{c}}_1\boldsymbol{s} + \hat{\boldsymbol{c}}_2\boldsymbol{s}^2]_q = \Delta[\boldsymbol{m}_1\boldsymbol{m}_2]_t + \boldsymbol{v}'' \bmod q$$
3. Relinearization: $(\hat{\boldsymbol{c}}_0 + \hat{\boldsymbol{c}}_2\boldsymbol{s}^2, \hat{\boldsymbol{c}}_1) \xrightarrow{\boldsymbol{s} \text{ private}} (\hat{\boldsymbol{c}}_0 + \hat{\boldsymbol{c}}_2(\boldsymbol{s}^2 + \boldsymbol{e} + \boldsymbol{a}\boldsymbol{s}), \hat{\boldsymbol{c}}_1 - \boldsymbol{a}\hat{\boldsymbol{c}}_2)$
   - Large noise growth $\|\hat{\boldsymbol{c}}_2 \times \boldsymbol{e}\|_\infty < q \times nB_{\mathtt{err}} \rightarrow$ Original solution is to...
   - Decompose $\hat{\boldsymbol{c}}_2 = \boldsymbol{b}_0 + \boldsymbol{b}_1\omega + \ldots + \boldsymbol{b}_\ell\omega^{\ell-1}$ in radix ω
   - Public relinearization key: $([\boldsymbol{s}^2 \cdot (1, \omega, \ldots, \omega^{\ell-1}) + \vec{\boldsymbol{e}} + \vec{\boldsymbol{a}}\boldsymbol{s}]_q, -\vec{\boldsymbol{a}})$

Full RNS variant of FV multiplication

Original homomorphic multiplication of $(\boldsymbol{c}_0, \boldsymbol{c}_1)$ by $(\boldsymbol{c}'_0, \boldsymbol{c}'_1)$

Issues in original process for an RNS variant

1. Product $(\tilde{\boldsymbol{c}}_0, \tilde{\boldsymbol{c}}_1, \tilde{\boldsymbol{c}}_2) = (\boldsymbol{c}_0\boldsymbol{c}'_0, \boldsymbol{c}_0\boldsymbol{c}'_1 + \boldsymbol{c}'_0\boldsymbol{c}_1, \boldsymbol{c}_1\boldsymbol{c}'_1)$ over $\mathbb{Z}$
2. Division + Round-off: $\hat{\boldsymbol{c}}_i = \lfloor \frac{t}{q} \cdot \tilde{\boldsymbol{c}}_i \rceil$
   $$\rightsquigarrow [\hat{\boldsymbol{c}}_0 + \hat{\boldsymbol{c}}_1\boldsymbol{s} + \hat{\boldsymbol{c}}_2\boldsymbol{s}^2]_q = \Delta[\boldsymbol{m}_1\boldsymbol{m}_2]_t + \boldsymbol{v}'' \bmod q$$
3. Relinearization: $(\hat{\boldsymbol{c}}_0 + \hat{\boldsymbol{c}}_2\boldsymbol{s}^2, \hat{\boldsymbol{c}}_1) \xrightarrow{\boldsymbol{s} \text{ private}} (\hat{\boldsymbol{c}}_0 + \hat{\boldsymbol{c}}_2(\boldsymbol{s}^2 + \boldsymbol{e} + \boldsymbol{a}\boldsymbol{s}), \hat{\boldsymbol{c}}_1 - \boldsymbol{a}\hat{\boldsymbol{c}}_2)$
   - Large noise growth $\|\hat{\boldsymbol{c}}_2 \times \boldsymbol{e}\|_\infty < q \times nB_{\mathtt{err}} \rightarrow$ Original solution is to...
   - Decompose $\hat{\boldsymbol{c}}_2 = \boldsymbol{b}_0 + \boldsymbol{b}_1\omega + \ldots + \boldsymbol{b}_\ell\omega^{\ell-1}$ in radix ω
   - Public relinearization key: $([\boldsymbol{s}^2 \cdot (1, \omega, \ldots, \omega^{\ell-1}) + \vec{\boldsymbol{e}} + \vec{\boldsymbol{a}}\boldsymbol{s}]_q, -\vec{\boldsymbol{a}})$
   - Smaller noise growth $\|(\boldsymbol{b}_0, \boldsymbol{b}_1, \ldots, \boldsymbol{b}_\ell) \cdot \vec{\boldsymbol{e}}\|_\infty < \ell\omega \times nB_{\mathtt{err}}$

