Homomorphic polynomial evaluation using Galois structure and applications to BFV bootstrapping

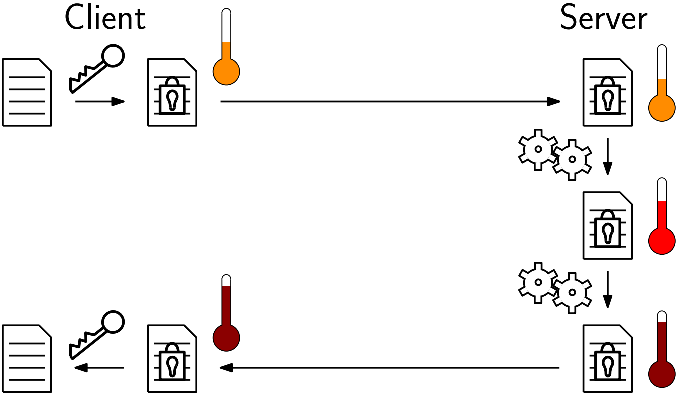
Hiroki Okada[2] Rachel Player[1] Simon Pohmann[1]

Royal Holloway, University of London, UK

KDDI Research, Japan

November 30, 2023

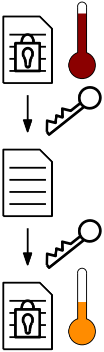
Homomorphic Encryption and Bootstrapping



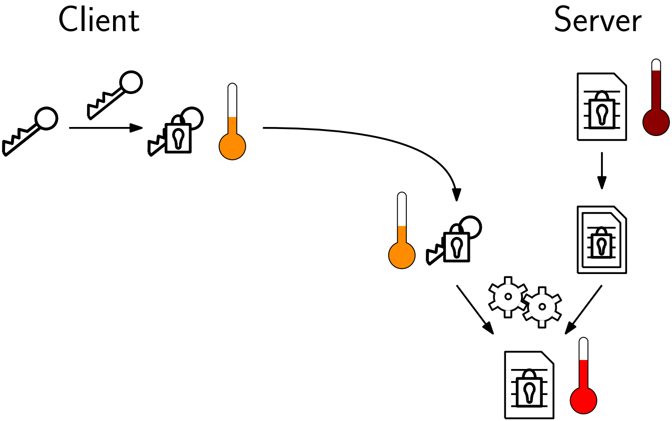
Homomorphic Encryption and Bootstrapping

Client

Server



Homomorphic Encryption and Bootstrapping



BGV/BFV

- RLWE-based
- Uses cyclotomic rings $R = \mathbb{Z}[X]/(\Phi_m(X))$

	BFV	BGV
Plaintexts	$m \in R_t$	
	possibly $R_t \cong S_1 \oplus ... \oplus S_n$	
Ciphertexts	$(c_0, c_1) \in R_q^2$	
Secret key	$s \in R_q$	
Hom. Operations	$+$, $\cdot$, action of $\mathrm{Gal}(R/\mathbb{Z})$	
Decryption	$\left\lfloor \frac{t}{q}(c_0 + c_1 s) \right\rceil$	$(c_0 + c_1 s) \mod t$

- Homormorphic computation of "digit extraction" necessary

Digit Extraction

- Assume $t = p$
- Extract least significant p-adic digit

$$\mathbb{Z}_{p^e} \to \mathbb{Z}_{p^e}, \quad \sum_{i=0}^{e-1} a_i p^i \mapsto a_0$$

Option 1

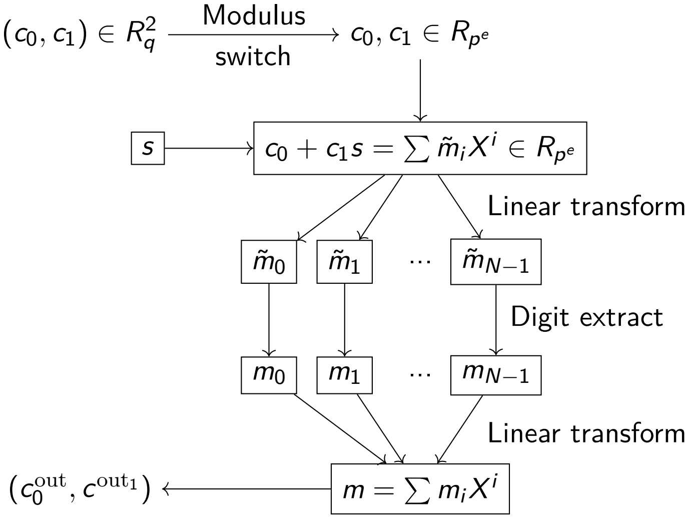
"Lifting polynomials" [HS21]

