Two round multiparty computation via Multi-key fully homomorphic encryption with faster homomorphic evaluations

NingBo Li[1#], TanPing Zhou[1#*], XiaoYuan Yang[1], YiLiang Han[1], Longfei Liu[1], WenChao Liu[1]

[1] Key Laboratory of Network & Information Security under the People's Armed Police, College of Cryptography Engineering, Engineering University of People's Armed Police, Xi'an, 710086, China

850301775@qq.com

Abstract. Multi-key fully homomorphic encryption (MKFHE) allows computations on ciphertexts encrypted by different users (public keys), and the results can be jointly decrypted using the secret keys of all the users involved. The NTRU-based scheme is an important alternative to post-quantum cryptography, but the NTRU-based MKFHE has the following drawbacks, which cause it inefficient in scenarios such as secure multi-party computing (MPC). One is the relinearization technique used for key switching takes up most of the time of the scheme's homomorphic evaluation, the other is that each user needs to decrypt in sequence, which makes the decryption process complicated. We propose an efficient leveled MKFHE scheme, which improves the efficiency of homomorphic evaluations, and constructs a two-round (MPC) protocol based on this. Firstly, we construct an efficient single key FHE with less relinearization operations. We greatly reduces the number of relinearization operations in homomorphic evaluations process by separating the homomorphic multiplication and relinearization techniques. Furthermore, the batching technique and a specialization of modulus can be applied to our scheme to improve the efficiency. Secondly, the efficient single-key homomorphic encryption scheme proposed in this paper is transformed into a multi-key vision according to the method in LTV12 scheme. Finally, we construct a distributed decryption process which can be implemented independently for all participating users, and reduce the number of interactions between users in the decryption process. Based on this, a two-round MPC protocol is proposed. Experimental analysis shows that the homomorphic evaluation of the single-key FHE scheme constructed in this paper is 2.4 times faster than DHS16, and the MKFHE scheme constructed in this paper can be used to implement a two-round MPC protocol effectively, which can be applied to secure MPC between multiple users under the cloud computing environment.

Keywords: Multi-key fully homomorphic encryption (MKFHE), NTRU, two-round MPC, relinearization

Authors contribute equally to this work

1 Introduction

People are increasingly inclined to store large amounts of data on powerful cloud servers and outsource the cumbersome and complicated data computing process to the cloud. Although the cloud facilitates the storage and computation of big data, it is also vulnerable to be attacked by illegal organizations and users [1], thus triggering a serious issue that cannot be ignored about how to protect users' individual privacy and data security?

Fully homomorphic encryption (FHE) [2] allows arbitrary computation on encrypted data without access to user's secret key, it has exchangeable properties for data encryption and computation, and can be used to protect the privacy and security processing of data in cloud computing environment.

Following the breakthrough blueprint [3] of FHE designed by Gentry in 2009, FHE has come a long way [4-9]. Traditional FHE is only suitable for scenarios where the computations of ciphertext involve a single user, since it requires all the input ciphertext to be encrypted under the same key. However, in many real-world scenarios, it is often necessary to perform computations on encrypted data corresponding to different users, while ensuring that the user's individual privacy is not exposed.

Multi-key fully homomorphic encryption (MKFHE) [10] allows computations on ciphertexts encrypted by different parties without trust, and the results can be jointly decrypted using the secret keys of all the users involved. Meanwhile, the process of computations on ciphertexts can be outsourced to the cloud offline, which avoid the interaction between the users, and can be applied to implement the secure multi-party computing (MPC) [11-14] of multi-users in the cloud computing environment.

Similar to traditional single-key FHE, the type of current MKFHE mainly include NTRU type, GSW type and BGV type.

In 2012, L´opez-Alt et al. first proposed the concept of MKFHE, and construct a MKFHE scheme [LTV12] based on a variant NTRU public key cryptosystem [15], which is a variant of the original NTRU scheme in [16]. Its security relies on two assumptions: the ring-learning with errors (RLWE) assumption and the decisional small polynomial ratio (DSPR) assumption. [17] improved the efficiency of [LTV12] by optimizing parameters, introducing a specialization of the ring structure and modulus. In PKC2017, Chongchitmate et al. proposed an NTRU-type MKFHE scheme [18], which can protect the circuit privacy. This scheme proposes a basic framework for constructing MKFHE with circuit privacy characteristics, and constructs a basic framework based on this. 3 rounds of on-the-fly MPC protocol.

Clear and McGoldrick [19] proposed the first GSW-type MKFHE scheme CM15 based on the learning with error (LWE) problem whose security can be reduced to the worst-case hardness of problems on ideal lattices. Mukherjee and Wichs [20] simplified [CM15] and gave a construction of MKFHE scheme MW16 based on LWE. [MW16] can be used to construct a simple 1-round threshold decryption protocol and a two-round MPC protocol. Both [19] and [20] need to determine the parties involved in homomorphic computation in advance and

any new party cannot be allowed to join in during the homomorphic computation. This type of MKFHE is called single-hop in [21], comparing to multi-hop MKFHE whose result ciphertext can be employed to further evaluation with new parties, i.e. any new party can dynamically join the homomorphic evaluation at any time. Another similar concept named fully dynamic MKFHE was proposed in [22], which means that the bound of number of users does not need to be input during the setup procedure.

In TCC2017, Chen et al. proposed a BGV-type multi-hop MKFHE scheme [23], which supports the Chinese Remainder Theorem (CRT)-based ciphertexts packing technique, and simplifies the ciphertext extension process in MKFHE. What's more, [23] admits a threshold decryption protocol and two-round MPC protocol.

Comparing to BGV-type and GSW-type MKFHE, NTRU-based MKFHE scheme is simple and much faster. Furthermore, the ciphertext of NTRU-type scheme is a polynomial, thus the implementation of NTRU-type scheme is efficient, and the process of ciphertext extension is not required when extending a single-key NTRU-type FHE scheme to a multi-key vision.

Contributions. In this paper, we propose an efficient leveled MKFHE scheme which improves the efficiency of homomorphic evaluations, and constructs a two-round multiparty computation (MPC) protocol based on this.

• Optimized the single-key leveled FHE scheme in DHS16. We reduce the number of relinearization operations in homomorphic evaluations process by separating the homomorphic multiplication and relinearization techniques.

• Construct a multi-key leveled FHE scheme. Comparing to LTV12, the relinearization process are implemented after evaluating two levels circuit, which can reduce the computational complexity significantly. Besides that, only the evaluation keys whose corresponding users are existed in at least two ciphertexts are employed in relinearization process, which is efficient and important in real applications.

• Construct an efficient two-round MPC based on the multi-key FHE scheme in this paper. We construct a distributed decryption process which can be implemented independently for all participating users, thus reduce the interaction processes between users in the decryption process.

2 Background

2.1 Preliminaries

In this paper, the bold upper case letters denote matrices, and the bold lower case letters denote vectors, and all the vectors are represented as columns. For a vector $\mathbf{a}$ we use $\mathbf{a}[i]$ to denote the i-th element in $\mathbf{a}$. For a positive integer $n \in \mathrm{N}$, let $[n] = \{1, ..., n\}$. For a distribution A, let $x \leftarrow A$ denote x is chosen according to a distribution A.