Full RNS variant of FV multiplication

Original homomorphic multiplication of (c_0, c_1) by (c'_0, c'_1)

Issues in original process for an RNS variant

1. Product $(\tilde{c}_0, \tilde{c}_1, \tilde{c}_2) = (c_0 c'_0, c_0 c'_1 + c'_0 c_1, c_1 c'_1)$ over $\mathbb{Z}$ (lift)
2. Division + Round-off: $\hat{c}_i = \lfloor \frac{t}{q} \cdot \tilde{c}_i \rceil$
   $$\rightsquigarrow [\hat{c}_0 + \hat{c}_1 s + \hat{c}_2 s^2]_q = \Delta [m_1 m_2]_t + v'' \bmod q$$
3. Relinearization: $(\hat{c}_0 + \hat{c}_2 s^2, \hat{c}_1) \xrightarrow{s \text{ private}} (\hat{c}_0 + \hat{c}_2(s^2 + e + as), \hat{c}_1 - a\hat{c}_2)$
   - Large noise growth $\|\hat{c}_2 \times e\|_\infty < q \times nB_{\mathtt{err}}$ → Original solution is to...
   - Decompose $\hat{c}_2 = b_0 + b_1\omega + \ldots + b_\ell \omega^{\ell-1}$ in radix ω
   - Public relinearization key: $([s^2 \cdot (1, \omega, \ldots, \omega^{\ell-1}) + \vec{e} + \vec{a}s]_q, -\vec{a})$
   - Smaller noise growth $\|(b_0, b_1, \ldots, b_\ell) \cdot \vec{e}\|_\infty < \ell\omega \times nB_{\mathtt{err}}$

Issues for RNS representation

Lift in $\mathbb{Z}$, division and rounding, using positional system in radix ω...

Full RNS variant of FV multiplication

- Problem 1: compute product $(\tilde{\boldsymbol{c}}_0, \tilde{\boldsymbol{c}}_1, \tilde{\boldsymbol{c}}_2)$ in $\mathbb{Z}$.
  $\rightarrow$ Solution: $\|\tilde{\boldsymbol{c}}_i\|_\infty <\sim nq^2$: no lift in $\mathbb{Z}$,
  1. Fast extension to convert residues in second base $\mathcal{B}_{sk} = \mathcal{B} \cup \{m_{sk}\}$;
  2. compute $(\tilde{\boldsymbol{c}}_0, \tilde{\boldsymbol{c}}_1, \tilde{\boldsymbol{c}}_2)$ in $q \cup \mathcal{B}_{sk}$.

Full RNS variant of FV multiplication

- Problem 1: compute product $(\tilde{c}_0, \tilde{c}_1, \tilde{c}_2)$ in $\mathbb{Z}$.
  $\longrightarrow$ Solution: $\|\tilde{c}_i\|_\infty <\sim nq^2$: no lift in $\mathbb{Z}$,
  1. Fast extension to convert residues in second base $\mathcal{B}_{sk} = \mathcal{B} \cup \{m_{sk}\}$;
  2. compute $(\tilde{c}_0, \tilde{c}_1, \tilde{c}_2)$ in $q \cup \mathcal{B}_{sk}$.

- Problem 2: division and round-off $\hat{c}_i = \lfloor \frac{t}{q} \cdot \tilde{c}_i \rceil$ in RNS.
  $\longrightarrow$ Solution: flooring instead of rounding in $\mathcal{B}_{sk}$,
  1. $\texttt{FastBconv}(\tilde{c}_i, q, \mathcal{B}_{sk})$ and computation of flooring in $\mathcal{B}_{sk}$;
  2. $\hat{c}_i \leftarrow \texttt{FastBconvSK}(\tilde{c}_i, \mathcal{B}_{sk}, q)$ (no extra multiple of $\mathcal{B}$).

