Towards Privacy-Preserving Computation on Gene: Construct Covid-19 Phylogenetic Tree using Homomorphic Encryption

Peiyi Han[1,2,3], Yunzhe Guo[2], Shaoming Duan[1], Chuanyi Liu*[1,2,3], and Yuxing Zhou[1]

[1]Department of computer science, Harbin Institute of Technology(Shenzhen), Shenzhen China
[2]Institute of Data Security, Harbin Institute of Technology(Shenzhen) and Qianxin, Shenzhen China
[3]Peng Cheng Laboratory, Shenzhen, China
{hanpeiyi, liuchuanyi}@hit.edu.cn, yunzhe.guo@outlook.com
shaomingduan@gmail.com, zhou978025302@alu.hit.edu.cn

***Abstract*—Constructing a phylogenetic tree is an important method to analyze the evolution of covid-19 virus. In the case of multiple entities holding different coronavirus gene data, it is a simple approach to aggregate all data to one entity and then calculate the phylogenetic tree. However,such method is difficult to carry out. Genetic data is very sensitive and has high economic value, it is usually impossible to directly copy between different entities, also the direct sharing of genetic data can also lead to data leaks or even legal problems.**

In this paper, we propose an homomorphic encryption based solution to tackle this problem, where two participants A and B both hold a part of covid-19 genetic data, and compute the gene distance matrix calculation of overall dataset without revealing the gene data held by both parties. After the computation,participant A can decrypt the final distance matrix from the encrypted result, then using plain-text result to construct the covid-19 phylogenetic tree. Experiment results show that the proposed method can process the genetic data accurately in a short time, and the phylogenetic tree generated by proposed solution has no loss on accuracy compared to plain-text calculation. In terms of engineering optimization, we propose an optimized encryption method, which can further shorten the encryption time of full dataset without reducing security level.

***Index Terms*—phylogenetic tree, homomorphic encryption, privacy-preserving computation**

I. Introduction

Nowadays, there are more than 10 million confirmed cases of covid-19 globally and still soaring with 300,000 new cases per day. The novel coronavirus is highly virulent, secretive, highly variability, and spreads rapidly. In the post-epidemic era, the analysis of the genome evolution of the novel coronavirus provides comprehensive and effective data support for assessing the risk of the epidemic, launching public health response measures, and formulating medical countermeasures. Constructing a phylogenetic tree of covid-19 virus genes is also an important method to analyze its evolution.

As a special form, data is easy to be redistributed or copied without authorization. Once sensitive data is leaked, various problems will arise, such as privacy leaks, legal issues, etc. As for medical data or genetic data, these data are very sensitive, and the impact of data leakage will be very serious, as a result, such data related to individuals are basically stored securely and will not be distributed to in an entity other than the data owner. Due to above reasons, data islands have been formed in the field of gene biology. Many institutions or entities have geneic data, but since they cannot directly share data with each other, it is difficult to mine the potential value on the overall data held by all entities.

In this paper, we use privacy-preserving computing to solve the problem of data islands in the genetic analysis, we consider a two-party scenario, and use homomorphic encryption. In the whole calculation process, we decomposed the gene distance matrix, clarified the scope of privacy protection calculation, and modified the gene distance comparison algorithm using homomorphic encryption. In the modified algorithm, neither of the two participants leak the data, nor can they infer the genetic data held by the other party through the intermediate results.

In algorithm selection, we use the paliiler homomorphic encryption scheme, which occupies less computing resources than other schemes, meanwhile, there is no accuracy loss in the entire calculation process, and the cipher-text calculation can obtain same result compared to the plain-text calculation. solved.

In the implementation, we used the python language in the entire operation process. We optimize the encryption process, we propose a cipher-text replacement method is used to speed up the encryption process of the full dataset, while ensuring that the security level is not degraded. In the algorithm modification, we used a special encoding method, participants can only calculate the correct distance of two gene sequence. Participants cannot infer whether there are missing bases in a specific position of the gene sequence nor can they infer the genetic data held by other participant.