For security parameter λ, let $d = d(\lambda)$ be a positive integer, and let $\varphi_d(x) = \varphi(x)=x^n + 1$ be the d-th cyclotomic polynomial, and the degree $n = \phi(d)$ is a power of 2, where $\phi(\cdot)$ is the Euler's totient function.

The polynomial ring used in our scheme is defined as $R = \mathrm{Z}(x)/\varphi(x)$, and all operations over the ciphertext are performed in $R_q = R/qR$ where the modulus $q = q(\lambda)$ is a prime or a power of prime. Elements in R_q are polynomials with coefficients in $[-q/2, q/2)$(except for $q = 2$). We also define a B=$B(\lambda)$-bound error distribution χ over the ring $R = \mathrm{Z}(x)/\varphi(x)$, which means that the coefficients of polynomial sampled from χ are at most B in absolute value. For $a \in R$, we use $\|a\|_\infty = \max_{0 \le i \le n-1} |a_i|$ to denote the standard $\|\cdot\|_\infty$-norm.

The security of our scheme is based on the ring learning with error (RLWE) assumption and the decisional small polynomial ratio (DSPR) assumption. Here we give a brief introduction to them.

Definition 1 (Ring Learning With Error (RLWE) Assumption). *The (decisional) RLWE assumption is a variant of learning with error (LWE) assumption. RLWE assumption states that it is infeasible to distinguish the following two distributions: First distribution is the uniform samples* $(\mathbf{a}_i, b_i) \in R_q^{n+1}$. *In the second distribution, sampled* $\mathbf{a}_i \leftarrow R_q^n$ *and* $\mathbf{s} \leftarrow R_q^n$ *uniformly,* $e_i \leftarrow \chi$, *and the second distribution is the specialization samples* $(\mathbf{a}_i, b_i) \in R_q^{n+1}$ *where* $b_i =< \mathbf{a}_i, \mathbf{s} > +e_i$.

Definition 2 (Decisional Small Polynomial Ratio (DSPR) Assumption). *Given the ring* R *and* R_q, *and a* B*-bound error distribution* χ *over* R, *the (decisional) DSPR assumption states that it is infeasible to distinguish the following two distributions:*

- *a polynomial* $h = tg/f$, *where* $f = tf' + 1$ *is invertible over* R_q, *and* $f', g \leftarrow \chi$.
- *a polynomial* h *sampled uniformly at random over* R_q.

Stehlé and Steinfeld states that the DSPR assumption is hard even for unbounded adversaries when the n-th cyclotomic polynomial's degree n is a power of 2, and the error distribution χ is a discrete Gaussian distribution $\mathrm{D}_{\mathrm{Z}^n,\sigma}$ for $\sigma > \sqrt{q} \cdot poly(n)$.

2.2 Two subroutines

Here introduce two subroutines BitDecomp$(\cdot)$ and Powersof $2(\cdot)$ which are widely used in FHE schemes. Let $\beta = \lfloor \log q \rfloor + 1$, and describe these two subroutines as follows.

BitDecomp$(\mathbf{x} \in R_q^n, q)$: Given a polynomial vector $\mathbf{x} \in R_q^n$, write it as $\mathbf{x} = \sum_{j=0}^{\beta-1} 2^j \mathbf{u}_j$ with all $\mathbf{u}_j \in R_2^n$, and output $\mathbf{U} = [\mathbf{u}_0, \mathbf{u}_1, ..., \mathbf{u}_{\beta-1}] \in \{0,1\}^{n \cdot \beta}$.

Powersof $2(\mathbf{y} \in R_q^n, q)$: Let $\mathbf{v}_j = 2^j \mathbf{y} \in R_q^n$, $j \in \{0, 1, ..., \beta - 1\}$, and output $\mathbf{V} = [\mathbf{v}_0, \mathbf{v}_1, ..., \mathbf{v}_{\beta-1}] \in R_q^{n \cdot \beta}$.

It's obviously to verify that $\langle \text{BitDecomp}(\mathbf{x}, q), \text{Powersof } 2(\mathbf{y}, q) \rangle = \langle \mathbf{x}, \mathbf{y} \rangle \bmod q$.

2.3 Basic NTRU homomorphic encryption scheme

Here we give an introduction of the basic NTRU homomorphic encryption scheme in [10], which is a variant of the original NTRU scheme in [15, 16] to support homomorphisms and improve its security.

Setup(1^λ) : For security parameter, given the following parameters which have been defined above: a integer $n = n(\lambda)$, the prime modulus $q = q(\lambda)$, the polynomial ring $R = \mathrm{Z}(x)/x^n + 1$ and $R_q = R/qR$, and a B=$B(\lambda)$-bound error distribution χ over the ring R.

KeyGen(1^λ) : Sample polynomials $f', g \leftarrow \chi$, and set $f = 2f' + 1 \in R$ (If f is not invertible over R_q, resample f') so that $f \equiv 1 \bmod 2$. Set the public key $pk := h = 2gf^{-1} \in R_q$, and the secret key $sk := f \in R$.

$Enc(pk, m)Sample$s,e$\leftarrow \chi$

$$c := hs + 2e + m \in R_q$$

$Enc(pk, m)$Sample $s, e \leftarrow \chi$, and output the ciphertext

$$c := hs + 2e + m \in R_q$$

$Dec(sk, c)$Compute $\mu = fc \in R_q$ as

$$\begin{aligned} \mu &= fc(\mathrm{mod}q) \\ &= f(hs + 2e + m)(\mathrm{mod}q) \\ &= 2(gs + fe) + fm \in R_q \end{aligned}$$

Since $f \equiv 1 \bmod 2$ and $|2(gs + fe) + fm| < q/2$, the message is recovered by $m' = \mu \bmod 2$.

Given two ciphertexts $c_1 = h_1s_1 + 2e_1 + m_1 \in R_q$ and $c_2 = h_2s_2 + 2e_2 + m_2 \in R_q$ that encrypt messages m_1 and m_2 under the same secret key f, it's not difficult to verify its homomorphism properties of addition and multiplication.

$$\begin{aligned} f(c_1 + c_2) &= f(h_1s_1 + 2e_1 + m_1 + h_2s_2 + 2e_2 + m_2) \\ &= 2(g_1s_1 + g_2s_2 + f(e_1 + e_2)) + f(m_1 + m_2) \end{aligned}$$

$$\begin{aligned} f^2 \cdot c_1c_2 &= f^2[(h_1s_1 + 2e_1 + m_1)(h_2s_2 + 2e_2 + m_2)] \\ &= 2E_{error} + f^2 \cdot m_1m_2 \end{aligned}$$

Here we mainly introduce the multikey homomorphic properties which the basic NTRU scheme naturally has.

