applied sciences

MDPI

Article

Securely Computing the Manhattan Distance under the Malicious Model and Its Applications

Xin Liu [1,2], Xiaomeng Liu [2], Ruiling Zhang [2], Dan Luo [1,*], Gang Xu [3,4,*] and Xiubo Chen [5]

[1] Department of Computer Science and Technology, Tianjin Ren'ai College, Tianjin 733299, China
[2] School of Information Engineering, Inner Mongolia University of Science and Technology, Baotou 014010, China
[3] School of Information Science and Technology, North China University of Technology, Beijing 100144, China
[4] Beijing Key Laboratory of Security and Privacy in Intelligent Transportation, Beijing Jiaotong University, Beijing 100044, China
[5] Information Security Center, State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China
* Correspondence: rdjeans@gmail.com (D.L.); gx@ncut.edu.cn (G.X.); Tel.: +86-13619961021 (D.L.); +86-13241634886 (G.X.)

Abstract: Manhattan distance is mainly used to calculate the total absolute wheelbase of two points in the standard coordinate system. The secure computation of Manhattan distance is a new geometric problem of secure multi-party computation. At present, the existing research secure computing protocols for Manhattan distance cannot resist the attack of malicious participants. In the real scene, the existence of malicious participants makes it necessary to study a solution that can resist malicious attacks. This paper first analyzes malicious attacks of the semi-honest model protocol of computing Manhattan distance and then designs an advanced protocol under the malicious model by using the Goldwasser–Micali encryption system and Paillier encryption algorithm, and utilizing some cryptographic tools such as the cut-choose method and zero-knowledge proof. Finally, the real/ideal model paradigm method is used to prove the security of the malicious model protocol. Compared with existing protocols, the experimental simulation shows that the proposed protocol can resist malicious participant attacks while maintaining high efficiency. It has practical value.

Keywords: secure multi-party computation; Manhattan distance; malicious model; cut-choose method; real/ideal model paradigm

Citation: Liu, X.; Liu, X.; Zhang, R.; Luo, D.; Xu, G.; Chen, X. Securely Computing the Manhattan Distance under the Malicious Model and Its Applications. *Appl. Sci.* **2022**, *12*, 11705. https://doi.org/10.3390/app122211705

Academic Editors: Howon Kim and Thi-Thu-Huong Le

Received: 24 October 2022
Accepted: 15 November 2022
Published: 17 November 2022

1. Introduction

With the innovation and development of new generation information technologies such as the blockchain, big data, and artificial intelligence, data information in various fields of society is constantly enriched and information sharing value is increasing. However, information privacy leakage is getting more and more attention. Secure multi-party computation (MPC) is one of the important tools to protect information data privacy and finish collaborative computing [1–5].

The earliest MPC problem is the Millionaire problem proposed by Professor Yao [6]. Goldreich and other scholars [7,8] are also involved in the research on MPC. There are many aspects of researching MPC, including secure computing applications [9,10], secure data mining [11,12], secure scientific computing [13–18], secure multi-party computational geometry problems [19,20], etc. MPC is widely used to solve many practical problems [21–24]. As tools in cryptography, homomorphic secret sharing (HSS) [25,26], oblivious transfer (OT) [27], and the cut-choose method [28] have become effective tools to combat malicious participants in MPC.

The MPC protocol of Manhattan distance is a typical problem of secure multi-party computational geometry problems [19,20,29–31]. In machine learning, Manhattan distance

is often necessary to compute a similarity measure across different samples when making a classification, which requires computing the "distance" across samples, and it is a practical method for distance calculation. In early computer graphics, screens were constructed from pixel dots whose coordinates were generally integers and it was costly to carry out float operations, but using Manhattan distance, which requires only addition and subtraction, can greatly improve the operation speed. Therefore, the study of Manhattan distance has extremely important theoretical value and practical significance.

At present, there is some research on the secure computing of Manhattan distance [29–31], and the existing protocols are designed in the semi-honest model. Consequently, it is particularly significant to study the MPC protocol of Manhattan distance to resist malicious attacks. In this paper, an MPC protocol for calculating the Manhattan distance is designed. The contributions are as follows:

1. Firstly, the protocol in Reference [29] is analyzed and found that some situations may be attacked by malicious participants.
2. According to the possible attacks of malicious participants, we design a new MPC protocol for computing the Manhattan distance can resist malicious attacks. In the process of designing the protocol, we use the Goldwasser–Micali and Paillier encryption algorithm, the cut-choose and zero-knowledge proof methods are used. The secure computation of the Manhattan distance will be converted into the Millionaires' problem to further improve the efficiency of the protocol.
3. The real/ideal model paradigm method is used to prove the security of the proposed malicious model MPC protocol. The performance and efficiency of the protocol are analyzed and simulated by experiments compared with existing protocols.

2. Related Work

Manhattan distance is a secure multi-party computational geometry problem. Manhattan distance can be seen everywhere in real life, which can be called the CityBlock distance or the taxi distance, that is, the actual distance as taxis pass from one crossroads to another. In a standard coordinate system, the sum of the absolute wheelbase of two points, which is the distance of two points in the north-south direction plus the distance in the east-west direction, which is the Manhattan distance between two points, can be noted as $d = |X_2 - X_1| + |Y_2 - Y_1|$. As shown in Figure 1.

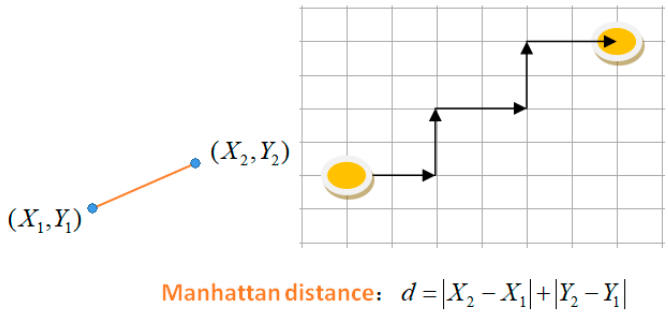

Figure 1. Manhattan distance.

In Fang [29], the authors designed a new coding method. With the help of homomorphism, the problem of calculating the Manhattan distance between two points is flexibly and skillfully transformed into calculating the Hamming distance of two vectors. This conversion idea not only improves the protocol efficiency but also prevents the disclosure of intermediate information. However, the protocol cannot resist malicious attacks.

Reference [30] invoked the absolute value of the difference and designed the MPC of Manhattan distance under different restrictions. The protocol's performance is poor.

Reference [31] designed a new graph encryption scheme for shortest distance queries based on a 2-hop cover labeling, which uses symmetric-key primitives. This scheme can

obtain the shortest distance between any two points in the graph, but it cannot resist malicious attacks.

In the above references, the MPC protocols of Manhattan distance is designed based on the semi-honest model, which cannot resist the attack of malicious participants. The protocol for secure computing Manhattan distance under the malicious model needs to be designed. The protocol proposed in reference [32] is a classic problem in the malicious model, but it is used to solve the Millionaire problem, which is different from the problem scenario we solve.

3. Preparatory Knowledge

3.1. Paillier Encryption Algorithm

Paillier proposed a probabilistic encryption algorithm [33] to solve the problem of composite residue classes. It has additive homomorphism.

Preparation: Select two large prime numbers p and q, satisfying $\gcd((pq,(p-1)(q-1)) = 1$, calculate $N = pq$, $\lambda(N) = lcm(p-1,q-1)$, $S_N = \left\{\mu < N^2 | \mu \equiv 1 \bmod N\right\}$, and defining function $L(\mu) = \frac{\mu-1}{N} (\forall \mu \in S_N)$. A random number $g \in Z_{N^2}^*$ is selected, with (N,g) as the public key, and λ as the private key.

