Recent Advances in Fully Homomorphic Encryption

Hyeongmin Choe

Seoul National University

@Ruhr University Bochum

Jan. 21st, 2025

Table of Contents

Introduction to FHE

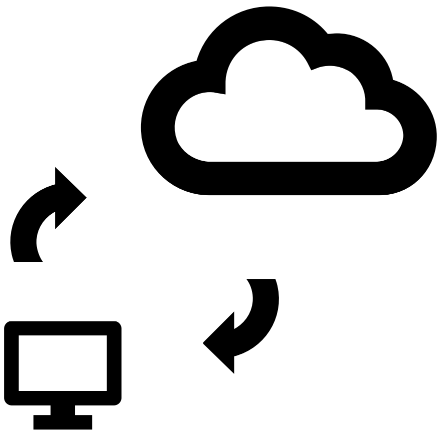
Motivation

- **Privacy Issues**
  - Personalized services
  - Cloud computing services
  - Data abuse

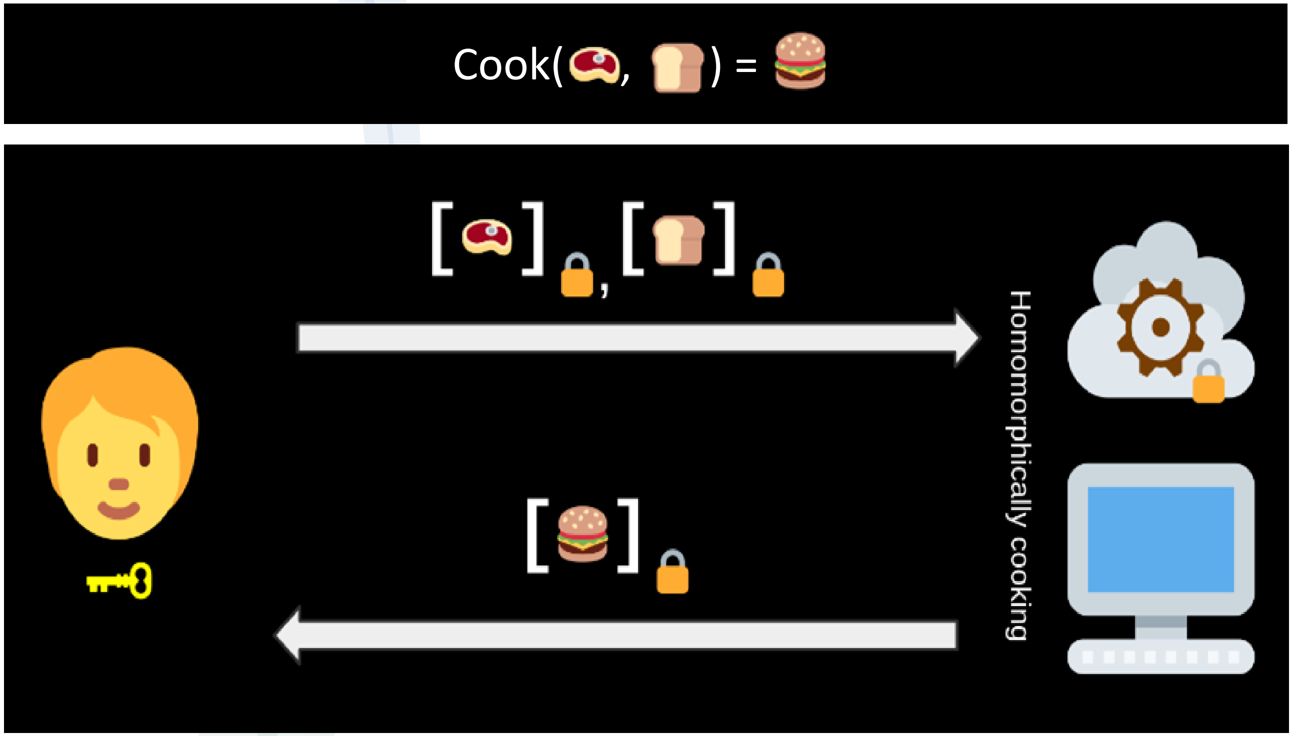
- **Data Policies and Regulations**
  - HIPPA (US), GDPR (EU), Data Three Rules (Korea), …

➔ Privacy Enhancing Technologies (PETs)
  - MPC, FHE, DP, Confidential Computing, …

Introduction:
Fully Homomorphic Encryption



- **Allow computation delegation**
  - Secure Outsourced Computation

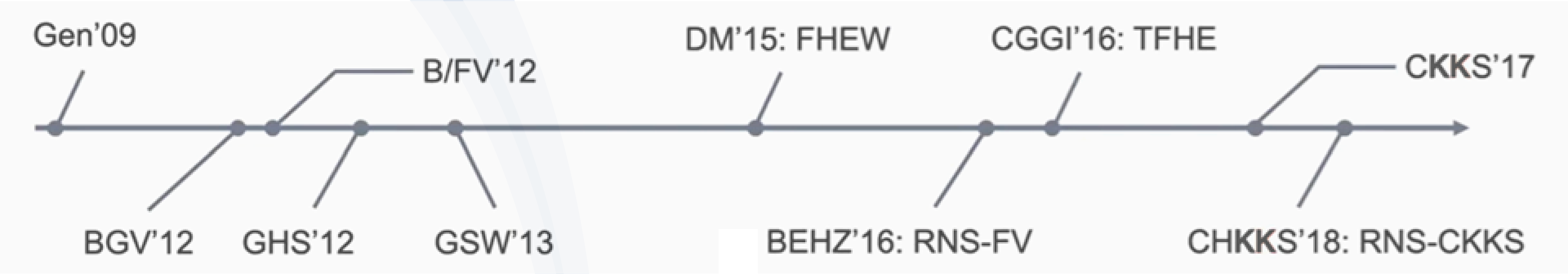


* Figure adapted from Elias Suvanto, CryptoLab Inc.

