Introduction to Fully Homomorphic Encryption

Part 2: leveled FHE and bootstrapping

Jean-Sébastien Coron

University of Luxembourg

Overview

- Previous lecture: basic techniques for fully homomorphic encryption
  - First generation of FHE, the DGHV scheme
  - Overview of bootstrapping
  - LWE-based encryption. Relinearization for ciphertext multiplication
- This lecture: leveled FHE, bootstrapping
  - Modulus switching
  - Leveled FHE
  - Bootstrapping

Four generations of FHE

- First generation: bootstrapping, slow
  - Breakthrough scheme of Gentry [G09], based on ideal lattices.
  - FHE over the integers: [DGHV10]
- Second generation: [BV11], [BGV11]
  - More efficient, (R)LWE based. Relinearization, depth-linear construction with modulus switching.
- Third generation [GSW13]
  - No modulus switching, slow noise growth
  - Improved bootstrapping: [BV14], [AP14]
- Fourth gen: [CKKS17]
  - Approximate floating point arithmetic

- Homomorphic encryption based on polynomial evaluation
  - Homomorphism: $\delta : \mathbb{Z}_q[\vec{x}] \to \mathbb{Z}_q[x]$ given by evaluation at secret $\vec{s} = (s_1, \ldots, s_n)$

$$\begin{tikzcd}
\text{Ciphertexts} & \mathbb{Z}_q[\vec{x}] \times \mathbb{Z}_q[\vec{x}] \arrow[r, "{+,\times}"] \arrow[d, "{\delta,\delta}"] & \mathbb{Z}_q[\vec{x}] \arrow[d, "\delta"] \\
\text{Plaintexts} & \mathbb{Z}_q \times \mathbb{Z}_q \arrow[r, "{+,\times}"] & \mathbb{Z}_q
\end{tikzcd}$$

- One must add some noise, otherwise broken by linear algebra.
  - $f(\vec{s}) = 2e + m \bmod q$, for some small noise $e \in \mathbb{Z}_q$
- LWE assumption [R05]
  - Linear polynomials $f_i(\vec{x})$ with $|f_i(\vec{s}) \bmod q| \ll q$ are comp. indist. from random $f_i(\vec{x})$ modulo q.

Second generation: LWE-based encryption

- Homomorphic encryption based on polynomial evaluation
  - Homomorphism: $\delta : \mathbb{Z}_q[\vec{x}] \to \mathbb{Z}_q[x]$ given by evaluation at secret $\vec{s} = (s_1, \ldots, s_n)$

$$\begin{array}{lccc}
\text{Ciphertexts} & \mathbb{Z}_q[\vec{x}] \times \mathbb{Z}_q[\vec{x}] & \xrightarrow{+,\times} & \mathbb{Z}_q[\vec{x}] \\
 & \Big\downarrow \delta,\delta & & \Big\downarrow \delta \\
\text{Plaintexts} & \mathbb{Z}_q \times \mathbb{Z}_q & \xrightarrow{+,\times} & \mathbb{Z}_q
\end{array}$$

- One must add some noise, otherwise broken by linear algebra.
  - $f(\vec{s}) = 2e + m \bmod q$, for some small noise $e \in \mathbb{Z}_q$
- LWE assumption [R05]
  - Linear polynomials $f_i(\vec{x})$ with $|f_i(\vec{s}) \bmod q| \ll q$ are comp. indist. from random $f_i(\vec{x})$ modulo q.

Second generation: LWE-based encryption

- Homomorphic encryption based on polynomial evaluation
  - Homomorphism: $\delta : \mathbb{Z}_q[\vec{x}] \to \mathbb{Z}_q[x]$ given by evaluation at secret $\vec{s} = (s_1, \ldots, s_n)$

$$
\begin{array}{lccc}
\text{Ciphertexts} & \mathbb{Z}_q[\vec{x}] \times \mathbb{Z}_q[\vec{x}] & \xrightarrow{+,\times} & \mathbb{Z}_q[\vec{x}] \\
 & \Big\downarrow \delta,\delta & & \Big\downarrow \delta \\
\text{Plaintexts} & \mathbb{Z}_q \times \mathbb{Z}_q & \xrightarrow{+,\times} & \mathbb{Z}_q
\end{array}
$$

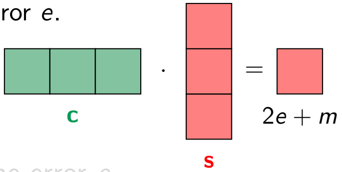
- One must add some noise, otherwise broken by linear algebra.
  - $f(\vec{s}) = 2e + m \bmod q$, for some small noise $e \in \mathbb{Z}_q$
- LWE assumption [R05]
  - Linear polynomials $f_i(\vec{x})$ with $|f_i(\vec{s}) \bmod q| \ll q$ are comp. indist. from random $f_i(\vec{x})$ modulo q.

LWE-based encryption [R05]

- Key generation
  - Secret-key: $\mathbf{s} \in (\mathbb{Z}_q)^n$
- Encryption of $m \in \{0, 1\}$
  - A vector $\mathbf{c} \in \mathbb{F}_q$ such that

$$\langle \mathbf{c}, \mathbf{s} \rangle = 2e + m \pmod{q}$$

  - for a small error e.