In the experiment, we implement the encryption operation of the full dataset (L3) within 8 minutes on the Alibaba Cloud 4-core ECS . The cipher-text calculation of a gene sequence with a length of 20,000 is completed in 3 seconds, and the gene distance calculation in about 2 minutes. In

memory management, the variables involved in encryption will be persisted to the hard disk after the encryption operation is completed, encrypted results are read from the disk and then decrypted, the ciphertext in memory will be freed after decryption. In the execution of L3 dataset, the whole memory occupation is less than 400MB.

II. Related Work

A. Privacy-Preserving Computation

Privacy-Preserving computation, also called privacy computation or multi-party computation, is a is a subfield of cryptography. Privacy-Preserving computation allows data holders to perform specific computation of a function without leaking their own data.

At present, there are various technical routes to achieve privacy-Preserving computation. Researchers usually adopt different technical routes according to different computing purposes, such as using federated learning to achieve machine learning modeling, using homomorphic encryption to perform arithmetic operations, etc.

B. Homomorphic Encryption

Homomorphic encryption allows arithmetic operations to be performed directly on encrypted data, and the decryption result of the operation is the same as the result of the plaintext calculation. In this paper, we use additively homomorphic encryption scheme such as Paillier**??**, which is a public key cryptosystem. Using paillier, participants can use the public key for encryption operations, and decryption operations can only be done by the participant holding the private key. Additively homomorphic encryption provides addition between ciphertexts and multiplication between plaintext and ciphertext, these two operations can be done without leaving the encrypted space.

C. Federated Learning

Google first proposed the concept of federated learning based on the parameter server [1], [2], and implemented a federated learning system based on next word prediction in smartphones [3], which is generalized to horizontal federated learning; in this type of federated learning system, the participants have similar datasets and same data features, but the amount of data is insufficient. Participants use federated learning to achieve logical data aggregation and train a federated model. The performance of this federated model is stronger than the local model trained by participants using their own data. Federated learning platform still face attacks [4] to steal data privacy, and its defense work can be a future study.

Vertical federated learning was later proposed [5], [6]. In a vertical federated learning system, each participant usually belongs to different profession and has different types of data, only one participant has the label, which is called guest; other participants, called host, has more feature; guest can improve the performance of her machine learning model by introducing other features from host parties.

The purpose of federated learning is to use data from multiple parties to train machine learning models. In case of genetic analysis, calculations need to be performed according to specific rules, rather than training machine learning models. In the scenario of this paper, federated learning techniques are not applicable.

D. The Phylogenetic Tree Problem

Phylogenetic tree, also known as an evolutionary tree, is a tree that shows the evolutionary relationship between species that are thought to have a common ancestor,it is blongs to cladogram method. In the tree, each node represents the nearest common ancestor of its branches, and the length of the line segment between nodes represents the evolution distance.

In this paper, we ues neighbor-joining(NJ) method compute the phylogenetic tree [7]–[9] using COVID-19 virus data.

III. Prerequisites

This section introduces the preliminary knowledge of this paper. We first introduce the general description and the security model, then we present the basic flow of constructing a phylogenetic tree and indicate the detailed operations using privacy-preserving computation, finally the security model of this paper is given.

A. General Description

We briefly describe the participants in this paper, we consider the following scenario:

- There are two participants involved in the calculation of the phylogenetic tree, and each participant holds a part of the data;
- There isn't a trusted third participant in this scenario to assist the computation;
- We assume that two participants named A and B respectively, after the calculation is completed, the phylogenetic tree is generated by participant A;
- The public key and private key in the entire calculation are produced by participant A, the private key is always held by participant A, while public keys are distributed to participant B;
- In calculation, participant B locally uses the encrypted data of A and the data held by itself to perform operations (such as substraction) to obtain the encrypted result then send it back to A;

Therefore, in the above calculation process, the content sent by participant A to participant B and returned to participant A by participant B does not contain the gene plaintext data held by both participants, and neither participant can infer the plaintext gene sequence through the transmission of the content. The specific security discussion is presented in section VI.