Introduction:
Fully Homomorphic Encryption

- **Computations as exact as in plaintext**
  - Not like Differential Privacy (DP)
- **Round optimality & Ciphertext reusability**
  - Not like MPC
- **Security proven under hardness assumptions**
  - Not like Confidential Computing

Introduction:
Fully Homomorphic Encryption



* Figure adapted from Prof. Miran Kim, Hanyang University.

- **SotA FHE schemes**
  - **BGV, BFV**: Integer (finite field) arithmetic (+, x)
  - **DM, CGGI**: Boolean (AND, OR, NAND, XOR, …)
  - **CKKS**: Real/Complex numbers $(\mathbb{R}, +, \times)$ or $(\mathbb{C}, +, \times)$

➔ Arbitrary circuits by composing the unit operations

Introduction:
Fully Homomorphic Encryption

- **SotA FHE schemes**
  - **BGV, BFV, CKKS:** RLWE-based
    - Ciphertext:

$$(a, b = -as + \Delta m + e) \in R_Q^2$$

for $R = \mathbb{Z}[x]/(x^N + 1)$,

      - $Q \approx$ 400~2900-bit integer
      - $N \approx 2^{13\sim17}$ sized integer
    - Plaintext space = vectors:
      - Add/Mult in parallel ($\approx 2^{12\sim16}$)
      - Coordinate-wise rotation
  - **DM, CGGI:** LWE-based
    - Ciphertext:

$$(a, b = -as + \Delta m + e) \in \mathbb{Z}_Q^2$$

      - $Q \approx$ 32~64-bit integer
      - $N \approx 2^{9\sim11}$ sized integer
    - Plaintext space = bits:
      - Boolean Gates

Introduction:
Fully Homomorphic Encryption

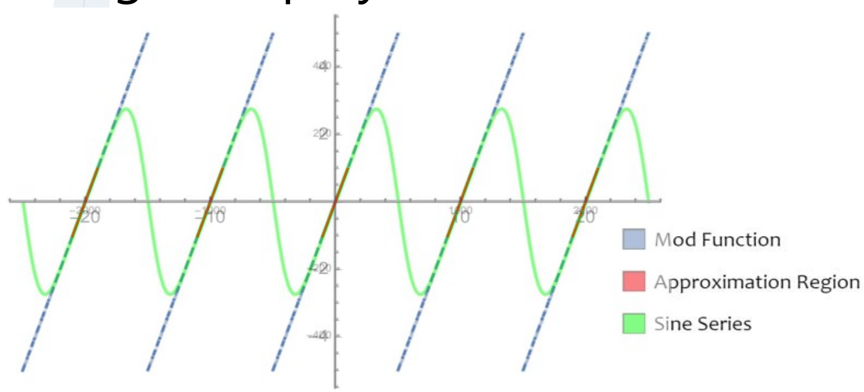
- **SotA FHE schemes**
  - **BGV, BFV, CKKS:** RLWE-based
    - Level-based:
      - Mult consumes 1 level
      - Add/Rot consume 0 level
      - Bootstrapping regains level
  - **DM, CGGI:** LWE-based
    - No levels:
      - Bootstrapping required after every (or several) gate operations

Moderate, one-core CPU	CKKS Bootstrapping	TFHE Bootstrapping
(Amortized) Time	$\sim$7s/ 2^{16} real numbers of 22-bit fixed-point $\approx$ 0.1ms/ real number	$\sim$10ms/ bit

* Timings borrowed from Dr. Damien Stehlé, CryptoLab Inc.

Introduction:
RLWE-based FHEs

- **Homomorphic Evaluations** via (+, x)
  - Linear Algebra
    - Matrix, vector multiplications
  - Polynomials
    - Minimax, Remez, Chebyshev approximations
    - Depth $\lfloor \log_2 d \rfloor$ for degree d polynomial



* Figure adapted from Dr. Damien Stehlé, CryptoLab Inc.

