Biotechnology: An Indian Journal

Research Article | Vol 18 Iss 3

Secure Genomic Data Processing by Homomorphic Encryption Techniques

Abinaya B*and Santhi S
Department of Biotechnology, Anna University, Chennai, Tamil Nadu, India

***Corresponding author:** Abinaya B, Department of Biotechnology, Anna University, Tamil Nadu, India, Tel: 9597913567; E-mail: abinayabtech01@gmail.com

Received date: 18-May-2022, Manuscript No. tsbt-22-64126; **Editor assigned date:** 20-May-2022, PreQC No. tsbt-22-64126; **Reviewed date:** 03-Jun-2022, QC No. tsbt-22-64126; **Revised date:** 18-Jul-2022, Manuscript No. tsbt-22-64126; **Published date:** 25-Jul-2022, DOI: 10.37532/0974-7435.22.18(3).001

Abstract

Background: Now a day, the processing of genomic data is growing tremendously and some concerns of outsourcing the genomic data are lack of security as it contains sensitive information about the individuals, computation time, and execution time is high due to the storage size.

Objective: Traditional cryptographic techniques require an unencrypted genomic dataset for computations. Here, there will be no control over the genomic data that is being processed in plaintext. To solve this, Homomorphic Encryption (HE) techniques are used for computation that takes place on encrypted data. This will give the same result as if it was processed in the plaintext.

Methods: In this paper, an encryption technique based on homomorphic encryption is used to secure genomic data for computation by comparing the techniques to achieve less execution time.

Results: The results obtained prove that BFV is more secure as it cannot be identified by intruders. It is fast when compared to other homomorphic encryption techniques.

Conclusion: The comparative study shows that BFV scheme is more efficient and gives quantum security.

***Keywords:** Data security; Genomic data processing; Partially homomorphic encryption; Fully homomorphic encryption*

Introduction

Genomic data processing is very sensitive as it reveals the individual's health information during the data sharing stage. It needs some privacy/security solutions using some cryptographic techniques. When the encryption method is applied to data during data sharing on the cloud using traditional cryptography, data has to be decrypted before using the data. A trusted third party may use the data illegally. Genome sequencing in speed led to an era in growth rate. The whole genome sequencing cost will reach $1 k in the future, which allows the individual to access large genome datasets on the internet. There are several popular projects like PGP and Hap Map that show genotypic information in public databases so that the information will be publicly available and accessible. Genome is used in a variety of applications that includes research purposes, the healthcare sector, and forensics. The data can be misused during process execution, leads to violating the data privacy. Even when explicit identifications like name, address, etc. are taken off from the dataset, one can easily find the identity, so data should be always considered with care [1-5].

However, there are various researches done to protect the genomic data using traditional cryptographic techniques. Specifically, homomorphic encryption is used, which allows processes to be carried out on the encrypted text so that no one can reveal the information to the third-party/researcher. In the below sections, BFV (Fully homomorphic encryption) is discussed in detail with approximate results [6-10]. The publicly available database is used for research purposes. Databases are not taken as plaintext and are encrypted for computation, as they may leak information. A few sequences are considered here because a small piece of DNA is enough to find the personality of an individual while sharing in the cloud. To achieve the proposed work, the BFV cryptosystem and its characteristics are given below [11-15].

Citation: Abinaya B, Santhi S. Secure Genomic Data Processing by Homomorphic Encryption Techniques**.** Biotech Ind J. 2022; 18 (3):001

- **Efficiency:** Homomorphic Encryption (BFV) is based on RLWE, which is more powerful than a hard solution designed against quantum computers attack.
- **Attacks resistance:** Material resources are considered and a very high number of parameters of RLWE offer high security.
- **Unpredictable:** Coefficients of random polynomial public and private keys are used for BFV cryptosystems to become unpredictable and cannot be based on an adversary.

Firstly, Brakerski Fan Vercauterian (BFV) was implemented and is compared with Partially Homomorphic Encryption (PHE). After analyzing the results, duplicate elements are encrypted several times, which leads to high computation time. To overcome the issue, a HaskMap structure is used with (key, value). The proposed method had many advantages; some of them are storage efficiency, high retrieval of data, and indexing of the genomic data position for query and both the encryption and decryption process. Additionally, BFV was compared to PHE, which performs all the possible operations on encrypted data so that plaintext is not revealed. PHE is not efficient during query processing and computation time. It is also not secure when compared to fully homomorphic encryption.

The step-by-step process of the contribution is as follows:

- Genomic data is considered here as an input. Dataset are available online publicly.
- In order to provide security to the outsourcing genomic data, data need to be encrypted using Homomorphic encryption techniques.
- Here, homomorphic encryption techniques are compared and a final solution on efficiency is concluded.

