Bootstrapping with RMFE for Fully Homomorphic Encryption

Khin Mi Mi Aung[1], Enhui Lim[2], Jun Jie Sim,

Benjamin Hong Meng Tan[1], Huaxiong Wang[2]

[1] Agency for Science, Technology and Research (A*STAR), Singapore
[2] Nanyang Technological University (NTU), Singapore

Galois Rings

Intuition: Generalisation of finite fields to characteristic p^r

- Let $f(X) \in \mathbb{Z}_{p^r}[X]$ be a monic irreducible polynomial with degree d.
- Then $\mathbb{Z}_{p^r}[X]/(f(X))$ is a Galois ring, denoted $GR(p^r, d)$.

Tower structure

- Let $d = sw$. Then $GR(p^r, s) \subseteq GR(p^r, d)$, and w is the relative extension degree
- Frobenius automorphism σ on $GR(p^r, d)$ fixes $\mathbb{Z}_{p^r}$

Extra operations

- Mod reduction by powers of p, division by p

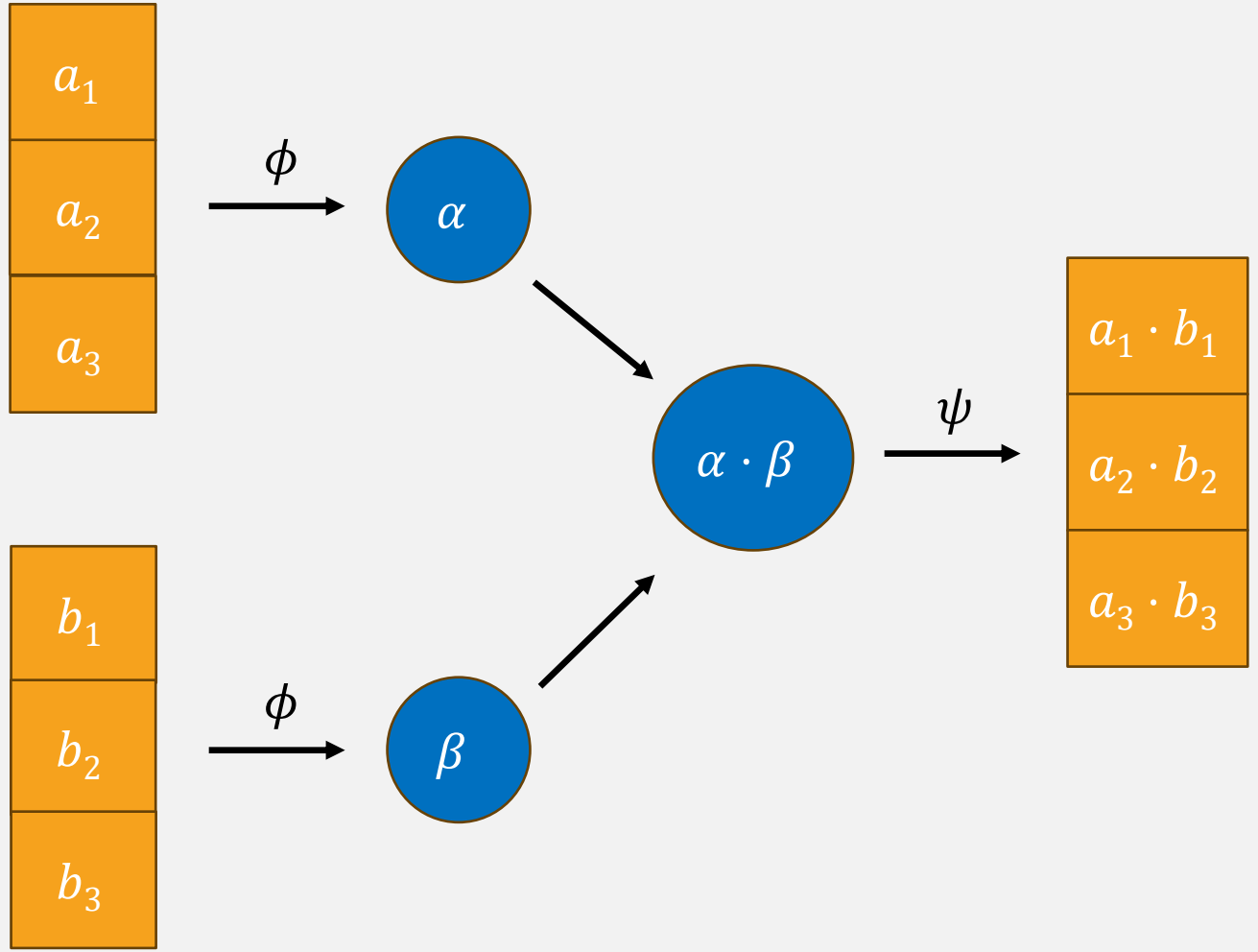
Reverse Multiplication-Friendly Embeddings (RMFE)

