A Full RNS Variant of FV like Somewhat Homomorphic Encryption Schemes

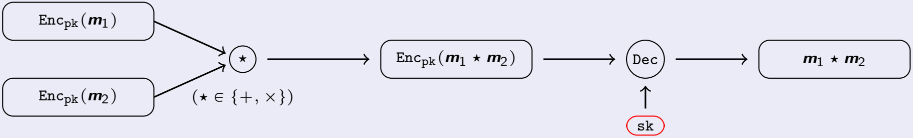
Jean-Claude Bajard[1], Julien Eynard[2]
M. Anwar Hasan[2], Vincent Zucca[1]

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

10-12 August 2016

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).

Purpose of this work

Arithmetical optimization of a certain type of SHE schemes.

Outline

- Introducing Residue Number Systems (RNS) and `FV` scheme
- Full RNS variant of `FV` decryption
- Full RNS variant of `FV` multiplication
- Experiments
- Conclusion

Introducing RNS and FV

Chinese Remainder Theorem

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

$$\varphi : \mathbb{Z}_q \xrightarrow{\sim} \mathbb{Z}_{q_1} \times \ldots \times \mathbb{Z}_{m_k} \text{ (isomorphism)}$$

Residue Number Systems

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

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

- **Fast** but approximate: x in $\mathcal{B}_q \rightarrow |x|_q + \alpha q$ in $\mathcal{B}$.
- Sometimes, possible to add an **extra modulus** m **to correct** α efficiently.

Introducing RNS and FV

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

- $[x]_q$ is (x mod q) in $[-q/2, q/2)$ (centered remainder),
- $|x|_q$ is (x mod q) in $[0, q)$ (classical non negative remainder).

Introducing RNS and FV

Context: $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^n + 1)$

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)

$$\mathbb{Z}_q \simeq \mathbb{Z}_{q_1} \times \dots \times \mathbb{Z}_{q_k}$$

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

Optimized polynomial product (for n a power of 2): $\mathcal{O}(n \log_2(n))$
(**matches with RNS** representation)

Introducing RNS and FV

"Ladder" of representations

Cost to climb up/down	Integers	Polynomials	Main purposes
	positional	coefficient	comparison, division, round-off on coeff.
$\mathcal{O}(n\log_2(q)^2)$	$\uparrow\downarrow$		
	RNS	coefficient	optimized integer arithmetic
$\mathcal{O}(n\log_2(n))$		$\uparrow\downarrow$	
	RNS	NTT	optimized polynomial arithmetic

Introducing RNS and FV

"Ladder" of representations

Cost to climb up/down	Integers	Polynomials	Main purposes
used in Dec and Mult →	positional	coefficient	comparison, division, round-off on coeff.
$\mathcal{O}(n \log_2(q)^2)$	↑↓		
	RNS	coefficient	optimized integer arithmetic
$\mathcal{O}(n \log_2(n))$		↑↓	
	RNS	NTT	optimized polynomial arithmetic

Introducing RNS and FV

"Ladder" of representations

Cost to climb up/down	Integers	Polynomials	Main purposes
this work →	~~positional~~	~~coefficient~~	~~comparison, division, round-off on coeff.~~
$\mathcal{O}(n \log_2(q)^2)$			
	RNS	coefficient	optimized integer arithmetic
$\mathcal{O}(n \log_2(n))$			
	RNS	NTT	optimized polynomial arithmetic

Introducing RNS and FV

χ_{key} and χ_{err}: "**small**" distributions on $\mathcal{R}_q$; $\mathcal{U}$: **uniform** distrib. 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 $(\boldsymbol{c}_0, \boldsymbol{c}_1) = ([\Delta[\boldsymbol{m}]_t + \boldsymbol{p_0}\boldsymbol{u} + \boldsymbol{e_1}]_q, [\boldsymbol{p_1}\boldsymbol{u} + \boldsymbol{e_2}]_q)$ (with $\Delta = \lfloor \frac{q}{t} \rfloor$)

$$[\boldsymbol{c}_0 + \boldsymbol{c}_1 \boldsymbol{s}]_q = [\Delta[\boldsymbol{m}]_t + \boldsymbol{v}]_q$$

$\boldsymbol{v}$: "fresh noise"