The execution time is calculated to check computation overhead and privacy concerns. Here, BFV and BGV are evident and securely violated by all partially homomorphic encryption techniques.

Materials and Methods

HE

The cryptosystem performs an operation on the encrypted data instead of the plaintext. All the computations are performed on encrypted data.

Let message space (M, 0) be finite. Let the parameter of security be σ. M is massage used in homomorphic encryption is quadruple of probabilistic (K, E, D, A) for polynomial operation have some functionalities, they are as follows

Generation of key value: Let 1σ be input for algorithm K gives output as key pair (ke, kd)=k ∈ K, ke, kd=k ∈ K, where K identifies key space.

Encrypting the message M: Input 1σ, ke and m ∈ M, E is encryption will give output as cipher text c ∈ C.c ∈ C, where C is an encrypted data space (cipher text).

Plaintext formation (decryption): Deterministic D is the decryption algorithm. 1σ, k and an element c ∈ C are taken as input, and its output is an element in M (message) for all m ∈ M it holds. If c=E(1σ, ke, m) then Prob(D(.k.c)≠m) is negligible, *i.e.*) it holds $\text{Prob}(D(.k.c)\neq m) \leq 2-\sigma \text{Prob}(D(.k.c)\neq m) \leq 2-\sigma$.

Property for homomorphic encryption scheme: Let A be an algorithm with input values 1σ, ke and elements c_1, $c_2 \in C$ and its output be $c_3 \in C$, for all m1, $m_2 \in M$ it holds, if $M_3=m_1.m_2 m_3=m_1.m_2$ and $C_1=E(1\sigma, ke, m_1)$ and $C_2=E(1\sigma,ke,m_2)$ then $\text{Prob}(D(A(1\sigma,ke,C_1,C_2)))\neq M_3$ is negligible. Homomorphic algorithm with the additional property gives an efficient algorithm to compute N encryption for two messages with a public key. If message M is additive, then the message is known as additively homomorphic encryption, where algorithm A is called Add. If message M is multiplicative, then it is called multiplicative homomorphic where algorithm A is called Mult. Homomorphic is efficient and crucial that the size of cipher text is polynomially bounded in σ (security) in repeated computation. HE is sometimes known as Partial HE (PHE), Somewhat HE (SWHE) and Fully HE (FHE).

PHE: It takes only one operation at a time either addition of two cipher texts or multiplication of two cipher texts. Some

of the encryption schemes are described below.

RSA: RSA is the best and widely known deterministic multiplicatively homomorphic encryption algorithm on M=(Z/NZ,.)M=(Z/NZ,.). Where N is product of two large primes.

Cipher text C=(Z/NZ,)C=(Z/NZ,)
Key K={(ke, kd)=((N, e), d) | N=pq, ed ≡1 mod φ(N)}
Encryption of message M, m∈M is given as Eke(m)= me mod N
Decryption of cipher text c,
Eke(m)=c ∈ C, need to be computed by Dke,
kd(c)=cd mod N=m mod N.
The encryption of the product of two messages can be efficiently computed by multiplying the corresponding cipher texts, *i.e.*). Eke ($m_1.m_2$)=($m_1.m_2$) e mod N=(m_1e mod n) (m_2e mod N)=Eke(m_1). Eke(m_2) where m_1, m_2∈M.
Therefore, algorithm for Mult can be easily realized as follows, Mult (Eke(m_1), Eke(m_2))=Eke(m_2).Eke(m_2)
Usually in RSA and other cryptosystems are faced with difficulty factoring the security parameter σ is the bit length of N. For instance, σ =1024 is a common security parameter.

Elliptic Curve Cryptography (ECC): It is a public key cryptography approach based on the algebraic structure of elliptic curves over finite fields. There are two types of finite fields where the elliptic curve is defined

- Binary field and prime fields. Prime fields Fp where p is a large prime number
- Binary fields F2 m.

An elliptic curve over finite field GF (p) is the set of points described by the equation: Ep (a,b):y2 =x3+ax+b mod p

Where 4a3 +27b2 ≠0, p is a prime number.
The security of ECC based cryptographic protocol is based on the Elliptic curve discrete logarithm problem (ECDLP). The ECDLP can be defined as the problem of finding the scalar k such that Q=kP given Q and P (generator point).

Goldwasser-Micali scheme: The Goldwasser-Micali scheme is an example of a probabilistic, additively homomorphic cryptosystem on M= (Z/2Z, +) M= (Z/2Z, +) with the cipher text space C=Z= (Z/NZ) C=Z= $(Z/NZ)^*$

Where N=pq N=pq is the product of two large primes.

