applied sciences

MDPI

*Article*

# Privacy Protection Based on Special Identifiers of Intersection Base Computing Technology

**Ping Zhang [1,2], Tengfei Ma [1,*], Qing Zhang [1], Ji Zhang [1,2] and Jiechang Wang [3]**

[1] School of Mathematics and Statistics, Henan University of Science and Technology, Luoyang 471023, China; zping@haust.edu.cn (P.Z.); zhangq@stu.haust.edu.cn (Q.Z.); zhangji@haust.edu.cn (J.Z.)
[2] Intelligent System Science and Technology Innovation Center, Longmen Laboratory, Luoyang 471023, China
[3] Sports Big Data Center, Department of Physical Education, Zhengzhou University, Zhengzhou 450001, China; wangjiechang@126.com
* Correspondence: 210321100699@stu.haust.edu.cn

**Abstract:** Private Set Intersection Cardinality (PSI-CA) and Private Set Union Cardinality (PSU-CA) are two cryptographic primitives whereby two or more parties are able to obtain the cardinalities of the intersection and the union of their respective private sets, and the privacy of their sets is preserved. In this paper, we propose a new privacy protection intersection cardinality protocol, which can quickly deal with set inequality and asymmetry problems and can obtain 100% correct results, and, in terms of efficiency, we are much faster than using the polynomial method. Our protocol adopts the Paillier addition homomorphic encryption scheme and applies the identifier guidance technology, using identifier determination, to the semi-homomorphic encryption ciphertext environment, excluding a large number of different options and quickly finding the base of the intersection of two sides.

**Keywords:** semi-homomorphic encryption; PSI; MPC; PSI-CA



**Citation:** Zhang, P.; Ma, T.; Zhang, Q.; Zhang, J.; Wang, J. Privacy Protection Based on Special Identifiers of Intersection Base Computing Technology. *Appl. Sci.* **2024**, *14*, 813. https://doi.org/10.3390/app14020813

Academic Editors: Libin Yang and Honglong Chen

Received: 25 October 2023
Revised: 4 January 2024
Accepted: 11 January 2024
Published: 18 January 2024



## 1. Introduction

In today's digital age, more and more companies are reaping great rewards by collecting data and using them based on their own legitimate needs, such as intelligent AD recommendation systems, privacy data queries, and more. In the above application scenarios, the user's personal privacy is difficult to protect, and a large amount of personal information will be disclosed during the process, such as various marketing activities, so data privacy computing technology becomes more and more important. In various countries around the world, legislation has been enacted to protect data privacy security, such as the HIPAA, GLBA, COPPA, DPP, and so on, so data privacy computing technology has gradually boomed in the current academic research.

However, at present, it is either difficult to guarantee security with the international privacy protection protocol, or its efficiency is particularly low. If the calculation costs are too large, they will bring a major burden to the user's privacy protection process, and if security is difficult to guarantee, then the privacy protection will become a joke. Therefore, with a reasonable security protocol, there will be huge benefits.

The two-party PSI problem is the most basic kind of problem in the two-party computing model of security. In the two-party PSI problem, we assume that the two parties are Alice and Bob, and we assume that X and Y are any set of strings held by the two parties. At the completion of a series of interactions, we require that at least one party is able to obtain the intersection XY of both parties, and that no one participant is able to know the elements of the other party's set.

The two-party PSI-CA/PSU-CA problem is an extension of the two-party PSI problem. One only needs to calculate the cardinality of the intersection of the two parties, without revealing any elements of the other party's set in the process, and this problem is useful

for many privacy computing scenarios in reality. For example, in a social network, two users can calculate their degree of social compatibility by comparing the proportion of their mutual friends without revealing their specific friends. In health, customers with private genetic data can confidently interact with public risk genetic databases to know their probability of contracting a disease. The purpose of this article is to discuss the related protocols for solving the PSI-CA/PSU-CA problem.