Given two ciphertexts $c_1 = h_1s_1 + 2e_1 + m_1 \in R_q$ and $c_2 = h_2s_2 + 2e_2 + m_2 \in R_q$ that encrypt messages m_1 and m_2 under the public key h_1 and h_2 respectively, and the corresponding secret key are f_1 and f_2. Some simple algebraic expressions (see the expressions [1] and [2]) show that the sum or multiplication of c_1 and c_2 can be decrypted by their "jointly secret key" f_1f_2:

$$
\begin{aligned}
f_1f_2(c_1+c_2) &= f_1f_2(h_1s_1+2e_1+m_1+h_2s_2+2e_2+m_2)\\
&= 2(f_1f_2(e_1+e_2)+f_2g_1s_1+f_1g_2s_2)+f_1f_2(m_1+m_2)\\
&= 2e_{add}+f_1f_2(m_1+m_2)\\
f_1f_2(c_1c_2) &= f_1f_2(h_1s_1+2e_1+m_1)(h_2s_2+2e_2+m_2)\\
&= 2(2g_1s_1g_2s_2+2f_2g_1s_1e_2+f_2g_1s_1m_2+2f_1g_2s_2e_1\\
&+ f_1g_2s_2m_1)+f_1f_2(4e_1e_2+2e_1m_2+2e_2m_1)+f_1f_2m_1m_2\\
&= 2e_{mult}+f_1f_2(m_1m_2)
\end{aligned}
$$

This shows that decrypting c_1+c_2 and c_1c_2 using the joint secret key f_1f_2 results in the sum and product of the two messages, assuming that the error e_{add} and e_{mult} do not grow to be too large. Furthermore, as for some complicated circuit, we can observe that the ciphertext c resulting from evaluating a multivariate polynomial function on the input ciphertext of N users can be decrypted by the jointly secret key $\prod_{i=1}^{N} f_i^{d_i}$, where d_i denotes the degree of the i-th variable in the polynomial function. In other words, the secret key required to decrypt the result ciphertext is not only dependent on the involved users, but also the evaluating circuit. For example, the secret key to jointly decrypt the evaluated ciphertext ${c_1}^2+c_2$ is $f_1^2f_2$, which is related to the involved users and the evaluating function $f(x_1,x_2)={x_1}^2+x_2$.

With the assumption of DSPR, the security of the basic NTRU homomorphic encryption scheme can be proveR_qd under the RLWE assumption following two steps: 1. Based on the hardness of DSPR assumption, the public key $h=2gf^{-1}$ can be replaced by $2h'$ for a uniformly sampled h'. 2. Once the step 1 is done, we can change the ciphertext $c{=}hs+2e+m$ to $c^*{=}u+m$ with the RLWE assumption, where u is uniformly sampled from R_q. As for arbitrary adversary, the advantage of distinguish c^* and c is both $1/2$ since u is uniformly distributed in R_q, which is independent of the message m.

2.4 Cryptographic Definitions of Multi-key FHE

Definition 3 (Multi-key FHE scheme). *A leveled multi-key FHE scheme consists of a set of algorithms described as follows:*

- *Setup*$(1^\lambda,1^K,1^L)$ *Given the security parameter* λ*, a bound* K *on the number of keys, a bound* L *on the circuit depthoutput the public parameter* pp *.*
- *Gen*(pp)*Given the public parameter* pp*output the public key and secret key of party* $i(i=1,...,K)$*and output the materials which are required for the generation of evaluation keys* evk*.*
- *Enc*(pp,pk_i,m)*Given the public key* pk_i *of party* i *and a message* μ*output the ciphertext* ct_i *which contains the index of the corresponding secret key and the level tag.*
- *Dec*$(pp,(sk_{i_1},sk_{i_2},...,sk_{i_k}),ct_S)$*Given a ciphertext* ct_S *corresponding to a set of parties* $S=\{i_1,i_2,...,i_k\}\subseteq[K]$*and their secret keys* $sk_S=\{sk_{i_1},sk_{i_2},...,sk_{i_k}\}$*, output the message* μ_j

– $Eval(pp, evk, C, (ct_{S_1}, pk_{S_1}), ..., (ct_{S_t}, pk_{S_t}))$ *Given t tuples* $\{(ct_{S_i}, pk_{S_i})\}_{i=1,...,t}$ *and a boolean circuit* $\mathcal{C}$ *which is needed to be evaluatedeach tuple contains a ciphertext* ct_{S_i} *corresponding to a set of secret keys indexed by* $S_i = i_1, ..., i_{k_i} \subseteq [K]$*and a set of public keys* $pk_{S_i} = \{pk_j, \forall j \in S_i\}$*. Output a ciphertext ct corresponding to a set of secret keys indexed by*$S = \cup_{i=1}^{t} S_i \subseteq [K]$*¡*

If the input ciphertext of $Eval(\cdot)$ can be fresh ciphertext or intermediate results after any homomorphic operation, the MKFHE scheme satisfies the multi-hop property.

Definition 4 (Correctness). *A leveled multi-hop MKFHE scheme is correct if for any circuit* $\mathcal{C}$ *of depth at most L with t input wires and a set of tuples* $\{(ct_{S_i}, pk_{S_i})\}_{i\in\{1,...,t\}}$*, letting* $\mu_i = Dec(sk_{S_i}, ct_{S_i})$*, where* $sk_{S_i} = \{sk_j, \forall j \in S_i\}$*,*$i = 1, ..., t$*, it holds that*

$$\Pr[Dec(sk_S, Eval(\mathcal{C}, (ct_{S_1}, pk_{S_1}), ..., (ct_{S_t}, pk_{S_t}))) \neq \mathcal{C}(\mu_1, ..., \mu_t)] = negl(\lambda)$$

Where $S = \cup_{i=1}^{t} S_i \subseteq [K]$*,* $pp \leftarrow Setup(1^\lambda, 1^K, 1^L)(pk_j, sk_j) \leftarrow Gen(pp)$ *for* $j \in [S]$*.*

Definition 5 (Compactness). *A leveled multi-hop MKFHE scheme is compact if there exists a polynomial* $poly(\cdot, \cdot, \cdot)$ *such that* $|ct| \leq poly(\lambda, K, L)$*, which means that the length of ct is independent of the circuit* $\mathcal{C}$*, but can depend of* λ*,* K *and* L*.*

3 Efficient leveled NTRU-based FHE scheme

3.1 Motivation

Recall the basic NTRU-type homomorphic encryption scheme in section 2.3. What we already know is that the secret key required to decrypt the result NTRU ciphertext is not only dependent on the involved users, but also the evaluating circuit,

In order to eliminate the influence of the circuit evaluated on the decryption and maintain the consistency of the decrypting form, we can use the relinearization technology introduced in [5] to transform the ciphertext into one that can be jointly decrypted by the unified secret key $\prod_{i=1}^{N} f_i$ of all the involved users after homomorphic operation.

In RLWE-based leveled FHE scheme BV11a, the ciphertext form is a polynomial vector, thus the ciphertext dimension after homomorphic multiplication will expand quadratically, which may cause trouble for the storage and operation of the ciphertext. To handle it, relinearization technique was proposed in [5] to reduce the dimension of the result ciphertext to the original extent.

However, relinearization is by far the most expensive operation in a leveled FHE scheme. We noticed that in the NTRU-type FHE scheme, the ciphertext form is a simple polynomial in R_4, so the homomorphic operations on the ciphertexts does not cause dimension expansion. In other words, we do not have to do

relinearization after every homomorphic operation. However, in order to maintain the consistency of decryption, it is still necessary to perform relinearization operation to transform the result ciphertext to one under the secret key .

In this paper, we consider to perform relinearization operation on the ciphertext after k4 homomorphic evaluations to reduce the number of complicated relinearization operations, thus reducing the computational complexity of homomorphic evaluations and improving the scheme's efficiency. It should be noted that the modulus-switching operation is still necessary after each homomorphic evaluation to control the noise in the ciphertext, i.e. we separate the relinearization technique from the modulus -switching technique to make them no longer bundle.