K={(ke, kd)=((N, a), (p, q)) | N=pq, a ∈(Z/NZ)K=(ke, kd=N, a, p, q | N=pq, a ∈ $(Z/NZ)^*$:(ap)=(aq)=
−1}ap= aq= -1}
Since this scheme is probabilistic, the encryption algorithm gets as additional input a random value r ∈ Z. Let us define Eke (m,r)=amr2 mod NEkem, r=amr2 mod N and D(ke kd)=0D(ke kd)=0 if c is a square and=1 otherwise.

The following relation therefore holds good:
Eke (m_1, r_1).Eke (m_2, r_2)= Eke (m_1+m_2, r_1r_2) Eke (m_1, r_1). Eke (m_2, r_2)= Eke (m_1+m_2, r_1r_2) The algorithms Add can, therefore, be efficiently implemented as follows:
Add (Eke (m_1, r_1), Eke(m_2, r_2), r_3)= Eke (m_1, r_1).Eke (m_2, r_2).r_{23} mod N= Eke (m_1 + m_2, $r_1r_2r_3$) Add (Eke (m_1, r_1), Eke (m_2, r_2), r_3)= Eke (m_1, r_1).Eke (m_2, r_2).r_{32} mod N= Eke (m_1+ m_2, $r_1r_2r_3$) In the above equation, r_{23} mod Nr_{32} mod N is equivalent to Eke (0, r_3) Eke (0, r_3).

Also, m_1, m_2 ∈ Mm_1, m_2 ∈ M and r_1, r_2r_1, r_2,r_3 ∈ Z.r_3 ∈ Z.
Note that this algorithm should be probabilistic, since it obtains a random number r_3r_3 as an additional input.

Benaloh's scheme: Benaloh is a generalization of the GM scheme that enables one to manage inputs of l(k)l(k) bits, k being a prime satisfying some specified constraints.
Encryption is similar as in GM scheme (encrypting a message m ∈{0,….,k−1}m ∈{0,….,k-1} is tantamount to picking an integer r ∈ Z^*nr ∈ Zn^* and computing c=gmrk mod n),c=gmrk mod n. However, the decryption phase is more complex. If the input and output sizes are l(k)l(k) and l(n)l(n) bits respectively, the expansion is equal to l(n)/l(k)l(n)/l(k). The value of expansion obtained in this approach is less than that achieved in GM. This makes the scheme more attractive. Moreover, the encryption is not too expensive as well. The overhead in the decryption process is estimated to be O(k−−√.l(k))O(k.l(k)) for pre-computation which remains constant for each dynamic decryption step. This implies that the value of k has to be taken very small, which in turn limits the gain obtained on the value of expansion.

Okamoto-Uchiyama scheme: To improve the performance of the earlier schemes on homomorphic encryption, Okamoto and

Uchiyama changed the base group G (Okamoto and Uchiyama, 1998). By taking n= p^2qn= p^2q, p and q being two large prime numbers as usual, and the group G= $Z^*p^2G= Zp^{2*}$, the authors achieve k=pk=p. The value of the expansion obtained in the scheme is 3. One of the biggest advantages of this scheme is that its security is equivalent to the factorization of n. However, a chosen-cipher text attack has been proposed on this scheme that can break the factorization problem. Hence, currently it has a limited applicability. However, this scheme was used to design the EPOC systems, which is accepted in the IEEE standard specifications for public-key cryptography (IEEE P1363).

Paillier scheme: One of the most well-known homomorphic encryption schemes is due to Paillier. It is an improvement over the earlier schemes in the sense that it is able to decrease the value of expansion from 3 to 2. The scheme uses n=p.qn=p.q with gcd(n, ϕ(n))=1 gcdn, ϕn=1. As usual p and q are two large primes. However, it considered the group G= Z^*n2G= Zn2* and a proper choice of H led to k=l(n)k=l(n). While the cost of encryption is not too high, decryption needs one exponentiation modulo n2n2 to the power λ(n)λ(n), and a multiplication modulo n. This makes decryption a bit heavyweight process. The author has shown how to manage decryption efficiently using the famous Chinese Remainder Theorem. With smaller expansion and lower cost compared with the other schemes, this scheme found great acceptance. In 2002, Cramer and Shoup proposed a general approach to achieve higher security against adaptive chosen-cipher text attacks for certain cryptosystems with some particular algebraic properties. They applied their propositions on Paillier's original scheme and designed a stronger variant of homomorphic encryption. Bresson proposed a slightly different version of a homomorphic encryption scheme that is more accurate for some applications [17-18].

Key generation:
Step 1: n= pq, the RSA modulus
Step 2: λ=lcm (p−1, q−1)
Step 3: g ϵ Z /n2 Z s.t. n|or dn 2 (g)
Step 4: Public-key: (n, g), secret key: λ, μ 2.
Encryption of message m:
Step 1: m ϵ {0, 1... n−1}, a message
Step 2: h ϵR Z/n Z
Step 3: c=gm h n mod n_2, a cipher text 3.
Decryption of c: m=L (cλ mod n^2) L (gλ mod n^2)$^{-1}$ mod n
The constant parameter, L (gλ mod n^2)$^{-1}$ mod n or L (gα mod n^2)$^{-1}$ mod n where g=1+ n mod n^2 can also be recomputed once for all.