Full RNS variant of FV decryption

The decryption process

$(\boldsymbol{c}_0, \boldsymbol{c}_1)$ encrypting $[\boldsymbol{m}]_t$, with noise $\boldsymbol{v}$: $[\boldsymbol{c}_0 + \boldsymbol{c}_1\boldsymbol{s}]_q = \Delta[\boldsymbol{m}]_t + \boldsymbol{v} + q\boldsymbol{r}$

1. scale-down: $\frac{t}{q} \cdot [\boldsymbol{c}_0 + \boldsymbol{c}_1\boldsymbol{s}]_q = [\boldsymbol{m}]_t + \frac{\boldsymbol{v}'}{q} + t\boldsymbol{r}$
2. round-off: $\lfloor [\boldsymbol{m}]_t + \frac{\boldsymbol{v}'}{q} + t\boldsymbol{r} \rceil = [\boldsymbol{m}]_t + \lfloor \frac{\boldsymbol{v}'}{q} \rceil + t\boldsymbol{r}$

Bound on noise: $\|\boldsymbol{v}\|_\infty = \max(|\boldsymbol{v}_i|) < \frac{\Delta - |q|_t}{2} \Rightarrow \lfloor \frac{\boldsymbol{v}'}{q} \rceil = 0.$

$$\mathtt{Dec}(\boldsymbol{c}, \mathbf{sk}) = [\lfloor \tfrac{t}{q} \cdot [\boldsymbol{c}_0 + \boldsymbol{c}_1\boldsymbol{s}]_q \rceil]_t = [[\boldsymbol{m}]_t + t\boldsymbol{r}]_t = [\boldsymbol{m}]_t.$$

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

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

Full RNS variant of FV decryption

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 $\{t\}$:

$$\sum_{i=1}^{k} |tx\frac{q_i}{q}|_{q_i} \cdot |\frac{q}{q_i}|_t \bmod t = |tx|_q + \alpha q \bmod t \quad (\alpha \in [0,k) \cap \mathbb{Z})$$

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

Error E corrected $\Rightarrow$ correct decryption.

Full RNS variant of FV decryption

Correcting the error

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

If gap $\varepsilon > 0$ ($-\frac{q}{2} + \varepsilon \leqslant [tx]_q \leqslant \frac{q}{2} - \varepsilon$) then **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 $\gamma\varepsilon \geqslant k + \frac{1}{2}$ then $|\lfloor \gamma \frac{[tx]_q}{q} \rceil - E| < \frac{\gamma}{2}$

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

Strategy

1. Compute $\lfloor \gamma \frac{t}{q} \cdot x \rceil$ modulo t **and** modulo 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: $\quad \boldsymbol{s}^{(m)} \leftarrow \sum_{i=1}^{k} |\gamma t \frac{q_i}{q} \cdot (\boldsymbol{c_0} + \boldsymbol{c_1 s})|_{q_i} \times | - \frac{q}{q_i} q^{-1}|_m \bmod m$

3: **end for**

4: $\tilde{\boldsymbol{s}}^{(\gamma)} \leftarrow [\boldsymbol{s}^{(\gamma)}]_\gamma$

5: $\boldsymbol{m}^{(t)} \leftarrow [(\boldsymbol{s}^{(t)} - \tilde{\boldsymbol{s}}^{(\gamma)}) \times |\gamma^{-1}|_t]_t$

6: **return** $\boldsymbol{m}^{(t)}$

Contributions

- 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

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

Full RNS variant of FV multiplication

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. Computing $(\tilde{\boldsymbol{c}}_0, \tilde{\boldsymbol{c}}_1, \tilde{\boldsymbol{c}}_2) = (\boldsymbol{c}_0\boldsymbol{c}'_0, \boldsymbol{c}_0\boldsymbol{c}'_1 + \boldsymbol{c}'_0\boldsymbol{c}_1, \boldsymbol{c}_1\boldsymbol{c}'_1)$ over $\mathbb{Z}$ (lift).
2. division+round-off: $\hat{\boldsymbol{c}}_i = \left\lfloor \frac{t}{q} \cdot \tilde{\boldsymbol{c}}_i \right\rceil$.
   ("degree-2" ciphertext: $\hat{\boldsymbol{c}}_0 + \hat{\boldsymbol{c}}_1\boldsymbol{s} + \hat{\boldsymbol{c}}_2\boldsymbol{s}^2 = \Delta[\boldsymbol{m}_1\boldsymbol{m}_2] + \boldsymbol{v} \bmod q$)