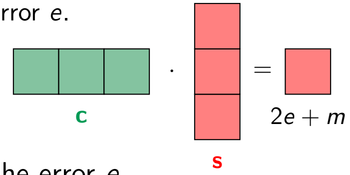
- Distribution of the error e
  - One can take the centered binomial distribution χ with parameter κ.
  - Let $e = h(u) - h(v)$ where $u, v \leftarrow \{0, 1\}^\kappa$, where h is the Hamming weight function.
- Decryption
  - Compute $m = (\mathbf{c} \cdot \mathbf{s} \bmod q) \bmod 2$
  - Decryption works if $|e| < q/4$

LWE-based encryption [R05]

- Key generation
  - Secret-key: $\mathbf{s} \in (\mathbb{Z}_q)^n$
- Encryption of $m \in \{0,1\}$
  - A vector $\mathbf{c} \in \mathbb{F}_q$ such that

$$\langle \mathbf{c}, \mathbf{s} \rangle = 2e + m \pmod q$$

  - for a small error e.

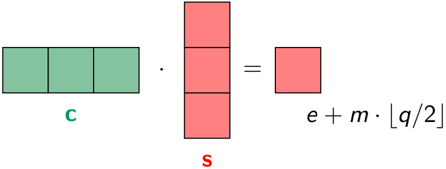


- Distribution of the error e
  - One can take the centered binomial distribution χ with parameter κ.
  - Let $e = h(u) - h(v)$ where $u, v \leftarrow \{0,1\}^\kappa$, where h is the Hamming weight function.
- Decryption
  - Compute $m = (\mathbf{c} \cdot \mathbf{s} \bmod q) \bmod 2$
  - Decryption works if $|e| < q/4$

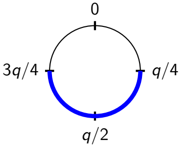
LWE-based encryption: alternative encoding

- The message m can also be encoded in the MSB.
- Encryption of $m \in \{0, 1\}$
  - A vector $\mathbf{c} \in \mathbb{F}_q$ such that

$$\langle \mathbf{c}, \mathbf{s} \rangle = e + m \cdot \lfloor q/2 \rfloor \pmod q$$



- Decryption
  - Compute $m = \text{th}(\langle \mathbf{c}, \mathbf{s} \rangle \bmod q)$
  - where $\text{th}(x) = 1$ if $x \in (q/4, 3q/4)$, and 0 otherwise.

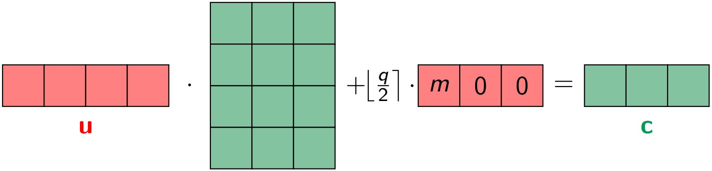


LWE-based public-key encryption

- Key generation
  - Secret-key: $\mathbf{s} \in (\mathbb{Z}_q)^n$, with $s_1 = 1$.
  - Public-key: $\mathbf{A}$ such that $\mathbf{A} \cdot \mathbf{s} = \mathbf{e}$ for small $\mathbf{e}$
    - Every row of $\mathbf{A}$ is an LWE encryption of 0.
- Encryption of $m \in \{0, 1\}$

$$\mathbf{c} = \mathbf{u} \cdot \mathbf{A} + (m \cdot \lfloor q/2 \rceil, 0, \ldots, 0)$$

  - for a small $\mathbf{u}$



A

- Decryption
  - Compute $m = \text{th}(\langle \mathbf{c}, \mathbf{s} \rangle \bmod q)$

Homomorphic addition

- LWE ciphertexts can be added
  - with a small increase in the noise

$$
\begin{aligned}
\langle \mathbf{c}_1, \mathbf{s} \rangle &= e_1 + m_1 \cdot (q+1)/2 \pmod q \\
\langle \mathbf{c}_2, \mathbf{s} \rangle &= e_2 + m_2 \cdot (q+1)/2 \pmod q \\
\langle \mathbf{c}_1 + \mathbf{c}_2, \mathbf{s} \rangle &= e_1 + e_2 + (m_1 + m_2) \cdot (q+1)/2 \pmod q
\end{aligned}
$$

Homomorphic multiplication

- Homomorphic multiplication of two ciphertexts is more complex, with 3 steps:
  - 1) Tensor product
    - We obtain a ciphertext in $\mathbb{Z}_q^{n^2}$, under a new key $\mathbf{s} \times \mathbf{s}$.
  - 2) Binary decomposition
    - We obtain a binary ciphertext in $\{0,1\}^{n^2 \cdot n_q}$, under a new key $\mathbf{s}' = \mathsf{PowerOfTwo}(\mathbf{s} \times \mathbf{s})$, with $n_q = \lceil \log_2 q \rceil$
  - 3) Key switching
    - We switch the key from $\mathbf{s}'$ back to the original key $\mathbf{s}$.