Let $G \subset G'$ be Galois rings

An RMFE is a pair of maps (ϕ, ψ):

$\phi: G^k \to G'$ (Encode)
$\psi: G' \mapsto G^k$ (Decode)
Preserves a finite number of multiplications

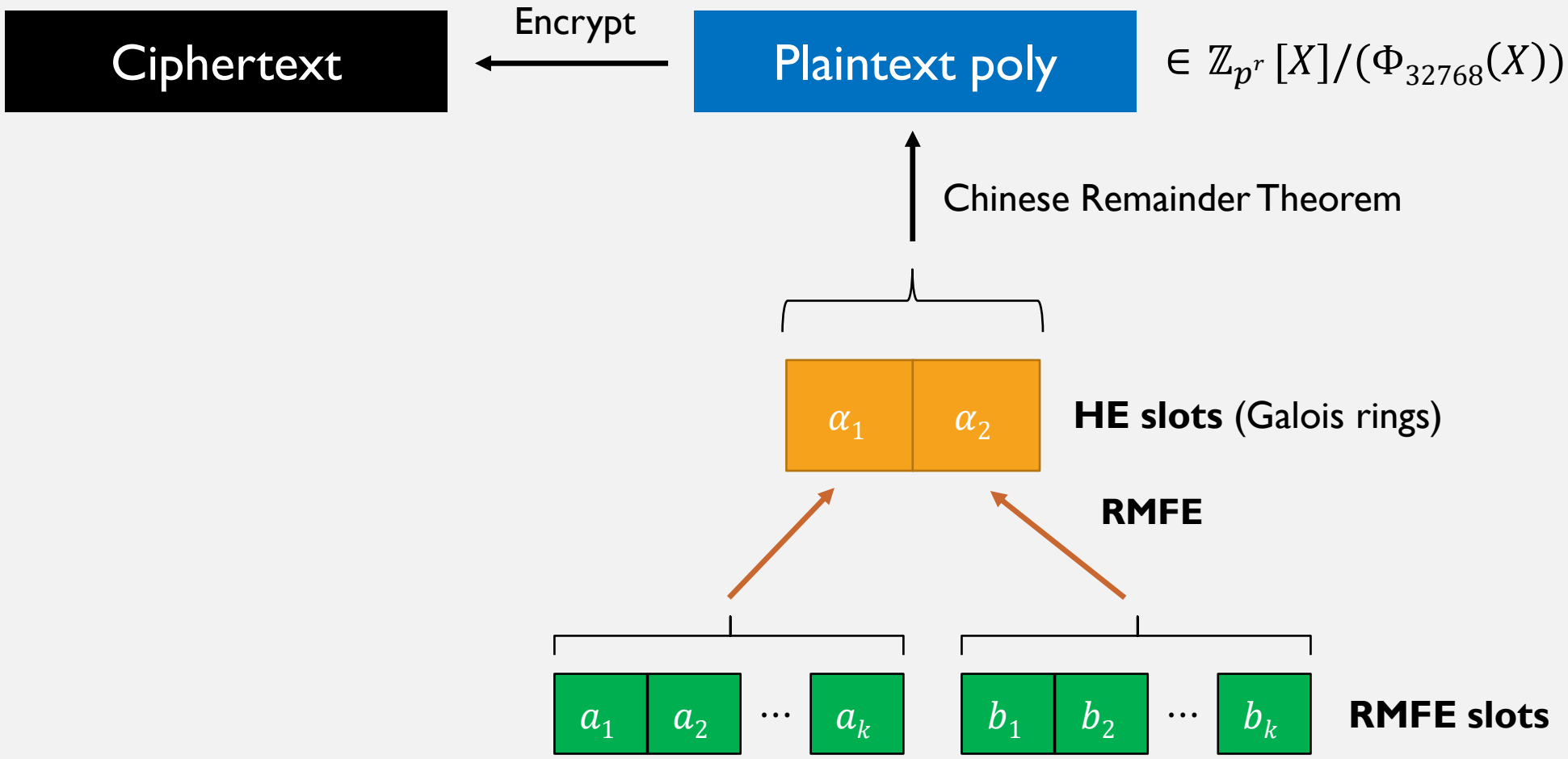
For more multiplications:
Recode $\pi = \phi \circ \psi: G' \to G'$



