Large Domain Homomorphic Evaluation in Levelled Mode

Jean-Philippe Bossuat   Malika Izabachène

jeanphilippe.bossuat@gmail.com

malika.izabachene@gmail.com

Fully Homomorphic Encryption Conference - 2024

Disclaimer: The views and opinions expressed in this talk are our own and they do not necessarily represent the views and opinions of our employers.

TABLE OF CONTENTS

INTRODUCTION

- Functional bootstrapping[1] allows to evaluate arbitrary discretized functions which are *a priori* encoded
- However high precision/large domain functional bootstrapping quickly becomes prohibitive

In this talk, we propose a solution based on the *Split-Domain* approach from Iliashenko et al. [IIMP22], used for the computation of private scores over large domains.

[1] [BR15, BDF18, CIM19, BGGJ18], etc

OVERARCHING GOAL

Client $\leftrightarrow$ Server interaction:

1. User: selects $m_0, m_2, \cdots, m_{N-1} \in \mathbb{Z}_t$ with N large (or a subset of them where each one is taken for a very large domain);

2. Server: holds an arbitrary function $f : \mathbb{Z}_t \to \mathbb{Z}_{t'}$ with $t' \leq t$;

At the end of the interaction, the user receives the evaluations on the selected points $f(m_0), \cdots, f(m_{N-1})$, with the server learning nothing about the selected points.

NOTATIONS

- $\mathcal{R}_q = \mathbb{Z}_q[X]/(X^N + 1)$
- $a(X) = \sum_{i=0}^{N-1} a_i \cdot X^i$
- $\mathsf{coeffs}(a(X)) = \mathbf{a} = (a_0, a_1, \ldots, a_{N-1})$
- $\mathsf{RLWE}_s(m(X)) = (a, b) \in \mathcal{R}_q^2$, s.t. $as + b = m(X) + e(X)$
- $\mathsf{LWE}_s(m) = (\mathbf{a}, b) \in \mathbb{Z}_q^{N+1}$ s.t. $\langle a, s \rangle + b = m + e$

Split Domain approach, [IIMP22]

Allows the user to obtain the successive evaluations:

$$\mathsf{RLWE}(X^{f(m_0)}), \cdots, \mathsf{RLWE}(X^{f(m_{N-1})})$$

- $1_{\mathtt{H}}$: one-hot encoding over $[0, N-1]$ and $f : \mathbb{Z}_N \to \mathbb{Z}_N$.
- Matrix representation of f:

$$\mathbf{U}_f = \begin{pmatrix} 1_{\mathtt{H}}(f(0)) \\ 1_{\mathtt{H}}(f(1)) \\ \vdots \\ 1_{\mathtt{H}}(f(N-1)) \end{pmatrix}$$

Note that:

$$1_{\mathtt{H}}(m) \times \mathbf{U}_f = 1_{\mathtt{H}}(f(m)) = \mathsf{coeffs}(X^{f(m)})$$

- As $1_{\mathtt{H}}(m) = \mathsf{coeffs}(X^m)$, given $\mathsf{RLWE}(X^m)$ and $\mathbf{U}_f$, we could obtain the evaluations $\mathsf{RLWE}(X^{f(m)})$.

Split Domain approach, [IIMP22]

How to obtain RLWE$(X^{f(m)})$ given RLWE$(X^m) = (a, b) \in \mathcal{R}_q^2$ and $\mathbf{U}_f$?

$(a, b) \in \mathcal{R}_q^2$ can be expressed as $(\mathbf{A}, \mathbf{b}) \in \mathbb{Z}_q^{N\times N, 1\times N}$ with:

- $\mathbf{A}$: the anti-circulant Vandermond matrix representation of a
- $\mathbf{b}$: coeffs(b)

Given RLWE$(X^m) = (a, b)$ and $\mathbf{U}_f$, the server evaluates

$$(\mathbf{A}, \mathbf{b}) \times \mathbf{U}_f = (\mathbf{A}\mathbf{U}_f, \mathbf{b}\mathbf{U}_f) \in \mathbb{Z}_q^{N\times N, 1\times N}$$

This still decrypts correctly under coeffs(s), BUT $(\mathbf{A}\mathbf{U}_f, \mathbf{b}\mathbf{U}_f)$ is not a valid RLWE ciphertext because $\mathbf{A}\mathbf{U}_f$ is not an anti-circulant Vandermond matrix anymore.

