An Efficient Protocol for Privately Determining the Relationship Between Two Straight Lines

Ledi Fang[1], Shundong Li[1], Wenli Wang[2]
(Corresponding author: Shundong Li)
School of Computer Science, Shaanxi Normal University[1]
No. 620 Xi Chang'an Street, Xi'an 710119, China
(Email: shundong@snnu.edu.cn)
School of Mathematics and Information Science, Shaanxi Normal University[2]
(Received July 13, 2017; revised and accepted Oct. 22, 2017)

Abstract

Secure multiparty computation (SMC) is now a research focus in the international cryptographic community. SMC makes participants perform secure computation without revealing their own private data. In this paper, we discuss a secure computational geometry problem, that is, to privately determine whether two straight lines intersect. This is a basic and important SMC problem. Almost all protocols addressing this problem are applicable for integers, which limits their applications. So, we propose an efficient scheme for rational numbers. We proved that the protocol is secure under the semi-honest model by using the simulation paradigm. In addition, we propose a protocol which can be applied to space problems. This protocol can be used as a building block to construct new protocols to solve some space problems. Finally, we analyze the computational complexity and communication complexity of the protocol, and present an experimental result.

Keywords: Secure Multi-party Computation; Simulation Paradigm; Straight Line Intersection

1 Introduction

Secure Multiparty Computation (SMC) [3] was initially introduced by Yao as the millionaires' problem [16] in 1982 for two parties. Then Ben and Goldwasser [2] extended SMC to multiple parties and established the theoretical basis of SMC [5, 13]. The heart of SMC is that parties can cooperatively compute a function of their own private data without disclosing any private information. Hence, the paries are able to maximize their interests while protecting their data privacy.

Privacy-preserving computational geometry is a promising research area of SMC. It mainly focuses on protecting the security of computational geometry. Many privacy-preserving computational geometry problems have been studied, such as point-inclusion, intersection of two convex polygons, convex hulls. Du [1] introduced the problem of the intersection of two straight lines. Later, Luo [10] presented and solved the problem.

The relationship between two straight lines has significant application in practice. For instance, the spy of Country A observes activity an route L_1 while another spy in Country B observes an activity route L_2. They are willing to cooperate to figure out whether L_2 is relevant to L_1 and the result is helpful for both countries to understand the trend of the target's behaviors, such as some suspected terroristic organizations, the military dynamics of a dangerous country. However, neither A nor B wants to disclose its observation to each other because they don't believe each other. It is possible that Country B exploits the intelligence information of Country A (or sells it to the target) to expose the spy of Country A, resulting in the spy being persecuted. So the problem of the relationship between two straight lines is of great significance.

A number of scholars have proposed protocols for this problem. For example, Luo introduced and solved the problem of the intersection of two straight lines. The protocol is helpful, but it only works for integer points on the lines [10]. In our real life, we usually choose some rational points on the lines to meet the needs of numerous practical applications. So we propose an efficient protocol based on the plane geometry to solve this problem. Then we use the simulation paradigm to prove the security of the protocol. In addition, we propose a protocol which can be applied to some space problems. This protocol can be used as a building block to construct new protocols to solve some space problems. Finally, we present the computation and communication complexity of different protocols and show an experimental result.

2 Preliminaries

2.1 Security

Two-party Computation. Two-party computation is a random mapping process where a random input pair is mapped to a random output pair, which is represented below:

$$f : \{0,1\}^* \times \{0,1\}^* \longrightarrow \{0,1\}^* \times \{0,1\}^*$$

That is to say, for an arbitrary given input pair, the function will output a pair of random variables $(f_1(x,y), f_2(x,y))$. The function is denoted as

$$(x,y) \longrightarrow (f_1(x,y), f_2(x,y))$$

Semi-honest parties [8]. Our work assumes that all parties are semi-honest. loosely speaking, a semi-honest party is one that follows the protocol properly, except that it keeps a records of all its intermediate computations and might try to derive the other parties' private inputs from the record. Goldreich [4] proved that, given a protocol that privately computes functionality f in the semi-honest model, we can construct a protocol by introducing macros that force each party either to behave in the semi-honest manner or to be detected, by which case we can privately compute functionality f in the malicious model. The semi-honest model is not merely an important methodological locus but may also provide a good model for many settings. It suffices to prove that a protocol is secure in the semi-honest setting.