Homomorphic Encryption

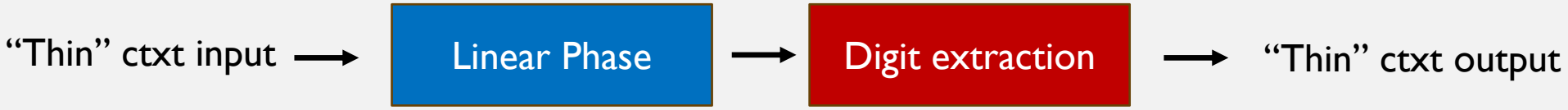
- Enables computation on ciphertext without decrypting
  - $Enc(m_1) + Enc(m_2) = Enc(m_1 + m_2)$
  - $Enc(m_1) \cdot Enc(m_2) = Enc(m_1 \cdot m_2)$
- **BGV** [2] and **BFV** [1,4] schemes enable Galois ring arithmetic (in particular arithmetic over $\mathbb{Z}_{p^r}$)
- **Bootstrapping** needed to control ciphertext noise, for more multiplications

Field Instruction Multiple Data (2022)



This work: How to bootstrap an RMFE-packed ciphertext?

- "Thin" bootstrapping: only integers in HE slots

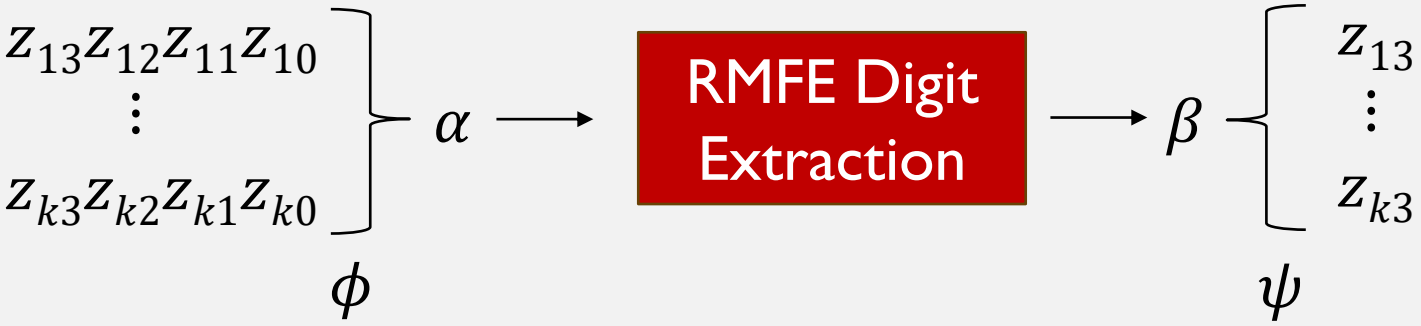


- **Goal**: adapt this pipeline to work for RMFE-packed input
- **Challenge**: adapt Digit Extraction to work on RMFE-packed Galois ring elements