3.2 Parameters selection in DHS16

In [17], Doröz et al. optimize the parameters selection in the light of recent theoretical and experimental results in the field of lattice reduction so as to reduce the size of public key significantly. Beyond that, the secret key remain the same for all levels so that the evaluation keys can be computed by the initial evaluation keys at level 0. In this paper, we follow the method of parameters selection in DHS16 to improve scheme's efficiency comparing to LTV12, and give a simple introduction to it in this section.

Assume that the modulus of each level of the circuit is a decreasing sequence $q_0 > q_1 > \cdots > q_L$, and set $q_i = p^{t-i}$ for $i = 0, \cdots, t-1$, where $p \in \mathbb{Z}$ is a prime integer. The secret key $f \in R_{q_0}$ remains the same for all levels and is invertible in all rings $\mathbb{Z}_{q_i}$.

Lemma 1 (Lemma 3 in [17]). Let p be a prime, and let f be a polynomial. If f is a unit in R_p, then f is a unit in R_{p^k} for $k \geqslant 1$. According to lemma 1, the evaluation keys for level i can be simply computed by $\zeta^{(i)} = \zeta^{(0)} \bmod q_i$.

DHS16 introduces batching technology [24] to package multiple input plaintext into the same ciphertext and realize simultaneous input of multiple plaintext, thus improving the encryption efficiency and reducing the number of ciphertext required for homomorphic operation. Batching technology can also be applied to our scheme to improve its efficiency. The details of batching process are presented in section 4.1 in [17].

3.3 Optimized leveled single-key FHE scheme

In this paper, we construct an leveled single-key FHE scheme, which can homomorphically evaluate the circuit more efficiently than DHS16.

- **Setup**(1^λ) : Given the security parameter λ, an integer $n = n(\lambda)$, a prime integer p=$p(\lambda)$, the prime modulus $q = q(\lambda)$, the polynomial ring $R = \mathbb{Z}(x)/x^n+1$ and $R_q = R/qR$, and a B=$B(\lambda)$-bound error distribution χ over the ring R. Define a series of decreasing modulus q_0=$p^t > q_1 > ... > q_{t-1}$, one modulus per circuit level, and require that q_i=p^{t-i} for $i \in \{0, ..., t-1\}$.

- **KeyGen**(1^λ) : Choose polynomials $f', g \leftarrow \chi$, and set $f = 2f'+1$ so that $f \equiv 1 \bmod 2$. Set $h = 2g/f \in R_{q_0}$ (If f is not invertible over R_{q_0}, resample f'). Sample $\mathbf{s}, \mathbf{e} \leftarrow \chi^{\lceil \log q_0 \rceil}$, let $\zeta^{(0)} := h\mathbf{s}+2\mathbf{e}_\zeta+\text{Powersof } 2((f)^3) \in R_{q_0}^{\lceil \log q_0 \rceil}$, and $\zeta^{(i \to i+2)} \triangleq \zeta^{(0)} \bmod q_{i+1}$. Let $f^{(i)}(x) = f(x)^{-1}(\bmod q_i)$, thus $f(x)f^{(i)}(x) = 1(\bmod q_i)$.
  Output:$sk := f \in R_{q_0}$(the secret key f remains the same for all levels), and $pk := \left\{h, \zeta^{(0)}\right\}$, where $\zeta^{(0)}$ denotes the evaluation key in level 0, and evaluation keys of other levels can be computed by $\zeta^{(0)}$ and the modulus.
- **Enc**(h, m): Input the message m, sample $s^{(0)}$, $e^{(0)} \leftarrow \chi$, and output the ciphertext
$$c^{(0)} := hs^{(0)} + 2e^{(0)} + m \in R_{q_0}$$
- **Dec**$(f, c^{(l)})$: Input the ciphertext $c^{(l)} \in R_{q_l}$, compute
$$\mu := f \cdot c^{(l)} \in R_{q_l}$$
  and output the message $m' := \mu \bmod 2$.
- (1) Add$(c_1^{(i-1)}, c_2^{(i-1)})$:Input two ciphertexts $c_1^{(i-1)}$ and $c_2^{(i-1)}$ at level $i-1$.
  (a) Addition: $\tilde{c}_{add}^{(i-1)} = c_1^{(i-1)} + c_2^{(i-1)}$.
  (b) Modulus switching: $\tilde{c}_{add}^{(i)} \lfloor (q_i/q_{i-1}) \cdot \tilde{c}_{add}^{(i-1)} \rceil_2$, where $\lfloor \cdot \rceil_2$ denotes $\tilde{c}_{add}^{(i)} = \tilde{c}_{add}^{(i-1)} \bmod 2$ (Relinearization is not required as addition doesn't change the secret key).
- (2) Mult$(c_1^{(i-2)}, c_2^{(i-2)}, c_3^{(i-2)}, c_4^{(i-2)})$:input $i-2$ level ciphertext $c_1^{(i-2)}$, $c_2^{(i-2)}$, $c_3^{(i-2)}$, $c_4^{(i-2)}$ at level $(i-2)$.
  (a) Multiplication: $\tilde{c}_1^{(i-2)}=c_1^{(i-2)} \times c_2^{(i-2)} (\bmod q_{i-2})$;$\tilde{c}_2^{(i-2)}=c_3^{(i-2)} \times c_4^{(i-2)} (\bmod q_{i-2})$
  (b) Modulus switching: $\tilde{c}_1^{(i-1)} = \lfloor (q_{i-1}/q_{i-2}) \cdot \tilde{c}_1^{(i-2)} \rceil_2$,
  $\tilde{c}_2^{(i-1)} = \lfloor (q_{i-1}/q_{i-2}) \cdot \tilde{c}_2^{(i-2)} \rceil_2$, where $\lfloor \cdot \rceil_2$ denotes $\tilde{c}_1^{(i-1)} = \tilde{c}_1^{(i-2)} \bmod 2$ and $\tilde{c}_2^{(i-1)} = \tilde{c}_2^{(i-2)} \bmod 2$.
  (c) Multiplication: $\tilde{\tilde{c}}_1^{(i-1)} = \tilde{c}_1^{(i-1)} \cdot \tilde{c}_2^{(i-1)} (\bmod q_{i-1})$
  (d) Relinearization: $\tilde{\tilde{c}}^{(i)} = \left\langle \text{BitDecomp}(\tilde{\tilde{c}}^{(i-1)}), \zeta^{(i-2 \to i)} \right\rangle (\bmod q_{i-1}) \in R_{q_{i-1}}$
  (e) Modulus switching: $\tilde{c}^{(i)} = \lfloor (q_i/q_{i-1}) \cdot \tilde{\tilde{c}}^{(i)} \rceil_2$

Analysis.

(1)Correctness analysis

Here we mainly analyze the correctness of ciphertext multiplication.

Lemma2: The noise growth of evaluating two levels as a block under average case is:

$$B_{i,average} \approx v^4 n^{1.5} B_{i-2}^4 \kappa^3 + 2v^2 n^2 (2B+1)^2 B_{i-2}^2 \kappa^2 + n \log q_i (2B^2 + 6B^3)\kappa + \sqrt{n}(2B+1)$$

where n is the degree, B is the bound of error distribution χ over the ring R, B_i denotes the error bound of the ciphertext at level i, κ is the reduction scale of modulus switching.