Privacy by simulation [8]. Intuitively, a protocol is private if what a party can efficiently compute by participating in the protocol can also be efficiently computed from its input and output only. This assumption is formalized by the simulation paradigm, which requires that a party's view in a protocol execution can be simulated by its input and output only. If so, the parties learn nothing from the protocol execution itself, and the protocol is secure.

Definition 1. *For a functionality f, π privately computes f if there exist probabilistic polynomial-time algorithms, denoted by S_1 and S_2 such that*

$$\{S_1(x, f_1(x,y)), f_2(x,y)\}_{x,y}$$

$$\stackrel{c}{\equiv} \{view_1^{\pi}(x,y), output_2^{\pi}(x,y)\}_{x,y} \quad (1)$$

and

$$\{f_1(x,y)), S_2(y,(x,y))\}_{x,y}$$

$$\stackrel{c}{\equiv} \{view_1^{\pi}(x,y), output_2^{\pi}(x,y)\}_{x,y} \quad (2)$$

where $\stackrel{c}{\equiv}$ denotes computational indistinguishability, $view_1^{\pi}(x,y)$ and $view_2^{\pi}(x,y)$, $output_1^{\pi}(x,y)$ and $output_2^{\pi}(x,y)$ are related random variables, defined as a function of the same random execution.

2.2 A Symmetric Cryptographic Solution to Determine Whether Two Numbers Are Equal

Li *et al.* [9] proposed a secure solution to determine whether two numbers are equal by using XOR operations. This cryptographic protocol is much more efficient than others because the computational complexity of symmetric encryption is much lower than that of public key encryption. We use this scheme as a basic module to design Protocol 3 in Section 3. Li's protocol is as follows:

Protocol 1: A symmetric cryptographic solution to determine whether two numbers are equal.

Inputs: Alice has a number a, Bob has a number b.

Output: Whether $a = b$.

Setup: Alice and Bob choose random numbers $r \in \{0,1\}^m$ and $s \in \{0,1\}^n (m, n > 64)$, respectively, and compute $c = a \oplus r$, $d = b \oplus s$. Then exchange c and d.

Encryption Process: Alice and Bob compute $a' = d \oplus r = b \oplus s \oplus r$, $a' = c \oplus s = a \oplus r \oplus s$, respectively. Then they use hash to compute $hash(a')$ and $hash(b')$, respectively. Finally, they exchange $hash(a')$ and $hash(b')$.

Decryption Process: Alice and Bob judge whether $hash(a') = hash(b')$. If it holds, then $a = b$; otherwise $a \neq b$.

2.3 Area of the Triangle In the Plane

There are three points $P_1(x_1, y_1)$, $P_2(x_2, y_2)$ and $P_3(x_3, y_3)$ with rational coordinates. The area of the triangle constituted by these three points can be computed as follows:

$$S_{\triangle P_1 P_2 P_3} = \frac{1}{2}[y_1(x_3 - x_2) + x_1(y_2 - y_3) + x_2 y_3 - x_3 y_2] \quad (3)$$

If $P_1(x_1, y_1)$, $P_2(x_2, y_2)$ and $P_3(x_3, y_3)$ is in counterclockwise order, then the area value is positive; otherwise the area value is negative.

2.4 Protocol Based on the Formula for the Area of the Triangle In the Plane

Li *et al.* proposed a protocol [7] for securely computing the area of a triangle, but the protocol discloses the slope. Then they improved and developed the protocol [6]. We use this protocol as a basic module to design Protocol 3 in Section 3. The protocol is as follows:

Protocol 2: Securely compute Area of a Triangle in the Plane.

Inputs: Alice' s input is point $P_1(x_1, y_1)$, and Bob' s inputs are point $P_2(x_2, y_2)$ and point $P_3(x_3, y_3)$.

Output: $S_{\triangle P_1P_2P_3}$.

Setup: Paillier's homomorphic encryption scheme (G, E, D), Bob runs $G(\tau)$ (τ is the given security parameter) to generate a key pair (p_k, s_k).