Recent Advances in FHE

Recent Advances in FHE: Topics under the spotlight

Applications

- SVM, PCA [EP:CCJ+23]
- CNN, **DNN** [BMC:HPCCC22]
- LM, **LMM**
- **Linear Algebra**
- **Protocols using FHE**

Security

- **IND-CPA**^{D} [CCS:CCP+24]
- IND-CVA
- Func-CVA
- IND-CPA^{C}

New Functionalities

- **Bit/Integer-CKKS**
- **High-precision**
- Ring switching

Acceleration

- **CPU**
  - New KeySwitchings
  - **New Arithmetic** [EP:CCK+24]
- **GPU/FPGA**
  - **NTT, BTS workloads**

Threshold

- Threshold security [CCS:CCP+24]
- Distributed KeyGen
- **Distributed Dec**
  - **Smaller modulus**
  - **New definitions** [CCSDS:Choe24]

Recent Advances in FHE: Acceleration: GPU

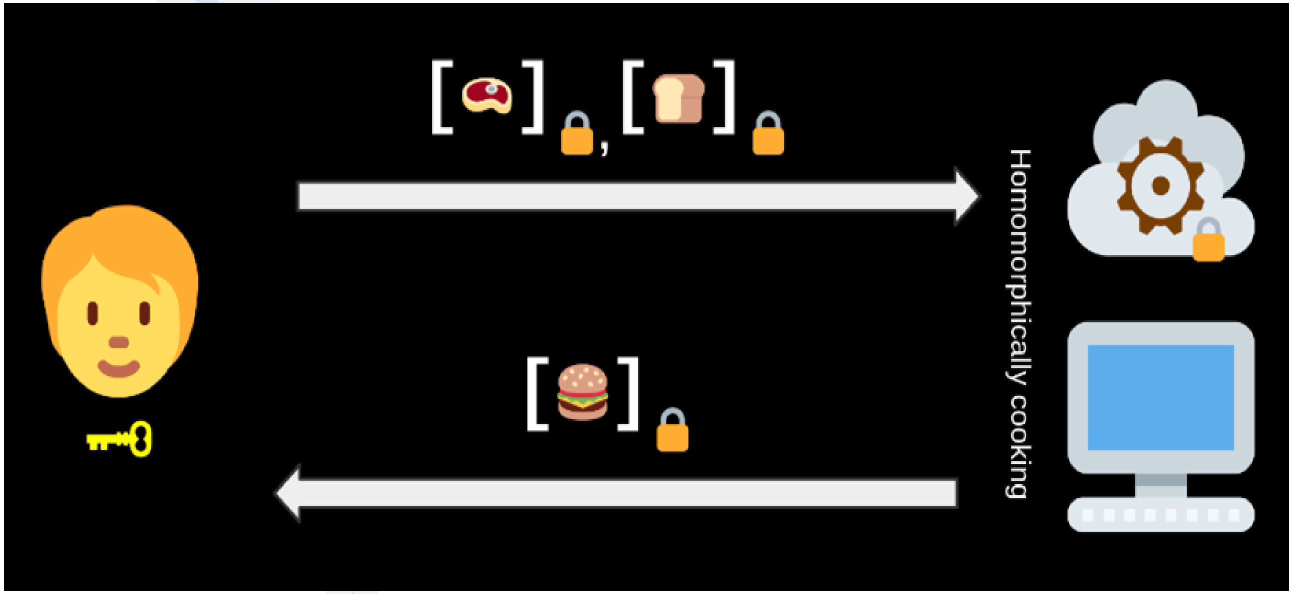
- **Some numbers for CKKS [HEaaN]**
  - 22-bit **Bootstrapping**, 2^{16} real numbers
    - [CPU] 6.9s in Intel Xeon Gold 6342 ≈ 0.1ms/ real number
    - [GPU] 61ms in NVIDIA GeForce RTX 4090 ≈ 0.9μs/ real number
      - GPUs 100x~, FPGAs 1,300x~
  - 22-bit **Multiplication** takes 73.6ns/ real number in GPU

Recent Advances in FHE: Application: LMM [RKP+24]

- **Some numbers for Language Model**
  - **BERT fine-tuning**
    - 5-17 hours in 8 GPUs for most of the downstream tasks
    - some accuracy degradation
  - **LLAMA2-7B**
    - 181.5 seconds for one token generation in 8 GPUs

Task	Plaintext	under HE
	Full+SM	Full+GK
CoLA (Matthews corr. ↑)	0.2688	0.1575
MRPC (F1 ↑)	0.8304	0.8147
RTE (Accuracy ↑)	0.5884	0.5993
STSB (Pearson ↑)	0.8164	0.7997
SST-2 (Accuracy ↑)	0.8991	0.8188
QNLI (Accuracy ↑)	0.8375	0.7827
Average	0.7068	0.6621

Recent Advances in FHE:
Security: IND-CPA^D Attack [CCS:CCP+24]



* Figure adapted from Elias Suvanto, CryptoLab Inc.