Option 2

"Digit retain polynomials" [CH18]

$\Rightarrow$ Have to evaluate a polynomial $f \in \mathbb{Z}[X]$ in $x \in \mathbb{Z}_{p^e}$

Bootstrapping



Improving polynomial evaluation

The setting

- Given $f \in \mathbb{Z}[X]$ and $x \in \mathbb{Z}_{p^e}$, (homomorphically) compute $f(x)$
- We are in a "plaintext slot" S_i (if $e = 1$ then $S_i \cong \mathbb{F}_{p^d}$)

$$R_{p^e} \cong S_1 \oplus \ldots \oplus S_n$$

The norm

$$N(\alpha) := \prod_{i=0}^{d-1} \pi^i(\alpha)$$

Frobenius automorphism

Improving polynomial evaluation

The setting

- Given $f \in \mathbb{Z}[X]$ and $x \in \mathbb{Z}_{p^e}$, (homomorphically) compute $f(x)$
- We are in a "plaintext slot" S_i (if $e = 1$ then $S_i \cong \mathbb{F}_{p^d}$)

$$R_{p^e} \cong S_1 \oplus \ldots \oplus S_n$$

The norm

$$N(\alpha) := \prod_{i=0}^{d-1} \pi^i(\alpha)$$

Frobenius automorphism

Observation 1: If $x \in \mathbb{F}_p$ and $\alpha \in \mathbb{F}_{p^d}$, then

$$N(\alpha - x) = \mathrm{MinPoly}(\alpha)(x)$$

Improving polynomial evaluation

The setting

- Given $f \in \mathbb{Z}[X]$ and $x \in \mathbb{Z}_{p^e}$, (homomorphically) compute $f(x)$
- We are in a "plaintext slot" S_i (if $e = 1$ then $S_i \cong \mathbb{F}_{p^d}$)

$$R_{p^e} \cong S_1 \oplus \ldots \oplus S_n$$

The norm

$$N(\alpha) := \prod_{i=0}^{d-1} \pi^i(\alpha)$$

Frobenius automorphism

Observation 2: We can compute $N(\alpha)$ as

$$\left.\begin{aligned} \alpha_0 &:= \alpha \\ \alpha_1 &:= \alpha_0 \cdot \pi(\alpha_0) = \alpha \cdot \pi(\alpha) \\ \alpha_2 &:= \alpha_1 \cdot \pi^2(\alpha_1) = \alpha \cdot \pi(\alpha) \cdot \pi^2(\alpha) \cdot \pi^3(\alpha) \\ &\vdots \end{aligned}\right\} \text{log } d \text{ mults!}$$

Improved evaluation

Observation 1

$N(\alpha - x) = \mathrm{MinPoly}(\alpha)(x)$

Observation 2

Can compute $N(\alpha)$ with $\log(d)$ multiplications

If we find $\alpha \in \mathbb{F}_{p^d}$ such that
$\mathrm{MinPoly}(\alpha) =$ Lifting Poly

- Requires deg(poly) $\leq d$
  - For lifting polynomials: $\deg(\mathrm{poly}) = p$
- Digit extraction in $\log(p)$ mults!
- Paterson Stockmeyer needs $2\sqrt{p}$ mults
- Can be used for many polynomials!

It is faster!

Setting

"power-of-two cyclotomics" $\Phi_m = X^N + 1$

- previously considered by [CH18]
- less slots/higher rank than other cases
- better performance

$$\Phi_m = X^{2^{15}} + 1, \quad p = 257, \quad d = 256, \quad n = 128, \quad e = 2$$

	Key switches	Time (our impl)	Time [CH18]
Lin. Transform 1	22	7.9s	-
Lin. Transform 2	30	8.6s	-
Digit Extract	17	5.6s	-
Total	69	22.1s	36.8s

(timings for slim bootstrapping)

Future directions

Other parameter settings!

- Digit retain polynomials
- Recent optimizations [CH18; Gee+23]

Other applications!

- Evaluating multiple polynomials

Thank you for your attention!

[CH18] Hao Chen and Kyoohyung Han. “Homomorphic Lower Digits Removal and Improved FHE Bootstrapping”. 2018.

[Gee+23] Robin Geelen, Ilia Iliashenko, Jiayi Kang, and Frederik Vercauteren. “On Polynomial Functions Modulo p^e and Faster Bootstrapping for Homomorphic Encryption”. 2023.

[HS21] Shai Halevi and Victor Shoup. “Bootstrapping for HElib”. (2021).