B. Basic Flow of Constructing a Phylogenetic Tree

We describe the flow of constructing a phylogenetic tree when genetic data is held by single participant. The flow is mainly divided into 4 steps: data preparation, genetic data encryption, calculation of gene distance matrix based on

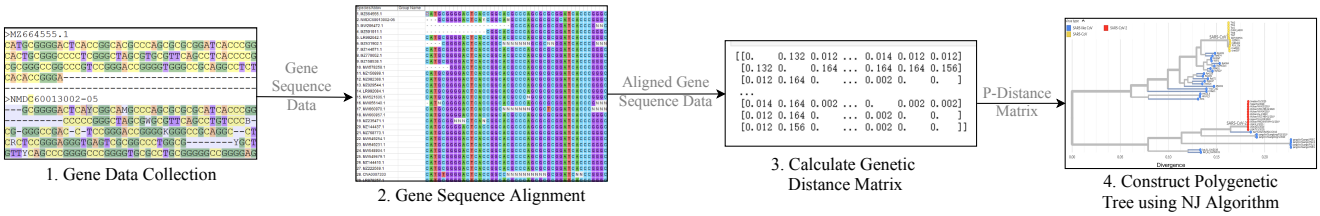

Fig. 1. Basic Flow of Constructing a Phylogenetic Tree.

encrypted data, calculation of phylogenetic tree, shown in Figure 1, each step will be described below.

In genetic data collection, each participant will collect its raw genetic data and organize the data into a unified fasta format [10]. A gene sequence is composed of multiple bases. Base is the basic units of DNA or RNA, and their constituent elements contain nitrogen, also known as "nitrogenous bases". In this paper, the covod-19 virus belongs to the RNA virus, and the bases constitute the RNA chain of the virus.

In genetic data alignment, according to the actual situation, each participant negotiate the length of the gene sequence after alignment, then start the alignment. We assume that all participants have enough data to complete the alignment and achieve optimal results. We leave the gene sequence alignment algorithm across participants using privacy-preserving computation as future work.

In genetic distance matrix calculation, when a single participant holds all the gene sequence data, the genetic distance matrix can be calculated directly using the gene sequence data. A genetic distance matrix is a two-dimensional array with the same width and height(square matrix), N pieces of genetic data can output a matrix of size N * N. The elements of the matrix represent the distances between genes, i.e. the elements in row i and column j represent the distances between gene i and gene j. The matrix calculated by the two participants in this paper are shown in Figure 3, we will describe the information in the figures in detail in Section refsection:decompose.

In phylogenetic tree construction, using distance matrix, the participant can construct the polyngenetic tree using the neighbor-joining method [7]. A phylogenetic tree [11] is a tree structure showing the evolutionary relationships among various species based on the similarities or distance of their genetic characteristics. In this paper, researchers can use the polyngenetic tree to explore the evolutionary path of the covid-19 virus for other biological studies.

C. Flow of Constructing a Phylogenetic Tree with Privacy-Preserving Computation

In this section, we describe the in the calculation process using privacy preserving computation and the operations that need to be changed.

In this paper, the most notable feature is that the data is distributed among two participants, and the participants can only tell the number of data and the ID name of the gene, but cannot reveal the base data of the gene sequence to other participants. For data collection and genetic data alignment, these operations do not change under multiple participants scenario because they do not need to interact with other participants. For the calculation of the gene distance matrix, since the content of the matrix needs to compare the gene sequence data owned by different participants, this part needs to be completed using privacy protection calculation, as shown in Figure 2. After participant A get the plain-text gene distance matrix, he/she can directly calculate the covid-19 gene phylogenetic tree. This process has no difference from the situation where single participant owns all gene sequence data.

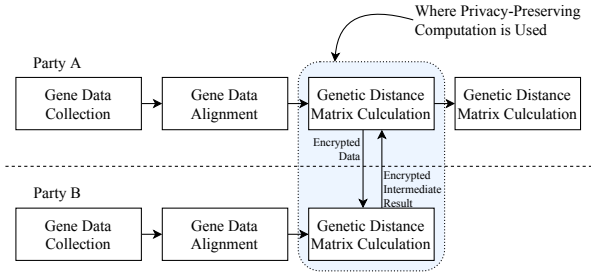

Fig. 2. In two-participant scenario, the steps that participants A and B need to perform and the scope of application of privacy calculations.