Tensor product

- LWE ciphertexts can be multiplied by tensor product.

$$2\langle \mathbf{c}_1, \mathbf{s}\rangle \cdot \langle \mathbf{c}_2, \mathbf{s}\rangle = 2\left(\sum_{i=1}^{n} c_{1,i} s_i\right)\left(\sum_{i=1}^{n} c_{2,i} s_i\right)$$
$$= 2(e_1 + (q+1)/2 \cdot m_1) \cdot (e_2 + (q+1)/2 \cdot m_2)$$

- This gives

$$\sum_{i=1}^{n}\sum_{j=1}^{n} 2c_{1,i}c_{2,j} \cdot s_i s_j = e + m_1 m_2 \cdot (q+1)/2 \pmod q$$

  - for a new eroor $e = 2e_1e_2 + m_1e_2 + m_2e_1$
- Therefore $\mathbf{c}' = (2c_{1,i} \cdot c_{2,j})_{i,j} \in \mathbb{Z}_q^{n^2}$ is a new LWE ciphertext
  - for the secret-key $\mathbf{s}' = (s_i \cdot s_j)_{i,j} \in \mathbb{Z}_q^{n^2}$

$$\langle \mathbf{c}', \mathbf{s}'\rangle = e + m_1 m_2 \cdot (q+1)/2 \pmod q$$

- The bitsize of the noise has roughly doubled.
  - We get a ciphertext with n^2 components instead of n.

Binary decomposition

- We want to have a ciphertext with binary components only.
  - We use binary decomposition. For any $0 \leq a, b < q$, we have, using $n_q = \lceil \log_2 q \rceil$:

$$\begin{aligned} a \cdot b &= \sum_{i=0}^{n_q - 1} a_i \cdot 2^i b \pmod{q} \\ &= \langle \mathsf{BitDecomp}(a), \mathsf{PowerOf2}(b) \rangle \end{aligned}$$

  - $\mathsf{BitDecomp}(a) = (a_0, \ldots, a_{n_q - 1})$ and $\mathsf{PowerOf2}(b) = (b, 2^1 b, \ldots, 2^{n_q - 1} b)$.
  - We extend BitDecomp and PowerOf2 to vectors, by concatenation
- New binary ciphertext from $\mathbf{c} \in \mathbb{Z}_q^m$ and $\mathbf{s} \in \mathbb{Z}_q^m$
  - Let $\mathbf{c}' = \mathsf{BitDecomp}(\mathbf{c})$, and $\mathbf{s}' = \mathsf{PowerOf2}(\mathbf{s})$

$$\langle \mathbf{c}', \mathbf{s}' \rangle = \langle \mathsf{BitDecomp}(\mathbf{c}), \mathsf{PowerOf2}(\mathbf{s}) \rangle = \langle \mathbf{c}, \mathbf{s} \rangle$$

  - The new binary ciphertext $\mathbf{c}'$ encrypts the same message under the new secret-key $\mathbf{s}'$.

Key switching

- How to switch keys ?
  - Start with a binary ciphertext $\mathbf{c} \in \{0,1\}^m$ under key $\mathbf{s} \in \mathbb{Z}_q^m$.
  - We write $u = \langle \mathbf{c}, \mathbf{s} \rangle = \sum_{i=1}^{m} c_i \cdot s_i \pmod{q}$
  - Let $\mathbf{s}' \in \mathbb{Z}_q^n$ be another key.
  - We consider LWE pseudo-encryptions $\mathbf{t}_i$ of each s_i under the new key $\mathbf{s}'$, with $\langle \mathbf{t}_i, \mathbf{s}' \rangle = f_i + s_i \pmod{q}$ for small errors f_i.
- Generating the new ciphertext under $\mathbf{s}'$
  - We can write:

$$u = \sum_{i=1}^{m} c_i \left(\langle \mathbf{t}_i, \mathbf{s}' \rangle - f_i \right) = \left\langle \sum_{i=1}^{m} c_i \mathbf{t}_i, \mathbf{s}' \right\rangle - \sum_{i=1}^{m} c_i \cdot f_i \pmod{q}$$

  - We can define a new ciphertext $\mathbf{c}' = \sum_{i=1}^{m} c_i \mathbf{t}_i \pmod{q}$ and we get for a small error f:

$$\langle \mathbf{c}', \mathbf{s}' \rangle = \langle \mathbf{c}, \mathbf{s} \rangle + f \pmod{q}$$

  - $\Rightarrow$ the two ciphertexts encrypt the same message

Summary of homomorphic multiplication

- Homomorphic multiplication of two ciphertexts has 3 steps:
  - 1) Tensor product
    - We obtain a ciphertext in $\mathbb{Z}_q^{n^2}$, under a new key $\mathbf{s} \times \mathbf{s}$.
  - 2) Binary decomposition
    - We obtain a binary ciphertext in $\{0,1\}^{n^2 \cdot n_q}$, under a new key $\mathbf{s}' = \text{PowerOfTwo}(\mathbf{s} \times \mathbf{s})$, with $n_q = \lceil \log_2 q \rceil$
  - 3) Key switching
    - We switch the key from $\mathbf{s}'$ back to the original key $\mathbf{s}$.