Split Domain approach, [IIMP22]

- Until now we assumed that $\text{Dom}(f) \leq N$
- If $\text{Dom}(f) > N$, $\text{Dom}(f)$ can be expressed as a union of functions with disjoint domains, i.e. $\text{Dom}(f) = \bigcup_{i=0}^{k-1} \text{Dom}(f_i)$, with $k = \left\lceil \frac{\text{Dom}(f)}{N} \right\rceil$.
- Then the client sends a k sized vector with $\text{RLWE}(X^{m \mod N})$ at the $\lfloor m/k \rfloor$-th position and $\text{RLWE}(0)$ at the others.
- And the server evaluates $(\mathbf{A}\mathbf{U}_f, \mathbf{b}\mathbf{U}_f) = \sum_{i=0}^{k-1} (\mathbf{A}_i\mathbf{U}_{f_i}, \mathbf{b}_i\mathbf{U}_{f_i})$.

SOME OBSERVATIONS

- To convert back $(\mathbf{A}\mathbf{U}_f, \mathbf{b}\mathbf{U}_f)$ to an RLWE ciphertext, a *format-fixing* key composed of N switching keys (one for each bit of the secret) is required
- One key-switch is evaluated for each row of $\mathbf{A}\mathbf{U}_f$, acting as a homomorphic decryption coefficient by coefficient

SOME OBSERVATIONS

- The *format-fixing* step requires $\mathcal{O}(N)$ key-switching operations per point
- The calculation approach comes from the fact that the result is retrieved as in the exponent as RLWE$(X^{f(m)})$
- This format enables dense packing of the count of each point: RLWE$([\#f(m) = 0], [\#f(m) = 1] \cdot X, \ldots, [\#f(m) = N-1] \cdot X^{N-1})$, $\mathcal{O}(\sqrt{q})$ counts can be stored per coefficients[2]

[2] Assume $q \approx 2^k$ and $m = 1 \cdot 2^{k'} + e$, then we can perform $\mathcal{O}(2^{k'})$ additions before $e \geq 2^{k'}$ and $\mathcal{O}(2^{k-k'})$ additions before the message overflows q. The number of additions is maximized when $2^{k-k'} = 2^{k'} = \sqrt{q}$

OUR PROPOSAL

MOTIVATION

Our approach retrieves the successive evaluations as :

$$\mathsf{RLWE}(f(m_0)), \cdots, \mathsf{RLWE}(f(m_{N-1}))$$

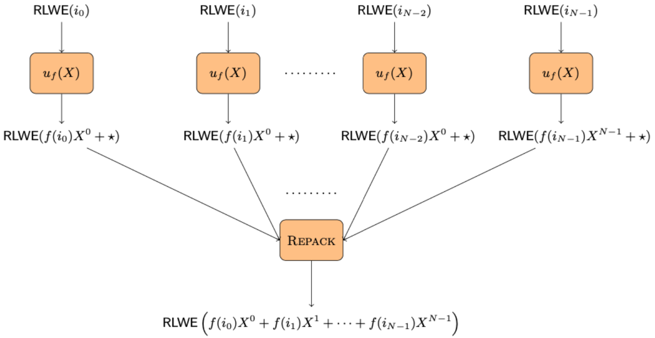
We will see how this enables:

- A reduced number of key-switching operations per point
- A reduced number of key-switching keys

HIGH LEVEL OVERVIEW

For a batch of N points

1. Define a test vector polynomial u_f
2. Evaluate u_f on each point
3. Repack the N points in a single RLWE ciphertext



CUSTOMIZED TEST VECTOR

- Step 1: The user encrypts $c_m = \mathsf{RLWE}(X^m)$ for each targeted element m_i and sends the ciphertext to the server.

- Step 2: The server defines a polynomial representation of the function f to be evaluated as follows:

$$u_f = f(0) - \sum_{i=1}^{N-1} f(N-i) \cdot X^i.$$

- Note that for each ciphertext sent by the client, we have:

$$c_m \cdot u_f = \mathsf{RLWE}(X^m) \cdot u_f = \mathsf{RLWE}(f(m)X^0 + \star)$$

- **Split Domain**: the same technique as [IIMP22] can be used

Repack: RLWE $\rightarrow$ LWE CONVERSION

- Let $m = \sum_{i=0}^{N-1} m_i \cdot X^i$
- Recall that $\text{RLWE}_s(m) = (a, b) \in \mathcal{R}_q^2$ can be expressed as $(\mathbf{A}, \mathbf{b}) \in \mathbb{Z}_q^{N\times N, 1\times N}$ with $\mathbf{A}$ the anti-circulant Vandermon matrix representation of a and $\mathbf{b} = \text{coeffs}(b)$
- Instead we can view $\text{RLWE}_s(m) = (a, b) \in \mathcal{R}_q^2$ as a structured set of N ciphertexts of the form $\text{LWE}_{\tilde{s}}(m_i)$, with $\tilde{s} = (s_0, -s_{N-1}, \dots, s_1)$
- Then the evaluation can be written as

$$\begin{pmatrix} \text{LWE}(\text{coeff}(a \cdot X^0), b[0]) \\ \vdots \\ \text{LWE}(\text{coeffs}(a \cdot X^{N-1}), b[N-1]) \end{pmatrix} \cdot \mathbf{U}_f$$