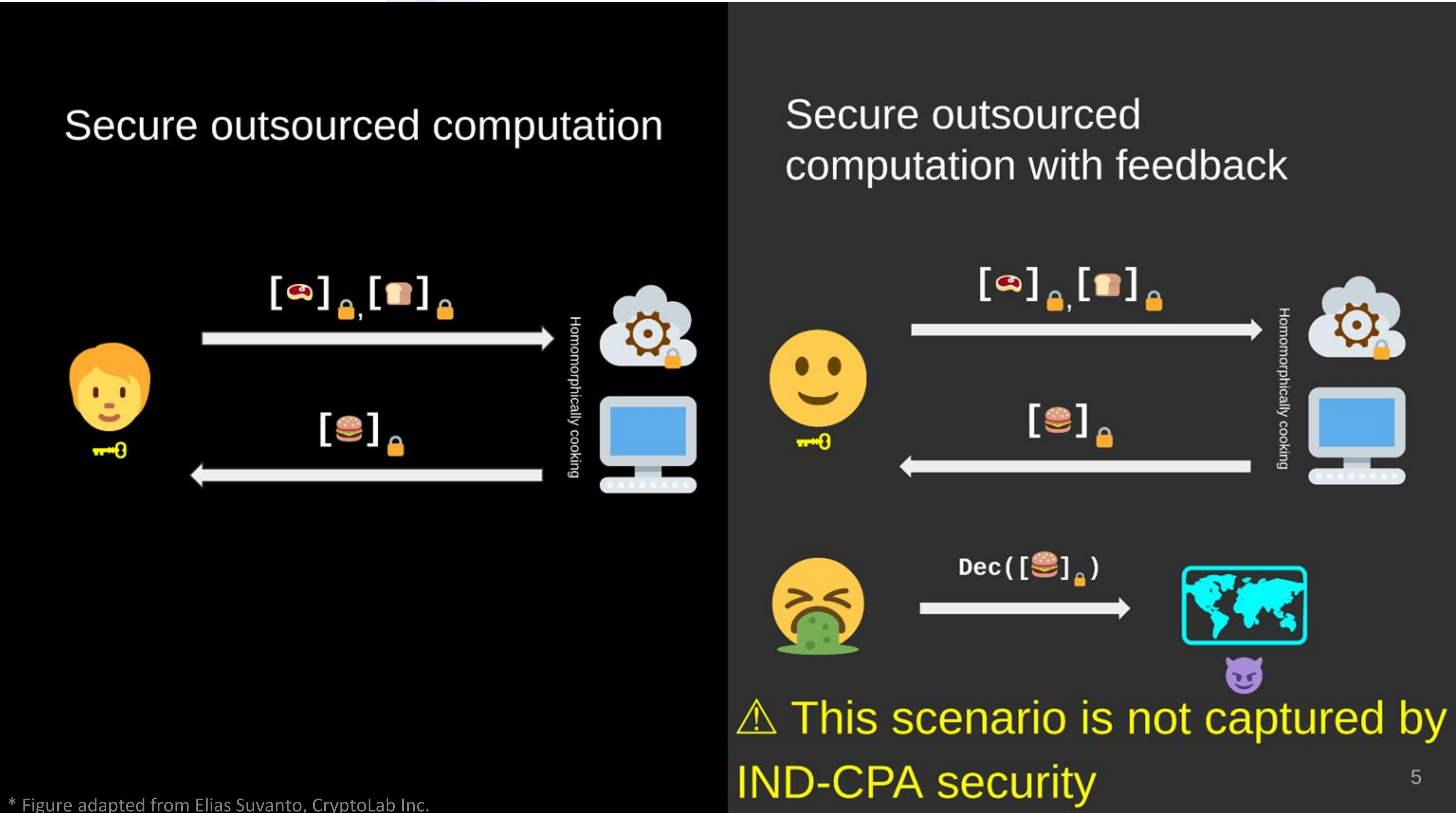
IND-CPA security:

The [*] do not leak any information about `msg`, , and

IND-CPA^D security:

Even if is shared, the [*] do not leak any additional information

Recent Advances in FHE:
Security: IND-CPA^D Attack [CCS:CCP+24]



* Figure adapted from Elias Suvanto, CryptoLab Inc.

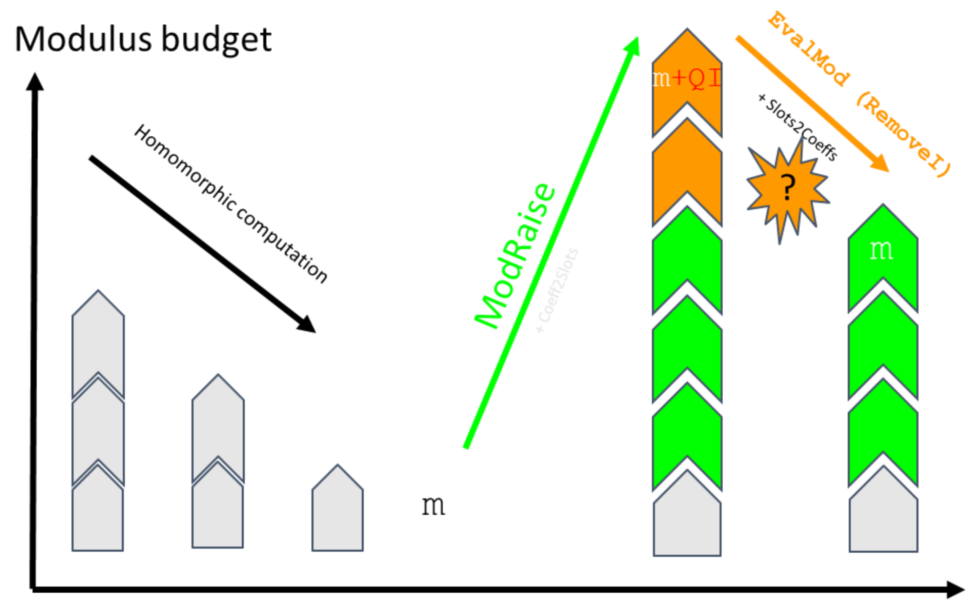
Recent Advances in FHE:
Security: IND-CPA^D Attack [CCS:CCP+24]

- **Bootstrapping (BTS) in CKKS**
  - BTS is basically ModSwitch from Q to $Q' \gg Q$, and evaluating "Mod Q" function
  - $b + as = \Delta m + e \bmod Q$

→ $b + as = \Delta m + e + QI$ for some I.