Modulus switching

- Consider a ciphertext modulo q

$$\begin{aligned}\langle \mathbf{c}, \mathbf{s} \rangle &= \lfloor q/2 \rfloor \cdot m + e \pmod q \\ &= q/2 \cdot m + \varepsilon + e + \lambda \cdot q\end{aligned}$$

  - for $|\varepsilon| \leq 1/2$ and $\lambda \in \mathbb{Z}$
- Switching to a ciphertext modulo $p < q$

$$\langle \mathbf{c} \cdot \frac{p}{q}, \mathbf{s} \rangle = p/2 \cdot m + \varepsilon \cdot \frac{p}{q} + e \cdot \frac{p}{q} + \lambda \cdot p$$

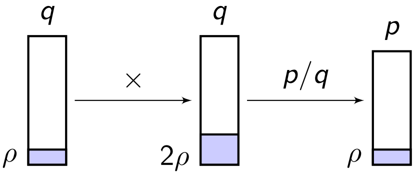
  - Write $\mathbf{c}' = \lfloor \mathbf{c} \cdot p/q \rceil = \mathbf{c} \cdot p/q + \mathbf{u}$ where $\|\mathbf{u}\|_\infty \leq 1/2$. Then

$$\langle \mathbf{c}', \mathbf{s} \rangle = \lfloor p/2 \rceil \cdot m + e' \pmod p$$

  - where $|e'| \leq e \cdot p/q + 1 + \frac{1}{2} \cdot \|\mathbf{s}\|_1$
- We get a new ciphertext $\mathbf{c}'$ modulo p encrypting the same m
  - with scaled error $e' \simeq e \cdot p/q$.

The BGV scheme: modulus switching [BGV11]

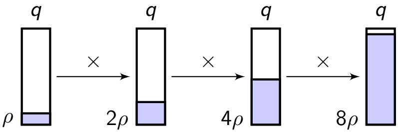
- Modulus switching from $\mathbf{c}$ modulo q to $\mathbf{c}'$ modulo $p < q$
  - Encrypts the same message m, but with error scaled by p/q
- Application: reducing noise growth. Assume $p/q = 2^{-\rho}$.



  - Noise reduction without bootstrapping !

Leveled fully homomorphic encryption

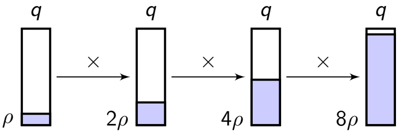
- Previous model: exponential growth of noise



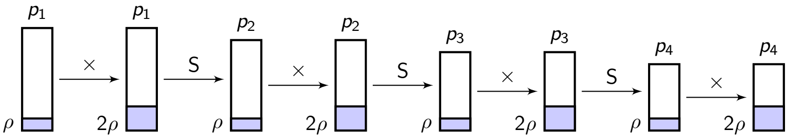
  - Only bootstrapping can give FHE
- New model: modulus switching after each multiplication layer
  - with a ladder of moduli p_i such that $p_{i+1}/p_i = 2^{-\rho}$

- Leveled FHE
  - Size of p_1 linear in the circuit depth
  - Parameters depend on the depth
  - Can accommodate polynomial depth

- Previous model: exponential growth of noise



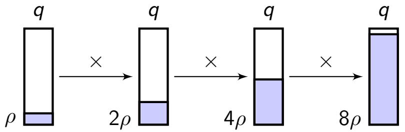
  - Only bootstrapping can give FHE
- New model: modulus switching after each multiplication layer
  - with a ladder of moduli p_i such that $p_{i+1}/p_i = 2^{-\rho}$



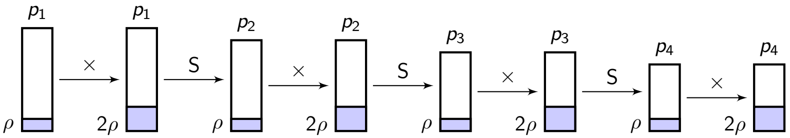
- Leveled FHE
  - Size of p_1 linear in the circuit depth
  - Parameters depend on the depth
  - Can accommodate polynomial depth

Leveled fully homomorphic encryption

- Previous model: exponential growth of noise



  - Only bootstrapping can give FHE
- New model: modulus switching after each multiplication layer
  - with a ladder of moduli p_i such that $p_{i+1}/p_i = 2^{-\rho}$



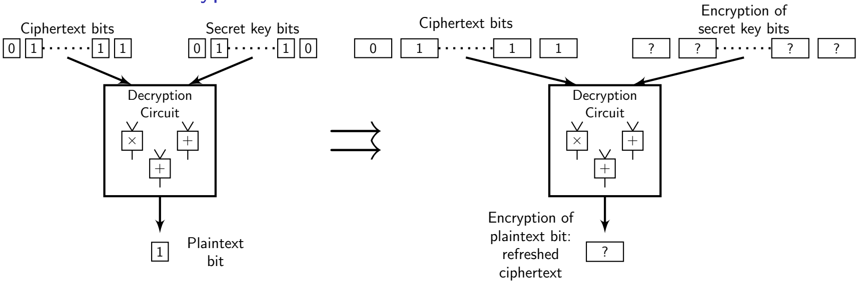
- Leveled FHE
  - Size of p_1 linear in the circuit depth
  - Parameters depend on the depth
  - Can accommodate polynomial depth