Full RNS variant of FV multiplication

- Problem 1: compute product $(\tilde{\boldsymbol{c}}_0, \tilde{\boldsymbol{c}}_1, \tilde{\boldsymbol{c}}_2)$ in $\mathbb{Z}$.
  $\rightarrow$ Solution: $\|\tilde{\boldsymbol{c}}_i\|_\infty <\sim nq^2$: no lift in $\mathbb{Z}$,
  1. Fast extension to convert residues in second base $\mathcal{B}_{sk} = \mathcal{B} \cup \{m_{sk}\}$;
  2. compute $(\tilde{\boldsymbol{c}}_0, \tilde{\boldsymbol{c}}_1, \tilde{\boldsymbol{c}}_2)$ in $q \cup \mathcal{B}_{sk}$.

- Problem 2: division and round-off $\hat{\boldsymbol{c}}_i = \left\lfloor \frac{t}{q} \cdot \tilde{\boldsymbol{c}}_i \right\rceil$ in RNS.
  $\rightarrow$ Solution: flooring instead of rounding in $\mathcal{B}_{sk}$,
  1. $\texttt{FastBconv}(\tilde{\boldsymbol{c}}_i, q, \mathcal{B}_{sk})$ and computation of flooring in $\mathcal{B}_{sk}$;
  2. $\hat{\boldsymbol{c}}_i \leftarrow \texttt{FastBconvSK}(\tilde{\boldsymbol{c}}_i, \mathcal{B}_{sk}, q)$ (no extra multiple of $\mathcal{B}$).

- Problem 3: decompose $\hat{\boldsymbol{c}}_2$ in radix ω in base q.
  $\rightarrow$Solution: decompose $\hat{\boldsymbol{c}}_2$ in an RNS way,
  1. $\hat{\boldsymbol{c}}_2 = \left(|\hat{\boldsymbol{c}}_2 \cdot \frac{q_1}{q}|_{q_1}, \cdots, |\hat{\boldsymbol{c}}_2 \cdot \frac{q_k}{q}|_{q_k} \right)$;
  2. $\mathbf{evk}_{RNS} = \left(\left[\left(|\boldsymbol{s}^2 \frac{q}{q_1}|_q, \cdots, |\boldsymbol{s}^2 \frac{q}{q_k}|_q \right) + \vec{\boldsymbol{e}} + \vec{\boldsymbol{a}}\boldsymbol{s} \right]_q, \vec{\boldsymbol{a}} \right)$.

Full RNS variant of FV multiplication

Results

- Prior costly operations over $\mathbb{Z} \rightsquigarrow$ fast RNS base extensions.
- Fairly equivalent noise growth.
- Same number of polynomial products $\Rightarrow$ same asymptotic complexity.
- Better complexity for operations on coefficients.
- Well suited for parallelization.

Experiments

Software implementation

- C++,
- NFLlib (dedicated to RNS polynomial arith. in $\mathcal{R}$ with NTT),
- compared with[a] standard approach with NFLlib + GMP 6.1.0,
- laptop under Fedora 22 with i7-4810MQ CPU @ 2.80GHz, g++ 5.3.1, Hyper-Threading and Turbo Boost turned off.