Full RNS variant of FV multiplication

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. Computing $(\tilde{\boldsymbol{c}}_0, \tilde{\boldsymbol{c}}_1, \tilde{\boldsymbol{c}}_2) = (\boldsymbol{c}_0\boldsymbol{c}'_0, \boldsymbol{c}_0\boldsymbol{c}'_1 + \boldsymbol{c}'_0\boldsymbol{c}_1, \boldsymbol{c}_1\boldsymbol{c}'_1)$ over $\mathbb{Z}$ (lift).
2. division+round-off: $\hat{\boldsymbol{c}}_i = \left\lfloor \frac{t}{q} \cdot \tilde{\boldsymbol{c}}_i \right\rceil$.
   ("degree-2" ciphertext: $\hat{\boldsymbol{c}}_0 + \hat{\boldsymbol{c}}_1\boldsymbol{s} + \hat{\boldsymbol{c}}_2\boldsymbol{s}^2 = \Delta[\boldsymbol{m}_1\boldsymbol{m}_2] + \boldsymbol{v} \bmod q$)
3. Relinearizing: $(\hat{\boldsymbol{c}}_0 + \hat{\boldsymbol{c}}_2\boldsymbol{s}^2, \hat{\boldsymbol{c}}_1) \xrightarrow{\boldsymbol{s}\ \ private} (\hat{\boldsymbol{c}}_0 + \hat{\boldsymbol{c}}_2(\boldsymbol{s}^2 + \boldsymbol{e} + \boldsymbol{a}\boldsymbol{s}), \hat{\boldsymbol{c}}_1 - \boldsymbol{a}\hat{\boldsymbol{c}}_2)$.
   - But large noise $\|\hat{\boldsymbol{c}}_2 \times \boldsymbol{e}\|_\infty < q \times nB_{\mathtt{err}}$! Original solution is to...

Full RNS variant of FV multiplication

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. Computing $(\tilde{\boldsymbol{c}}_0, \tilde{\boldsymbol{c}}_1, \tilde{\boldsymbol{c}}_2) = (\boldsymbol{c}_0\boldsymbol{c}'_0, \boldsymbol{c}_0\boldsymbol{c}'_1 + \boldsymbol{c}'_0\boldsymbol{c}_1, \boldsymbol{c}_1\boldsymbol{c}'_1)$ over $\mathbb{Z}$ (lift).
2. division+round-off: $\hat{\boldsymbol{c}}_i = \lfloor \frac{t}{q} \cdot \tilde{\boldsymbol{c}}_i \rceil$.
   ("degree-2" ciphertext: $\hat{\boldsymbol{c}}_0 + \hat{\boldsymbol{c}}_1\boldsymbol{s} + \hat{\boldsymbol{c}}_2\boldsymbol{s}^2 = \Delta[\boldsymbol{m}_1\boldsymbol{m}_2] + \boldsymbol{v} \bmod q$)
3. Relinearizing: $(\hat{\boldsymbol{c}}_0 + \hat{\boldsymbol{c}}_2\boldsymbol{s}^2, \hat{\boldsymbol{c}}_1) \xrightarrow{\boldsymbol{s} \text{ private}} (\hat{\boldsymbol{c}}_0 + \hat{\boldsymbol{c}}_2(\boldsymbol{s}^2 + \boldsymbol{e} + \boldsymbol{a}\boldsymbol{s}), \hat{\boldsymbol{c}}_1 - \boldsymbol{a}\hat{\boldsymbol{c}}_2)$.
   - But large noise $\|\hat{\boldsymbol{c}}_2 \times \boldsymbol{e}\|_\infty < q \times nB_{\mathtt{err}}$! Original solution is to...
   - decompose $\hat{\boldsymbol{c}}_2 = \boldsymbol{b}_0 + \boldsymbol{b}_1\omega + \ldots + \boldsymbol{b}_{\ell-1}\omega^{\ell-1}$ in radix ω.