→ $b + as = \Delta m + e + QI \mod Q'$.

1. The integer **I** comes from $\langle ct, s \rangle$
2. EvalMod is correct iff $-K < I < K$
3. Incorrectness means $|I| \geq K$
4. Hint: highly likely that $ct \,//\, s$



* Figure adapted from Elias Suvanto, CryptoLab Inc.

Recent Advances in FHE:
Security: IND-CPA^D Attack [CCS:CCP+24]

	Plaintext space	IND-CPA^D status belief	In many libraries	Reasons
BFV/BGV (2012)		✓	✗	Incorrect noise upper bound
DM/CGGI (2015)	small integers	✓	✗	High failure probability
discrete-CKKS (2024)	small integers	✓	✗	High failure probability
CKKS (2017)		✗	✗	High failure probability
CKKS (+ noise flooding)		✓	✗	High failure probability

* Table adapted from Elias Suvanto, CryptoLab Inc.

Recent Advances in FHE: New Functionalities: Bit/Integer-CKKS

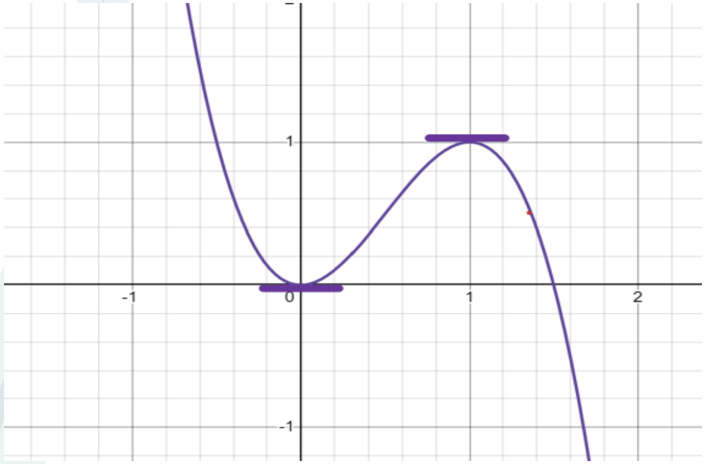
Problem

- **Operation type** highly affects performance
  - Bits, small integers → DM/CGGI
  - Large integers, finite field → BGV/BFV
  - Hard to go back and forth between different types of operations.

Recent Advances in FHE:
New Functionalities: Bit/Integer-CKKS

How to?

- **Binary gate operations using CKKS**
  - Encode $b \in \{0,1\}$ into $b + \varepsilon \in \mathbb{R}$
  - Cleaning $b + \varepsilon$ into $b + \varepsilon^*$, where $\varepsilon^* \ll \varepsilon$ using low-degree polynomial

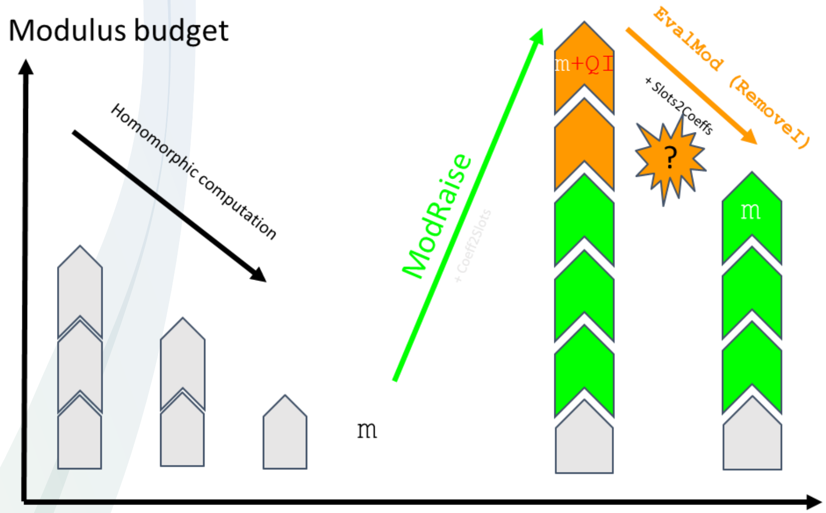


* Figure adapted from Dr. Damien Stehlé, CryptoLab Inc.

Recent Advances in FHE:
New Functionalities: Bit/Integer-CKKS

How to?

- **Bootstrapping (BTS) in CKKS**
  - BTS is basically evaluating "Mod Q" function
    - $b + as = \Delta m + e \bmod Q \rightarrow b + as = \Delta m + e + QI$ for some I.



* Figure adapted from Elias Suvanto, CryptoLab Inc.

Recent Advances in FHE:
New Functionalities: Bit/Integer-CKKS

How to?