Encryption: A random number $r < N$ is selected to perform the encryption operation $c = E(m) = g^m r^N \bmod N^2$.

Decryption: The private key λ is used for decryption to obtain the ciphertext: $m = \frac{L(c^\lambda \bmod N^2)}{L(g^\lambda \bmod N^2)} \bmod N$.

Addition homomorphism: $E(m_1)E(m_2) = g^{m_1} r^N \bmod N^2 \cdot g^{m_2} r^N \bmod N^2 = g^{m_1+m_2} r^N \bmod N^2 = E(m_1+m_2)$.

3.2. Goldwasser–Micali Encryption Algorithm

The Goldwasser–Micali (GM) encryption algorithm [34] has XOR homomorphism.

Preparation: Two large prime numbers p and q are selected to get $n = pq$. And $t \in Z_n^1$ (where Z_n^1 is a subset of Jacobi containing elements of Z_n^*) is selected as part of the public key. The private key of the algorithm is (p,q) and the public key is (n,t).

Encryption: For $m = m_1 m_2 \cdots m_s (m_i \in \{0,1\})$ in binary representation, the random number r is selected to encrypt the message m_i:

$$E(m_i) = t^{m_i} r_i^2 \bmod n = \begin{cases} t r_i^2 \bmod n, & m_i = 1 \\ r_i^2 \bmod n, & m_i = 0 \end{cases}.$$

Decryption: The private key (p,q) is used for decryption to obtain the ciphertext:

$$m_i = \begin{cases} 0, \ (\frac{E(m_i)}{p}) = (\frac{E(m_i)}{q}) = 1 \\ 1, \ (\frac{E(m_i)}{p}) = (\frac{E(m_i)}{q}) = -1 \end{cases}.$$

Among them, $(\frac{a}{p})$ is defined as follows:

$$(\frac{a}{p}) = \begin{cases} 1, p \text{ cannot divide } a, \ p \text{ is the sec ond residue of } a; \\ -1, \ p \text{ is not divisible by } a, p \text{ is a quadratic non-residue of } a; \\ 0, p \ \text{ can divide } a. \end{cases}$$

XOR homomorphism: $E(m_1) \cdot E(m_2) = E(m_1 \oplus m_2)$.

3.3. Zero-Knowledge Proof

The zero-knowledge proof means that in the process of interaction between the certifier and the verifier, when the certifier does not provide effective information, the verifier believes that the conclusion is correct through the interaction of both parties, then we say that the process has realized the zero-knowledge proof. In the process of interaction between the two sides, the information obtained by the verifier is only the right and wrong of the conclusion. For example, the zero-knowledge proof protocol $DLEQ(g_1,h_1,g_2,h_2)$ [35] is as follows, where $h_1 = {g_1}^\alpha$, $h_2 = {g_2}^\alpha$.

1. The certifier selects random numbers w and c, calculates $C = H(g_1^w, g_2^w)$, $r = w - \alpha \cdot c$, and finally publishes (r, c, C).
2. The verifier can verify whether $C = H(g_1^r h_1^c, g_2^r h_2^c)$ is established. If it is true, the verifier believes that the conclusion is correct, that is, the certifier knows the secret α.

3.4. Encoding

To study the MPC protocol of Manhattan distance, the encoding method is used to further simplify the research problem. The following methods are the coding rules and calculation principles of this protocol.

3.4.1. Encoding Rule

A universal set $U = \{u_1, \cdots, u_n\}$, $i \in \{1, 2, \cdots, n\}$, where $u_1 < u_2 < \cdots < u_n$. And the abscissa and ordinate of points $P(x_1, y_1)$ and $Q(x_2, y_2)$ are $\{x_1, x_2, y_1, y_2\} \in U$. Next, take $P(x_1, y_1)$ as an example to introduce the coding method.

Point P can be constructed as an array $A = (a_{11}, \cdots, a_{1k}, \cdots, a_{1n}, a_{21}, \ldots, a_{2l}, \cdots, a_{2n})$ according to the full set U, and its construction method is as follows: If $x_1 = u_k, k \in \{1, 2, \cdots, n\}$, set $a_{11} =, \cdots, = a_{1k} = 1$, $a_{1(k+1)} =, \cdots, = a_{1n} = 0$; if $y_1 = u_l$, $l \in \{1, 2, \cdots, n\}$, set $a_{21} =, \cdots, = a_{2l} = 1$, $a_{2(l+1)} =, \cdots, = a_{2n} = 0$.

Then $P(x_1, y_1)$ can be coded as $A(P) = (a_{11}, \ldots, a_{1k}, \cdots, a_{1n}, a_{21}, \ldots, a_{2l}, \cdots, a_{2n})$.

3.4.2. Calculation Principle

The universal set $U = \{u_1, \cdots, u_n\}$, Alice holds $P(x_1, y_1)$, Bob holds $Q(x_2, y_2)$, and $\{x_1, x_2, y_1, y_2\} \in U$.

P is coded as $A = \{a_{11}, \cdots, a_{1n}, a_{21}, \cdots, a_{2n}\}$; Q is coded as $B = (b_{11}, \cdots, b_{1n}, b_{21}, \cdots, b_{2n})$.

Then, the Manhattan distance between points P and Q can be calculated according to the following formula:

$$d(P, Q) = |x_1 - x_2| + |y_1 - y_2| = A \oplus B = \sum_{i=1}^{2} \sum_{j=1}^{n} (a_{ij} \oplus b_{ij}) \tag{1}$$

3.5. Security Definition under the Malicious Model

Under the malicious model, the widely accepted security definition is the real /ideal model paradigm method [32].

Definition 1. *For $\bar{B} = (B_1, B_2)$ in the ideal protocol, if $\bar{A} = (A_1, A_2)$ can be recognized in the actual protocol, so that:*

$$\left\{ IDEAL_{F, \bar{B}(z)}(x, y) \right\}_{x,y,z} \stackrel{c}{\equiv} \left\{ REAL_{\Pi, \bar{A}(z)}(x, y) \right\}_{x,y,z} \tag{2}$$

At this time, the protocol can be said to safely calculate the function F, where $\bar{A} = (A_1, A_2)$ and $\bar{B} = (B_1, B_2)$ are the probability polynomials constructed under the actual protocol and the ideal protocol respectively; x and y are the information owned by both parties; F is the function of executing $\{0,1\}^* \times \{0,1\}^* \to \{0,1\}^* \times \{0,1\}^*$; and z is the auxiliary input information. $IDEAL_{F, \bar{B}(z)}(x, y)$ refers to that in the ideal situation the participant uses strategy $\bar{B}$ to calculate $F(x, y)$ with the participation of auxiliary input information z. During the interaction between $A_1(x, z)$ and $A_2(y, z)$, the output result generated is recorded as $REAL_{\Pi, \bar{A}(z)}(x, y)$.

4. The MPC Protocol of Computing Manhattan Distance under the Semi-Honest Model

In Reference [29], the method of calculating the Manhattan distance between two points is converted into the Hamming distance between vectors. The specific MPC protocol is as follows (Algorithm 1):

Algorithm 1 Securely computing the Manhattan distance under the semi-honest model

Input: Alice owns point $P(x_1, y_1)$ and Bob owns point $Q(x_2, y_2)$.
Output: $|x_2 - x_1| + |y_2 - y_1|$.
Preparation: Alice and Bob construct the vectors $A = (P_{11}, \cdots, P_{1n}, P_{21}, \cdots, P_{2n})$ and $B = (Q_{11}, \cdots, Q_{1n}, Q_{21}, \cdots, Q_{2n})$ corresponding to point P and point Q respectively using the coding rules.