Fully homomorphic encryption: Fully Homomorphic Encryption (FHE) allows all the possible operations at a time. In 2009, Gentry described first plausible construction of a fully homomorphic cryptosystem that supports both addition and multiplication. Gentry's proposed FHE consists of various steps.

- It constructs a somewhat homomorphic encryption to evaluate low-degree polynomials on encrypted data.
- It describes decryption process so it can be defined as low-degree polynomial.
- Finally, bootstrapping is applied to form FHE scheme.

This scheme is too used to evaluate polynomials of high-degree using decryption that can be expressed in polynomial of low-degree. Once the polynomial degree is evaluated by the scheme that exceeds the degree of decryption polynomial by factor of two then the process is known as boots trappable and it can then be converted into a FHE. Gentry defined a secret key; even private key is in short basis of random ideal lattice. Generating the public and secret bases pair with right distributions for worst-case to average-case reduction is technically quite complicated. Significant research has been devoted to increase the efficiency of its implementation.

A parallel line of work that use lattices in cryptography are back to NTRU cryptosystem. This approach uses lattices for cryptographic derivations with efficiency. The structure of lattices is compared to ordinary lattices and found that it makes their representation more powerful and faster in computation motivated by the work in. A significant number of works has been proposed for efficiency constructions of several cryptographic primitives whose security can be reduced to hardness of vector problems in ideal lattices.

Lyubashevsky proposed Ring Learning With Errors (RLWE) assumption that is their ring counterpart of Regev Learning With Errors (LWE) assumption. In a nutshell, polynomial assumptions are over ring of the form (ai, aid+ei)

ai,ais+ei, where s is the random secret ring ai is distributed uniformly and randomly in ring and ea. is small ring, it is impossible for an adversary to find the sequence of samples from random pairs of ring elements. Authors have shown that assumptions are efficiently reduced to worst case for hardness of short vector problems on ideal lattices. They have also shown on how to construct an efficient ring counterpart to Regev's public key encryption technique, as well as counterpart to identify based encryption scheme presented in by using sampling techniques in. Scheme described in is very elegant and efficient as it is not dependent on any complex computations over ideal lattices.

Brakerski and Vaikuntanathan questioned that whether the approaches explained above can effectively exploit so benefits of both approaches can be achieved at the same time, namely functional powerfulness in one hand, and simplicity and efficiency in another hand. They have constructed a somewhat homomorphic encryption scheme based on RLWE. Somewhat homomorphic encryption scheme is constructed based on RLWE. The scheme inherits simplicity and efficiency as well as worst case relation to ideal lattices. Moreover, scheme is key dependent message security; it can securely encrypt polynomial function using its own secret key. Authors argue that all known constructions of fully homomorphic encryption used bootstrapping technique that enforces the public key of scheme to grow liberally with Kaushal depth of evaluated circuits. The drawback is usability and efficiency. However, size of public key can be made independent of circuits depth of the somewhat homomorphic encryption scheme can securely encrypt using secret key.

In order to cocaine the noise problem, Henry introduced FHE in 2009; bootstrapping mechanism is introduced to increase with each operation (addition or multiplication). Decryption only will face the impact of noise level exceeds the threshold value set during the process. Other patterns like BFV are developed by Brakershi, Gentry and Vaikuntanathan, followed by BFV. According to the consortium taking place in Europe, BFV is best method for practical homomorphic encryption.

They have committed the circular security with respect to the representation of secret key as a ring-based element where bootstrapping requires circular security with respect to bit-wise secret key. Since there is no prior work that studies a possible coexistence between somewhat homomorphic encryption with circular security.

Learning With Error over ring (RLWE): Generalization of Learning with Error is known as Ring Learning With Error (RLWE). RLWE is introduced in. It is the new homomorphic encryption algorithm that proves the efficiency of FHE. The polynomial tuples are described in (a_i(x), b_i(x)) where the coefficient of a_i is uniformly chosen from R_q and (x) $is\ calculated\ as$ (x) =(x) ·$s_i(x)$ + $e_i(x)\in$ R_q Where s $\in R_q$ is a secret polynomial.

Brakerski Fan Vercauteren (BFV): Brakerski Fan Vercauteren name comes from the founder's name Brakerski, Jun Feng Fan and Frederik Vercauteren. BFV is based on RLWE, introduced in 2012. It is a fully homomorphic encryption and has different phases and are discussed below. Only the addition operations are considered here and implemented without a bootstrapping step. The detailed description of the different phases is as follows,

Setting

Set of parameters considered are λ, n, q, t, σ.