In general, all privacy computing problems can theoretically be solved with common secure computing protocols (e.g., GMW protocol [1], obfuscated circuit [2]).But these generic schemes require high computing and communication costs. Therefore, for specific secure computing problems, we usually use dedicated efficient protocols. Specifically, for solving the PSI-CA/PSU-CA problem, from the accuracy of the output results to classify, we can be divided into the following two categories.

1. The first type of protocol is the perfect computation protocol, which outputs accurate results. Taking the work in literature [3–8] as an example, it uses the evaluation method of fuzzy polynomials, selects a polynomial to represent the input set, and combines the intersection of the evaluation set using homomorphic encryption technology.
2. The second type of protocol is the imperfect calculation protocol, and the output result of the protocol [9,10] allows a certain amount of error. When faced with a small amount of data, it is often difficult to strike a good balance between efficiency and availability with this type of protocol, and it is often abandoned because of the large errors.

**Our contribution:**

Although imperfect computing protocols have good applications in many application scenarios, imperfect protocols tend to perform particularly poorly for certain small data sets, because in the case of small data sets, imperfect computing protocols may lead to consistent matching errors at a certain probability, so as to completely affect the final results. The efficiency gap between an imperfect protocol and a perfect protocol is not particularly obvious, but the accuracy is obviously different. Therefore, this paper constructs a class of perfect computing protocol for small data sets. Our contributions are as follows:

1. We propose a new privacy protection intersection cardinality protocol, and this protocol can quickly obtain the union of both sides.
2. This protocol has extremely strong compatibility for the intersection of two sets of elements, it can accept any type of data, without knowing any information from either side, and can be 100% accurate in calculating the intersection base of both privacy sets.
3. The protocol only needs two rounds of communication to complete, and the efficiency in the offline phase is much better than for the polynomial intersection cardinality protocol.

This paper is divided into five parts, of which the first part is the introduction part, which mainly introduces the background and development prospects of the paper, as well as the contribution made by this paper. The second part introduces some preparatory knowledge for this article, including giving a security definition, and some important knowledge for this article. The third part is mainly the protocol design, which is divided into two stages, which are the offline stage and online stage. The fourth part is an efficiency analysis, which explains the contribution of this paper using qualitative and quantitative analysis. The fifth part is a summary and prospects.

## 2. Materials and Methods

### *2.1. Security Definition*

Semi-honest ideal reality model: Executing the protocol under security parameter $\kappa$, each party $P_i$ will honestly execute the agreement using their own private input $x_i$. Let

$V_i$ be the final perspective of participant $P_i$ and let $y_i$ be the final output of the player: $\text{Real}_\pi(\kappa, \mathcal{C}; x_1, \ldots, x_n)$ input $P_i$

$$Ideal_{\mathcal{F},\text{Sim}}(\kappa, \mathcal{C}; x_1, \ldots, x_n): \text{ compute } (y_1, \ldots, y_n) \leftarrow \mathcal{F}(x_1, \ldots, x_n)$$
$$\text{Input Sim}(\mathcal{C}, \{(x_i, y_i)|i \in \mathcal{C}\}), (y_1, \ldots, y_n)$$

If the perspective of the attacker in the ideal world is indistinguishable from the perspective of the attacker in the real world, then the protocol is safe from a semi-honest attacker.

**Definition 1.** *Given the protocol $\pi$, if there exists an emulator Sim, such that for all subsets of the compromised participant set $\mathcal{C}$, for all inputs $x_1, \ldots, x_n$, the probability distribution*

$$\text{Real}_\pi(\kappa, \mathcal{C}; x_1, \ldots, x_n) \text{ and } \text{Ideal}_{\mathcal{F},\text{Sim}}(\kappa, \mathcal{C}; x_1, \ldots, x_n)$$

*is (under $\kappa$) indistinguishable, then the protocol is safely implemented $\mathcal{F}$ in the presence of a semi-honest attacker.*

### *2.2. Paillier Homomorphic Encryption System*

Paillier homomorphic encryption [11]: Paillier homomorphic encryption is a public key encryption method that satisfies the addition of homomorphism, and the scheme has been proved secure, specifically described as follows:

Key generation: Given the security parameter $\kappa$, generate two $\kappa$ primes $p, q$ and $p \neq q$ (this property guarantees that two primes have the same length), and compute $N = pq$, $\lambda = lcm(p-1, q-1)$.