Full RNS variant of FV multiplication

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. Computing $(\tilde{\boldsymbol{c}}_0, \tilde{\boldsymbol{c}}_1, \tilde{\boldsymbol{c}}_2) = (\boldsymbol{c}_0\boldsymbol{c}'_0, \boldsymbol{c}_0\boldsymbol{c}'_1 + \boldsymbol{c}'_0\boldsymbol{c}_1, \boldsymbol{c}_1\boldsymbol{c}'_1)$ over $\mathbb{Z}$ (lift).
2. division+round-off: $\hat{\boldsymbol{c}}_i = \left\lfloor \frac{t}{q} \cdot \tilde{\boldsymbol{c}}_i \right\rceil$.
   ("degree-2" ciphertext: $\hat{\boldsymbol{c}}_0 + \hat{\boldsymbol{c}}_1\boldsymbol{s} + \hat{\boldsymbol{c}}_2\boldsymbol{s}^2 = \Delta[\boldsymbol{m}_1\boldsymbol{m}_2] + \boldsymbol{v} \bmod q$)
3. Relinearizing: $(\hat{\boldsymbol{c}}_0 + \hat{\boldsymbol{c}}_2\boldsymbol{s}^2, \hat{\boldsymbol{c}}_1) \xrightarrow{\boldsymbol{s}\ \textit{private}} (\hat{\boldsymbol{c}}_0 + \hat{\boldsymbol{c}}_2(\boldsymbol{s}^2 + \boldsymbol{e} + \boldsymbol{a}\boldsymbol{s}), \hat{\boldsymbol{c}}_1 - \boldsymbol{a}\hat{\boldsymbol{c}}_2)$.
   - But large noise $\|\hat{\boldsymbol{c}}_2 \times \boldsymbol{e}\|_\infty < q \times nB_{\mathtt{err}}$! Original solution is to...
   - decompose $\hat{\boldsymbol{c}}_2 = \boldsymbol{b}_0 + \boldsymbol{b}_1\omega + \ldots + \boldsymbol{b}_{\ell-1}\omega^{\ell-1}$ in radix ω.
   - Public key: $\mathtt{rlk} = (\boldsymbol{s}^2 \cdot (1, \omega, \ldots, \omega^{\ell-1}) + \vec{\boldsymbol{e}} + \vec{\boldsymbol{a}}\,\boldsymbol{s}, -\vec{\boldsymbol{a}})$.

Full RNS variant of FV multiplication

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. Computing $(\tilde{\boldsymbol{c}}_0, \tilde{\boldsymbol{c}}_1, \tilde{\boldsymbol{c}}_2) = (\boldsymbol{c}_0 \boldsymbol{c}'_0, \boldsymbol{c}_0 \boldsymbol{c}'_1 + \boldsymbol{c}'_0 \boldsymbol{c}_1, \boldsymbol{c}_1 \boldsymbol{c}'_1)$ over $\mathbb{Z}$ (lift).
2. division+round-off: $\hat{\boldsymbol{c}}_i = \left\lfloor \frac{t}{q} \cdot \tilde{\boldsymbol{c}}_i \right\rceil$.
   (“degree-2” ciphertext: $\hat{\boldsymbol{c}}_0 + \hat{\boldsymbol{c}}_1 \boldsymbol{s} + \hat{\boldsymbol{c}}_2 \boldsymbol{s}^2 = \Delta[\boldsymbol{m}_1 \boldsymbol{m}_2] + \boldsymbol{v} \bmod q$)
3. Relinearizing: $(\hat{\boldsymbol{c}}_0 + \hat{\boldsymbol{c}}_2 \boldsymbol{s}^2, \hat{\boldsymbol{c}}_1) \xrightarrow{\boldsymbol{s}\ \textit{private}} (\hat{\boldsymbol{c}}_0 + \hat{\boldsymbol{c}}_2(\boldsymbol{s}^2 + \boldsymbol{e} + \boldsymbol{a}\boldsymbol{s}), \hat{\boldsymbol{c}}_1 - \boldsymbol{a}\hat{\boldsymbol{c}}_2)$.
   - But large noise $\|\hat{\boldsymbol{c}}_2 \times \boldsymbol{e}\|_\infty < q \times nB_{\mathtt{err}}$! Original solution is to...
   - decompose $\hat{\boldsymbol{c}}_2 = \boldsymbol{b}_0 + \boldsymbol{b}_1 \omega + \ldots + \boldsymbol{b}_{\ell-1}\omega^{\ell-1}$ in radix ω.
   - Public key: $\mathtt{rlk} = (\boldsymbol{s}^2 \cdot (1, \omega, \ldots, \omega^{\ell-1}) + \overrightarrow{\boldsymbol{e}} + \overrightarrow{\boldsymbol{a}}\boldsymbol{s}, -\overrightarrow{\boldsymbol{a}})$.
   - Smaller noise $\|(\boldsymbol{b}_0, \boldsymbol{b}_1, \ldots, \boldsymbol{b}_\ell) \cdot \overrightarrow{\boldsymbol{e}}\|_\infty < \ell\omega \times nB_{\mathtt{err}}$.