1. Alice gets the public-private key $pk(A)$ and $sk(A)$ of GM encryption algorithm and sends $pk(A)$ to Bob. Alice encrypts A to obtain $E(A) = (E(P_{11}), \cdots, E(P_{1n}), E(P_{21}), \cdots, E(P_{2n}))$, and sends $E(A)$ to Bob.
2. Bob encrypts vector B, calculates $R = (E(P_{11})E(Q_{11}), \cdots, E(P_{1n})E(Q_{1n}), E(P_{21})E(Q_{21}), \cdots, E(P_{2n})E(Q_{2n}))$, disturbs the order of elements in R to get $\hat{R}$, and sends it to Alice.
3. Alice decrypts $\hat{R}$ using the private key $sk(A)$, obtains $D(\hat{R}) = (d_{11}, \cdots, d_{2n})$, calculates $y = d_{11} + \cdots + d_{2n}$, and sends y to Bob.

The protocol ends.

This protocol is secure for Alice and Bob under the semi-honest model. However, if either Alice or Bob is malicious, the protocol will no longer be secure. Solutions need to be designed for possible malicious behavior.

5. The MPC Protocol of Computing Manhattan Distance under the Malicious Model

Ideas: This part first analyzes the possible malicious attacks of the semi-honest model protocol, designs the corresponding countermeasures to resist the malicious attacks, and finally makes the malicious participant unable to attack or be found (Note: the case where participants provided incorrect input cannot be considered, because this could not be avoided under the ideal model).

Possible malicious attacks in Algorithm 1 (as shown in Figure 2):

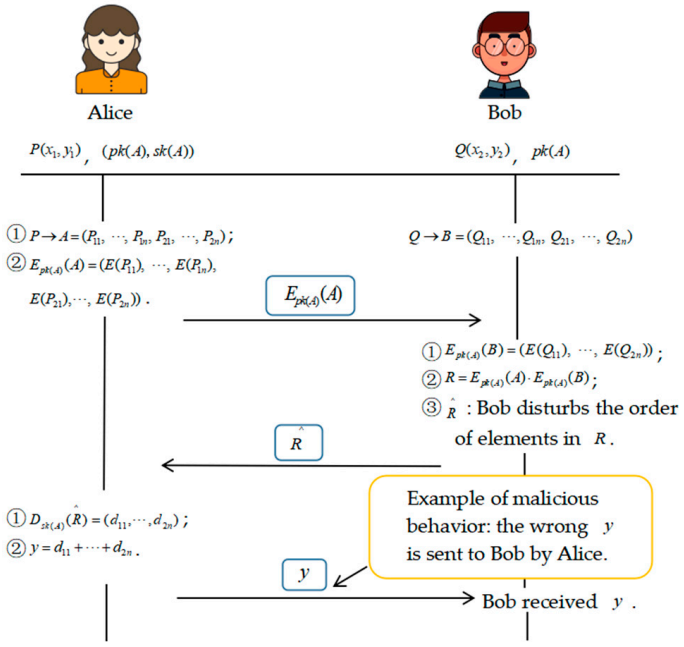

Figure 2. Example of malicious attacks in Algorithm 1.

1. In Step 3 of Algorithm 1, the result can only be calculated by Alice (Bob has no private key), which is unfair to Bob.
2. There may be a malicious attack in step 3, that is, Alice may tell Bob the wrong calculation result or terminate the protocol. In the end, Alice gets the right result, while Bob may get the wrong result or not.

To prevent the above attacks, the solution is to use the GM and Paillier encryption algorithm, utilizing the zero-knowledge proof and cut-choose method.

5.1. Specific Protocol

The specific protocol is as follows (Algorithm 2):

Algorithm 2 Securely computing the Manhattan distance under the malicious model

Input: Alice owns point $P(x_1, y_1)$ and Bob owns point $Q(x_2, y_2)$.
Output: $f_1(P,Q) = f_2(P,Q) = |x_1 - x_2| + |y_2 - y_1|$.
Preparation stage:

(a) Alice and Bob respectively generate the public and private keys $pk_{(A)}/sk_{(A)}$ and $pk_{(B)}/sk_{(B)}$ of the GM encryption system and send their public keys to each other. Alice generates the public key (g_a, N_a) and private key λ_a of Paillier encryption system and calculates $\mu = g_a^{\lambda_a} \bmod N_a^2$. Similarly, Bob generates (g_b, N_b), λ_b, and calculates $\nu = g_b^{\lambda_b} \bmod N_b^2$. Alice and Bob exchange (g_a, N_a, μ) and and (g_b, N_b, ν).
(b) Alice constructs the vector $A = (P_{11}, \cdots, P_{1n}, P_{21}, \cdots, P_{2n})$ of point P according to the coding rules.
(c) Bob constructs the vector $B = (Q_{11}, \cdots, Q_{1n}, Q_{21}, \cdots, Q_{2n})$ of point Q according to the coding rules.

Processing steps:

1. Alice encrypts vector A with $pk_{(A)}$ to get $E_{pk_{(A)}}(A) = (E(P_{11}), \cdots, E(P_{1n}), E(P_{21}), \cdots, E(P_{2n}))$, and sends $E_{pk_{(A)}}(A)$ to Bob.
2. Bob encrypts vector B with Alice's GM public key to obtain: $E_{pk_{(A)}}(B) = (E(Q_{11}), \cdots, E(Q_{1n}), E(Q_{21}), \cdots, E(Q_{2n}))$, calculates $E_{pk_{(A)}}(A) \cdot E_{pk_{(A)}}(B)$, and obtains the ciphertext: $E_{pk_{(A)}}(A \oplus B) = E_{pk_{(A)}}(A) \cdot E_{pk_{(A)}}(B)$ based on the GM encryption system. The $\hat{R}(E_{pk_{(A)}}(A \oplus B))$ is obtained by randomly disrupting the sequence of $E_{pk_{(A)}}(A \oplus B)$ and then sent to Alice.
3. Alice decrypts $\hat{R}(E_{pk_{(A)}}(A \oplus B))$ term by term using the GM private key to get $\hat{R}(A \oplus B) = (d_{11}, \cdots, d_{2n})$ and calculates $x = d_{11} + \cdots + d_{2n}$.
4. Bob encrypts vector B with $pk_{(B)}$ to get: $E_{pk_{(B)}}(B) = (E(Q_{11}), \cdots, E(Q_{1n}), E(Q_{21}), \cdots, E(Q_{2n}))$, and sends $E_{pk_{(B)}}$ to Alice.
5. Alice encrypts vector A with Bob's GM public key to obtain: $E_{pk_{(B)}}(A) = (E(P_{11}), \cdots, E(P_{1n}), E(P_{21}), \cdots, E(P_{2n}))$, calculates $E_{pk_{(B)}}(A) \cdot E_{pk_{(B)}}(B)$, and then obtains ciphertext: $E_{pk_{(B)}}(A \oplus B) = E_{pk_{(B)}}(A) \cdot E_{pk_{(B)}}(B)$. The $\hat{R}(E_{pk_{(B)}}(A \oplus B))$ is obtained by randomly disrupting the sequence of $E_{pk_{(B)}}(A \oplus B)$ and then sending it to Bob.
6. Bob decrypts $\hat{R}(E_{pk_{(B)}}(A \oplus B))$ term by term with his GM private key to obtain $\hat{R}(A \oplus B) = (e_{11}, \cdots, e_{2n})$ and calculates and calculates $y = e_{11} + \cdots + e_{2n}$.
7. Alice and Bob select m random numbers $a_i, b_i (i = 1, \cdots, m)$ respectively, calculate $(C_{1a}^i, C_{2a}^i) = (g_a^{a_i x} \bmod N_a^2, g_a^{a_i} \bmod N_a^2)$, $(C_{1b}^i, C_{2b}^i) = (g_b^{b_i y} \bmod N_b^2, g_b^{b_i} \bmod N_b^2)$, and publish (C_{1a}^i, C_{2a}^i) and (C_{1b}^i, C_{2b}^i) respectively.
8. Using the cut-choose method, Alice selects $m/2$ groups (C_{1b}^i, C_{2b}^i) from m groups (C_{1b}^i, C_{2b}^i). After Bob publishes $b_i y$, Alice verifies $g_b^{b_i y} \bmod N_b^2 = C_{1b}^i$. If the verification passes, she continues to execute the protocol, if not, terminates the protocol.
9. Bob selects $m/2$ groups (C_{1a}^i, C_{2a}^i) from m groups (C_{1a}^i, C_{2a}^i). After Bob publishes $a_i x$, Alice verifies $g_a^{a_i x} \bmod N_a^2 = C_{1a}^i$. If the verification passes, he continues to execute the protocol, otherwise terminate the protocol.
10. Alice and Bob randomly select one (C_{1a}^j, C_{2a}^j) and (C_{1b}^j, C_{2b}^j) from the remaining $m/2$ groups (C_{1a}^i, C_{2a}^i) and (C_{1b}^i, C_{2b}^i), and select $a \in Z_b^*$ and $b \in Z_a^*$.
11. Alice calculates $C_b = E_b(ab_j(x-y)) = (C_{2b}^j)^{ax} (C_{1b}^j)^{-a} r_1^{N_b} \bmod N_b^2 = g_b^{ab_j(x-y)} r_1^{N_b} \bmod N_b^2$ and sends it to Bob.
12. Bob calculates $C_a = E_a(a_i b(x-y)) = (C_{1a}^i)^b (C_{2a}^i)^{-by} r_2^{N_a} \bmod N_a^2 = g_a^{a_i b(x-y)} r_2^{N_a} \bmod N_a^2$ and sends it to Alice.
13. Alice uses λ_a to calculate $m_a = C_a^{\lambda_a} \bmod N_a^2$, Bob uses λ_b to calculate $m_b = C_b^{\lambda_b} \bmod N_b^2$, and publishes m_a and m_b respectively.
14. Both parties verify the correctness of the calculation with the help of the zero-knowledge proof. Alice proves $\log_{C_a} m_a = \log_{g_a} \mu$ and Bob proves $\log_{C_b} m_b = \log_{g_b} \nu$. If one of them fails to pass the proof, he or she is the malicious participant.
15. If the certificate is passed, Bob calculates $L(m_a)/L(\mu)$ to obtain $a_i b(x-y)$, and then obtains $a_i(x-y)$. When $a_i(x-y) = 0$, then $x = y$..
16. Alice calculates $L(m_b)/L(\nu)$ to obtain $ab_j(x-y)$, and then gets $b_j(x-y)$. When $b_j(x-y) = 0$, then $x = y$.
17. If $x = y$, then $f_1(P,Q) = f_2(P,Q) = x = y$. Alice and Bob get $f_1(P,Q)$ and $f_2(P,Q)$ respectively.

The Protocol ends.

5.2. Correctness Analysis

The following is a correctness analysis of Algorithm 2:

1. The main purpose of the first six steps in Algorithm 2 is that Alice and Bob calculate the Manhattan distance respectively. In this process, Alice and Bob decrypt using their own GM private key, so they cannot get each other's information. At the same time, in order to prevent the other party from obtaining its own information, the obtained ciphertext is randomly scrambled in steps 2 and 5 and then sent to the other party.
2. In 7–16 steps, the problem of calculating Manhattan distance has been skillfully transformed into the socialist millionaires' problem.
3. In step 13, Alice must calculate m_a correctly, otherwise, it cannot be proved by the zero-knowledge, that is, cheating is impossible. If $a_i x$ in the remaining $m/2$ group (C_{1a}^i, C_{2a}^i) also satisfies $a_i x < N_a/2$ and Bob selects $b < N_b/2$, Bob can calculate $F(x, y)$ after publishing m_a.
4. In this process, if Alice wants to successfully implement the malicious behavior, she can only select a a_i that does not meet the requirements, which is not found in the verification in step 8 and is selected by Bob in step 10, so Bob will not get the correct result. But Alice cannot get y, because $a_i b(x - y)$ is unsolvable (an equation has two unknown numbers).
5. If Alice uses the above method to cheat, the maximum success rate of deception is that in m group (C_{1a}^i, C_{2a}^i), $m - 1$ groups meet the requirements, and only one group does not meet the requirements, that is, the maximum probability is $1/m$. When the probability of success is the largest (i.e., one group does not meet the requirements), when the probability of success of $m = 10$ deception is $\frac{C_9^5}{C_{10}^5} \times \frac{1}{5} = \frac{1}{10}$ and five groups do not meet the requirements, the probability of success of deception is reduced to $\frac{C_5^5}{C_{10}^5} = \frac{1}{252}$. When $m = 50$, these two probabilities are reduced to 2×10^{-2} and 7.9×10^{-15} respectively. If the group greater than $1/2$ does not meet the requirements, the probability of successful deception will be reduced to 0 (it will always be found in the verification stage). Similarly, Bob's probability of successful malicious behavior is the same as Alice's. Therefore, the protocol is secure.

5.3. Example Description

Suppose Alice holds $P(5, 9)$, Bob holds $Q(8, 12)$, and the full set is $U = \{5, 6, 7, 8, 9, 10, 11, 12, 13, 14\}$. The output should be $f_1(P, Q) = f_2(P, Q) = |x_1 - x_2| + |y_2 - y_1| = 6$.

Preparation:

Alice and Bob respectively generate the public and private keys $pk_{(A)}/sk_{(A)}$ and $pk_{(B)}/sk_{(B)}$ of the GM encryption system and send their public keys to each other. Alice generates the public key (g_a, N_a) and private key λ_a of Paillier encryption system and calculates $\mu = g_a^{\lambda_a} \mathrm{mod} N_a^2$. Similarly, Bob generates (g_b, N_b), λ_b, and calculates $\nu = g_b^{\lambda_b} \mathrm{mod} N_b^2$. Alice and Bob exchange (g_a, N_a, μ) and (g_b, N_b, ν).

Convert points to vectors:

Alice constructs the vector A = (1,0,0,0,0,0,0,0,0,0, 1,1,1,1,1, 0,0,0,0,0) of point P according to the coding rules. Bob constructs the vector B = (1,1,1,1,0,0,0,0,0,0, 1,1,1,1,1,1,1,1,0,0) of point Q according to the coding rules.