(a)Multiplication: $\tilde{c}_1^{(i-2)} = c_1^{(i-2)} \times c_2^{(i-2)}; \tilde{c}_2^{(i-2)} = c_3^{(i-2)} \times c_4^{(i-2)} (\bmod q_{i-2})$

Suppose that $c_k^{(i-2)}f = 2E_k^{(i-2)} + fm_k(modq_{i-2})$ for $k \in \{1,2,3,4\}$. As the corresponding secret key of $\tilde{c}_1^{(i-1)}$ and $\tilde{c}_1^{(i-2)}$ is f^2, we have

$$\tilde{c}_1^{(i-2)}(x)f^2 = 2\tilde{E}_1^{(i-2)} + f^2m_1m_2(modq_{i-2})$$
$$\tilde{c}_2^{(i-2)}(x)f^2 = 2\tilde{E}_2^{(i-2)} + f^2m_3m_4(modq_{i-2})$$

(b)Modulus switching

Lemma3: Let q and p be two odd integer, $c \in R_q$, denote $c' = \lfloor (p/q) \cdot c \rceil_2$, where $\lfloor \rceil_2$ denotes $c' = c \bmod 2$. For arbitrary f that satisfies $||[f^2c]_q||_\infty < q/2 - (q/p) \cdot ||f^2||_1$, we have $[f^2c']_p = [f^2c]_q \bmod 2$ and $||[f^2c]_p||_\infty < (p/q)||[f^2c]_q||_\infty + ||f^2||_1$.

According to lemma3, we have

$$\tilde{c}_1^{(i-1)}(x) = \lfloor (q_{i-1}/q_{i-2}) \cdot \tilde{c}_1^{(i-2)}(x) \rceil_2 (\mathrm{mod} q_{i-1}) \Rightarrow$$
$$[f^2\tilde{c}_1^{(i-1)}(x)]_{q_{i-1}} = [f^2\tilde{c}_1^{(i-2)}(x)]_{q_{i-2}} \bmod 2 = m_1m_2;$$
$$\tilde{c}_2^{(i-1)}(x) = \lfloor (q_{i-1}/q_{i-2}) \cdot \tilde{c}_2^{(i-2)}(x) \rceil_2 (\mathrm{mod} q_{i-1}) \Rightarrow$$
$$[f^2\tilde{c}_2^{(i-1)}(x)]_{q_{i-1}} = [f^2\tilde{c}_2^{(i-2)}(x)]_{q_{i-2}} \bmod 2 = m_3m_4$$

(c)Multiplication: $\tilde{\tilde{c}}^{(i-1)} \triangleq \tilde{c}_1^{(i-1)} \times \tilde{c}_2^{(i-1)} (\mathrm{mod} q_{i-1})$

As the secret key of $\tilde{\tilde{c}}^{(i-1)}$ is f^4, we have

$$\tilde{\tilde{c}}^{(i-1)}f^4 \triangleq (\tilde{c}_1^{(i-1)}f^2)(\tilde{c}_2^{(i-1)}f^2) \triangleq \tilde{\tilde{E}}^{(i-1)} + f^4m_1m_2m_3m_4 (\mathrm{mod} q_{i-1})$$

(d)Relinearization: $\tilde{\tilde{c}}^{(i)} = \left\langle \mathrm{BitDecomp}(\tilde{\tilde{c}}^{(i-1)}), \zeta^{(i-2\to i)} \right\rangle (\mathrm{mod} q_{i-1}) \in R_{q_{i-1}}$

As the evaluation key $\zeta_\tau^{(i-2\to i)}$ is a ciphertext at level $(i-1)$, i.e.

$$\zeta_\tau^{(i-2\to i)}f = 2E_{\zeta_\tau^{(i-2\to i)}} + 2^\tau f^4 (\mathrm{mod} q_{i-1})$$

then we can get

$$\begin{aligned}
\tilde{\tilde{c}}^{(i)}(x)f &= f(\sum\nolimits_{\tau=0}^{\lfloor \log q_{i-1} \rfloor} \tilde{\tilde{c}}_\tau^{(i-1)}(x)\zeta_\tau^{(i-2\to i)}) (\mathrm{mod} q_{i-1}) \\
&= \sum\nolimits_{\tau=0}^{\lfloor \log q_{i-1} \rfloor} \tilde{\tilde{c}}_\tau^{(i-1)}(x)(2E_{\zeta_\tau^{(i-2\to i)}} + 2^\tau f^4) (\mathrm{mod} q_{i-1}) \\
&= 2\sum\nolimits_{\tau=0}^{\lfloor \log q_{i-1} \rfloor} \tilde{\tilde{c}}_\tau^{(i-1)}(x)E_{\zeta_\tau^{(i-2\to i)}} + \sum\nolimits_{\tau=0}^{\lfloor \log q_{i-1} \rfloor} \tilde{\tilde{c}}_\tau^{(i-1)}(x) \cdot 2^\tau f^4 (\mathrm{mod} q_{i-1}) \\
&= 2\sum\nolimits_{\tau=0}^{\lfloor \log q_{i-1} \rfloor} \tilde{\tilde{c}}_\tau^{(i-1)}(x)E_{\zeta_\tau^{(i-2\to i)}} + f^4 \cdot \tilde{\tilde{c}}^{(i-1)}(x) (\mathrm{mod} q_{i-1}) \\
&= 2\sum\nolimits_{\tau=0}^{\lfloor \log q_{i-1} \rfloor} \tilde{\tilde{c}}_\tau^{(i-1)}(x)E_{\zeta_\tau^{(i-2\to i)}} + \tilde{\tilde{E}}^{(i-1)} + f^4m_1m_2m_3m_4 (\mathrm{mod} q_{i-1}) \\
&= \tilde{\tilde{E}}'^{(i-1)} + f^4m_1m_2m_3m_4 (\mathrm{mod} q_{i-1})
\end{aligned}$$

(e)Modulus switching:$c^{(i)}(x) = \lfloor (q_i/q_{i-1}) \cdot \tilde{\tilde{c}}^{(i)}(x) \rceil_2$

$$\begin{aligned} c^{(i)} f &= \tilde{\tilde{c}}^{(i)}(x) f(\mathrm{mod} q_{i-1})(mod 2) \\ &= 2\tilde{E}'^{(i-1)} + f^4 m_1 m_2 m_3 m_4 (mod 2) \\ &= m_1 m_2 m_3 m_4 \end{aligned}$$

(2) Noise growth

In this paper, we set $B_i = 2^{12}, B = 2, \delta = 1.0066$, and compute the noise growth of evaluating one block (one level per block in DHS16).The results are presented in table1.

Table 1. Comparsion of noise growth between DHS16 and our scheme

Log(n)	Log(q)	Noise growth in DHS(bit)	Noise growth in our scheme(bit)
12	155	8.322	8.321
13	331	8.822	8.821
14	622	9.322	9.321
15	1244	9.822	9.821
16	2488	10.322	10.321
17	4976	10.823	10.822

Analysis shows that in the case of two levels per block, the noise growth in our scheme is almost same as DHS16. Therefor we choose the same number of levels and modulus as in DHS16.