Encryption Process:

Bob executes the following:

1) Computes $a = x_3 - x_2$, $b = y_2 - y_3$, $c = x_2y_3 - x_3y_2$. It is straightforward that the signs of a and b are different. We assume that $a > 0$, $b < 0$.
2) Chooses a random number r ($r \in Z_n^*$) such that $b_1 = b + r > 0$.
3) Uses the public key p_k to encrypt a and b_1, the results are denoted by $E(a)$ and $E(b_1)$, and then sends $E(a)$, $E(b_1)$, r and p_k to Alice.

Alice computes

$$\begin{aligned} E(S_1) &= E(ay_1 + b_1x_1) = E(a)^{y_1} \cdot E(b_1)^{x_1} \\ R &= rx_1 \end{aligned}$$

and then sends $E(S_1)$ to Bob.

Decryption Process:

Bob decrypts $E(S_1)$

$$S_1 = ay_1 + b_1x_1 = ay_1 + b_1x_1 + rx_1$$

and computes

$$S_2 = S_1 + c = ay_1 + b_1x_1 + c + rx_1$$

Bob sends S_2 to Alice.

Alice computes

$$S_{\triangle P_1P_2P_3} = \frac{1}{2}(S_2 - R) = \frac{1}{2}(ay_1 + b_1x_1 + c)$$

Alice tells Bob the result.

3 Determining the Relationship Between Two Straight Lines and Its Extension

In this section, we aim at solving the problem of privately determining the relationship between two straight lines. That is, Alice and Bob desire to determine the relationship between their own lines without disclosing the lines' information. This problem can be generalized as follows. Alice has $L_1 : y = k_1x + b_1$ and Bob has $L_2 : y = k_2x + b_2$. They want to know whether these two lines intersect. In addition, they want to know whether $L_1 // L_2$ or $L_1 \perp L_2$ without disclosing information about the lines. Many protocols have been put forward in recent years to solve this problem. Luo [10] put a scheme with high computational complexity. Yang [15] improved the protocol by using Paillier homomorphic encryption [14]. The Paillier public key encryption has additively homomorphic [11] property. Yang's protocol is of high computational complexity due to many modular exponentiation operations. Although Luo and Yang's protocols solved the problem, their protocols are only limited to the integer field. In real life, $k_1(k_2)$ or $b_1(b_2)$ are more likely to be rational numbers. The existing protocols are not applicable. So it's necessary to design a protocol to meet this requirement and we propose a protocol to solve this problem.

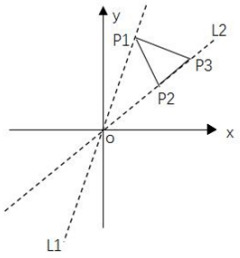

Figure 1: Two lines intersect

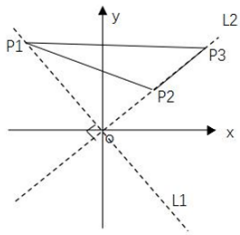

Figure 2: Two perpendicular lines

3.1 An Efficient Protocol for Determining the Relationship Between Two Straight Lines

Alice has a private line $L_1 : y = k_1x + b_1$ where k_1 and b_1 are rational numbers, Bob has a private line $L_2 : y = k_2x + b_2$ where k_2 and b_2 are also rational numbers. They can separately and secretly compare the slopes and intercepts. The two lines are parallel if they have the same slopes and different intercepts. They are coincident if they have the same slopes and intercepts. Otherwise, the two lines intersect or may be perpendicular.

In the latter situation, Alice and Bob separately shift L_1 and L_2 to go through the origin. Alice randomly chooses a point denoted by $P_1(x_1, y_1)$ on L_1, Bob randomly chooses two points denoted as $P_2(x_2, y_2)$ and $P_3(x_3, y_3)$ on L_2. These three points constitute a triangle $\triangle P_1P_2P_3$. We denote the height of $\triangle P_1P_2P_3$ by h, the area of $\triangle P_1P_2P_3$ by $S_{\triangle P_1P_2P_3}$. $h = \frac{2S_{\triangle P_1P_2P_3}}{\overline{P_2P_3}}$. We determine whether $h = \overline{op_1}$ (See Figure 1 and Figure 2).

If $h = \overline{op_1}$, L_1 and L_2 are perpendicular.