Protocol Start:

1. Alice encrypts vector A with $pk(A)$ to get $E_{pk_{(A)}}(A) = (E(1),\ E(0),\ \cdots,\ E(0),\ E(1),\ E(1),\ \cdots,\ E(0))$, and sends $E_{pk_{(A)}}(A)$ to Bob.
2. Bob encrypts vector B with $pk(A)$ to obtain: $E_{pk_{(A)}}(B) = (E(1), E(1),\ \cdots, E(0),\ E(1), E(1), \cdots, E(0))$, and calculates the ciphertext $E_{pk_{(A)}}(A \oplus B) = E_{pk_{(A)}}(A) \cdot E_{pk_{(A)}}(B) = \{E(1) \oplus E(1),.\ \ E(0) \oplus E(1), \ldots, E(0) \oplus E(0),\ \ E(1) \oplus E(1), E(1) \oplus E(1), \ldots,\ E(0) \oplus E(0)\}$ based on the XOR homomorphism of GM encryption system. Bob randomly

disturbs the sequence of elements in $E_{pk_{(A)}}(A \oplus B)$ to get $\hat{R}(E_{pk_{(A)}}(A \oplus B)) = \{E(0) \oplus E(1), E(1) \oplus E(0), \ldots, E(0) \oplus E(0), \ldots, E(1) \oplus E(1)\}$ and sends it to Bob.

3. Alice decrypts $\hat{R}(E_{pk_{(A)}}(A \oplus B))$ term by term using $sk(A)$ to get $\hat{R}(A \oplus B) = \{1, 0, \ldots, 1\}$ and calculates: $x = 1 + 0 + \ldots + 1$.
4. Bob encrypts vector B with $pk(B)$ to get: $E_{pk_{(B)}}(B) = (E(1),\ E(1),\ \cdots,\ E(0),\ E(1),\ E(1),\ \cdots,\ E(0))$, and sends $E_{pk_{(B)}}(B)$ to Alice.
5. Alice encrypts vector A with $pk(B)$ to obtain: $E_{pk_{(A)}}(A) = (E(1),\ E(0),\ \cdots,\ E(0),\ E(1),\ E(1), \cdots,\ E(0))$, and calculates the ciphertext $E_{pk_{(B)}}(A \oplus B) = E_{pk_{(B)}}(A) \cdot E_{pk_{(B)}}(B) = \{E(1) \oplus E(1),.\ E(0) \oplus E(1), \ldots, E(0) \oplus E(0),\ E(1) \oplus E(1), E(1) \oplus E(1), \ldots, E(0) \oplus E(0)\}$. Alice randomly disturbs the sequence of elements in $E_{pk_{(B)}}(A \oplus B)$ to get $\hat{R}(E_{pk_{(B)}}(A \oplus B)) = \{E(1) \oplus E(0), E(0) \oplus E(0),.\ \ldots, E(1) \oplus E(1), \ldots, E(0) \oplus E(1)\}$ and then sent it to Bob.
6. Bob decrypts $\hat{R}(E_{pk_{(B)}}(A \oplus B))$ term by term with $sk(B)$ to obtain $\hat{R}(A \oplus B) = \{0, 0, \ldots, 1\}$ and calculates $y = 0 + 0 + \ldots + 1$.
7. In steps 7–12, Alice and Bob follow Algorithm 2, which will not be described in detail.

In Algorithm 2, Alice and Bob have the same operations, and the status of both parties is fair. Assume Alice is a malicious participant to explain. Alice's possible malicious behaviors include: using the wrong x to calculate C_b in step 11, using the wrong λ_a to calculate m_a in step 13, etc. Then Alice cannot prove through the zero-knowledge proof in step 14. Bob knows that Alice is a malicious participant, and Algorithm 2 is terminated. If Algorithm 2 is not finished by step 15 and $x = y$ is obtained, it can be proved that both parties have implemented the protocol in a semi-honest way, and Bob will get the correct conclusion that $f_2(P, Q) = y = 6$ in step 17.

5.4. Security Proof

This paper uses the real/ideal model paradigm method to prove the security.

In steps 3 and 6 of Algorithm 2, Alice and Bob get x and y respectively. We can take x and y as the input data for further execution of Algorithm 2 in the first six steps. Executing the protocol with the false x and y is equivalent to providing the false input, which is unavoidable under an ideal protocol, so it is not considered. Therefore, only steps 7–17 need to be proven secure.

Theorem 1. *Algorithm 2 (mentioned as* Π*) is secure.*

Proof of Theorem. The Algorithm Π is feasible only if at least one of the two parties in the protocol is not a malicious participant, that is, there are two cases that need to be proven secure.

(Case I) A_1 is honest, A_2 is dishonest. In this situation, A_1 and A_2 execute Π, then:

$$REAL_{\Pi,\bar{A}}(x, y) = \left\{F(x, A_2(y), A_2(C_{1a}^i, C_{2a}^i), m_a, S\right\} \tag{3}$$

where S is the sequence message received by A_2 through the zero-knowledge proof.

A_1 will execute the protocol honestly, and B_1 is determined. B_2 of the ideal model is indistinguishable from A_2 of the actual protocol needs to be proven. The output of $\bar{B} = (B_1, B_2)$ is indistinguishable from $REAL_{\Pi,\bar{A}(x,y)}$ in the actual model (note: A_2 is the actual executor of Π. Therefore, when proving, the security of Π needs to be verified according to A_2's behavior, that is $A_2(y)$).

1. A_1 in the actual protocol is honest, so B_1 will send right x to TTP. According to dishonest A_2's strategy, B_2 decides what information to send to TTP. Consequently, the input of B_2 is $A_2(y)$.

2. TTP obtains the $(x, A_2(y))$ and calculates $F(x, A_2(y))$.
3. B_2 dresses up as A_1 and executes Π with A_2, that is, B_2 selects x' and makes $F(x', A_2(y)) = F(x, A_2(y))$.

   ① B_2 and A_2 execute Π and send the information $\left\{A_2(C_{1a}^{i'}, C_{2a}^{i'})\right\}$ to A_2 in step 7 based on Algorithm 2.
   ② In step 8, B_2 verifies the information he asked A_2 to publish.
   ③ In step 9, B_2 publishes the information required by A_2.
   ④ In steps 10–12, B_2 selects, calculates, and publishes the information according to Π.
   ⑤ In step 13, m_a' is calculated and published.
   ⑥ In step 14, the information sequence S' is obtained.

   In the process of Π, the following can be got:

$$IDEAL_{F,\bar{B}}(x, y) = \left\{F(x, A_2(y)), A_2(C_{1a}^{i'}, C_{2a}^{i'}), m_a', S'\right\} \tag{4}$$

In steps 7–14, Π adopts the same encryption algorithm, $(C_{1a}^{i}, C_{2a}^{i}) \stackrel{c}{\equiv} (C_{1a}^{i'}, C_{2a}^{i'})$, $m_a' \stackrel{c}{\equiv} m_a$, and the zero-knowledge proof ensures $S' \stackrel{c}{\equiv} S$, so:

$$\left\{IDEAL_{F,\bar{B}}(x, y)\right\} \stackrel{c}{\equiv} \left\{REAL_{\Pi,\bar{A}}(x, y)\right\} \tag{5}$$

(Case II) A_1 is dishonest, A_2 is honest. There are two cases:

1. Alice gets the result and ignores TTP, then $\perp$ is sent to Bob by TTP, so:

$$REAL_{\Pi,\bar{A}}(x, y) = \left\{A_1(C_{1b}^{i} C_{2b}^{i}), m_b, S, \perp\right\} \tag{6}$$

2. Conversely, $f_2(x, y)$ is sent to Bob by TTP, then:

$$REAL_{\Pi,\bar{A}}(x, y) = \left\{A_1(C_{1b}^{i} C_{2b}^{i}), m_b, S, F(A_1(x), y)\right\} \tag{7}$$

where S is the sequence message received by A_1.

A_2 is honest, B_2 is determined. B_1 of the ideal model is indistinguishable from A_1 of the actual protocol needs to be proven. That is, the output of strategy pair $\bar{B} = (B_1, B_2)$ is indistinguishable from $REAL_{\Pi,\bar{A}(x,y)}$ needs to be proven (note: A_1 is the actual executor of Π. Therefore, when proving, the security of Π needs to be verified according to A_1's behavior, that is $A_1(x)$).