3.4 Leveled multi-key FHE scheme

In this section, we extend the leveled single-key FHE scheme in section 3.1 to a leveled multi-key FHE scheme based on the framework of LTV12.

- Setup(1^λ): For security parameter λ, given the following parameters which have been defined above: a integer $n = n(\lambda)$, the prime modulus $q = q(\lambda)$, the polynomial ring $R = \mathbb{Z}(x)/x^n + 1$ and $R_q = R/qR$, and a $B = B(\lambda)$-bound error distribution χ over the ring R. Define a series of decreasing modulus $q_0 = p^t > q_1 > \cdots > q_{t-1}$, one modulus per circuit level, and require that $q_i = p^{t-i}$ for $i \in \{0, ..., t-1\}$.
- KeyGen(1^λ): Choose polynomials $f', g \leftarrow \chi$, and set $f = 2f' + 1$(If f is not invertible over R_q, resample f') so that $f \equiv 1 \bmod 2$.
  Sample $s_\tau, e_\tau \leftarrow \chi$, and for $j \in \{1, 2, 3\}$, $\tau \in \{0, ..., \lfloor \log q_0 \rfloor\}$, compute

$$\zeta_{j,\tau}^{(0)} := hs_\tau + 2e_\tau + 2^\tau f^j \in R_{q_0}$$

  and $\zeta_{j,\tau}^{(i-2\to i)}$ can be computed by $\zeta_{j,\tau}^{(i-2\to i)} \triangleq \zeta_{j,\tau}^{(0)} \bmod q_{i-1}$.

Output $sk := f \in R_{q_0}$(the secret key f remains the same for all levels), and $pk := \left\{h, \zeta_{j,\tau}^{(0)}\right\}_{j\in\{1,2,3\},\tau\in\{0,\ldots,\lfloor \log q_0 \rfloor\}}$.

– Enc(pk, m) Input the message m, sample $s^{(0)}$, $e^{(0)} \leftarrow \chi$, and output the ciphertext

$$c^{(0)} := hs^{(0)} + 2e^{(0)} + m \in R_{q_0}$$

– Dec$(f_1, ..., f_N, c)$: Input the ciphertext $c \in R_p$, suppose the corresponding secret keys of the involved users are $f_1, ..., f_N$, compute

$$\mu := (f_1 \cdots f_N) \cdot c(\mathrm{mod} p)(\mathrm{mod} 2)$$

and output the message μ.

– Eval$(C_\times, c_1^{i-2}, c_2^{i-2}, c_3^{i-2}, c_4^{i-2})$: Here we show how to homomorphically multiply four ciphertexts $c_1^{i-2}, c_2^{i-2}, c_3^{i-2}$ and c_4^{i-2} at level $(i-2)$. We assume that the users associated with each ciphertext is denoted by K_1, K_2, K_3, and K_4 respectively. The public-key set of a fresh encryption is simply the set $\{pk\}$ containing the public key under which it was encrypted, and we set $K_1 \cup K_2 \cup K_3 \cup K_4 = \{pk_1, \ldots, pk_r\}$.
(1) Multiplication: $\tilde{c}_1^{(i-2)} = c_1^{(i-2)} \times c_2^{(i-2)} (\mathrm{mod}\ q_{i-2})$; $\tilde{c}_2^{(i-2)} = c_3^{(i-2)} \times c_4^{(i-2)} (\mathrm{mod}\ q_{i-2})$.
(2) Modulus switching: $\tilde{c}_1^{(i-1)} = \lfloor (q_{i-1}/q_{i-2}) \cdot \tilde{c}_1^{(i-2)} \rceil_2$; $\tilde{c}_2^{(i-1)} = \lfloor (q_{i-1}/q_{i-2}) \cdot \tilde{c}_2^{(i-2)} \rceil_2$ (3) Multiplication: $\tilde{\tilde{c}}^{(i-1)} = \tilde{c}_1^{(i-1)} \cdot \tilde{c}_2^{(i-1)} (\mathrm{mod}\, q_{i-1})$ (4) Relinearization For $v = 1, ..., r$ and $\tau \in \{0, ..., \lfloor \log q_{i-1} \rfloor\}$, define $\tilde{\tilde{c}}_{v-1,\tau}^{(i-1)}$ so that

$$\tilde{\tilde{c}}_{v-1}^{(i-1)} = \sum_{\tau=0}^{\lfloor \log q_{i-1} \rfloor} \tilde{\tilde{c}}_{v-1,\tau}^{(i-1)} 2^\tau$$

is the binary representation of $\tilde{\tilde{c}}_{v-1}^{(i-1)}$, and $\tilde{\tilde{c}}_0^{(i-1)} = \tilde{\tilde{c}}^{(i-1)}$.
(a) If $pk_v \in \{K_1 \cap K_2 \cap K_3 \cap K_4\}$, let

$$\tilde{\tilde{c}}_v^{(i-1)} = \sum_{\tau=0}^{\lfloor \log q_{i-1} \rfloor} \tilde{\tilde{c}}_{v-1,\tau}^{(i-1)} \zeta_{j_v=3,\tau}^{(i-2\to i)}$$

(b) If $pk_v \notin \{K_1 \cap K_2 \cap K_3 \cap K_4\}$, and pk_v is exist in any three sets of K_1, K_2, K_3, K_4, let

$$\tilde{\tilde{c}}_v^{(i-1)} = \sum_{\tau=0}^{\lfloor \log q_{i-1} \rfloor} \tilde{\tilde{c}}_{v-1,\tau}^{(i-1)} \zeta_{j_v=2,\tau}^{(i-2\to i)}$$

(c) If $pk_v \notin \{K_1 \cap K_2 \cap K_3 \cap K_4\}$, and pk_v is exist in any two sets of K_1, K_2, K_3, K_4, let

$$\tilde{\tilde{c}}_v^{(i-1)} = \sum_{\tau=0}^{\lfloor \log q_{i-1} \rfloor} \tilde{\tilde{c}}_{v-1,\tau}^{(i-1)} \zeta_{j_v=1,\tau}^{(i-2\to i)}$$

Finally, we can get $\tilde{\tilde{c}}_r^{(i-1)}$ iteratively.

(5) Modulus switching: $\tilde{c}^{(i)} = \lfloor (q_i/q_{i-1}) \cdot \tilde{\tilde{c}}_r^{(i-1)} \rceil_2$, and output the result ciphertext $\tilde{c}^{(i)}$, whose corresponding secret key is $f_1 \cdot f_2 \cdots f_r$.

To see why relinearization works, see the proof below.

Proof : Suppose that the evaluation key of v-th user is $\zeta_{j_v,\tau}^{(i-2\to i)} = hs_\tau + 2e_\tau + 2^\tau (f_v)^{j_v}$, thus the corresponding secret key of $\tilde{\tilde{c}}_{v-1}^{(i-1)}$ can be represented by $f' \cdot (f_v)^{j_v+1}$, where f'generally denotes the combination of other users' secret keys, and we show that the new secret key after 1-round relinearization by the evaluation key $\zeta_{j_v,\tau}^{(i-2\to i)}$ is changed to $f' \cdot f_v$.