Adapting Digit Extraction

Before:

Our work:



Digit Extraction

Halevi-Shoup [6]

Input integer $z \xrightarrow{F} F(z) \xrightarrow{F} F^2(z) \xrightarrow{F} F^3(z)$

$z' \xrightarrow{F} F(z') \xrightarrow{F} F^2(z')$

$z'' \xrightarrow{F} F(z'')$

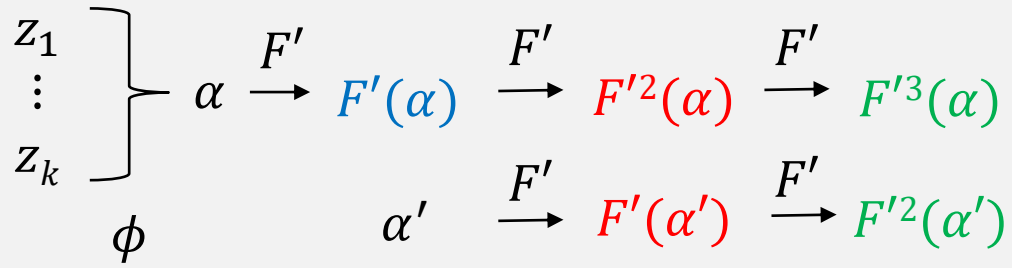
Return β

- $z' := \frac{z - F(z)}{p}$
- $z'' := \frac{\frac{z - F^2(z)}{p} - F(z')}{p}$
- $\beta := \frac{\frac{\frac{z - F^3(z)}{p} - F^2(z')}{p} - F(z'')}{p}$

RMFE Digit Extraction (Naïve)

Just RMFE recode after every evaluation of F

- $F' := \pi \circ F$



Return β

- $\alpha' := \frac{\alpha - F'(\alpha)}{p}$
- $\alpha'' := \frac{\frac{\alpha - F'^2(\alpha)}{p} - F'(\alpha')}{p}$
- $\beta := \frac{\frac{\frac{\alpha - F'^3(\alpha)}{p} - F'^2(\alpha')}{p} - F'(\alpha'')}{p}$

RMFE Digit Extraction (Our Solution)

Correction maps: for canonical vectors $e_1, \dots, e_k$

$$L_i : \phi(e_j) \mapsto \phi(e_j)^{p^i}$$

$$\begin{matrix} z_1 \\ \vdots \\ z_k \end{matrix} \Big\} \; \phi \qquad \alpha \xrightarrow{F} F(\alpha) \xrightarrow{F} F^2(\alpha) \xrightarrow{F} F^3(\alpha)$$

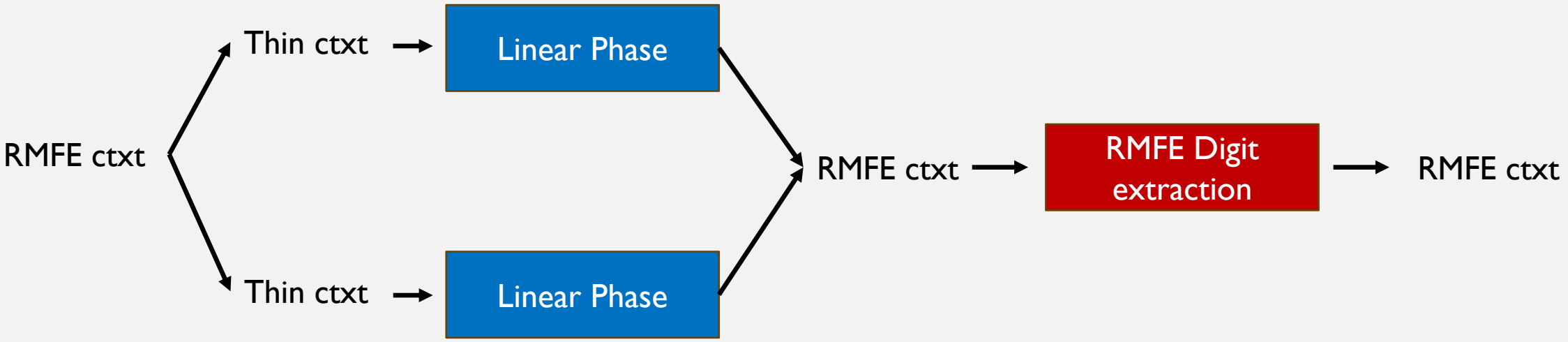
$$\alpha' \xrightarrow{F} F(\alpha') \xrightarrow{F} F^2(\alpha')$$