- **Cleaning + Bootstrapping**
  - [EC:BCKS24] Bits: For $b \in \{0,1\}$, $\frac{b}{2} + \varepsilon + I \rightarrow b + O(\varepsilon^2)$:
    - $\frac{1}{2}\left(1 + \sin\left(2\pi x - \frac{\pi}{2}\right)\right) = b + O(\varepsilon^2)$ for $x = \frac{b}{2} + \varepsilon + I$ and $b \in \{0,1\}$.
  - [Integers] For $m \in \mathbb{Z}_t$,
    - $\frac{1}{t} \cdot m + \varepsilon + I \rightarrow \mathrm{e}^{2\pi\left(\frac{1}{t} \cdot m + \varepsilon + I\right)i} = \mathrm{e}^{2\pi\left(\frac{1}{t} \cdot m + \varepsilon\right)i} \rightarrow \text{Imag part} \approx \frac{2\pi}{t} \cdot m + \varepsilon^*$

	CGGI	[DMPS24]	[BCKS24]	[BKSS24]
Amortized Binary gate time	~10ms	27.7µs	17.6µs	7.39µs

Recent Advances in FHE: Acceleration: Grafting [EP:CCK+24]

Problem

- RLWE-based schemes use modulus of 700~2900 bits

→ Need efficient polynomial operations in R_Q for $Q = q_0 q_1 \cdots q_{\ell-1}$ ($q_i \approx \Delta$).

- **Residue Number System (RNS)**
  - Relatively prime q_i → $R_Q \cong R_{q_0} \times \cdots \times R_{q_\ell}$ based on CRT
    - $\mathcal{O}(\log^2 Q)$ →$\mathcal{O}(\sum_i \log^2 q_i) \approx \mathcal{O}(\ell \cdot \log^2 Q^{1/\ell}) \approx \mathcal{O}\left(\frac{1}{\ell} \cdot \log^2 Q\right)$

- **Number Theoretic Transform (NTT)**
  - For NTT prime $q_i \equiv 1 \; mod \; 2N$ → efficient polynomial mult.
    - $\mathcal{O}(N^2)$ → $\mathcal{O}(N \log N)$

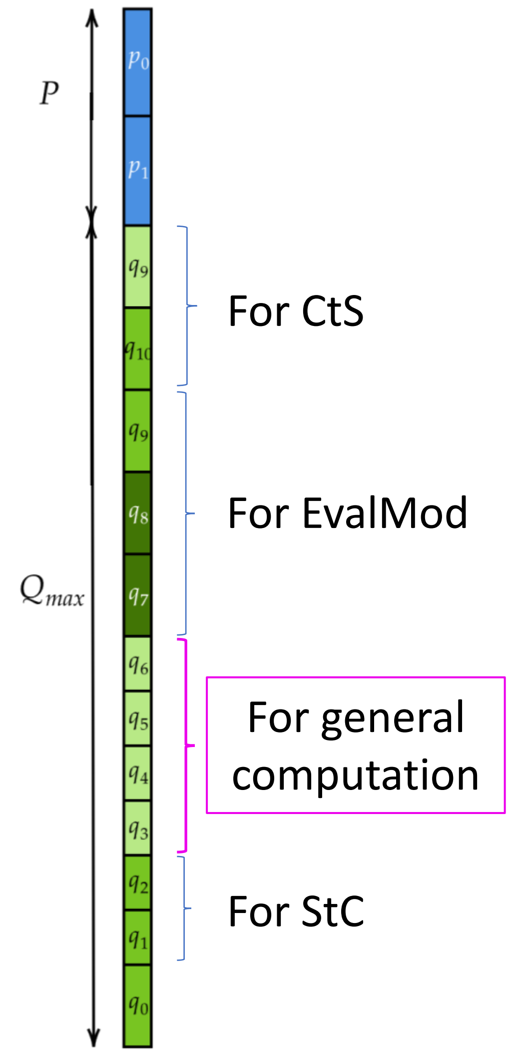
Recent Advances in FHE: Acceleration: Grafting [EP:CCK+24]

Problem

- Use **NTT primes** as RNS moduli, **40~60 bit** in 64-bit CPU.
- **Reserved, special-sized moduli for BTS**
  - CtS, EvalMod: e.g., $q_i \approx \Delta \approx 2^{45}$
  - StC: e.g., $q_i \approx \Delta \approx 2^{35}$

➔ **Optimized** modulus consumption & performance for target precision

- e.g., for 20-bit BTS:



Recent Advances in FHE:
Acceleration: Grafting [EP:CCK+24]

Problem

- #RNS moduli matters the performance & memory
  - Costs O(#RNS moduli) or O (#RNS moduli)^2