- To build a FHE scheme, start from the somewhat homomorphic scheme, that is:
  - Only a polynomial f of small degree can computed homomorphically, for $\mathcal{F} = \{f(b_1, \ldots, b_t) : \deg f \leq d\}$
  - $V_{pk}(f, E_{pk}(b_1), \ldots, E_{pk}(b_t)) \to E_{pk}(f(b_1, \ldots, b_t))$

$$\begin{tikzcd}
\text{Ciphertexts} & \mathcal{C}^t \arrow[r, "V_{pk}(f{,}\cdots)"] \arrow[d, "D_{sk}(\cdots)"] & \mathcal{C} \arrow[d, "D_{sk}(\cdot)"] & f \in \mathcal{F} \\
\text{Plaintexts} & (\mathbb{Z}_2)^t \arrow[r, "f"] & \mathbb{Z}_2 &
\end{tikzcd}$$

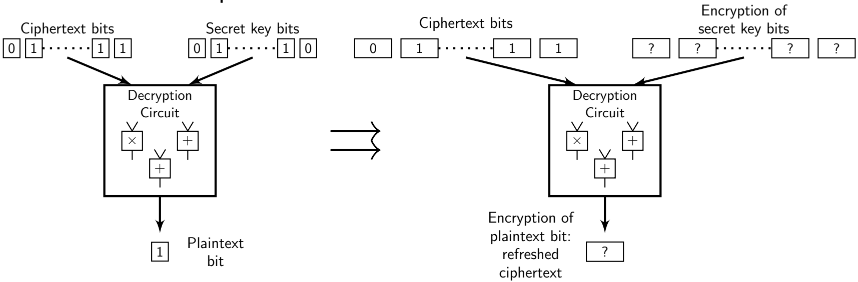
Ciphertext refresh: bootstrapping

- Gentry's breakthrough idea: refresh the ciphertext using the decryption circuit homomorphically.
  - Evaluate the decryption polynomial not on the bits of the ciphertext c and the secret key sk, but homomorphically on the encryption of those bits.

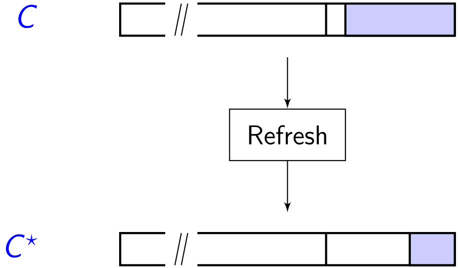


Ciphertext refresh: bootstrapping

- Gentry's breakthrough idea: refresh the ciphertext using the decryption circuit homomorphically.
  - Instead of recovering the bit plaintext *m*, one gets an encryption of this bit plaintext, *i.e.* yet another ciphertext for the same plaintext.

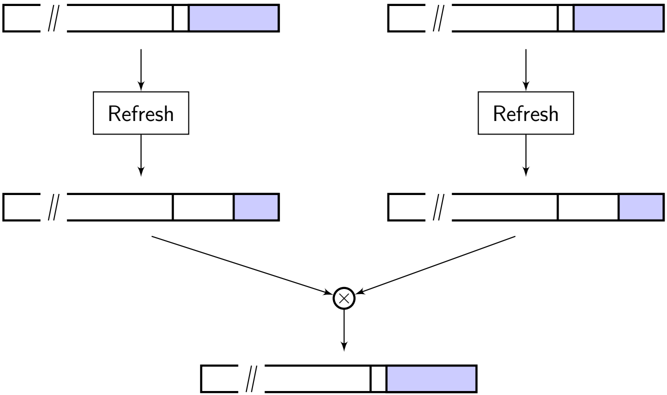


- Refreshed ciphertext:
  - If the degree of the decryption polynomial $D(\cdot,\cdot)$ is small enough, the resulting noise in the new ciphertext can be smaller than in the original ciphertext.



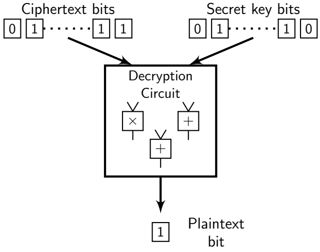
Fully homomorphic encryption

- Fully homomorphic encryption
  - Using this "ciphertext refresh" procedure, the number of homomorphic operations becomes unlimited
  - We get a fully homomorphic encryption scheme.



Bootstrapping LWE ciphertexts

- Building the decryption circuit
  - Takes as input the bits of the ciphertext, and the bits of the secret-key.
  - Outputs the decrypted message $m \in \{0, 1\}$



- Easier to switch to encryption modulo 2^k, instead of q
  - We perform a modulus switching to modulo 2^k using previous technique.