If $h \neq \overline{op_1}$, L_1 and L_2 intersect.
In order to describe clearly, we define

$$P(L_1, L_2) = \begin{cases} 0, & L_1, L_2 \text{ intersect} \\ 1, & L_1, L_2 \text{ are parallel} \\ 2, & L_1, L_2 \text{ are coincident} \\ 3, & L_1, L_2 \text{ are perpendicular} \end{cases}$$

Protocol 3: An Efficient Protocol for Determining the Relationship between Two Straight Lines

Inputs: Alice' s private line $L_1 : y_1 = k_1 x + b_1$, Bob' s private line $L_2 : y_2 = k_2 x + b_2$.

Output: $P(L_1, L_2)$

Setup: Suppose that $k_1 = \frac{u_1}{v_1}$ where $\gcd(u_1, v_1) = 1$, and $k_2 = \frac{u_2}{v_2}$ where $\gcd(u_2, v_2) = 1$. Alice and Bob use Protocol 1 to compare whether $u_1 = u_2$, $v_1 = v_2$, respectively. If $u_1 = u_2$ and $v_1 = v_2$, then $k_1 = k_2$. Similarly, Alice and Bob determine whether $b_1 = b_2$. If $k_1 = k_2$ and $b_1 = b_2$, then L_1 and L_2 are coincident. Otherwise, Alice and Bob do the following.

Alice and Bob separately shift L_1 and L_2 to go through the origin. Alice randomly chooses a point $P_1(x_1, y_1)$ on L_1, and Bob randomly chooses two points denoted by $P_2(x_2, y_2)$ and $P_3(x_3, y_3)$ on L_2. These three points constitute a triangle $\triangle P_1P_2P_3$.

Encryption Process: Alice and Bob use Protocol 2 to privately compute the area of $\triangle P_1P_2P_3$.

Decryption Process: Bob computes h, and Alice computes $\overline{op_1}$. Then they use protocol 1 to determine whether $h = \overline{op_1}$. If

$$h = \overline{op_1}$$

L_1 and L_2 are perpendicular. Otherwise, L_1 and L_2 intersect. Then they can get the result of $P(L_1, L_2)$.

Thus, it's important for us to use the idea, but almost all protocols used to address this problem only work for planes. This limits their applications. Thus, we propose an efficient protocol for spaces.

3.2 A Secure Computational Protocol for Triangle Area in Spaces

By the formula, the area of the triangle constituted by three rational points $P_1(x_1, y_1, z_1)$, $P_2(x_2, y_2, z_2)$ and $P_3(x_3, y_3, z_3)$ in spaces is as follows:

$$S_{\triangle P_1P_2P_3} = \frac{1}{2}\sqrt{\begin{vmatrix} y_1 & z_1 & 1 \\ y_2 & z_2 & 1 \\ y_3 & z_3 & 1 \end{vmatrix}^2 + \begin{vmatrix} z_1 & x_1 & 1 \\ z_2 & x_2 & 1 \\ z_3 & x_3 & 1 \end{vmatrix}^2 + \begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix}^2} \quad (4)$$

which yields:

$$\begin{aligned} S_{\triangle P_1P_2P_3} &= \frac{1}{2}\{[x_1(y_2 - y_3) + y_1(x_3 - x_2) + y_2x_3 - y_3x_2]^2 \\ &\quad + [y_1(z_2 - z_3) + z_1(y_3 - y_2) + y_2z_3 - y_3z_2]^2 \\ &\quad + [z_1(x_2 - x_3) + x_1(z_3 - z_2) + z_2x_3 - z_3x_2]^2\}^{\frac{1}{2}}. \end{aligned}$$

We design a protocol to securely computes the area of a space triangle, and it is shown as follows.

Protocol 4: An efficient protocol for computing the space triangle area

Inputs: Private rational points $P_1(x_1, y_1, z_1)$, $P_2(x_2, y_2, z_2)$ and $P_3(x_3, y_3, z_3)$, where Alice has P_1 and Bob has P_2, P_3.

Output: $S = \frac{1}{2}\{[x_1(y_2 - y_3) + y_1(x_3 - x_2) + y_2x_3 - y_3x_2]^2 + [y_1(z_2 - z_3) + z_1(y_3 - y_2) + y_2z_3 - y_3z_2]^2 + [z_1(x_2 - x_3) + x_1(z_3 - z_2) + z_2x_3 - z_3x_2]^2\}^{\frac{1}{2}}$

Encryption Process:

Bob does the following:

1) Randomly chooses a random number $k \in Z_n^*$. (k is accurate to three decimal places.)
2) Computes

$$\begin{aligned} a &= y_3 - y_2, b = x_3 - x_2, c = y_2x_3 - y_3x_2 \\ d &= z_3 - z_2, e = y_2z_3 - y_3z_2, f = z_2x_3 - z_3x_2 \end{aligned}$$

and constitutes vectors

$$A = ((-ak, bk), (-dk, ak), (-bk, dk))$$

3) Sends A to Alice.