We focus on the calculation of the gene distance matrix, because the gene distance matrix needs to compare the distances of all genes, and the entire genetic dataset is held by 2 participants. We suppose participant A holds N gene sequences, and participant B holds M gene sequences. The length and width of the gene distance matrix is N+M, as shown in Figure 3. We can divide the entire gene distance matrix into 4 parts, of which matrix 1 in Figure 3 is an N*N matrix, and all the gene sequence data involved in the distance calculation come from participant A. Obviously, matrix 1 can be calculated by participant A using its own data. Similarly, matrix 4 is an M*M matrix, all the gene sequence data used in the distance calculation come from participant B, matrix 4 can be calculated by participant B using its own data. In gene distance data, the original data of the gene sequence will not be leaked or be inferred, so in the calculation process, participant B can directly send it to participant A for the gene phylogenetic tree calculation after the calculation of matrix 4 is completed.

The calculation of matrix 2 and matrix 3 in Figure 3 require the participation of privacy-preserving computation. Matrix 2 is a matrix with length M and width N, which contains

the gene distance information between N elements owned by participant A and M elements owned by participant B. It can be noted that matrix 3 can be obtained by transposing matrix 2, so in the privacy calculation process, two participants need to use privacy-preserving computation to calculate matrix 2. After participant A decrypts the plain text matrix 2. Matrix 3 can be generated without consuming extra time to compute matrix 3. The content of privacy-preserving computation using homomorphic encryption is described in detail in Section IV.

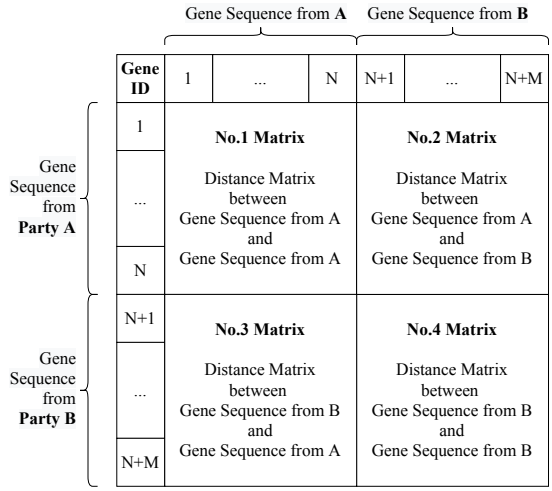

Fig. 3. Gene distance matrix decomposition: Matrix 1 and matrix 4 are calculated by the two participants using their own data, matrix 2 needs to be calculated using privacy-preserving computation, and matrix 3 can be obtained by transposing matrix 2.

D. Security Model

The proposed scheme follows the semi-honest adversary model, where both A and B perform the computation process according to the designed steps, but A and B may save the results of the intermediate state, and try infer the other participant's gene sequence data from intermediate results.

IV. Privacy-Preserving Computation Methodology

In this section, we first dive into the calculation of gene distance, and analyze several principles that gene distance calculation needs to meet in the multi-party context. Then we analyze these principles need to be satisfied, and give the calculation method of gene distance using homomorphic encryption, finally we summarize the calculation flow using privacy-preserving computation.

A. Dive into P-distance Calculation

We first dive into the logic of the gene comparison (calculation distance) function, as shown in the Algorithm 1.

Suppose there are two bases A and B. The distance calculation method that comes to mind is subtraction or division: if $A - B == 0$ or $A/B == 1$, then two bases are the same. Since the operation of subtraction is simpler than that of division, in this paper, we take subtraction as an example, and leave division to future work. There should also be a corresponding solution for division, the follow-up problems encountered by division may similar.

Algorithm 1 Calculate P-Distance between Gene Sequences

Input: Gene Sequence from participant A: seq_A; Gene Sequence from participant B: seq_B, which has same length with seq_A;
Output: Gene Distance $distance$;

let $distance$ = 0;
let snp = 0;
for each base index $i = 1$ to $len(seq\_A)$ **do**
  let $base\_A$ = seq_A[i];
  let $base\_B$ = seq_B[i];
  if $base\_A$!= $base\_B$ **and**
    $base\_A$ is not missing **and**
    $base\_A$ is not missing **then** $snp = snp + 1$
  else $snp = snp + 0$
  end if
end for
$distance = snp/len(seq\_A)$;
return Gene Distance: $distance$;

After analysising the p-distance calculation function described in Algortithm 1, we found that the comparison of gene sequences needs to meet the following three principles:

1) When both bases in two sequence are not missing, and the bases are the same, then the variable snp in Algorithm 1 remains unchanged ($snp = snp + 0$);
2) Both sequences are valid bases and the bases are different, then $snp = snp + 1$;
3) If any base position of the two sequences is missing, the snp remains unchanged ($snp = snp + 0$);

As a gene sequence, all bases appeared in form of characters, but as the input data of homomorphic encryption, it must be encoded as a number. We initially encode the gene as shown in Figure 4 in a common way. However, we found that if this common encoding is used, principle 1 described in this section can be fulfilled, while both principles 2 and 3 cannot be satisfied, this can lead to mistakes in the calculation of the snp value.

```
A = 1  G = 2  C = 3  T = 4
Y = 5  M = 6  R = 7  V = 8
W = 9  B = 10 K = 11 S = 12

N = 13
```

Fig. 4. Initial encoding of the gene bases, where N represents missing base.

B. Gene Base Encoding

For the 3 principles mentioned in section IV-A, the principles 2 and 3 cannot be solved directly by coding. In this section, we analyze the reasons and propose methods that can adapt to the two principles.

First, we need to carry out the next step analysis of the current gene comparison algorithm. For principle 2, we ideally should convert res to 1 when res != 0, and count it into the accumulation of snp, that is, as long as res!= 0, we should take snp += 1 operation.

Since in the current homomorphic encryption scheme, there is no comparison operation for ciphertext without a private key, we consider two solutions:

- In the first solution, we consider using multiple sigmoid functions to form a band-pass filter, as shown in Figure x, to convert the calculation result of res to 1 or 0. The sigmoid function can be Taylor expanded, and using fully homomorphic encryption to get the ciphertext of the Sigmoid result.
- In the second solution, participant A can decrypt the res in Algorithm 1. According to the result of base subtraction (the res), it is used to judge whether to accumulate snp in Algorithm 1.To ensure that the genetic data held by participant B is not leaked, after the base is subtracted, the result is shuffled, then sent to participant A. Participant A cannot infer the base infomation at each base position from shuffled result.

In this paper, we use the second scheme for gene distance calculation, and we set the encoding of missing N to 1015. For the first scheme, we leave it to future work.

C. Privacy-Preserving Computation using Homomorphic Encryption

This section presents the step 3 in Figure 2. Figure 5 shows the gene distance calculation method of chance homomorphic encryption. The input of proposed method is gene sequence data from two participants, which is the same as the algorithm 1. It is divided into 4 steps, namely data preparation, data encryption, genetic data calculation, result decryption and gene distance calculation.

In Step 3.1, two participants prepare genetic data for the following steps. Participant A generates the public key and private key required for the entire calculation process, the public key is distributed to Participant B, and the private key is always located in Participant A.

In Step 3.2, participant A encrypts a piece of genetic data with the public key of homomorphic encryption, and then transmits the encrypted data to Participant B. In Figure 5, the encrypted data is in square brackets, i.e. the cipher text of a_i is represented as $[a_i]$. For this step, we propose an optimized dataset encryption scheme to avoid directly encrypting all bases, thus saving execution time. Since the types of bases are limited, we first encrypt all bases, and each base encrypts cnt ciphertext bases (in homomorphic encryption, due to the existence of random variables, the same plaintext each The ciphertexts obtained from the second encryption are different), and randomly select one of the encrypted ciphertext sets for the base, and place its serial number in the encryption result to participate in the following step as the encrypted ciphertext.

In Step 3.3, participant B performs the actual distance calculation, and subtracts the genetic data held by himself