The new ciphertext $\tilde{\tilde{c}}_v^{(i-1)} = \sum_{\tau=0}^{\lfloor \log q_{i-1} \rfloor} \tilde{\tilde{c}}_{v-1,\tau}^{(i-1)} \zeta_{j_v,\tau}^{(i-2\to i)}$, and the decrypting process can be represented by:

$$
\begin{aligned}
&(f' \cdot f_v) \cdot \tilde{\tilde{c}}_v^{(i-1)} \\
&= (f' \cdot f_v) \sum_{\tau=0}^{\lfloor \log q_{i-1} \rfloor} \tilde{\tilde{c}}_{v-1,\tau}^{(i-1)} \zeta_{j_v,\tau}^{(i-2\to i)} (\mathrm{mod} q_{i-1}) \\
&= f' \cdot \sum_{\tau=0}^{\lfloor \log q_{i-1} \rfloor} \tilde{\tilde{c}}_{v-1,\tau}^{(i-1)} (2E_\zeta + 2^\tau (f_v)^{j_v+1}) \\
&= 2f' \sum_{\tau=0}^{\lfloor \log q_{i-1} \rfloor} \tilde{\tilde{c}}_{v-1,\tau}^{(i-1)} E_\zeta + f' \cdot (f_v)^{j_v+1} \sum_{\tau=0}^{\lfloor \log q_{i-1} \rfloor} \tilde{\tilde{c}}_{v-1,\tau}^{(i-1)} 2^\tau \\
&= 2f' \sum_{\tau=0}^{\lfloor \log q_{i-1} \rfloor} \tilde{\tilde{c}}_{v-1,\tau}^{(i-1)} E_\zeta + \tilde{\tilde{c}}_{v-1}^{(i-1)} \left(f' \cdot (f_v)^{j_v+1} \right) (\mathrm{mod} q_{i-1}) \\
&= 2f' \sum_{\tau=0}^{\lfloor \log q_{i-1} \rfloor} \tilde{\tilde{c}}_{v-1,\tau}^{(i-1)} E_\zeta + \left(2\tilde{E}_{v-1}^{(i-1)} + f' \cdot (f_v)^{j_v+1} \cdot C(m_1, ..., m_r) \right) (\mathrm{mod} q_{i-1}) \\
&= 2\tilde{E}'^{(i-1)}_{v-1} + f' \cdot (f_v)^{j_v+1} \cdot C(m_1, ..., m_r) (\mathrm{mod} q_{i-1}) \\
&= C(m_1, ..., m_r) (\mathrm{mod} 2)
\end{aligned}
$$

Finally we can get $\tilde{\tilde{c}}_r^{(i-1)}$ iteratively whose secret key is $\prod_{v=1}^r f_v$.

The optimized multi-key FHE scheme in this section has the following advantages comparing to LTV12:

(1) Relinearization process are implemented after evaluating two levels circuit, which can reduce the computational complexity significantly.

(2) Only the evaluation keys whose corresponding users are existed in at least two ciphertexts are employed in relinearization process, which is efficient and important in real applications.

(3) An efficient two-round MPC can be constructed based on the multi-key FHE scheme in this section.

4 Two-round multiparty computation

MKFHE schemes can be used to construct secure MPC protocols. When executing a MPC protocol, each user involved in homomorphic evaluations is

usually required to decrypt in sequence, which may cause complicated decryption process and inevitable interaction between users. However, in many actual scenarios, we do not want too much interaction, and prefer the final decryption process is completed by each user independently. That is to say, distributed decryption, in most cases, is the preferred decryption method in real life.

4.1 Construction

In this paper, according to the particularity of ciphertext form in NTRU-based leveled FHE scheme, we construct a distributed decryption process which can be implemented by the involved users independently. In our construction, each user will firstly receive the result evaluated ciphertext from the cloud server, and compute the ciphertext with his (her) own secret key to obtain an "intermediate ciphertext". Secondly, all the "intermediate ciphertext" will be sent to the user or organization who requires the evaluated results for decryption.

The whole process is relatively simple and intuitive, and we give a formalized explanation below to demonstrate the feasibility of constructing the distributed decryption by the NTRU-type FHE scheme.

Suppose that the result ciphertext after homomorphically evaluating the circuit C is denoted by $c \in R_{q_L}$, the user set involved in c is $S = \{i_1, ..., i_N\}$, their corresponding secret keys $sk_{i_j} = f_{i_j}$, and the corresponding message is m_{i_j}, $j \in [N]$. Then it holds that

$$(\prod_{j=1}^{N} f_{i_j}) \cdot c = 2E_{error} + C(m_{i_1}, ..., m_{i_N}) (\mathrm{mod} q_L)$$

Step 1. After the cloud return the result ciphertext $c \in R_{q_{t-1}}$ to involved users, each user firstly semi-decrypt the ciphertext using his own secret key and get a semi-ciphertext $c'_{i_j} = f_{i_j} \cdot c$. As the corresponding secret key of c is $\prod_{j=1}^{N} f_{i_j}$, c'_{i_j} doesn't reveal any information of users' messages.

Step 2. After all the users return the semi-ciphertexts c'_{i_j} to the user or organization who are going to decrypt, the user or organization firstly compute $c^{-(N-1)} \in R_{q_{t-1}}$, and then compute

$$\begin{aligned} c^{-(N-1)} \cdot \prod_{j=1}^{N} {c'}_{i_j} &= c^{-(N-1)} \cdot \prod_{j=1}^{N} (f_{i_j} \cdot c) \\ &= c^{-(N-1)} \cdot c^{N-1} \cdot (\prod_{j=1}^{N} f_{i_j}) \cdot c \\ &= (\prod_{j=1}^{N} f_{i_j}) \cdot c (\mathrm{mod} q_L) (\mathrm{mod} 2) \\ &= 2E_{error} + C(m_{i_1}, ..., m_{i_N}) \prod_{j=1}^{N} {c'}_{i_j} (\mathrm{mod} 2) \\ &= C(m_{i_1}, ..., m_{i_N}) \end{aligned}$$

Note that the modulus switching process is needed to reduce the noise at the final process of step 2 to ensure correct decryption.

4.2 Applications in secure genomic diagnoses

Here we present an application of MKFHE for secure genomic diagnoses without revealing patient genomes, which is an optimized vision of [25].

Genes are codes that direct human activity, thousands of monogenic diseases have been diagnosed to be caused by "malignant" mutations in certain genes. In the diagnosis of such diseases, it is necessary to find the exact location where these genetic variants occur, and this often requires comparison between the genes of patients and healthy people. However, genetic data is individual's privacy, genome sharing enables discrimination [25], and even causes crime once genome information is leaked or stolen by malicious and illegal person or organization. How to analyze the patient's genetic data while protecting the privacy of genetic data is a problem worth considering in the current cloud environment.

In [25], Jagadeesh K A et al. apply a cryptographic method called Yao's protocol to perform the desired computation without revealing any participant's input, and introduce a "two-cloud" model to extend the protocol to N parties. However, they need to assume there are two non-colluding cloud servers that facilitate the protocol execution, which is weak in real cloud and is vulnerable to collusion attacks.

MKFHE can effectively solve the problem of colluding cloud, because the cloud can only access and compute the ciphertext of the patient's genetic data.

The process is as follows:

Step1. The medical institution performs gene sequencing on the patients, after that, each individual will get a variant vector of all possible rare missense and nonsense variants in the human genome, and privately denote "1" or "0" to indicate whether they have the specific mutation or not respectively, so that each individual will get a bit string.