Key generation: In key generation, secret key SK is represented by the polynomial and the coefficient of polynomials is drawn from $^{\chi}$key

$SK=s$

Public key PK is represented as a pair of polynomials (p_0,p_1),

$PK = (P_0, P_1) = ([-(a \cdot s + e)]_q, a)$

Where,

a $\leftarrow R_q \leftarrow \chi err$

Encryption Method: Message m is encoded for integer t (where t≥2) by calculating the cipher text c obtained from a pair of polynomials c0, c1 are as follows,

$c_0=[\Delta \cdot m + p_0 \cdot u + e_1] \in R_q$

$c_1=[p_1 \cdot u + e_2] \in R_q$

Where,

$\Delta=\lfloor q/t \rfloor$

$u \leftarrow \chi key;\ \ e1, e2 \leftarrow (\chi err)_2$

Every plaintext is represented in binary format so encoding takes place as a polynomial.

Decryption method: The decryption took place by obtaining the message m,

$c_0 + c_1 \cdot s = \Delta \cdot m + p0 \cdot u + e_1 + (p1 \cdot u + e_2) \cdot s$

$c_0 + c_1 \cdot s = \Delta \cdot m + (-(a \cdot s + e) + a \cdot s) \cdot u + e_1 + s \cdot e_2$

$$m = \left\lfloor (c_0 + c_1 \cdot s) \times \frac{t}{q} \right\rceil \in R_t$$

m is obtained by performing a scale and round. The important thing is BFV models scalar product.

BGV: Dealing with integer vectors (whose security is dependent on the hardness of decisional LWE (Learning with Errors) [19]) and dealing with the integer polynomials (whose security is dependent on the hardness of the decisional R-LWE (Ring LWE) [20]) are two versions of the cryptosystem. BGV is an asymmetric encryption scheme which can be used for the encryption of the bits.

Key generation: In key generation, secret key SK is represented by the polynomial and the coefficients of polynomials are drawn from χ key.

SK=s

Public key PK is represented as a pair of polynomials (p_0, p_1), PK=p_0, p1=-a·s+eq.t,a Encryption Method: Encrypt (Plaintext m, Public Key Pub): Cipher text c

Message m is encoded for integer t (where t≥2) by calculating the cipher text c obtained from a pair of polynomials c_0, c_1 are as follows,

$c_0=[m+p_0 \cdot u+t.e_1]qL$ $c_1=[p_1 \cdot u+t.e_2]qL$

Decryption method: Decrypt (Cipher text c, Private Key Priv): Plaintext m the decryption took place by obtaining the message m, $c_0+c_1 \cdot s = m+p_0 \cdot u+t.e_1+(p_1 \cdot u+t.e_2) \cdot s$

$c_0+c_1 \cdot s = m+(-(a \cdot s+t.e)+a \cdot s) \cdot u+t.e1+s \cdot e_2$ $m=[\lfloor (c_0+c_1 \cdot s)qL \rceil t$

m is obtained by performing a scale and round. The important thing is BFV models scalar products. Level shifting operations Rescale (Cipher text c): Cipher text c'

Homomorphic operations Add (Cipher text c_1, Cipher text c_2) Cipher text csum

Mu_l (Cipher text c_1, Cipher text c_2): Cipher text cmu_l (Tables 1-3)

TABLE 1. **Shows the time required for the key generation in partially homomorphic encryption.**

S No.	Dataset	ECC	Key Generation Time				
			Elgamal	RSA	Paillier	GM	OU
1	GCF_000001405.10	0.0714	30.21	64.932	0.0022	0.0807	3.2673
2	GCF_000001405.11	0.0999	141.17	60.8726	0.0019	0.0191	2.3342
3	GCF_000001405.12	0.123	140.62	69.784	0.0018	0.0384	4.0897
4	GCF_000001405.16	0.0602	70.644	132.768	0.0033	0.0881	3.2658
5	GCF_000001405.38	0.0379	40.802	108.308	0.0029	0.0132	2.6575
6	GCF_000001405.39	0.0353	12.446	133.119	0.0029	0.0182	4.064
7	GCF_000001405.40	0.457	160.52	37.502	0.0053	0.1936	3.0981

TABLE 2**. Shows the execution time of the encryption techniques of partially homomorphic encryption.**