producing a new set of $n \leq N$ $\text{LWE}_{\tilde{s}}(m'_i)$ ciphertexts

Repack: LWE $\rightarrow$ RLWE CONVERSION

- We can convert an LWE ciphertext $\mathsf{LWE}_{\tilde{s}}(m_i) = (\mathbf{a}, b) \in \mathbb{Z}_q^{N+1}$ back to an RLWE ciphertext $\mathsf{RLWE}_s(m_i + \sum_{i=1}^{N} \star \cdot X^i) = (a, b)$ by setting

$$(a, b) = (\sum_{i=0}^{N-1} a_i X^i, b + \sum_{i=1}^{N-1} 0 \cdot X^i) \in \mathcal{R}_q^2$$

- Once we have N RLWE ciphertexts of the form $\mathsf{RLWE}(m_i + \sum_{i=1}^{N-1} \star \cdot X^i)$, we can repack them into a single RLWE ciphertext $\mathsf{RLWE}(\sum_{i=0}^{N-1} m_i \cdot X^i)$

RLWEs $\rightarrow$ RLWE REPACKING

$$\sum_{i=0}^{N-1} \mathsf{RLWE}(m_i + \sum_{i=1}^{N-1} \star \cdot X^i) \cdot X^i \xrightarrow{\mathsf{Repack}} \mathsf{RLWE}\left(\sum_{i=0}^{N-1} m_i \cdot X^i\right)$$

- RLWE repacking[3] requires $\mathcal{O}(N)$ automorphisms for N RLWE ciphertexts and makes use $\mathcal{O}(\log(N))$ key-switching keys

- Therefore for N points, this amortizes to $O(1)$ key-switching per point and a total of $\mathcal{O}(\log(N))$ switching keys, which is an asymptotic improvement for both the number of key-switching operations and key-switching keys:
  - Key-switching operations: $\mathcal{O}(N) \rightarrow \mathcal{O}(1)$
  - Key-switching keys: $\mathcal{O}(N) \rightarrow \mathcal{O}(\log N)$

[3] [CDKS21, KDE$^+$21]

RESPONSE FORMAT UPDATE

- The query size remains the same since the encoding of the points is unchanged: $\mathcal{O}(nkN\log(q))$, with e, n the number of points, $k = \left\lceil \frac{\mathsf{Dom}(f)}{N} \right\rceil$, N the ring degree and q the modulus

RESPONSE FORMAT UPDATE

- The response format is the main implication change of our modification, it changes from RLWE$(X^{f(m_i)})$ to RLWE$(f(m_i))$
- The response size is now proportional to $\lceil n/N \rceil N$ since each coefficient can only store one value, and we require that $q \geq |\mathsf{Img}(f)$. Therefor the response size from is changed from $\mathcal{O}(\lceil n/\sqrt{q} \rceil |\mathsf{Img}(f)| \log(q))$ to $\mathcal{O}(\lceil n/N \rceil N \log(|\mathsf{Img}(f)|))$
- In other words, the original solution is better when n is large and $|\mathsf{Img}(f)|$ is small, while the proposed solution is better when n is small (proportionally to q) and $|\mathsf{Img}(f)|$ is large (proportionally to N)

SUMMARY

Comparison summary of our split-domain approach with the original split-domain for a batch evaluation of n points. We let $k = |\text{Dom}(f)|/N$, where N is the ring degree of $\mathcal{R}$ and q is the ciphertext modulus.

	Query Size	Key-Switching Keys
Original [IIMP22]	$\mathcal{O}(nkN\log(q))$	$\mathcal{O}(N)$
Ours	$\mathcal{O}(nkN\log(q))$	$\mathcal{O}(\log(N))$

	Evaluation	Response Size
Original [IIMP22]	$\mathcal{O}_{\textsc{KeySwith}}(nN)$	$\mathcal{O}(\lceil n/\sqrt{q}\rceil \lvert\text{Img}(f)\rvert \log(q))$
Ours	$\mathcal{O}_{\textsc{KeySwith}}(n)$	$\mathcal{O}(\lceil n/N\rceil N\log(\lvert\text{Img}(f)\rvert))$

EXPERIMENTS

EXPERIMENTS - IMPLEMENTATION

We implemented our solution based on the revisited *Split-Domain* approach from [IIMP22] using the LATTIGO library

https://github.com/tuneinsight/lattigo

and the code is available at

https://github.com/Pro7ech/fhe-org-2024

EXPERIMENTS - PERFORMANCE

- Performance[4] of our revisited split-domain approach for an univariate function $f(x)$;
- Timings and data size are reported for batches of 2048 points:

$\mathrm{Dom}(f)$	$\mathrm{Img}(f)$	Encryption [sec]	Query [MB]	Evaluation [sec]	Response [KB]	Keys [MB]
$\mathbb{Z}_{2^{12}}$	$\mathbb{Z}_{2^{12}}$	0.64	129	0.219	32	1.5
$\mathbb{Z}_{2^{13}}$	$\mathbb{Z}_{2^{13}}$	1.25	258	0.240		
$\mathbb{Z}_{2^{14}}$	$\mathbb{Z}_{2^{14}}$	2.46	516	0.268		
$\mathbb{Z}_{2^{15}}$	$\mathbb{Z}_{2^{15}}$	4.92	1033	0.320		
$\mathbb{Z}_{2^{16}}$	$\mathbb{Z}_{2^{16}}$	9.95	2066	0.487		

- [IIMP22][5] for $\mathbb{Z}_{15} \rightarrow \mathbb{Z}_{12}$: 0.414sec/point (or 847sec for 2048 points)

[4] i9-12900K, 32GB DDR4, Windows 11, Go 1.21

[5] Xeon E5-2630 v2

EXPERIMENTS - PERFORMANCE

- Performance[6] of our split-domain approach for a bivariate function $g(x, y) = \alpha_1 f_1(x) + \alpha_2 f_2(y)$
- Timings and data size are reported for batches of 2048 points

Dom(g)	Img(g)	Encryption [sec]	Query [MB]	Evaluation [sec]	Response [KB]	Keys [MB]
$\mathbb{Z}_{2^{12}} \times \mathbb{Z}_{2^{12}}$	$\mathbb{Z}_{2^{12}}$	1.26	258	0.235	32	1.5
$\mathbb{Z}_{2^{13}} \times \mathbb{Z}_{2^{13}}$	$\mathbb{Z}_{2^{13}}$	2.46	516	0.261		
$\mathbb{Z}_{2^{14}} \times \mathbb{Z}_{2^{14}}$	$\mathbb{Z}_{2^{14}}$	4.83	1033	0.317		
$\mathbb{Z}_{2^{15}} \times \mathbb{Z}_{2^{15}}$	$\mathbb{Z}_{2^{15}}$	9.58	2066	0.418		
$\mathbb{Z}_{2^{16}} \times \mathbb{Z}_{2^{16}}$	$\mathbb{Z}_{2^{16}}$	19.15	4133	0.700		

- [IIMP22][7] for $\mathbb{Z}_{15} \times \mathbb{Z}_{15} \rightarrow \mathbb{Z}_{12}$: 0.820sec/point (or 1679sec for 2048 points)

[6] i9-12900K, 32GB DDR4, Windows 11, Go 1.21

[7] Xeon E5-2630 v2

References I

- Guillaume Bonnoron, Léo Ducas, and Max Fillinger, *Large FHE gates from tensored homomorphic accumulator*, AFRICACRYPT 18 (Antoine Joux, Abderrahmane Nitaj, and Tajjeeddine Rachidi, eds.), LNCS, vol. 10831, Springer, Heidelberg, May 2018, pp. 217–251.
- Christina Boura, Nicolas Gama, Mariya Georgieva, and Dimitar Jetchev, *CHIMERA: Combining ring-LWE-based fully homomorphic encryption schemes*, Cryptology ePrint Archive, Report 2018/758, 2018, `https://eprint.iacr.org/2018/758`.
- Jean-François Biasse and Luis Ruiz, *FHEW with efficient multibit bootstrapping*, LATINCRYPT 2015 (Kristin E. Lauter and Francisco Rodríguez-Henríquez, eds.), LNCS, vol. 9230, Springer, Heidelberg, August 2015, pp. 119–135.

References II

Hao Chen, Wei Dai, Miran Kim, and Yongsoo Song, *Efficient homomorphic conversion between (ring) LWE ciphertexts*, ACNS 21, Part I (Kazue Sako and Nils Ole Tippenhauer, eds.), LNCS, vol. 12726, Springer, Heidelberg, June 2021, pp. 460–479.

Sergiu Carpov, Malika Izabachène, and Victor Mollimard, *New techniques for multi-value input homomorphic evaluation and applications*, CT-RSA 2019 (Mitsuru Matsui, ed.), LNCS, vol. 11405, Springer, Heidelberg, March 2019, pp. 106–126.

Ilia Iliashenko, Malika Izabachène, Axel Mertens, and Hilder V. L. Pereira, *Homomorphically counting elements with the same property*, PoPETs **2022** (2022), no. 4, 670–683.

References III

Andrey Kim, Maxim Deryabin, Jieun Eom, Rakyong Choi, Yongwoo Lee, Whan Ghang, and Donghoon Yoo, *General bootstrapping approach for RLWE-based homomorphic encryption*, Cryptology ePrint Archive, Report 2021/691, 2021, `https://eprint.iacr.org/2021/691`.