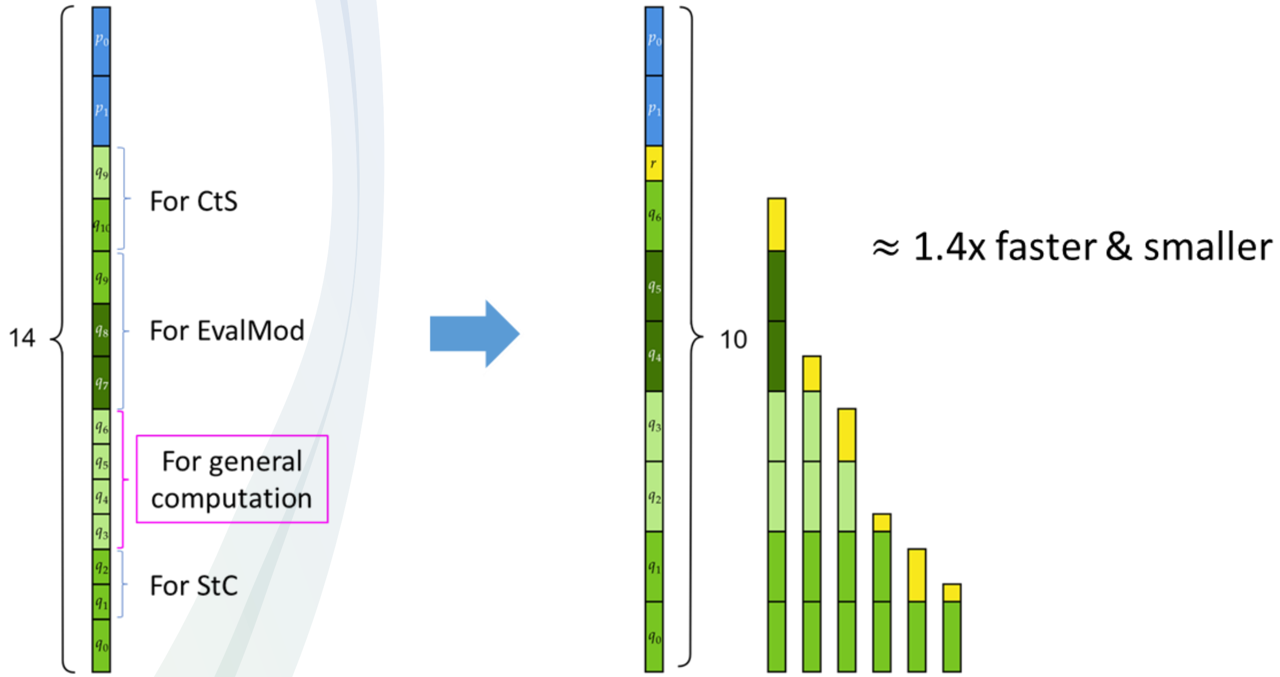
- But due to $q_i \approx \Delta$, we cannot have optimal,

$$\#\text{RNS moduli} \approx \frac{\log_2 \mathrm{PQ_{max}}}{\text{word}-\text{size}}$$

Recent Advances in FHE:
Acceleration: Grafting [EP:CCK+24]

How to?

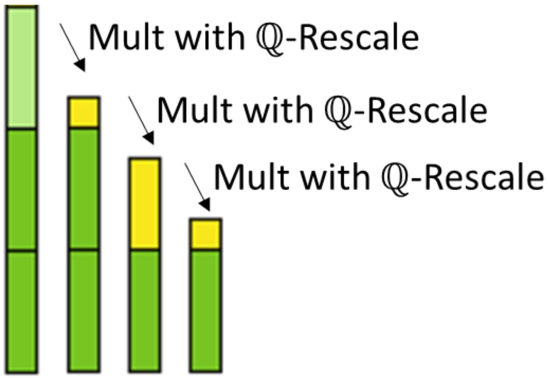
- **Fill the moduli chain with mostly the word-sizes, but also allow prior optimizations.**



Recent Advances in FHE:
Acceleration: Grafting [EP:CCK+24]

How to?

- **Rational Rescale**
  - $Q \to lcm(Q, Q') \to Q'$

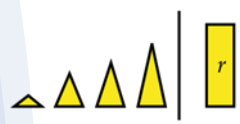


- **Key Switching** (part of Mult and Rotate)
  - We need a modulus for key (PQ_{max}) that is divisible by any possible Q.

Recent Advances in FHE:
Acceleration: Grafting [EP:CCK+24]

How to?

- **Sprout, a flexible part in Q**



"Sprout"

- E.g. sprout of $r = 2^{62}$, for where q_i are 62-bit RNS primes,
  - $Q = q_0 \cdot q_1 \cdots q_{\ell-1} \cdot 2^{\alpha}$
  - $Q_{max} = q_0 \cdot q_1 \cdots q_{L-1} \cdot 2^{62}$

  But, not so great for computing 2^{62} part

Recent Advances in FHE:
Acceleration: Grafting [EP:CCK+24]

How to?

- **Embedded NTT** [CHK+21] & **Composite NTT**
  - 2^{15}, $q_1 = 2^{16} + 1$, $q_2 = 30$-bit prime
    - $\mathbb{Z}_{q_1} \times \mathbb{Z}_{q_2} \cong \mathbb{Z}_{q_1 q_2}$ as $q_1 q_2 \approx 2^{46}$ ➔ NTT for $q_1 q_2$
    - Embed $\mathbb{Z}_{2^{15}}$ into $\mathbb{Z}_{q^*}$ for a 62-bit NTT prime q^*.