Step2. All patients encrypt their genome bits by MKFHE scheme and upload them to the cloud.

Step3. The research institution upload the homomorphic evaluating function of genome data to the cloud.

Step4. The cloud perform the homomorphic evaluating function on the encrypted genome data and return the result ciphertext to the patients.

Step5. The result ciphertext is jointly decrypted by all patients and return the result to the research institution.

5 Experimental results

In this section, we compare the efficiency of our single key holomorphic encryption scheme and DHS16. The results are presented in table 2.

Experimental results show that when we use two layers as a block, the speed of the 36-level homomorphic multiplication is 1.9 times that of the DHS scheme. When we use three layers as a block, the speed of the 36-level homomorphic multiplication is 2.4 times that of the DHS scheme. Therefore, our scheme can run the homomorphic circuit more efficiently.

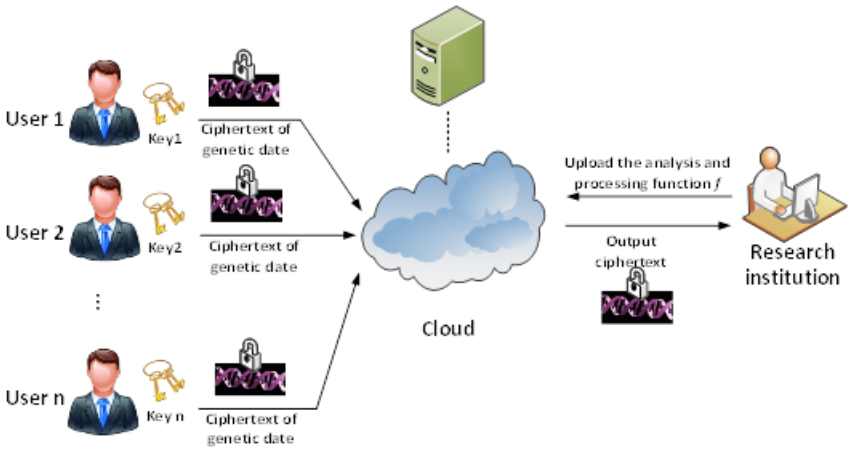

Fig. 1. The flowchart of the secure genomic diagnoses

Table 2. The comparison of homomorphic multiplication between DHS16 and our scheme

Scheme	Level(s) for one relinearization	Number of levels	Total Time for evaluating multiplication gate	Average time for evaluating multiplication gate
DHS16	1	36	168084	4669
Our scheme	2	36	86832	2412
	3	36	68400	1900

6 Conclutions

In this paper, we propose an efficient leveled MKFHE scheme, which improves the efficiency of homomorphic evaluations, and constructs a two-round multiparty computation (MPC) protocol based on this. We reduce the number of relinearization operations in homomorphic evaluations process by separating the homomorphic multiplication and relinearization techniques. We construct a distributed decryption process which can be implemented independently for all participating users, and avoid the interaction between users in the decryption process. Based on this, a two-round MPC protocol is proposed.

References

[1] T. Ristenpart, E. Tromer, H. Shacham, and S. Savage. Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds. In ACM Conference on Computer and Communications Security, pages 199-212, 2009

[2] Rivest R., Adleman L., Dertouzos M.: On Data Banks and Privacy Homomorphisms, pp. 169–177. Academic Press, New York (1978)

[3] Craig Gentry. Fully homomorphic encryption using ideal lattices. In STOC, volume 9, pages 169-178, 2009.

[4] Dijk M V, Gentry C, Halevi S, et al. Fully homomorphic encryption over the integers, International Conference on Theory and Applications of Cryptographic Techniques. Springer-Verlag, 2010:24-43.

[5] Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. Foundations of Computer Science Annual Symposium on, 2011(2):97-106, 2011.

[6] Zvika Brakerski and Vinod Vaikuntanathan. Fully homomorphic encryption from ring-lwe and security for key dependent messages. In Advances in Cryptology-CRYPTO 2011, pages 505-524. Springer, 2011.

[7] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. In Proceedings of the 3rd Innovations in Theoretical Computer Science Conference, pages 309-325.ACM, 2012.

[8] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In Advances in Cryptology-CRYPTO 2013, pages 75-92. Springer, 2013.

[9] Jacob Alperin-Sheriff and Chris Peikert. Faster bootstrapping with polynomial error. In Advances in Cryptology-CRYPTO 2014, pages 297-314. Springer, 2014.

[10] Adriana L´opez-Alt, Eran Tromer, and Vinod Vaikuntanathan. On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In Proceedings of the forty-fourth annual ACM symposium on Theory of computing, pages 1219-1234. ACM, 2012.

[11] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game or a completeness theorem for protocols with honest majority. In STOC, pages 218-229, 1987.
[12] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In STOC, pages 1-10, 1988.
[13] D. Chaum, C. Cr´epeau, and I. Damg˚ard. Multiparty unconditionally secure protocols (extended abstract). In STOC, pages 11-19, 1988.
[14] A. C.-C. Yao. Protocols for secure computations (extended abstract). In FOCS, pages 160-164, 1982.
[15] Damien Stehlé, Steinfeld R . Making NTRU as secure as worst-case problems over ideal lattices[J]. 2011.
[16] Hoffstein J., Pipher J., Silverman J.H. (1998) NTRU: A ring-based public key cryptosystem. In: Buhler J.P. (eds) Algorithmic Number Theory. ANTS 1998. Lecture Notes in Computer Science, vol 1423. Springer, Berlin, Heidelberg.
[17] Yarkın Doröz, Y. Hu , and B. Sunar .Homomorphic AES evaluation using the modified LTV scheme. Kluwer Academic Publishers, 2016.
[18] Chongchitmate W, Ostrovsky R. Circuit-private multi-key FHE[C]//IACR International Workshop on Public Key Cryptography. Springer, Berlin, Heidelberg, 2017: 241-270.
[19] Michael Clear and Ciaran McGoldrick. Multi-identity and multi-key leveled FHE from learning with errors. In Advances in Cryptology - CRYPTO 2015, Proceedings, Part II, pages 630-656, 2015.
[20] Pratyay Mukherjee and Daniel Wichs. Two round multiparty computation via multi-key FHE. In Advances in Cryptology - EUROCRYPT 2016 , Proceedings, Part II, pages 735-763, 2016.
[21] Chris Peikert and Sina Shiehian. Multi-key FHE from lwe, revisited. In Theory of Cryptography - 14th International Conference, TCC 2016-B, Proceedings, Part II, pages 217-238, 2016.
[22] BRAKERSKI Z, PERLMAN R. Lattice-based fully dynamic multi-key FHE with short ciphertexts. In: Advances in Cryptology—CRYPTO 2016. Springer Berlin Heidelberg, 2016: 190–213.
[23] Chen L, Zhang Z, Wang X. Batched Multi-hop Multi-key FHE from Ring-LWE with Compact Ciphertext Extension, Theory of Cryptography Conference. Springer, Cham, 2017:597-627.
[24] Smart N P, Vercauteren F. Fully homomorphic SIMD operations[J]. Designs, codes and cryptography, 2014, 71(1): 57-81.
[25] Jagadeesh K A, Wu D J, Birgmeier J A, et al. Deriving genomic diagnoses without revealing patient genomes[J]. Science, 2017, 357(6352): 692-695.