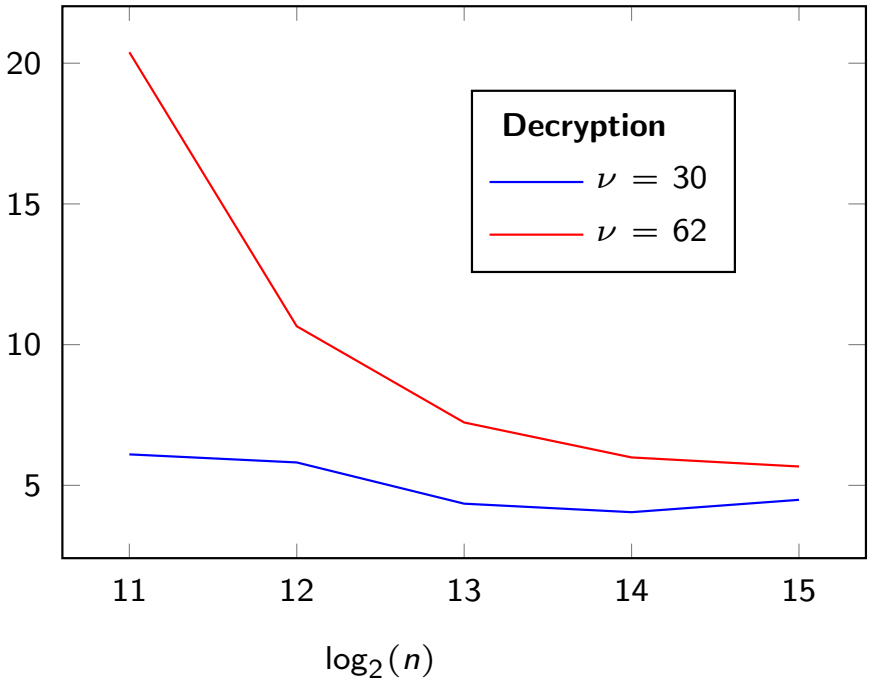
[a] `https://github.com/CryptoExperts/FV-NFLlib`

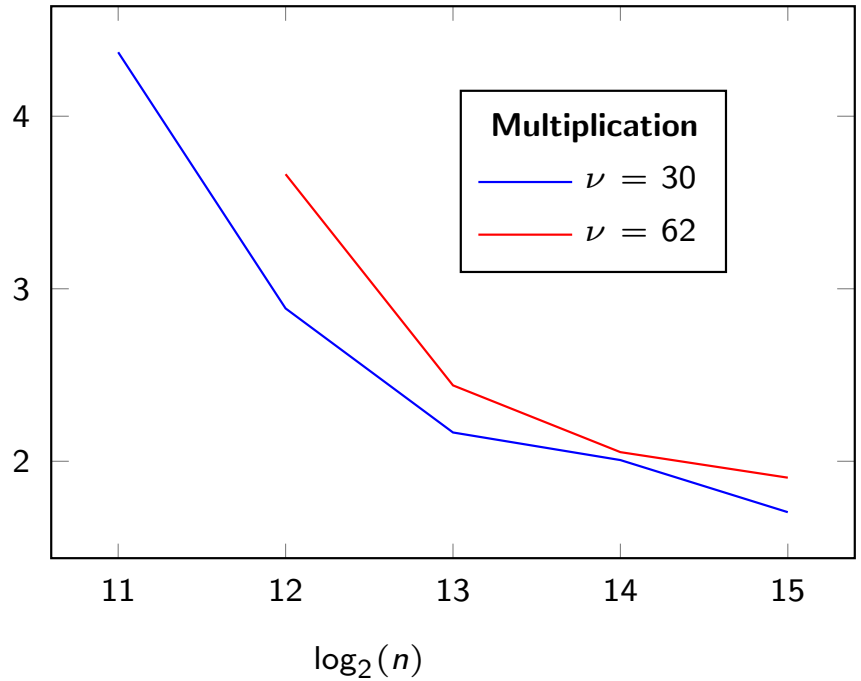
Experiments - Speed-up factors

ν bit-size of moduli	$\log_2(n)$	11	12	13	14	15
30	k	3	6	13	26	53
62	k	1	3	6	12	25

$t = 2^{10}$

$\gamma = 2^8$ (sufficient; practical)





$n \nearrow \Rightarrow$ NTT's dominate computational effort $\Rightarrow$ speed-up $\searrow$.

Conclusion and perspectives

- Optimization of arithmetic on polynomials at the coefficient level.
  - Benefits to SHE scheme like `FV`.
  - No more need of any positional system: only RNS.
- Greater noise growth, but not significant in practice.
- Opens the door to highly competitive parallel implementation of homomorphic encryption on accelerator cards such as GPUs and FPGAs, to take full advantage of the parallelization potential offered by RNS and NTT representation.

✎ Jean Claude Bajard, Julien Eynard, Anwar Hasan and Vincent Zucca. A Full RNS Variant of FV like Somewhat Homomorphic Encryption Scheme. Selected Areas of Cryptography 2016.

Thank you for your attention, any question ?