Alice randomly chooses three random numbers r_1,r_2,$r_3 \in Z_n^*$, and computes

$$\begin{aligned} T_1 &= -akx_1 + bky_1 + r_1 \\ T_2 &= -dky_1 + akz_1 + r_2 \\ T_3 &= -bkz_1 + dkx_1 + r_3 \end{aligned}$$

Then Alice sends T_1, T_2, T_3 to Bob.

Bob computes

$$\begin{aligned} T_1' &= T_1 + kc \\ T_2' &= T_2 + ke \\ T_3' &= T_3 + kf \end{aligned}$$

and sends T_1', T_2', T_3' to Alice.

Alice computes

$$\begin{aligned} D_1 &= T_1' - r_1 \\ D_2 &= T_2' - r_2 \\ D_3 &= T_3' - r_3 \\ T &= D_1^2 + D_2^2 + D_3^2 \end{aligned}$$

and tells T to Bob.

Decryption Process: $S = \frac{1}{2k}T^{\frac{1}{2}} = \frac{1}{2}\{[x_1(y_2-y_3)+y_1(x_3-x_2)+y_2x_3-y_3x_2]^2+[y_1(z_2-z_3)+z_1(y_3-y_2)+y_2z_3-y_3z_2]^2+[z_1(x_2-x_3)+x_1(z_3-z_2)+z_2x_3-z_3x_2]^2\}^{\frac{1}{2}}$ and tells the result to Alice.

Correctness: By Formula (4), the area of $\triangle P_1P_2P_3$ can be computed from $P_1(x_1, y_1z_1)$, $P_2(x_2, y_2, z_2)$ and $P_3(x_3, y_3, z_3)$. So Protocol 4 is correct.

Privacy: In order to analyse the security, we check whether each party can obtain the others' private information by executing Protocol 4. A brief analysis of the privacy of Protocol 4 is given as follows.

According to Protocol 4, Alice are supposed to receive a, b, d that contain Bob's unknown variables: $x_2, x_3, y_2, y_3, z_2, z_3$. It is obvious that the six unknown variables cannot be derived from three equations. Therefore, Alice cannot obtain Bob's secret points.

Bob can only gain Alice's information from three equations as follows:

$$T_1 = -akx_1 + bky_1 + r_1$$

$$T_2 = -dky_1 + akz_1 + r_2$$

$$T_3 = -bkz_1 + dkx_1 + r_3$$

Because of the random numbers r_1, r_2, r_3 which Alice adds, it is impossible for Bob to obtain Alice's secret point the six unknown variables$(x_1, y_1, z_1, r_1, r_2, r_3)$ from the three equations. Therefore, Bob cannot obtain Alice's secret points. This demonstrates that protocol 4 is private.

Thus, they can securely compute the area of a triangle in the space.

3.3 Applications

As mentioned above, Protocol 4 can be used as a building block to construct new protocols to solve some space problems such as the problem of the intersection of a line and a plane. This problem is as follows:

Alice has a line $L : \frac{x-x_0}{X} = \frac{y-y_0}{Y} = \frac{z-z_0}{Z}$. Bob has a plane $\pi : Ax+By+Cz+D = 0$. They want to determine the relationship between the line and the plane without revealing their private data.

Firstly Bob finds out the line L_3 of the normal vector of the plane. They can separately shift L and L_3 to go through the origin, and use Protocol 4 to determine the relationship between L_3 and L. Thereby the relationship between the line and the plane is obtained.

In order to describe clearly, we define

$$P(L,\pi) = \begin{cases} 0, & L, \pi \text{ intersect} \\ 1, & L, \pi \text{ are parallel or coincident} \\ 2, & L, \pi \text{ are perpendicular} \end{cases}$$

Protocol 5: A Scheme for Determining the Relationship between a line and a plane.