S No.	Dataset	ECC	Elgamal	Encryption Time			
				RSA	Paillier	GM	OU
1	GCF_000001405.10	0.2143	0.1324	0.5213	9.87688	0.2104	0.3454
2	GCF_000001405.11	0.2763	0.089	1.2136	78.5646	0.0697	2.3435
3	GCF_000001405.12	0.5203	0.1298	1.1045	27.5278	0.113	1.654
4	GCF_000001405.16	0.1895	0.0454	0.4629	10.2748	0.083	0.4681
5	GCF_000001405.38	0.1452	0.0464	0.4496	19.3251	0.096	0.4813
6	GCF_000001405.39	0.1513	0.1621	0.9864	10.3673	0.0379	1.3984
7	GCF_000001405.40	4.5861	0.2129	0.4904	76.5479	0.451	2.6834

TABLE 3. **Shows the execution time of the decryption techniques of partially homomorphic encryption.**

S No.	Dataset	ECC	Elgamal	Decryption Time			
				RSA	Paillier	GM	OU
1	GCF_000001405.10	0.05247	0.1323	0.5213	9.8768	0.2104	0.9843
2	GCF_000001405.11	0.0493	0.089	1.2136	78.5646	0.0697	0.892
3	GCF_000001405.12	0.0999	0.1298	1.1045	27.5278	0.113	1.2671
4	GCF_000001405.16	0.0434	0.0454	0.4629	10.2748	0.083	1.8765
5	GCF_000001405.38	0.0488	0.0464	0.4496	19.3251	0.0964	0.9675
6	GCF_000001405.39	0.0542	0.1621	0.9864	10.3673	0.0379	0.7654
7	GCF_000001405.40	0.4358	0.2129	0.4904	76.5479	0.451	10,638

Results and Discussion

Implementations are carried out using Python on a Linux OS based Lenovo computer with 8 GB RAM memory using an Intel Core i%-6300U-2.5 GHZ CPU. The position focused is retrieved from fasta record with python libraries. Therefore, all the positions are coded in binary form and represented as the degree of polynomials given during the setting phase of BFV. In order to increase the speed of the execution time, 24 threads are used. Adding of two cipher texts for retrieving information about positions when it is queried for genomic data. For the BFV method, coefficients of the polynomials (e, e_1, e_2) have binary numbers as value. Security level of encryption is depending in λ as a security parameter fully based on values (n, q and σ). Considering n value as a large number gives high security as it is difficult to break by intruders but have high execution time and require hardware resources to achieve best results. No procedure to be carried out for the exact values to be considered for the parameters (Table 1). Shows the key generation time for partially homomorphic encryption in that Paillier's cryptosystem is efficient when compared to other techniques [21-23].

In Tables 2 and 3, the execution time taken by the genomic data for encryption and decryption of partially homomorphic encryption techniques are shown. Table describes that the Paillier's has efficient and secured technique. The Paillier's cryptosystem is then compared with the fully homomorphic encryption techniques shown in figures. Data encrypted using fully homomorphic encryption cannot be cracked even by quantum computers. Large sets of genomic data need to be considered for more comparison between the fully homomorphic encryption with considerable execution time in future for the effective outsourcing of data with high security guarantees. In fully homomorphic encryption techniques, BFV and BGV is compared and in (Table 4). Key generation of the genomic data is shown with results. The BFV has efficient time when compared to BGV.

The encryption process applied on the BGV and BFV are shown in (Table 5). The BFV scheme is efficient with less

execution time when compared to the BGV. The security of BFV is also more secured when compared to the BGV. In Table 6, decryption process time is analyzed and concluded that the BFV is more efficient than BGV. The integrity of the BFV is more secure and efficient than BGV (Figure 1). Shows the performance evaluation of the BFV and BGV. The BFV has very less execution time with high security. In Figure 2, the comparison diagram is explained. In that the BFV, BGV, ECC, Elgamal, RSA, GM and Paillier cryptosystem are compared and the results prove that the BFV of integer value as input is more efficient and accurate that other techniques. Much research is still needed for better performances without compromising safety and efficiency. Good efficiency and robustness of the proposed technique is demonstrated by comparing BFV scheme with other homomorphic encryption techniques. The finding has shown that execution time of BFV is faster than the other cryptosystem shown in (Figure 4). For n= 64, λ=64, Approximately, BFV is 4 times stronger than that of the PHE. Speed of operation is higher in BFV as it reduces the complexity by using the polynomials. Here, complex process and execution time of the systems infer that the BFV model is strongest against the attacks occurred by the intruders when compared to the PHE. BFV scheme for genomic positions or data gives the high security and throughput shown in Figure 3.