$$\alpha'' \xrightarrow{F} F(\alpha'')$$

Return $\pi(\beta)$

- $\alpha' := \frac{\boldsymbol{L_1(\alpha)} - F(\alpha)}{p}$
- $\alpha'' := \frac{\frac{\boldsymbol{L_2(\alpha)} - F^2(\alpha)}{p} - F(\alpha')}{p}$
- $\beta := \frac{\frac{\frac{\boldsymbol{L_3(\alpha)} - F^3(\alpha)}{p} - F^2(\alpha')}{p} - F(\alpha'')}{p}$

RMFE Bootstrapping



Implementation

Implemented 4 RMFE bootstrapping workflows in C++ with HElib:

Two RMFE digit extraction strategies:

"Naïve" style – recode multiple times throughout the algorithm

- Better packing, impractically low multiplications, high latency

"Correction" style – introduce new *correction maps* to the algorithm

- More multiplications, lower latency, less efficient packing

Two recode optimisations (naively w-many automorphisms)

- BSGS: $\sim 2\sqrt{w}$ many automorphisms
- Trace-and-Merge: $\sim k \log w$ many automorphisms

Sample Results

Prime p	Workflow Style/Variant	#Slots $k \times \ell$	Remaining capacity	Remaining squarings	Latency (sec)	Throughput capacity × slots/latency
3	HE Thin	2	179	10	2.14	167
	Corr / TnM	14	136	7	14.9	128
	Corr / BSGS	14	150	8	16.7	126
	Naïve / BSGS	342	67	2	342	67
5	HE Thin	2	151	8	2.82	107
	Corr / TnM	10	101	5	11.3	89
	Corr / BSGS	10	115	6	15.0	76
	Naïve / BSGS	206	58	2	179	66

Future Work

Current work is foundational, not yet competitive with state of the art.

- "Hybrid" style using correction maps and recoding
- Do one linear phase instead of multiple
- Extend correction map strategy to more recent digit extraction algorithms
  - Chen-Han (2018) [3]
  - Geelen-Iliashenko-Kang-Vercauteren (2023) [5]

References

1. Brakerski, Z.: Fully homomorphic encryption without modulus switching from classical GapSVP. In: CRYPTO 2012. LNCS, vol. 7417, pp. 868–886 (2012)
2. Brakerski, Z., Gentry, C., Vaikuntanathan, V.: (Leveled) fully homomorphic encryption without bootstrapping. In: ITCS 2012. pp. 309–325. ACM (2012)
3. Chen, H., Han, K.: Homomorphic lower digits removal and improved FHE bootstrapping. In: EUROCRYPT 2018, Part I. LNCS, vol. 10820, pp. 315–337. Springer, Cham (2018)
4. Fan, J., Vercauteren, F.: Somewhat practical fully homomorphic encryption. Cryptology ePrint Archive, Paper 2012/144 (2012)
5. Geelen, R., Iliashenko, I., Kang, J., Vercauteren, F.: On polynomial functions modulo p^e and faster bootstrapping for homomorphic encryption. In: EUROCRYPT 2023, Part III. LNCS, vol 14006, pp. 257-286 (2023)
6. Halevi, S., Shoup, V.: Bootstrapping for HElib. In: EUROCRYPT 2015, Part I. LNCS, vol. 9056, pp. 641–670. Springer, Berlin, Heidelberg (2015), Updated in 2020 at https://eprint.iacr.org/2014/873

Thank You!