1. According to dishonest A_1's strategy, B_1 decides what information to send to TTP. Therefore, the input value B_1 sends to TTP is $A_1(x)$.
2. $(A_1(x), y)$ is obtained by TTP to calculate $F(A_1(x), y)$.
3. B_1 executes Π with A_1 by dressing up as A_2, that is, B_1 selects y' simulation protocol and makes $F(A_1(x), y') = F(A_1(x), y)$.

   ① B_1 and A_1 execute the protocol and send $\left\{A_1(C_{1b}^{i'}, C_{2b}^{i'})\right\}$ in step 7 to A_1.
   ② In step 9, B_1 verifies the information he asked A_1 to publish in step 8.
   ③ In steps 10–12, B_1 selects, calculates, and publishes the information according to Π.
   ④ In step 13, B_1 calculates m_b' and publishes it.
   ⑤ In step 14, the information sequence S' is obtained.

   In the process of the protocol, there are two situations:

1. A_1 receives the message and ignores TTP, so:

$$IDEAL_{F,\bar{B}}(x, y) = \left\{A_1(C_{1b}^{i'}, C_{2b}^{i'}), m_b', S', \perp\right\} \tag{8}$$

2. Conversely, that is:

$$IDEAL_{F,\overline{B}}(x,y) = \left\{ A_1(C_{1b}^{i'}, C_{2b}^{i'}), m_b', S', F(A_1(x), y') \right\} \tag{9}$$

In steps 7–14, Π uses the same encryption algorithm, so $(C_{1b}^{i}, C_{2b}^{i}) \overset{c}{\equiv} (C_{1b}^{i'}, C_{2b}^{i'})$, $m_b' \overset{c}{\equiv} m_b$, and $S' \overset{c}{\equiv} S$ ensured by the zero-knowledge proof, then:

$$\left\{ IDEAL_{F,\overline{B}}(x,y) \right\} \overset{c}{\equiv} \left\{ REAL_{\Pi,\overline{A}}(x,y) \right\} \tag{10}$$

Combining the above two cases it is proved that the output of the strategy to $\overline{B} = (B_1, B_2)$ in the ideal model is indistinguishable from $REAL_{\Pi,\overline{A}(x,y)}$ in the actual model, which meets Definition 1. Therefore, Algorithm 2 is secure. □

6. Performance Analysis

6.1. Overall Performance Comparison

This paper analyzes the performance of Algorithm 2 and protocols in other references [29–31], as shown in Table 1.

Table 1. Protocol's performance analysis.

Protocol	Fairness	Cryptography Tools	Resist Malicious Attacks
Fang [29]	unfair	GM	×
Dou [30]	unfair	Paillier	×
Liu [31]	unfair	Symmetric cryptography	×
Algorithm 2	fair	GM, Paillier	√

6.2. Computational Complexity

In Fang [29], the protocol adopts the GM encryption system. One-time encryption needs two modular multiplication operations, and one-time decryption needs $\log N$ modular multiplication operations. Alice's encryption and decryption need $2n$ times. Bob's encryption needs $2n$ times and the multiplication calculation is $2n$ times. Therefore, a total of $10n + 2n \log N$ modular multiplication operations are carried out.

In Dou [30], its protocol adopts the Paillier encryption algorithm. For the Paillier algorithm, one-time encryption or decryption needs $\log N$ modular multiplications. Alice's encryption and decryption times are $2mn$ and 1 respectively; Bob's encryption times are 1. Therefore, the protocol performs a total of $(2mn + 2) \log N$ modular multiplication operations.

The protocol proposed by Liu [31] uses the 2HCL and symmetric-key primitives, and its computational complexity cannot be expressed by modular multiplication. In reference [31], m is used to represent the length of the label in the 2HCL index and n is denoted as the number of vertices in the graph, and the computational complexity of the protocol is $O(mn)$.

In Algorithm 2 of this paper, the GM encryption system is used in steps 1–6. Alice and Bob perform encryption, decryption, and modular multiplication operations $4n$, $2n$, and $2n$ times respectively. Steps 7–14 transformed the judgment conditions into the socialist Millionaires' problem and carried out $10m \log N + 2$ times of modular multiplication. Therefore, the protocol performs a total of $[20n + (4n + 10m) \log N + 2]$ modular multiplication operations.

6.3. Communication Complexity

In Fang [29], the two parties conducted two rounds of communication. In Dou [30], the two parties conducted three rounds of communication. In Liu [31], the number of communication rounds between the client-side and the server-side are three rounds. In Algorithm 2, four rounds of communication are carried out. Table 2 shows the specific comparison.

Table 2. Efficiency comparison (based on modular multiplication).

Protocol	Computational Complexity	Rounds of Communication	Resist Malicious Attacks
Fang [29]	$10n + 2n \log N$	2	No
Dou [30]	$(2mn + 2) \log N$	3	No
Liu [31]	$O(mn)$	3	No
Algorithm 2	$20n + (4n + 10m) \log N + 2$	4	Yes

Note: the computational complexity of most MPC protocols is relatively high, due to the use of some cryptographic tools, such as the cut-choose method and zero-knowledge proof. These calculations do not reveal private data and can therefore be outsourced to improve the efficiency of the protocol.

6.4. Experimental Simulation

We use Python language to simulate Algorithm 2 and references [29,30], and the consistency of the universal set's potential and input information is maintained. For the above three protocols, 1000 experiments were performed based on different data lengths, and the average value of 10 execution times was randomly taken.

As shown in Figure 3, the computational complexity of reference [29] is slightly lower than that of Algorithm 2. In Dou [30], the proposed protocol is designed based on invoking the MPC protocol that takes the absolute value of the difference between two numbers, which greatly increases the amount of calculation, so its computational complexity is higher than Algorithm 2.

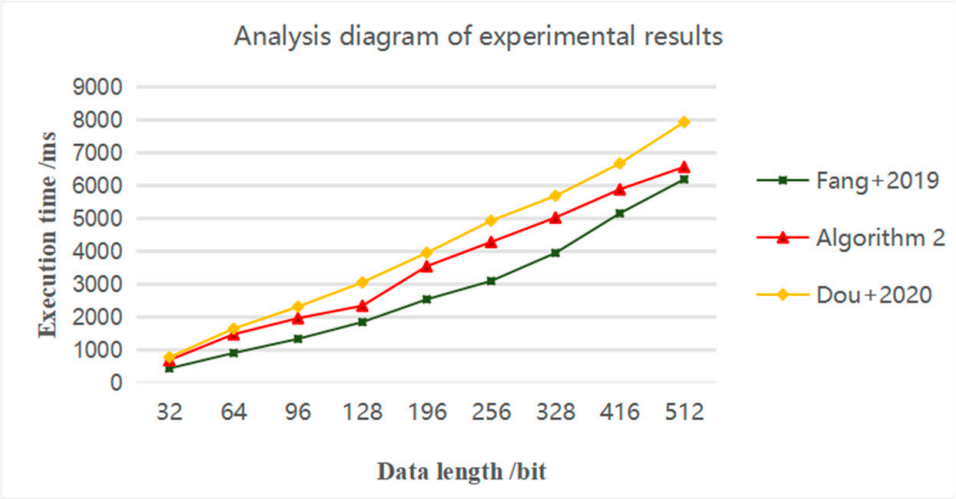

Figure 3. Time of executing protocols in Fang [29], Dou [30], and Algorithm 2.