Inputs: Alice' s private line $L : \frac{x-x_0}{X} = \frac{y-y_0}{Y} = \frac{z-z_0}{Z}$, Bob' s private plane $\pi : Ax + By + Cz + D = 0$.

Output: $P(L, \pi)$

Encryption Process:

Bob finds out the of normal vector L_3 of the plane. Then they separately shift L and L_3 to go through the origin. Bob randomly chooses two points $P_4(x_4, y_4, z_4)$ and $P_5(x_5, y_5, z_5)$ on L_3. Alice randomly chooses a point $P_6(x_6, y_6, z_6)$.

Decryption Process:

Alice and Bob use Protocol 4 to compute the area of $\triangle P_4P_5P_6$. And then they determine the relationship between L and L_3.

1) If L and L_3 are parallel or coincident , then L and π are perpendicular.
2) If L and L_3 intersect, then L and π intersect.
3) If L and L_3 are perpendicular, then L and π are parallel or L is in the π.

Similarly, we can utilize the idea to solve the problem of determining the relationship of two planes in spaces.

4 Security

In this section, we use the simulation paradigm to prove that Protocol 3 is secure.

Theorem 1. *Protocol 3 can securely determine the relationship of straight lines.*

Proof. Alice and Bob respectively construct two simulators, S_1 and S_2 which make Equations (1) and (2) hold.

In Protocol 3:

$$\begin{aligned} view_1^{\pi}(L_1, L_2) &= \{L_1, hash(\overline{k}), hash(\overline{b}), S_{\triangle}, \overline{op_1}, \\ &\quad hash(\overline{h}), P(L_1, L_2)\} \\ f_1(L_1, L_2) &= f_2(L_1, L_2) \\ &= output_1^{\pi}(L_1, L_2) \\ &= output_2^{\pi}(L_1, L_2) \\ &= P(L_1, L_2). \end{aligned}$$

L_1, L_2 are the inputs of Alice and Bob. Alice got $S_{\triangle}$ when Protocol 2 finished.

Bob sent $hash(\overline{k})$ and $hash(\overline{b})$ to the Alice when they comparing whether $k_1 = k_2$, $b_1 = b_2$. In addition, Bob sent $hash(\overline{h})$ to the Alice when they comparing whether $h = \overline{op_1}$.

Alice constructs S_1. S_1 performs the following simulation.

1) By $f(L_1, L_2)$, S_1 randomly chooses a line L_2' such that $P(L_1, L_2') = P(L_1, L_2)$. Suppose that $L_2' : y' = k_2'x + b_2'$.

2) Suppose that $k_2' = \frac{u_2'}{v_2'}$ where $\gcd(u_2', v_2') = 1$. S_1 compare whether $u_1 = u_2'$, $v_1 = v_2'$. Then S_1 randomly chooses two points $P_2'(x_2', y_2')$and $P_3'(x_3', y_3')$ on L_2'.

3) S_1 computes $S'_{\triangle} = \frac{1}{2}[y_1(x_3' - x_2') + x_1(y_2' - y_3') + x_2'y_3' - x_3'y_2']$.

Clearly, $S'_{\triangle} \neq 0$. Then S_1 computes h' and $\overline{op_1'}$. In addition, S_1 determine whether $h' = \overline{op_1'}$.

Let

$$S_1(L_1, f_1(L_1, L_2)) = \{L_1, hash(\overline{k}'), hash(\overline{b}'), S'_{\triangle}, \overline{op_1}, hash(\overline{h}'), P(L_1, L_2)\}.$$

Since the selected points are random points and Protocol 1 has been proved, then

$$hash(\overline{k}) \stackrel{c}{\equiv} hash(\overline{k}'),\ hash(\overline{b}) \stackrel{c}{\equiv} hash(\overline{b}')$$

$$S_{\triangle} \stackrel{c}{\equiv} S'_{\triangle},\ hash(\overline{h}) \stackrel{c}{\equiv} hash(\overline{h}')$$

thus,

$$\{S_1(L_1, P(L_1, L_2), f_1(L_1, L_2)), f_2(L_1, L_2)\}$$

$$\stackrel{c}{\equiv} \{view_1^{\pi}(L_1, L_2), output_2^{\pi}(L_1, L_2)\}$$

Similarly, the simulator such that Eq.(2) holds can be constructed analogously, thus,

$$\{f_1(L_1, L_2), S_2(L_2, f_2(L_1, L_2))\}$$

$$\stackrel{c}{\equiv} \{output_1^{\pi}(L_1, L_2), view_1^{\pi}(L_1, L_2)\}$$

This completes the proof of the theorem. □

5 Efficiency Analysis

5.1 Theoretical Analysis

Computational complexity . There are many protocols such as Luo's scheme and Yang's scheme [15] determining the relationship between two lines. Luo's scheme was put forward at first, then Yang greatly improved recently. So we compare our protocols with Luo's scheme and Yang's scheme.