Full RNS variant of FV multiplication

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. Computing $(\tilde{\boldsymbol{c}}_0, \tilde{\boldsymbol{c}}_1, \tilde{\boldsymbol{c}}_2) = (\boldsymbol{c}_0\boldsymbol{c}_0', \boldsymbol{c}_0\boldsymbol{c}_1' + \boldsymbol{c}_0'\boldsymbol{c}_1, \boldsymbol{c}_1\boldsymbol{c}_1')$ over $\mathbb{Z}$ (lift).
2. division+round-off: $\hat{\boldsymbol{c}}_i = \lfloor \frac{t}{q} \cdot \tilde{\boldsymbol{c}}_i \rceil$.
   ("degree-2" ciphertext: $\hat{\boldsymbol{c}}_0 + \hat{\boldsymbol{c}}_1\boldsymbol{s} + \hat{\boldsymbol{c}}_2\boldsymbol{s}^2 = \Delta[\boldsymbol{m}_1\boldsymbol{m}_2] + \boldsymbol{v} \bmod q$)
3. Relinearizing: $(\hat{\boldsymbol{c}}_0 + \hat{\boldsymbol{c}}_2\boldsymbol{s}^2, \hat{\boldsymbol{c}}_1) \xrightarrow{\boldsymbol{s}\ \text{private}} (\hat{\boldsymbol{c}}_0 + \hat{\boldsymbol{c}}_2(\boldsymbol{s}^2 + \boldsymbol{e} + \boldsymbol{a}\boldsymbol{s}), \hat{\boldsymbol{c}}_1 - \boldsymbol{a}\hat{\boldsymbol{c}}_2)$.
   - But large noise $\|\hat{\boldsymbol{c}}_2 \times \boldsymbol{e}\|_\infty < q \times nB_{\mathtt{err}}$! Original solution is to...
   - decompose $\hat{\boldsymbol{c}}_2 = \boldsymbol{b}_0 + \boldsymbol{b}_1\omega + \ldots + \boldsymbol{b}_{\ell-1}\omega^{\ell-1}$ in radix ω.
   - Public key: $\mathtt{rlk} = (\boldsymbol{s}^2 \cdot (1, \omega, \ldots, \omega^{\ell-1}) + \vec{\boldsymbol{e}} + \vec{\boldsymbol{a}}\boldsymbol{s}, -\vec{\boldsymbol{a}})$.
   - Smaller noise $\|(\boldsymbol{b}_0, \boldsymbol{b}_1, \ldots, \boldsymbol{b}_\ell) \cdot \vec{\boldsymbol{e}}\|_\infty < \ell\omega \times nB_{\mathtt{err}}$.

Issues for RNS representation

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

Full RNS variant of FV multiplication

Problem 1: Computing the products $(\tilde{\boldsymbol{c}}_0, \tilde{\boldsymbol{c}}_1, \tilde{\boldsymbol{c}}_2)$

$\|\tilde{\boldsymbol{c}}_i\|_\infty <\sim nq^2$: no lift in $\mathbb{Z}$, just use a larger base than $\{q_1, \ldots, q_k\}$

Full RNS variant of FV multiplication

Problem 1: Computing the products $(\tilde{\boldsymbol{c}}_0, \tilde{\boldsymbol{c}}_1, \tilde{\boldsymbol{c}}_2)$

$\|\tilde{\boldsymbol{c}}_i\|_\infty <\sim nq^2$: no lift in $\mathbb{Z}$, just use a larger base than $\{q_1, \ldots, q_k\}$