Building the decryption circuit

- First step: modulus switching to modulo 2^k
  - Let $\mathbf{c} \in \mathbb{Z}_q^n$ such that
$$\langle \mathbf{c}, \mathbf{s} \rangle = e + m \cdot (q+1)/2 \pmod q$$
  - From the previous modulus switching technique, we get
$$\langle \mathbf{c}', \mathbf{s} \rangle = 2^{k-1} \cdot m + e' \pmod{2^k}$$
  - where $|e'| \leq e \cdot 2^k / q + 1 + n/2$.
  - For correct decryption, we should have $|e'| \leq 2^{k-2}$.
  - Therefore we can take $k = \mathcal{O}(\log n)$.
- Second step: write the decryption circuit
  - Using only Xor and And gates
  - Starting from addition of two integers modulo 2^k.

Building the decryption circuit (2)

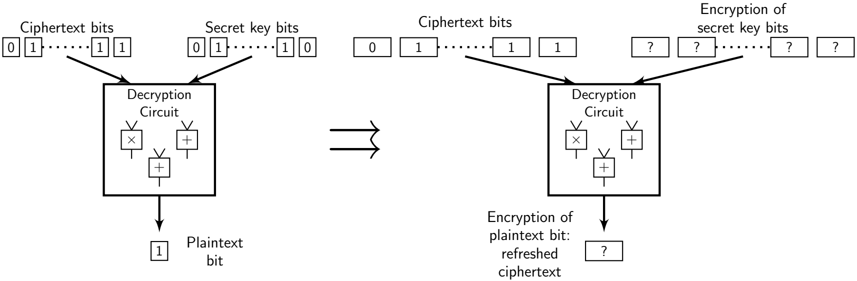
- We now have a ciphertext $\mathbf{c} \in \mathbb{Z}_{2^k}^n$ such that:

$$\langle \mathbf{c}, \mathbf{s} \rangle = \sum_{i=1}^{n} c_i \cdot s_i = 2^{k-1} \cdot m + e \pmod{2^k}$$

  - We want to write this operation with Xor and And gates only.
- 3 operations to compute with Xor and And gates:
  - Computing $c_i \cdot s_i$ with $c_i \in \mathbb{Z}_{2^k}$ and $s_i \in \{0, 1\}$
    - We compute a And between each the k bits of c_i and s_i.
  - Computing $a + b$ from $a, b \in \mathbb{Z}_{2^k}$
    - We use schoolbook addition, propagating the carry.
  - Extracting $m \in \{0, 1\}$ from $a = 2^{k-1} \cdot m + e$ with $|e| < 2^{k-2}$.
    - m is the xor of the most significant and second most significant bit of a

Bootstrapping achieved

- Bootstrapping
  - We perform the same operations as above, but homomorphically
  - Using an encryption of the secret-key bits



- Refreshed ciphertext $\mathbf{c}'$
  - The noise of $\mathbf{c}'$ only depends on the depth of the decryption circuit, not on the initial noise of $\mathbf{c}$.

Third generation of FHE: ciphertext matrices

- Homomorphic encryption with matrices [GSW13]
  - Ciphertexts are square matrices instead of vectors
  - Homomorphism: $\delta(C, \mathbf{v}) = \mu$ where μ is eigenvalue for secret eigenvector $\mathbf{v}$
  - Homomorphically add and multiply ciphertext using (roughly) matrix addition and multiplication

$$\begin{tikzcd}
\text{Ciphertexts} & \mathbb{Z}^{N\times N} \times \mathbb{Z}^{N\times N} \arrow[r, "{+,\times}"] \arrow[d, "{\delta,\delta}"] & \mathbb{Z}^{N\times N} \arrow[d, "\delta"] \\
\text{Plaintexts} & \mathbb{Z} \times \mathbb{Z} \arrow[r, "{+,\times}"] & \mathbb{Z}
\end{tikzcd}$$

- One must add some noise, otherwise broken by linear algebra
  - $C \cdot \mathbf{v} = \mu \cdot \mathbf{v} + \mathbf{e} \pmod{q}$
  - for message $\mu \in \mathbb{Z}$, for some small noise $\mathbf{e}$.
  - Security based on LWE problem.

Third generation of FHE: ciphertext matrices

- Homomorphic encryption with matrices [GSW13]
  - Ciphertexts are square matrices instead of vectors
  - Homomorphism: $\delta(C, \mathbf{v}) = \mu$ where μ is eigenvalue for secret eigenvector $\mathbf{v}$
  - Homomorphically add and multiply ciphertext using (roughly) matrix addition and multiplication

$$
\begin{array}{lccc}
\text{Ciphertexts} & \mathbb{Z}^{N\times N} \times \mathbb{Z}^{N\times N} & \xrightarrow{+,\times} & \mathbb{Z}^{N\times N} \\
 & \big\downarrow{\scriptstyle \delta,\delta} & & \big\downarrow{\scriptstyle \delta} \\
\text{Plaintexts} & \mathbb{Z} \times \mathbb{Z} & \xrightarrow{+,\times} & \mathbb{Z}
\end{array}
$$

- One must add some noise, otherwise broken by linear algebra
  - $C \cdot \mathbf{v} = \mu \cdot \mathbf{v} + \mathbf{e} \pmod q$
  - for message $\mu \in \mathbb{Z}$, for some small noise $\mathbf{e}$.
  - Security based on LWE problem.