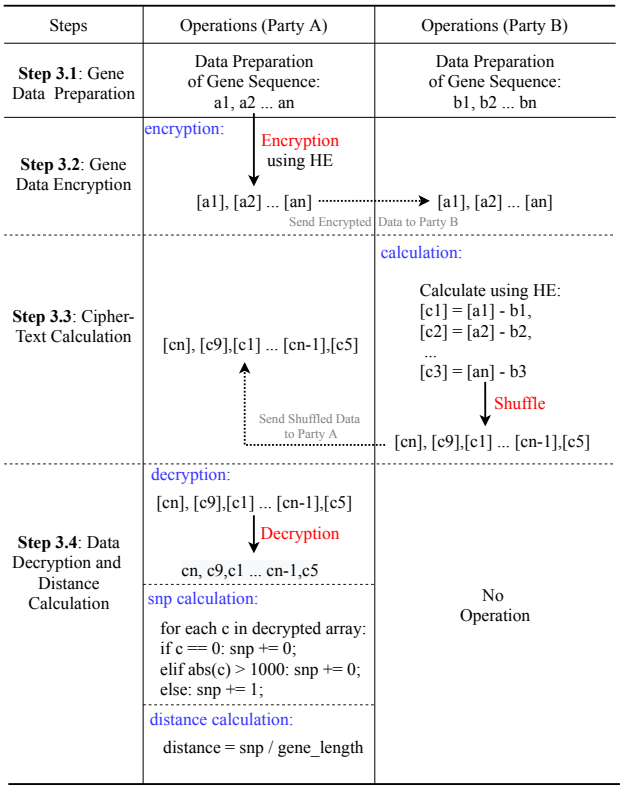

Fig. 5. Initial encoding of the gene bases, where N represents missing base.

from the encrypted data of Participant A for the base of each position. For a gene sequence, multiple subtractions are performed, and these results are randomly scrambled and transmitted back to participant A.

In Setp 3.4, participant A first needs to decrypt the calculation result of the ciphertext. After decryption, it needs to calculate the snp value according to the decryption result, and finally calculate the gene distance value. When the decrypted value is equal to 0, the snp does not accumulate; when its absolute value is greater than 1000, it means that one of the two parties contains missing bases and the other contains normal bases. At this time, according to the calculation rules in the algorithm x, The snp is also not accumulated; in other cases, both sides are normal bases, and the bases on both sides are not equal, the $snp + 1$ operation needs to be performed.

V. Experiments

In this section, we first introduce the used COVID-19 dataset, then introduce our implementation, and finally we present the experimental results and the analysis.

A. Experiment Setup

In this paper, we evaluate the performance of the proposed algorithm using three datasets with different volume. We use calculation accuracy, execution time(including dataset encryption, cipher-text calculation , decryption and gene distance calculation), memory usage as our indicator of model performance.

1) Gene Sequence Dataset: Table I shows the gene sequence datasets and their basic characteristics. The L1 dataset has 1000 gene sequence and each gene sequence contains 500 bases; the L2 dataset has 2000 gene sequence and each gene sequence contains 5000 bases; the L3 dataset has 20000 gene sequence and each gene sequence contains 20000 bases. All gene sequences of dataset L3 were aligned using MAFFT [12] to ensure the accuracy of sequence alignment, and then L1 and L2 datasets were cut out according to length requirements.

Dataset	# Size	# Gene Sequence Length
L1	1000	500
L2	2000	5000
L3	20000	20000

TABLE I
DATASET SPECIFICATION

2) Implementation: This paper use Biopython [13] for reading gene sequence data. Biopython is a python tools for computational molecular biology. It allows researchers to easily read and use various genetic data. We use phe [14] to implement partial homomorphic encryption operations. Phe is a python 3 library implementing the Paillier partial homomorphic encryption scheme [15], a is an additive homomorphic cryptosystem, we only use the addition operation between ciphertext and plaintext. In the whole calculation process, we only read one ciphertext gene sequence data (from participant A) and one plaintext gene sequence data (from participant B) from dataset each time, then persist the calculated ciphertext results to the file system , and sent to participant A by other programs to prevent occupying a lot of memory. Applying this method, even if the L3 dataset with the largest length(20000) is processed, the memory usage of the whole process is less than 400MB. We currently implement a single-threaded solution to test the actual performance of the modified algorithm. In real-world applications, multi-threading can further improve CPU resource usage and reduce program running time, which can be used as one of future engineering optimization.

B. Experiment Result

We use the 4-core ECS of Alibaba Cloud for performance test. The encryption operation refers to the time in seconds to complete the encryption of the full amount of data; the cipher-text calculation refers to the time to perform homomorphic subtraction of each gene sequence at the participant B, the distance calculation operation refers to the time to decrypt the subtraction result and calculate the genetic distance of each gene sequence.