Solutions: Introducing a second RNS base $\mathcal{B}$

Step	base $q_1, \ldots, q_k$		base $\mathcal{B} \cup \{\tilde{m}\}$
			to be reduced
0	$\boldsymbol{c}_i, \boldsymbol{c}'_j$		
1	$\boldsymbol{c}_i, \boldsymbol{c}'_j$	$\xrightarrow[\text{extension}]{\text{fast approximate}}$	$[\boldsymbol{c}_i]_q + q\boldsymbol{u}_i, [\boldsymbol{c}'_j]_q + q\boldsymbol{u}'_j$
2			$\hat{\boldsymbol{c}}_i \leftarrow \texttt{MRed\_q}_{\tilde{m}}([\boldsymbol{c}_i]_q + q\boldsymbol{u}_i)$
3	$\tilde{\boldsymbol{c}}_0 \leftarrow \boldsymbol{c}_0 \times \boldsymbol{c}'_0$, etc		$\tilde{\boldsymbol{c}}_0 \leftarrow \hat{\boldsymbol{c}}_0 \times \hat{\boldsymbol{c}}'_0$, etc

- Only **fast approximate** RNS base extensions to get $[\boldsymbol{c}_i]_q, [\boldsymbol{c}'_j]_q$ in $\mathcal{B}$,
- $+$ **low cost** Montgomery **reductions** of the approximations "$\boldsymbol{u}_i, \boldsymbol{u}'_j$".

Full RNS variant of FV multiplication

Problem 2: Division and round-off $\hat{c}_i = \lfloor \frac{t}{q} \cdot \tilde{c}_i \rceil$ in RNS

Context $\neq$ decryption: no large enough gap $\frac{q}{2} - \|[\tilde{c}_i]_q\| \geqslant \varepsilon > 0$, **no** guaranteed **correct RNS round-off**.

Full RNS variant of FV multiplication

Problem 2: Division and round-off $\hat{\boldsymbol{c}}_i = \left\lfloor \frac{t}{q} \cdot \tilde{\boldsymbol{c}}_i \right\rceil$ in RNS

Context $\neq$ decryption: no large enough gap $\frac{q}{2} - \|[\tilde{\boldsymbol{c}}_i]_q\| \geqslant \varepsilon > 0$, **no** guaranteed **correct RNS round-off**.

Solution: Fast approximate RNS round-off

Step	base $q_1, \ldots, q_k$		base $\mathcal{B} \cup \{m\}$
0	$\tilde{\boldsymbol{c}}_i$		$\tilde{\boldsymbol{c}}_i$
1	$t \cdot \tilde{\boldsymbol{c}}_i$	$\xrightarrow[\text{extension}]{\text{fast approximate}}$	$\left\|t \cdot \tilde{\boldsymbol{c}}_i\right\|_q + q\tilde{\boldsymbol{u}}_i$
2			$\hat{\boldsymbol{c}}_i \longleftarrow \frac{t \cdot \tilde{\boldsymbol{c}}_i - (\|t \cdot \tilde{\boldsymbol{c}}_i\|_q + q\tilde{\boldsymbol{u}}_i)}{q}$
3	$\hat{\boldsymbol{c}}_i = \left\lfloor \frac{t}{q} \cdot \tilde{\boldsymbol{c}}_i \right\rceil + \boldsymbol{e}_i$	$\xleftarrow[\text{extension}]{\text{efficient exact}}$	$\hat{\boldsymbol{c}}_i, \left\|\hat{\boldsymbol{c}}_i\right\|_m$
	$(\|\boldsymbol{e}_i\|_\infty \leqslant k)$		(use $\|\hat{\boldsymbol{c}}_i\|_m$ to correct the fast extension efficiently)

Approximate round-off $\Rightarrow$ **new** analysis of **noise growth** provided.

Full RNS variant of FV multiplication

Problem 3: Access to positional system

$\hat{\boldsymbol{c}}_2 = \boldsymbol{b}_0 + \boldsymbol{b}_1\omega + \ldots + \boldsymbol{b}_{\ell-1}\omega^{\ell-1}$ in positional system with radix ω

Recall: replace $\hat{\boldsymbol{c}}_2 \times \boldsymbol{e} \overset{\|\cdot\|}{\sim} q \times nB_{\mathtt{err}} \rightsquigarrow (\boldsymbol{b}_0, \boldsymbol{b}_1, \ldots, \boldsymbol{b}_\ell) \cdot \overrightarrow{\boldsymbol{e}} \overset{\|\cdot\|}{\sim} \ell\omega \times nB_{\mathtt{err}}$