Ciphertext matrices: slow noise growth

- Noise grow of ciphertext multiplication [GSW13]:
  - $C_1 \cdot \mathbf{v} = \mu_1 \cdot \mathbf{v} + \mathbf{e}_1 \pmod q$, $C_2 \cdot \mathbf{v} = \mu_2 \cdot \mathbf{v} + \mathbf{e}_2 \pmod q$
  - $(C_1 \cdot C_2) \cdot \mathbf{v} = C_1 \cdot (\mu_2 \cdot \mathbf{v} + \mathbf{e}_2) = (\mu_2 \cdot \mu_1) \cdot \mathbf{v} + \mathbf{e}_3$
  - with $\mathbf{e}_3 = \mu_2 \cdot \mathbf{e}_1 + C_1 \cdot \mathbf{e}_2$
- Slow noise growth:
  - Ensure $\mu_i \in \{0, 1\}$, using only NAND gates $\mu_3 = 1 - \mu_1 \cdot \mu_2$
  - Ciphertext flattening: ensure $C_i \in \{0, 1\}^{N \times N}$, using binary decomposition and $\mathbf{v} = (s_1, \ldots, 2^\ell s_1, \ldots, s_n, \ldots, 2^\ell s_n)$.
  - If $\|\mathbf{e}_1\|_\infty \leq B$ and $\|\mathbf{e}_2\|_\infty \leq B$, $\|\mathbf{e}_3\|_\infty \leq (N+1) \cdot B$
- Leveled FHE
  - At depth L, $\|\mathbf{e}\|_\infty \leq (N+1)^L \cdot B$
  - One can take $q > 8 \cdot B \cdot (N+1)^L$ and accommodate polynomial depth L.

Ciphertext matrices: slow noise growth

- Noise grow of ciphertext multiplication [GSW13]:
  - $C_1 \cdot \mathbf{v} = \mu_1 \cdot \mathbf{v} + \mathbf{e}_1 \pmod q$, $C_2 \cdot \mathbf{v} = \mu_2 \cdot \mathbf{v} + \mathbf{e}_2 \pmod q$
  - $(C_1 \cdot C_2) \cdot \mathbf{v} = C_1 \cdot (\mu_2 \cdot \mathbf{v} + \mathbf{e}_2) = (\mu_2 \cdot \mu_1) \cdot \mathbf{v} + \mathbf{e}_3$
  - with $\mathbf{e}_3 = \mu_2 \cdot \mathbf{e}_1 + C_1 \cdot \mathbf{e}_2$
- Slow noise growth:
  - Ensure $\mu_i \in \{0, 1\}$, using only NAND gates $\mu_3 = 1 - \mu_1 \cdot \mu_2$
  - Ciphertext flattening: ensure $C_i \in \{0, 1\}^{N \times N}$, using binary decomposition and $\mathbf{v} = (s_1, \ldots, 2^\ell s_1, \ldots, s_n, \ldots, 2^\ell s_n)$.
  - If $\|\mathbf{e}_1\|_\infty \leq B$ and $\|\mathbf{e}_2\|_\infty \leq B$, $\|\mathbf{e}_3\|_\infty \leq (N+1) \cdot B$
- Leveled FHE
  - At depth L, $\|\mathbf{e}\|_\infty \leq (N+1)^L \cdot B$
  - One can take $q > 8 \cdot B \cdot (N+1)^L$ and accommodate polynomial depth L.

Ciphertext matrices: slow noise growth

- Noise grow of ciphertext multiplication [GSW13]:
  - $C_1 \cdot \mathbf{v} = \mu_1 \cdot \mathbf{v} + \mathbf{e}_1 \pmod q$, $C_2 \cdot \mathbf{v} = \mu_2 \cdot \mathbf{v} + \mathbf{e}_2 \pmod q$
  - $(C_1 \cdot C_2) \cdot \mathbf{v} = C_1 \cdot (\mu_2 \cdot \mathbf{v} + \mathbf{e}_2) = (\mu_2 \cdot \mu_1) \cdot \mathbf{v} + \mathbf{e}_3$
  - with $\mathbf{e}_3 = \mu_2 \cdot \mathbf{e}_1 + C_1 \cdot \mathbf{e}_2$
- Slow noise growth:
  - Ensure $\mu_i \in \{0, 1\}$, using only NAND gates $\mu_3 = 1 - \mu_1 \cdot \mu_2$
  - Ciphertext flattening: ensure $C_i \in \{0, 1\}^{N \times N}$, using binary decomposition and $\mathbf{v} = (s_1, \ldots, 2^\ell s_1, \ldots, s_n, \ldots, 2^\ell s_n)$.
  - If $\|\mathbf{e}_1\|_\infty \leq B$ and $\|\mathbf{e}_2\|_\infty \leq B$, $\|\mathbf{e}_3\|_\infty \leq (N+1) \cdot B$
- Leveled FHE
  - At depth L, $\|\mathbf{e}\|_\infty \leq (N+1)^L \cdot B$
  - One can take $q > 8 \cdot B \cdot (N+1)^L$ and accommodate polynomial depth L.