The experiment result is shown in Table II, since the length of the gene sequences from L1 to L3 datasets increases, the calculation time consumed for L1 dataset is lower than that of the L2 and L3 datasets. We can also notice that in the calculation process, the distance calculation consumes a lot longer time than the cipher-text calculation, because the decryption operation of homomorphic encryption takes more amount of computation than the subtraction operation between the cipher-text and the plain-text.

Dataset	# Encryption	# Cipher-text Calculation (seconds / per sequence)	# Distance Calculation (seconds / per sequence)
L1	25s	0.0632s	5.42s
L2	30s	0.5649s	15.09
L3	7min23s	2.9411s	116.93s

TABLE II
EXPERIMENTS RESULT: CALCULATION TIME OF ENCRYPTION, ENCRYPT CALCULATION AND DISTANCE CALCULATION.

VI. DISCUSSION

In this section, we discuss the scheme proposed in this paper from two aspects, we first discuss the security of the scheme, and then we discuss what can be used for performance improvement.

A. Security

This section discusses security issues from three aspects. We first discuss the security of using subtraction, then we discuss whether coding schemes for the missing bases in the gene sequence may pose a security threat, finally we discuss the security of our data encryption schemes.

1) The Substraction Operation in Gene Distance Calculation: For subtraction operation, when base X and the subtraction result are known, base Y can be infered. While in this paper, the list of substraction result will be shuffled before sent back to participant A, the shuffle operation will not affect the calculation of snp, but make A unable to infer the gene sequence held by participant B. Therefore, such substraction will not cause data leakage.

If additional security measures are needed, a random number can be multiplied for each base subtraction result, so that the result of the base subtraction is not an integer, and the content cannot be inferred at all, and it will have a slight impact on the solution of the above principle 3 (When the multiplied random number is in the range of 3 to 10, the absolute value of the missing base should be higher than 1000 * 3 = 3000. This enhancement scheme is also implemented in our python code, but it is not enabled by default.

2) Encoding of Missing Base N: We use a different encoding range from the normal base to represent the missing base N, so as to ensure that the missing base is subtracted from the normal base, and its absolute value is higher than 1000, although the subtraction result between this result and the normal base is Obviously different, but it will not lead to the leakage of the data held by Party B, here are three reasons. First of all, B can only see the encrypted data of participant A's gene sequence, all sensitive information of participant A's gene sequence will not be leaked.

Secondly, when both bases in A and B's genetic sequence are missing (represented as N), or when the base of B is N, the gene compare result(which A can decrypted) should be

0. Notice that this result is the same as the subtraction result when both bases are non-missing bases.

Thirdly, when the base of B is not N and the base of A is N, participant A will get an absolute value greater than 1000 when decrypting. In this case, participant A can only infers that there is one base is missing in gene sequence, which is meaningless, since A knows the plaintext of his own gene sequence. Participant A can calculate the difference between the number of missing bases (the number of N) and the number of decrypted results greater than 1000. The difference is the number of bases where both A and B are N at the same time, but it cannot be used to infer any sensitive information. Since our security model is under a semi-honest assumption, participant A cannot maliciously change the data to detect B's sensitive information.

3) Encryption: In addition, due to the reduction of encryption operations, the encryption operation time of the full dataset can be greatly reduced. For each base, we generate each cnt different encrypted bases, and randomly select them during the replacement process. In the submission program, the each cnt variable is assigned a value of 42, that is, 15 bases correspond to 630 secret texts, and the situation of brute force cracking The down calculation amount is 630 ** 15, which meets the 128-bit security.

B. Potential Performance Improvements

There are several potential performance improvements in the current implementation.

Firstly, introducing multi-threading to make full use of CPU utilization. In Step 3.2, participant B can calculate the subtraction operation of multiple gene sequences at the same time. In Step 3.3, participant A can decrypt multiple calculation results at the same time, and perform necessary operations on snp variable in Algorithm 1.

Secondly, in addition to multi-threading, at present, we use Python language to realize homomorphic encryption operation. Changing to a more efficient programming language can also greatly reduce the time required for high computing and decryption.

VII. Summary and Conclusion

This is the first work using homomorphic encryption technology for sensitive data protection in the field of covid-19 phylogenetic tree construction. Homomorphic encryption can be used to calculate the genetic distance matrix of the two participants, and then use the decrypted gene distance matrix to construct a polylogenetic tree. We propose an optimization scheme for the encryption process of the dataset, which reduces the total execution time of the distance matrix calculation. The datasets used in this paper are aligned, how to perform privacy-preserving gene sequence alignment in the context of multiple participants is an interesting problem to consider.