Compared with references [29,30], the protocols have little difference in the number of communication rounds, but only Algorithm 2 proposed in this paper can resist malicious attacks. Although the efficiency of Algorithm 2 is not the highest, its complexity mainly comes from steps such as zero-knowledge proof, and this part of the calculation can be outsourced to improve efficiency.

7. Applications

The applications of Manhattan distance permeate many aspects of real life. Moreover, the research on MPC of Manhattan distance also has important practical application value,

such as privacy computation, computer graphics, data mining, and machine learning. The following are some specific application scenarios.

1. Computing distance is a measurement method. Manhattan distance is often used to measure the length of the path in many scientific studies. For example, in the study of biological cryptography, it is often necessary to judge whether the two biological templates are the same or similar. In data mining and machine learning, we often judge the analogy and similarity of individuals. When consuming products, it will also judge the similarity of consumers. Therefore, the similarity of different individuals can be deduced by calculating the distance value of individual feature vectors, that is, the protocol for MPC of Manhattan distance is the basic module for constructing the secret calculation vector similarity protocol.
2. The Manhattan distance between two points can better protect personal privacy and solve the constrained optimization problem. As shown in Figure 4, both military sides select military bases in an area, and neither side wants the other party to know where their military bases are located. At the same time, the military exchanges between the two sides are close, and the driving distance between the two military bases should be moderate, that is, the distance should be appropriate. In such an actual scene, the range size of the specified area is equivalent to the given complete set. Both parties just select the appropriate coordinate system in the area, and the location of the military bases of both parties is selected within the range of the complete set. For the suitability of the distance, both parties can jointly calculate the Manhattan distance between two points to make appropriate adjustments.

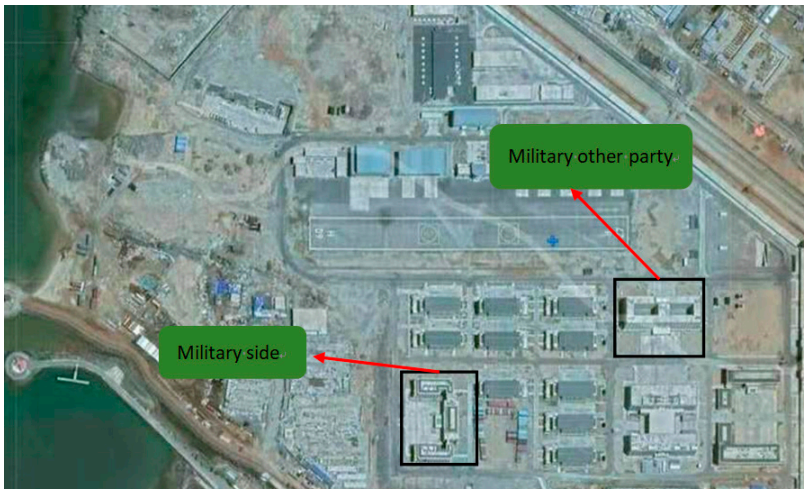

Figure 4. Site selection diagram of military base.

There are many similar scenarios, and there will be many constrained optimization problems in engineering practice, scientific research, and other fields. Therefore, to better solve the optimization constraint problem, it is particularly important to calculate the Manhattan distance between two points.

3. Today, with the development of information, information search and matching have practical application value. The location relationship between the security decision vector and the vector interval is the solution to solve the secure searching and matching. The problem of the relationship between the security decision vector and the vector set can also be solved by computing the Manhattan distance securely.

8. Conclusions

Securely computing Manhattan distance is a basic module for designing other MPC geometric protocols, so it has important theoretical significance and application value to study this problem. Given the shortcomings of existing protocols, combining Paillier's

algorithm with additive homomorphism and GM encryption algorithm with Xor homomorphism, this paper takes the lead in designing a protocol with high-security performance under the premise of resisting malicious participant attacks. Algorithm 2 used tools such as the cut-choose method to prevent deception. The real/ideal model paradigm method is used to prove the security of the malicious model protocol. Compared with existing protocols, the experimental simulation shows that only Algorithm 2 can resist malicious participants' attacks while maintaining high efficiency. The protocol is close to the reality of the existence of malicious participants and has more practical value. The next step is to improve Algorithm 2, using the homogeneous secret sharing to pass the ciphertext, to improve the protocol efficiency.

Author Contributions: Conceptualization, X.L. (Xin Liu) and X.L. (Xiaomeng Liu); funding acquisition, X.L. (Xin Liu); investigation, X.L. (Xin Liu); methodology, R.Z.; software, D.L. and G.X.; validation, G.X. and X.C.; writing—original draft, X.L. (Xiaomeng Liu); writing—review and editing, X.L. (Xin Liu), G.X., and X.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by NSFC, grant numbers 92046001 and 61962009; Inner Mongolia Natural Science Foundation, grant number 2021MS06006; 2022 Basic Scientific Research Project of Direct Universities of Inner Mongolia, grant number 20220101; 2022 Fund Project of Central Government Guiding Local Science and Technology Development, grant number 20220175; 2022 "Western Light" Talent Training Program "Western Young Scholars" Project; Inner Mongolia Discipline Inspection and Supervision Big Data Laboratory Open Project Fund, grant number IMDBD202020; Baotou Kundulun District Science and Technology Plan Project, grant number YF2020013; the 14th Five Year Plan of Education and Science of Inner Mongolia, grant number NGJGH2021167; Inner Mongolia Science and Technology Major Project, grant number 2019ZD025; 2022 Inner Mongolia Postgraduate Education and Teaching Reform Project, grant number 20220213; the 2022 Ministry of Education Central and Western China Young Backbone Teachers and Domestic Visiting Scholars Program, grant number 20220393; Basic Scientific Research Business Fee Project of Beijing Municipal Commission of Education, grant number 110052972027; Research Startup Fund Project of North China University of Technology, grant number 110051360002.

Institutional Review Board Statement: This article does not contain any studies with human participants performed by any of the authors.

Informed Consent Statement: Not applicable.

Data Availability Statement: The authors approve that data used to support the finding of this study are included in the article.

Acknowledgments: In the process of completing the manuscript, I got the help of many people. First of all, I should thank Baoshan Li for giving me many valuable opinions and providing me with the greatest support in thought and action. Second, I thank Yongxing Du for making a lot of amendments and suggestions to the manuscript. Finally, I want to thank my family, classmates, and friends for their support and encouragement.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

N	$N = pq$, Both p and q are large primes
m	Plaintext
c	Ciphertext
$pk(A)/sk(A)$	The public and private keys of Alice's GM encryption system
$pk(B)/sk(B)$	The public and private keys of Bob's GM encryption system
(g_a, N_a)	The public key of Alice's Paillier encryption system
(g_b, N_b)	The public key of Bob's Paillier encryption system
λ_a	The private key of Alice's Paillier encryption system
λ_b	The public key of Bob's Paillier encryption system

$E(\ )$	The process of converting encrypted plaintext into ciphertext
$D(\ )$	The process of decrypting ciphertext into plaintext
r_i	Random numbers
$\hat{R}(\ )$	Results for a random permutation of the ciphertext
$E_{pk(A)}$	Encrypted calculation with A's public key
$E_{pk(B)}$	Encrypted calculation with B's public key
$IDEAL_{F,B}(x,y)$	The function calculation results of x and y in the ideal case
$REAL_{\Pi,A}(x,y)$	The function calculation results of x and y in the practical case
$F(\ )$	Function calculation results