Fourth generation: homomorphic encryption for approximate numbers

- Homomorphic encryption for real numbers [CKKS17]
  - Floating point arithmetic, instead of exact arithmetic.
  - Starting point: Regev's scheme.
  - Homomorphism: $\delta : \mathbb{Z}_q[\mathbf{x}] \to \mathbb{Z}_q$ given by evaluation at $\mathbf{s}$

$$\begin{tikzcd}
\text{Ciphertexts} & \mathbb{Z}_q[\mathbf{x}] \times \mathbb{Z}_q[\mathbf{x}] \arrow[r, "{+,\times}"] \arrow[d, "{\delta,\delta}"] & \mathbb{Z}_q[\mathbf{x}] \arrow[d, "\delta"] \\
\text{Plaintexts} & \mathbb{Z}_q \times \mathbb{Z}_q \arrow[r, "{+,\times}"] & \mathbb{Z}_q
\end{tikzcd}$$

- One must add some noise, otherwise broken by linear algebra.
  - $f(\mathbf{s}) = m + e \bmod q$, for small $e \in \mathbb{Z}_q$
  - Noise only affects the low-order bits of m: approximate computation, as in floating point arithmetic.
  - Application: neural networks.

Fourth generation: homomorphic encryption for approximate numbers

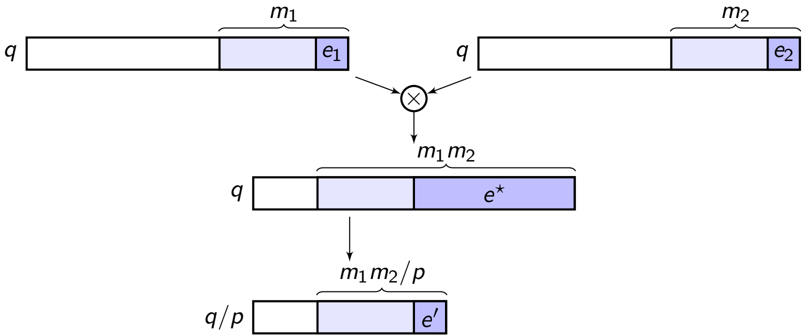
- Homomorphic encryption for real numbers [CKKS17]
  - Floating point arithmetic, instead of exact arithmetic.
  - Starting point: Regev's scheme.
  - Homomorphism: $\delta : \mathbb{Z}_q[\mathbf{x}] \to \mathbb{Z}_q$ given by evaluation at $\mathbf{s}$

$$\begin{tikzcd}
\text{Ciphertexts} & \mathbb{Z}_q[\mathbf{x}] \times \mathbb{Z}_q[\mathbf{x}] \arrow[r, "{+,\times}"] \arrow[d, "{\delta,\delta}"] & \mathbb{Z}_q[\mathbf{x}] \arrow[d, "\delta"] \\
\text{Plaintexts} & \mathbb{Z}_q \times \mathbb{Z}_q \arrow[r, "{+,\times}"] & \mathbb{Z}_q
\end{tikzcd}$$

- One must add some noise, otherwise broken by linear algebra.
  - $f(\mathbf{s}) = m + e \bmod q$, for small $e \in \mathbb{Z}_q$
  - Noise only affects the low-order bits of m: approximate computation, as in floating point arithmetic.
  - Application: neural networks.

[CKKS17]: ciphertext multiplication and rescaling

- Ciphertext multiplication $c(\mathbf{x}) = c_1(\mathbf{x}) \cdot c_2(\mathbf{x})$
  - $c(\mathbf{s}) = (m_1 + e_1) \cdot (m_2 + e_2) = m_1 m_2 + e^\star \pmod{q}$
  - with $e^\star = m_1 e_2 + e_1 m_2 + e_1 e_2$.
- Rescaling of ciphertext:
  - $c'(\mathbf{x}) = \lfloor \mathbf{c}(x)/p \rceil \pmod{q/p}$
  - Valid encryption of $\lfloor m/p \rceil$ with noise $\simeq e/p$
  - Similar to modulus switching



Conclusion

- Main challenge: make FHE pratical !
  - New primitives
  - Libraries (HElib)
  - Compiler to homomorphic evaluation
- Applications
  - Homomorphic machine learning: evaluate a neural network without revealing the weights.
  - Genome-wide association studies: linear regression, logistic regression.

References

BGV11 Zvika Brakerski, Craig Gentry, Vinod Vaikuntanathan. Fully Homomorphic Encryption without Bootstrapping. Electron. Colloquium Comput. Complex. 18: 111 (2011)

CKKS17 Jung Hee Cheon, Andrey Kim, Miran Kim, Yong Soo Song. Homomorphic Encryption for Arithmetic of Approximate Numbers. ASIACRYPT (1) 2017: 409-437

GSW13 Craig Gentry, Amit Sahai, Brent Waters. Homomorphic Encryption from Learning with Errors: Conceptually-Simpler, Asymptotically-Faster, Attribute-Based. CRYPTO (1) 2013: 75-92

Gen09 Craig Gentry. Fully homomorphic encryption using ideal lattices. STOC 2009: 169-178

R05 Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. STOC 2005: 84-93