TABLE 4**. Shows the execution time of key generation.**

S No.	Dataset	Key Generation Time	
		BGV	FV
1	GCF_000001405.10	0.000292	0.000787
2	GCF_000001405.11	0.009717	0.000808
3	GCF_000001405.12	0.011869	0.000763
4	GCA_000001405.16	0.00044	0.091384
5	GCF_000001405.38	0.000386	0.00079
6	GCF_000001405.39	0.000274	0.000736
7	GCF_000001405.40	0.066838	0.0007

TABLE 5. **Shows the execution time of the encryption techniques of partially homomorphic encryption.**

S No.	Dataset	Encryption Time	
		BGV	FV
1	GCF_000001405.10	0.197006	0.4297
2	GCF_000001405.11	0.718092	0.788259
3	GCF_000001405.12	0.488513	0.715013
4	GCA_000001405.16	0.982025	0.000742
5	GCF_000001405.38	0.796891	0.612095
6	GCF_000001405.39	0.165044	0.392338
7	GCF_000001405.40	0.270102	0.660797

TABLE 6. **Shows the execution time of the decryption techniques of partially homomorphic encryption.**

S No.	Dataset	Decryption Time	
		BGV	FV
1	GCF_000001405.10	0.039033	0.015965
2	GCF_000001405.11	0.014338	0.003882
3	GCF_000001405.12	0.007928	0.005997
4	GCA_000001405.16	0.01216	0.034093
5	GCF_000001405.38	0.007729	0.02761
6	GCF_000001405.39	0.021367	0.003477

7	GCF_000001405.40	0.049878	0.021504

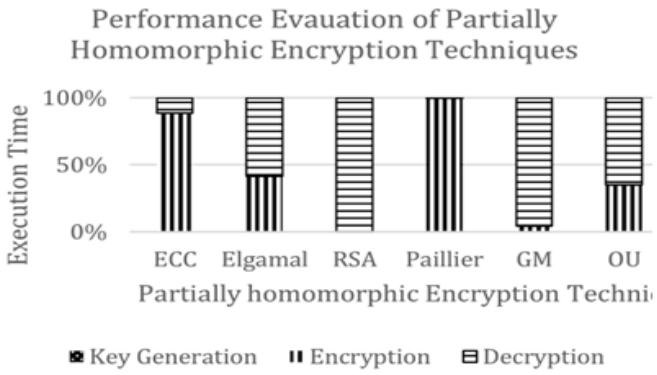

FIG.1. **Comparison results of partially homomorphic encryption techniques.**

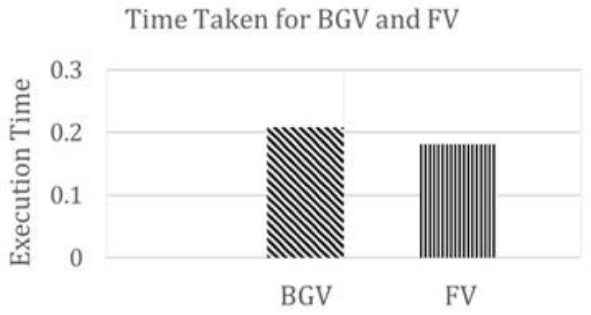

FIG.2. **Execution time of BGV and FV**.

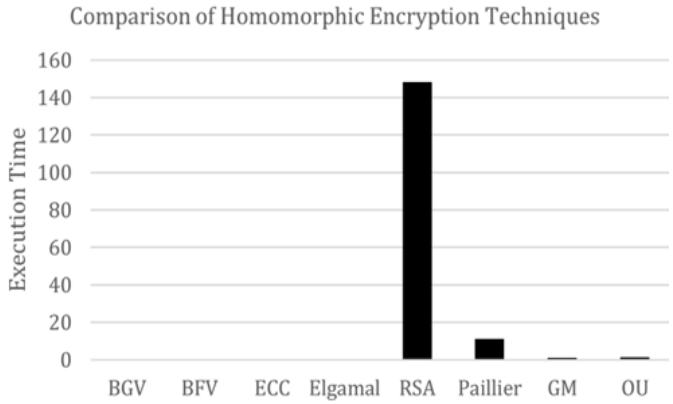

FIG.3. **Comparison chart of homomorphic encryption techniques.**

Conclusion

Asymmetric key cryptosystems are used for outsourcing the genomic data for the computation purposes. Along with public key cryptosystems, homomorphic encryption is applied to achieve efficiency, security/privacy, data availability and confidentiality. The execution time is faster in both encryption and decryption operations. Compared to PHE, the execution time for the BFV algorithm always remains large. Particularly by increasing the degree of the polynomial, the BFV algorithm

had less processing time when compared to other encryption techniques. The security of PHE is not ensured as it is more vulnerable to most of the attacks by intruders. Still the system needs improvements in finding prime numbers and large datasets. The cloud is not safe for storage and sharing due to third party issues. In future block chain is replaced instead of cloud storage.

Authors' contributions

Abinaya B collected the materials, prepared art works, edited, drafted and revised the manuscript. Santhi S has given the guidance for preparing the manuscript. Both the authors read and approved the final manuscript.