Luo's [10] scheme uses the scalar product protocol [1] for n times. The scalar product protocol utilizes an efficient oblivious transfer [12]. Suppose that the security parameter is p. Every invocation of scalar product protocol needs to use 1-out-of-k oblivious transfer p times. It needs $\lg k$ 1-out-of-2 oblivious transfer for a 1-out-of-k oblivious transfer. Each 1-out-of-2 oblivious transfer needs two modular exponentiation operations at least.

Therefore, Luo's Scheme needs at least $2p \lg k$ modular exponentiation operations. In order to meet the security requirement, Luo's scheme requires $p > 5$ and $k > 8$. So, Luo's scheme requires 30 modular exponentiation operations at least.

Yang's scheme uses the Paillier homomorphic encryption. Yang's protocol 3 (Yang 3) encrypts 3 times and decrypts 6 times to determine the relationship between two straight lines. That is to say, it uses 12 modular exponentiation operations in total.

Our Protocol 3 uses XOR operations so it greatly reduces the computational complexity. Protocol 3 uses at most 8 modular exponentiation operations for computing the area of the triangle. In addition, our protocols can be utilized in rational field while Luo's scheme and Yang's scheme does not work in rational field.

Communication Complexity. Communication complexity, i.e. communication rounds, is an important factor to evaluate secure multiparty computation solutions. Luo's scheme needs p rounds. Yang's scheme requires 2 round communications. Our protocols also require 2 round communications. Table 1 summarizes the comparison.

5.2 Simulation Result

In this section, we present an experimental result in terms of efficiency. Since Yang's scheme is much more efficient than Luo's scheme, we only compare our protocols with Yang's Scheme.

Experimental Settings: All the experiments are conducted on an HP PC with 3.30 GHz Intel Core i5-6600 processor with 8 GB RAM running a 64-bit Windows 10 Enterprise. The program code is written in Java.

Time Complexity Analysis: Our protocols can be used in the rational field. Supposed that Alice selects a point (16.5,13.2) on her line and Bob chooses two points (14.4,10.8) and (9.6,7.2) on his line. We run the experiment for 10000 times and randomly pick up 10 sets of data and the result is shown in Figure 3. Yang's scheme does not work in rational field, so we choose some integers to test. We assume that in the Paillier homomorphic encryption scheme the two large primes p and q are 256 bits. Suppose that Alice selects a point (17,13) on her line and Bob chooses two points (11,8) and (10,7) on his line. We run the experiment for 10000 times and randomly pick up 10 sets of data and the result is shown in Figure 3.

Table 1: Comparison of the computational and communication complexity

	Luo's scheme	Yang 3	Protocol 3
Computational Complexity	$2p \lg k$	12	8
Communication Complexity	p	2	2

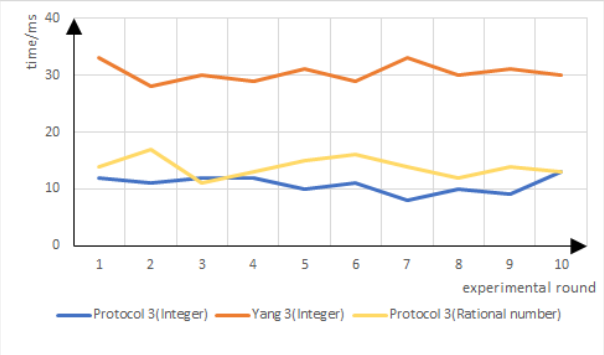

Figure 3: The comparison of Protocol 3 and Yang 3

The results of this experiment validate that our protocols are more efficient.