Full RNS variant of FV multiplication

Problem 3: Access to positional system

$\hat{\boldsymbol{c}}_2 = \boldsymbol{b}_0 + \boldsymbol{b}_1\omega + \ldots + \boldsymbol{b}_{\ell-1}\omega^{\ell-1}$ in positional system with radix ω

Recall: replace $\hat{\boldsymbol{c}}_2 \times \boldsymbol{e} \overset{\|.\|}{\sim} q \times nB_{\mathtt{err}} \rightsquigarrow (\boldsymbol{b}_0, \boldsymbol{b}_1, \ldots, \boldsymbol{b}_\ell) \cdot \vec{\boldsymbol{e}} \overset{\|.\|}{\sim} \ell\omega \times nB_{\mathtt{err}}$

Solution: Just use RNS representation...

If $\omega \sim q_i$ (i.e. $\ell = k$), use RNS representation for a fairly equivalent effect.

$$\begin{array}{rcll} \hat{\boldsymbol{c}}_2 & = & \boldsymbol{b}_0 + \boldsymbol{b}_1\omega + \ldots + \boldsymbol{b}_{\ell-1}\omega^{\ell-1} & (\boldsymbol{b}_i = |\lfloor \hat{\boldsymbol{c}}_2\omega^{-i} \rfloor|_\omega) \\ \hat{\boldsymbol{c}}_2 & = & \boldsymbol{d}_1\frac{q}{q_1} + \boldsymbol{d}_2\frac{q}{q_2} + \ldots + \boldsymbol{d}_k\frac{q}{q_k} \bmod q & (\boldsymbol{d}_i = |\hat{\boldsymbol{c}}_2\frac{q_i}{q}|_{q_i}) \end{array}$$

$$\|\boldsymbol{b}_i\|_\infty \sim \|\boldsymbol{d}_i\|_\infty \Rightarrow \|(\boldsymbol{b}_0, \boldsymbol{b}_1, \ldots, \boldsymbol{b}_{\ell-1}) \cdot \vec{\boldsymbol{e}}\|_\infty \sim \|(\boldsymbol{d}_1, \boldsymbol{d}_2, \ldots, \boldsymbol{d}_k) \cdot \vec{\boldsymbol{e}}\|_\infty$$

$$\Rightarrow \text{bound } \ell\omega \times nB_{\mathtt{err}} \text{ becomes } k\omega \times nB_{\mathtt{err}}$$

Public key: $\boldsymbol{s}^2 \cdot (1, \omega, \ldots, \omega^{\ell-1}) + \vec{\boldsymbol{e}} + \vec{\boldsymbol{a}}\,\boldsymbol{s} \rightsquigarrow \boldsymbol{s}^2 \cdot (\frac{q}{q_1}, \ldots, \frac{q}{q_k}) + \vec{\boldsymbol{e}} + \vec{\boldsymbol{a}}\,\boldsymbol{s}$

Full RNS variant of FV multiplication

Contributions

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

Experiments

Software implementation

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

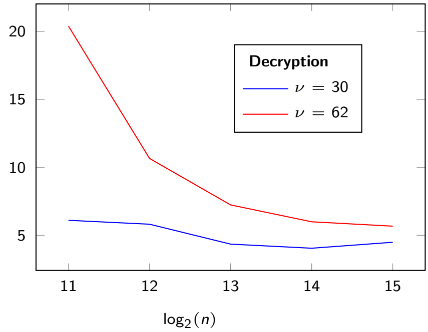
[a] https://github.com/CryptoExperts/FV-NFLlib

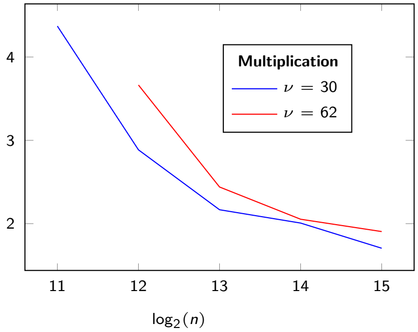
Experiments - Speed-up factors

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

- Optimization of arithmetic on polynomials at the coefficient level.
  - Benefits to SHE schemes like `FV`.
  - No more need of any positional system: only RNS.
- Possible greater noise growth, but not that significant in practice.
- Opens the door to highly competitive parallel implementation of homomorphic encryption.