References

1. Abinaya B, Santhi S. A survey on genomic data by privacy-preserving techniques perspective". Computational Biology and Chemistry. 2021; 93:1476-9271.
2. Gentry C, Sahai A, Waters B, et al. "Homomorphic Encryption from Learning with Error: Conceptually-Simpler, Asymptotically-Faster, Attribute-Based". In: Proc. of the 33rd Annual Cryptology Conference, Santa Barbara, California. 2013;75-92.
3. Bonnoron G. "A journey Towards Practical Fully Homomorphic Encryption". L'Ecole National Supérieure Mines-Telecom Atlantique. 2019.
4. Gentry S, Halevi, NP Smart, et al. "Homomorphic Evaluation of the AES Circuit". In: Proc. of the 32nd Annual Cryptology Conference, Santa Barbara, California, USA. 2012; 850-867.
5. Fan J, Vercauteren F. "Somewhat Practical Fully Homomorphic Encryption". Cryptography, Darmstadt, Germany. 2012;1-16.
6. Bajard JC, Eynard J, Hasan MA, et al. "A Full RNS Variant of BFV like Somewhat Homomorphic Encryption Schemes". In: Proc. of the 24th International Conference on Selected Areas in Cryptography, Ottawa, Canada, pp. 423–442, 2017.
7. Coron JS, Lepoint T, Tibouchi M, et al. "Scale-Invariant Fully Homomorphic Encryption over the Integers". In: Proc. of the 17th International Conference on Practice and Theory in Public-Key Cryptography, Buenos Aires, Argentina, 2014; 311-328.
8. Bos JW, Lauter K, Loftus J, et al. "Improved Security for an Rring-Based Fully Homomorphic Encryption Scheme". In: Proc. of the 14th IMA International Conference on Cryptography and Coding, Oxford, UK. 2013;45–64.
9. Karmakar A, Roy SS, Reparaz O, et al. "Constant-Time Discrete Gaussian Sampling". IEEE Transactions on Computers. 2018;67:1561–1571.
10. Knuth DE, Yao AC. "The Complexity of Non-Uniform Random Number Generation". In: Proc. of A Symposium on New Directions and Recent Results in Algorithms and Complexity, Pittsburgh, Pennsylvania, USA. 1976;357–428.
11. Dijk M, Gentry C, Halevi S, et al. "Fully Homomorphic Encryption over the Integers". In: Proc. of the 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, French Riviera, France. 2010;24-43.
12. Meiser LC, Koch J, Antkowiak PL, et al. DNA synthesis for true random number generation. Nat Commun 11, 2020; 5869.
13. Dwarakanath NC, Galbraith SD. "Sampling from Discrete Gaussians for Lattice-Based Cryptography on A Constrained Device". Applicable Algebra in Engineering, Communications and Computing. 2014; 25:159–180.
14. Paillier P. Public-key cryptosystems based on composite degree residuosity classes. Advances in Cryptology-EUROCRYPT 99-of Lecture Notes in Computer Science.1999;1592:223–238.
15. Rivest RL, Shamir A, Adleman L, et al. A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM. 1978;21:120–126.
16. Goldwasser S, Micali S. Probabilistic encryption and how to play mental poker keeping secret all partial information. Proceedings of the 14th ACM Symposium on the Theory of Computing. 1982;365–377.
17. Roy SS, Vercauteren F, Vliegen J, et al. "Hardware Assisted Fully Homomorphic Function Evaluation and ENcrypted Search". IEEE Transactions on Computers. 2017;66:1562-1572.
18. ElGamal T. A public key cryptosystem and a signature scheme based on discrete logarithms. Advances in Cryptology-CRYPTO'84, LNCS. 1985;10–18.
19. Lyubashevsky V, Peikert C, Regev O, et al. "On Ideal Lattices and Learning with Errors over Rings". ACM Journals. 2013;60.
20. Lyubashevsky V, Peikert C, Regev O, et al. "On ideal lattices and learning with errors over rings". In Advances in Cryptology-EUROCRYPT 2010. French Riviera, France. 2010;1–23.

21. Zucca V. Representations de Nombres pour les Algorithmes Contexte, Sorbonne Université, Paris, 2017. Z Brakerski and V Vaikuntanathan, “Efficient Fully Homomorphic Encryption form (standard) LWE”, In: Proc. Of 52nd Annual IEEE Symposium on Foundations of Computer Science, NW Washington DC, USA. 2011;97-106.
22. Brakerski Z, Gentry C, Vaikuntanathan V, et al. “Fully Homomorphic Encryption without Bootstrapping”. In: Proc of the 3rd Innovations in theoretical Computer Science Conference, Cambridge Massachusetts.2012;309-325.
23. Z Brakerski. “Fully Homomorphic Encryption without Modulus Switching from Classical GapSVP”. In: Proc. of the 32nd Annual Cryptology Conference, Santa Barbara, California, U.S.A. 2012;868–886.