6 Conclusion

In this paper, we proposed an efficient protocol to privately determine the relationship between two straight lines. The protocol improves the efficiency by utilizing XOR operations and the idea of computing the area of a triangle in the planes. Also, we presented a protocol to compute the area of a triangle in the spaces. In addition, the two protocols can be used in rational field. Then we utilized simulation paradigm to prove the security and did experiment to show the efficiency of Protocol 3. In the future, We will discuss the problem of the relationship between two straight lines in the malicious model.

References

[1] M. J. Atallah, W. L. Du, "Secure multi-party computational geometry," in *Algorithms and Data Structures, Springer Berlin Heidelberg*, pp. 165–179, 2001.

[2] M. Ben-Or, S. Goldwasser, A. Wigderson, "Completeness theorems for non-cryptographic fault-tolerant distributed computation," in *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*, pp. 1–10,1988.

[3] W. L. Du, M. J. Atallah, "Secure multi-party computation problems and their applications: A review and open problems," in *ACM Proceedings of the 2001 workshop on New security paradigms*, pp. 13–22, 2001.

[4] O. Goldreich, *Foundations of Cryptography: Basic Applications*, Cambridge University Press, London, 2004.

[5] S. N. Kumar, "Review on network security and cryptography," *International Transaction of Electrical and Computer Engineers System*, vol. 3, no. 1, pp. 1–11, 2015.

[6] S. D. Li, Y. M. Guo, S. F. Zhou, J. W. Dou, D. S. Wang, "Efficient protocols for the general millionaires' problem," *Chinese Journal of Electronics*, vol. 24, no. 4, pp. 598–604, 2015.

[7] S. D. Li, D. S. Wang, Y. Q. Dai, "Efficient secure multiparty computational geometry," *Chinese Journal of Electronics*, vol. 19, no. 2, pp. 324–328, 2010.

[8] S. D. Li, D. S. Wang, Y. Q. Dai, P. Luo, "Symmetric cryptographic solution to Yaos millionaires' problem and an evaluation of secure multiparty computations," *Information Sciences*, vol. 178, no. 1, pp. 244–255, 2008.

[9] S. D. Li, X. L. Yang, X. J. Zuo, S. F. Zhou, J. Kang, X. Liu, "Privacy protecting similitude determination for Graphics Similarity," *Chinese Journal of Electronics*, vol. 45, no. 9, pp. 2184–2189, 2017.

[10] Y. L. Luo, L. S. Huang, W. W. Jing, W. J. Xu, "Privacy protection in the relative position determination for two spatial geometric objects," *Computer Research and Development*, vol. 43, no. 3, pp. 410–416, 2006.

[11] Y. L. Luo, L. S. Huang, W. Yang, W. J. Xu, "An efficient protocol for private comparison problem," *Chinese Journal of Electronics*, vol. 18, no. 2, pp. 205–209, 2009.

[12] M. Naor, B. Pinkas, "Oblivious transfer and polynomial evaluation," in *ACM Proceedings of the Thirty-first Annual ACM Symposium on Theory of Computing*, pp. 245–254, 1999.

[13] R. Sharma, "Review paper on cryptography," *International Journal of Research*, vol. 2, no. 5, pp. 141–142, 2015.

[14] J. H. Wu, P. Zhang, X. B. Shi, "Research of MA protection based on addition-multiplication homomorphism and composite function technology," *Journal of Chinese Computer Systems*, vol. 33, no. 10, pp. 2223–2226, 2012.

[15] X. L. Yang, S. D. Li, X. J. Zuo, "Secure multi-party geometry computation," *Journal of Cryptologic Research*, vol. 3, no. 1, pp. 33–41, 2016.

[16] A. Yao, "Protocols for secure computations," in *IEEE Proceeding of the 23th IEEE Annual Symposium on Foundations of Computer Science*, pp. 160–164, 1982.

Biography

Ledi Fang was born in 1993. She is currently pursuing the M.S. degree with School of Computer Science in Shaanxi Normal University. Her research interests focus on secure multi-party computation and information security.

Shundong Li was born in 1963. He received the Ph.D. degree in Department of computer science and technology from Xian JiaoTong University in 2003. He is now a Professor with School of Computer Science in Shaanxi Normal University. His research interests focus on modern cryptography and secure multi-party computation.

Wenli Wang was born in 1991. She is currently pursuing the M.S. degree with School of Mathematics and Information Science in Shaanxi Normal University. Her research interests focus on modern cryptography and applied mathematics.