- Overall, we can achieve near-optimal,

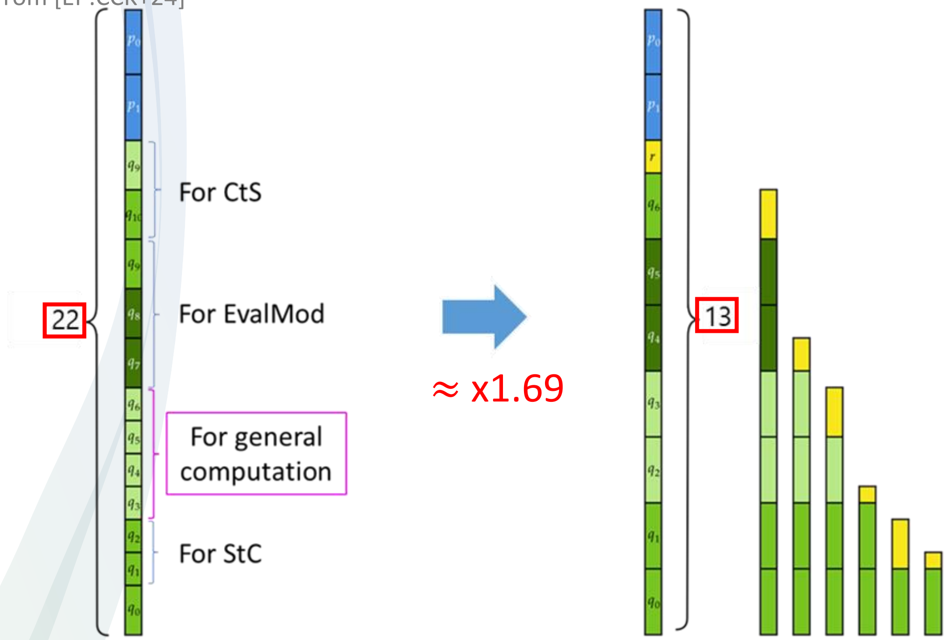
$$\#\text{RNS moduli} \approx \left\lceil \frac{\log_2 \mathrm{PQ_{max}}}{\text{word}-\text{size}} \right\rceil + 1$$

Recent Advances in FHE:
Acceleration: Grafting [EP:CCK+24]

How to?

$N = 2^{15}$, $\log PQ_{\max} = 777$	$\log q_i$					$\log p_i$	#	dnum
	Base	StC	Mult	EvalMod	CtS			
simple (FTa)	38	$32 + 28 \times 2$	28×5	38×8	41×3	42×2	22	10
grafted	$61 \times 10 + 45$					61×2	13	6

* Table borrowed from [EP:CCK+24]



Recent Advances in FHE: Acceleration: Grafting [EP:CCK+24]

How to?

Operations	Mult. (ms) Tensor	Relin.	Rescale	Total	Bootstrap. (ms) StC	CtS	EvalMod	Total
simple	9.77	310.09	38.48	358.34	649.43	7,632.32	3,940.44	16,607
grafted	5.17	109.93	24.74	139.84	741.36	2,990.17	1,649.86	6,814
Measured gain	1.89×	2.82×	1.56×	2.56×	0.88×	2.55×	2.39×	2.44×
Expected gain	1.82×	2.54×	1.82×		1×	2.54×	1.82×	2.07×

* Table borrowed from [EP:CCK+24]

Sizes	Ciphertext (KiB)	Switching key (KiB)
simple	10,240	112,640
grafted	6,144	43,008
Measured gain	↓ 40.0 %	↓ 61.8 %
Expected gain	↓ 40.0 %	↓ 61.8 %

* Table borrowed from [EP:CCK+24]

Thank You!

References

- [EP:CCJ+23] Cheon, J. H., Choe, H., Jung, S., Kim, D., Lee, D. H., & Park, J. H. (2023). Arithmetic PCA for Encrypted Data. Cryptology ePrint Archive.
- [BMC:HPCCC22] Hong, S., Park, J. H., Cho, W., Choe, H., & Cheon, J. H. (2022). Secure tumor classification by shallow neural network using homomorphic encryption. BMC genomics, 23(1), 284.
- [CCS:CCP+24] Cheon, J. H., Choe, H., Passelègue, A., Stehlé, D., & Suvanto, E. (2024, December). Attacks against the IND-CPAD security of exact FHE schemes. In Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security (pp. 2505-2519).
- [EP:CCK+24] Cheon, J. H., Choe, H., Kang, M., & Kim, J. (2024). Grafting: Complementing rns in ckks. Cryptology ePrint Archive.
- [CCSDS:Choe24] Choe, H. (2024, December). Toward Practical Threshold FHE: Low Communication, Computation and Interaction. In Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security (pp. 5107-5109).
- [HEaaN] CryptoLab Inc, HEaaN Library, 2022. Available at https://heaan.it/.
- [RKP+24] Rho, D., Kim, T., Park, M., Kim, J. W., Chae, H., Cheon, J. H., & Ryu, E. K. (2024). Encryption-friendly LLM architecture. arXiv preprint arXiv:2410.02486.
- [EC:BCKS24] Bae, Y., Cheon, J. H., Kim, J., & Stehlé, D. (2024, May). Bootstrapping Bits with CKKS. In Annual International Conference on the Theory and Applications of Cryptographic Techniques (pp. 94-123). Cham: Springer Nature Switzerland.
- [AC:BKSS24] Bae, Y., Kim, J., Stehlé, D., & Suvanto, E. (2025). Bootstrapping small integers with CKKS. In International Conference on the Theory and Application of Cryptology and Information Security (pp. 330-360). Springer, Singapore.