References

[1] G. Inc. Federated learning: Collaborative machine learning without centralized training data. https://ai.googleblog.com/2017/04/federated-learning-collaborative.html.

[2] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konecný, S. Mazzocchi, H. B. McMahan, T. V. Overveldt, D. Petrou, D. Ramage, and J. Roselander, "Towards federated learning at scale: System design," *CoRR*, vol. abs/1902.01046, 2019. [Online]. Available: http://arxiv.org/abs/1902.01046

[3] S. Ramaswamy, R. Mathews, K. Rao, and F. Beaufays, "Federated learning for emoji prediction in a mobile keyboard," *CoRR*, vol. abs/1906.04329, 2019. [Online]. Available: http://arxiv.org/abs/1906.04329

[4] B. Hitaj, G. Ateniese, and F. Pérez-Cruz, "Deep models under the GAN: information leakage from collaborative deep learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*, B. M. Thuraisingham, D. Evans, T. Malkin, and D. Xu, Eds. ACM, 2017, pp. 603–618. [Online]. Available: https://doi.org/10.1145/3133956.3134012

[5] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 2, pp. 12:1–12:19, 2019. [Online]. Available: https://doi.org/10.1145/3298981

[6] S. Hardy, W. Henecka, H. Ivey-Law, R. Nock, G. Patrini, G. Smith, and B. Thorne, "Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption," *CoRR*, vol. abs/1711.10677, 2017. [Online]. Available: http://arxiv.org/abs/1711.10677

[7] N. Saitou and M. Nei, "The neighbor-joining method: a new method for reconstructing phylogenetic trees." *Molecular biology and evolution*, vol. 4, no. 4, pp. 406–425, 1987.

[8] N. Chen, B. Shi, and N. Wang, "Constructing neighbor-joining phylogenetic trees with reduced redundancy computation," in *12th IEEE International Conference on Electronics, Circuits, and Systems, ICECS 2005, Gammarth, Tunisia, December 11-14, 2005*. IEEE, 2005, pp. 1–4. [Online]. Available: https://doi.org/10.1109/ICECS.2005.4633525

[9] M. Guo, J. Li, and Y. Liu, "Improving the efficiency of p-ecr moves in evolutionary tree search methods based on maximum likelihood by neighbor joining," in *Proceeding of the Second International Multi-Symposium of Computer and Computational Sciences (IMSCCS 2007), August 13-15, 2007, The University of Iowa, Iowa City, Iowa, USA*. IEEE Computer Society, 2007, pp. 60–67. [Online]. Available: https://doi.org/10.1109/IMSCCS.2007.90

[10] "What is fasta format?" https://seq2fun.dcmb.med.umich.edu//FASTA/, accessed: 2022-05-10.

[11] J. Felsenstein and J. Felenstein, *Inferring phylogenies*. Sinauer associates Sunderland, MA, 2004, vol. 2.

[12] K. Katoh, K. Misawa, K. Kuma, and T. Miyata, "MAFFT: a novel method for rapid multiple sequence alignment based on fast Fourier transform," *Nucleic Acids Research*, vol. 30, no. 14, pp. 3059–3066, 07 2002. [Online]. Available: https://doi.org/10.1093/nar/gkf436

[13] P. J. A. Cock, T. Antao, J. T. Chang, B. A. Chapman, C. J. Cox, A. Dalke, I. Friedberg, T. Hamelryck, F. Kauff, B. Wilczynski, and M. J. L. de Hoon, "Biopython: freely available Python tools for computational molecular biology and bioinformatics," *Bioinformatics*, vol. 25, no. 11, pp. 1422–1423, 03 2009. [Online]. Available: https://doi.org/10.1093/bioinformatics/btp163

[14] C. Data61, "Python paillier library," https://github.com/data61/python-paillier, 2013.

[15] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Proceedings of the 17th International Conference on Theory and Application of Cryptographic Techniques*, ser. EUROCRYPT'99. Berlin, Heidelberg: Springer-Verlag, 1999, p. 223–238.