References

1. Ishai, Y.; Rijmen, V. Advances in cryptology-eurocrypt 2019. In Proceedings of the 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, 19–23 May 2019; Springer: Berlin/Heidelberg, Germany, 2019; pp. 94–97.
2. Wang, Z.; Pang, X.; Chen, Y. Privacy-preserving crowd-sourced statistical data publishing with an untrusted server. *IEEE Trans. Mob. Comput.* **2018**, *18*, 1356–1367. [CrossRef]
3. Zhou, J.; Feng, Y.; Wang, Z.; Guo, D. Using secure multi-party computation to protect privacy on a permissioned blockchain. *Sensors* **2021**, *21*, 1540. [CrossRef] [PubMed]
4. Wu, D.; Si, S.; Wu, S. Dynamic trust relationships aware data privacy protection in mobile crowd-sensing. *IEEE Internet Things J.* **2017**, *5*, 2958–2970. [CrossRef]
5. Fălămaş, D.E.; Marton, K.; Suciu, A. Assessment of Two Privacy Preserving Authentication Methods Using Secure Multiparty Computation Based on Secret Sharing. *Symmetry* **2021**, *13*, 894. [CrossRef]
6. Yao, A.C. Protocols for secure computations. In Proceedings of the 23rd Annual Symposium on Foundation of Computer Science (SFCS 1982), Chicago, IL, USA, 3–5 November 1982.
7. Goldreich, O. *Foundations of Cryptography*; Basic Applications; Cambridge University Press: Cambridge, UK, 2009; Volume 2.
8. Cramer, R.; Damgard, I.B.; Nielsen, J.B. *Secure Multiparty Compution*; Cambridge University Press: Cambridge, UK, 2015.
9. Hunt, T.; Jia, Z.; Miller, V.; Szekely, A.; Hu, Y. Telekine: Secure computing with cloud {GPUs}. In Proceedings of the 17th USENIX Symposium on Networked Systems Design and Implementation (NSDI 2020), Santa Clara, CA, USA, 25–27 February 2020.
10. Jiang, J.; Tang, L.; Gu, K.; Jia, W. Secure computing resource allocation framework for open fog computing. *Comput. J.* **2020**, *63*, 567–592. [CrossRef]
11. Shen, H.; Liu, Y.; Xia, Z.; Zhang, M. An efficient aggregation scheme resisting on malicious data mining attacks for smart grid. *Inf. Sci.* **2020**, *526*, 289–300. [CrossRef]
12. Wang, J.; Wu, L.; Zeadally, S.; Khan, M.K.; He, D. Privacy-preserving data aggregation against malicious data mining attack for IoT-enabled smart grid. *ACM Trans. Sens. Netw.* **2021**, *3*, 1–25. [CrossRef]
13. Akram, A.; Giannakou, A.; Akella, V.; Lowe, J.; Peisert, S. Performance analysis of scientific computing workloads on general purpose TEEs. In Proceedings of the 35th IEEE International Parallel & Distributed Processing Symposium (IPDPS), Portland, OR, USA, 17–21 May 2021.
14. Rao, V.S.; Satyanarayana, N. Experimental analysis and comparative study of secure data outsourcing schemes in cloud. *Int. J. Cloud Comput.* **2019**, *1*, 83–101. [CrossRef]
15. Li, W.; Meng, P.; Hong, Y.; Cui, X. Using deep learning to preserve data confidentiality. *Appl Intell* **2020**, *50*, 341–353. [CrossRef]
16. Zhang, K.X.; Yang, C.; Li, S.D. Privacy preserving string matching. *J. Cryptologic Res.* **2022**, *9*, 619–632. [CrossRef]
17. Fakroon, M.; Alshahrani, M.; Gebali, F.; Traore, I. Secure remote anonymous user authentication scheme for smart home environment. *IoT* **2020**, *9*, 100158. [CrossRef]
18. Li, S.D.; Xu, W.T.; Wang, W.L. Secure Maximum (Minimum) Computation in Malicious Model. *Chin. J. Comput.* **2021**, *44*, 2076–2089.
19. Liu, X.; Xu, Y.; Xu, G. Secure Judgment of Point and Line Relationship Against Malicious Adversaries and Its Applications. *J. Internet Technol.* **2022**, *23*, 1019–1027.
20. Liu, X.; Zhang, R.L.; Xu, G. Securely determine the inclusion relation of a point and a convex polygon in malicious model. *J. Cryptologic Res.* **2022**, *9*, 524–534. [CrossRef]
21. Resende, A.; Railsback, D.; Dowsley, R. Fast privacy-preserving text classification based on secure multiparty computation. *IEEE Trans. Inf. Forensics Secur.* **2022**, *17*, 428–442. [CrossRef]
22. Wang, Q.; Guo, Y.; Wang, X.; Ji, T.; Yu, L.; Li, P. AI at the edge: Blockchain-empowered secure multiparty learning with heterogeneous models. *IEEE Internet Things J.* **2020**, *10*, 9600–9610. [CrossRef]
23. Tran, A.T.; Luong, T.D.; Karnjana, J. An efficient approach for privacy preserving decentralized deep learning models based on secure multi-party computation. *Neurocomputing* **2021**, *422*, 245–262. [CrossRef]
24. Shutty, N.; Wootters, M. Low-bandwidth recovery of linear functions of Reed-Solomon-encoded data. *arXiv* **2021**, arXiv:2107.11847. [CrossRef]

25. Fosli, I.; Ishai, Y.; Kolobov, V.I.; Wootters, M. On the download rate of homomorphic secret sharing. *arXiv* **2021**, arXiv:2111.10126. [CrossRef]
26. Roy, L.; Singh, J. Large message homomorphic secret sharing from DCR and applications. In Proceedings of the 41st Annual International Cryptology Conference (CRYPTO 2021), Virtual Event, 16–20 August 2021. [CrossRef]
27. Naor, M.; Pinkas, B. Efficient oblivious transfer protocols. In Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms (SODA '01), Washington, DC, USA, 7–9 January 2001.
28. Lindell, Y.; Pinkas, B. Secure two-party computation via cut-and-choose oblivious transfer. *J. Cryptology* **2012**, *25*, 680–722. [CrossRef]
29. Fang, L.; Li, S.; Dou, J. Secure manhattan distance computation. *J. Cryptologic Res.* **2019**, *4*, 512–525. [CrossRef]
30. Dou, J.; Ge, X.; Wang, Y. Secure Manhattan distance computation and its application. *J. Comput. Sci.* **2020**, *2*, 352–365. [CrossRef]
31. Liu, C.; Zhu, L.; He, X.; Chen, J. Enabling privacy-preserving shortest distance queries on encrypted graph data. *IEEE Trans. Dependable Secur. Comput.* **2018**, *18*, 192–204. [CrossRef]
32. Li, S.D.; Wang, W.; Du, R. Protocol for millionaires' problem in malicious model. *Sci. Sin. Inf.* **2021**, *1*, 75–88. [CrossRef]
33. Paillier, P. Public-key cryptosystems based on composite degree residuosity classes. In Proceedings of the international Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT'99), Prague, Czech Republic, 2–6 May 1999.
34. Goldwasser, S.; Micali, S. Probabilistic encryption & how to play mental poker keeping secret all partial information. In Proceedings of the fourteenth annual ACM symposium on Theory of computing (STOC'82), San Francisco, CA, USA, 5–7 May 1982. [CrossRef]
35. Chaum, D.; Pedersen, T.P. Transferred cash grows in size. In Proceedings of the workshop on the Theory and Application of Cryptographic Techniques (EUROCRYPT '92), Balatonfüred, Hungary, 24–28 May 1992.