Key generation: Given safety parameter $\kappa$, generate two prime numbers $p, q$ and $p \neq q$ that are particularly specifically large $\kappa$ (this property ensures that two prime numbers have the same length), and compute $N = pq$, $\lambda = lcm(p-1, q-1)$. In defining the Fractional Division Functions $L(x) = \frac{x-1}{N}$, select a positive integer $g \in Z_N^*$(such as : $g = n+1$), making $\gcd(L(g^\lambda \bmod N^2), N) = 1$. Then, the public key of the system is pk = $(g, N)$, and the private key is sk = $\lambda$. Paillier's plaintext space and ciphertext space are $Z_N$ and $Z_{N^2}^*$. In the following text, the encryption algorithm and decryption algorithm are denoted as $E$ and $D$.

Encryption process: To encrypt plaintext $m \in Z_N$, select the random number $r \in Z_N^*$, and calculate the ciphertext:

$$C = g^m r^N \bmod N^2$$

Decryption process: For the ciphertext $C \in Z_{N^2}^*$, calculate:

$$m = \frac{L(c^\lambda \bmod N^2)}{L(g^\lambda \bmod N^2)} \bmod N$$

Additive homomorphism:

$$\begin{aligned} E(m_1) \times E(m_2) &= g^{m_1} r_1^N g^{m_2} r_2^N \bmod N^2 \\ &= g^{m_1+m_2} (r_1 r_2)^N \bmod N^2 \\ &= E(m_1 + m_2) \bmod N^2 \end{aligned}$$

Scalar multiplication: $E(m_1)^{m_2} \bmod N^2 = E(m_1 m_2)$.

### *2.3. Learning Framework Based on Privacy Protection*

There is a lot of work in the process of privacy protection to construct a comprehensive and open privacy protection learning framework, among which the most famous framework of MPC mainly has two categories: one is the confusion circuit scheme proposed by the Mr. PSI protocol, which is a part of the above two protocols, such as in [12–15], etc. It is a privacy protection protocol constructed using a general framework. Although the

general-purpose framework is not as efficient as the dedicated protocol, it still has great advantages in terms of operational stability. For the current research on secure multi-party computing frameworks, see Table 1.

**Table 1.** Related mixed-protocol MPC frameworks with N parties, threshold t, and active (●) or passive (○) security.

| Framework | N | t | Security | Protocols | License |
|---|---|---|---|---|---|
| ABY [16] | 2 | 1 | ○ | A/B/Y | Lgpl-3.0 |
| PrivC [17] | 2 | 1 | ○ | A/B | -- |
| ABY3 [18] | 3 | 1 | ● or ○ | A/B/Y | No license |
| Sharemind [19] | 3 | 1 | ● or ○ | A/B | Payware3 |
| Trident [20] | 4 | 1 | ● | A/B/Y | |
| MP-SPDZ [21] | ≥2 | N − 1 | ● or ○ | A/B or Y | MIT-like |
| MOTION [22] | ≥2 | N − 1 | ○ | A/B/Y | MIT |

### *2.4. Computer Coding*

Computer coding refers to the mapping of plaintext information to ciphertext information, which is simply based on a known code, according to certain rules, and is converted into a string of numbers such as 0 and 1. Such coding technology is to facilitate computers recognizing the corresponding information because computer language is not interlinked with our human language. In order to enable computers to better recognize our language, people have formulated a set of rules, and the rules are combined with fixed lengths to represent numbers and characters. Thus was formed the earliest ASCII encoding rules (American Standard Code for Information Interchange). With the popularization of computing, computer coding schemes have also experienced development from localized coding to international coding, and finally formed Unicode's unified coding scheme.

## 3. Protocol Process and Proof of Security

**Problem description**: The two-party PSU/PSI-CA problem is an extension of the two-party PSI problem, which requires the final calculation of $|X \cup Y|/|X \cap Y|$ without revealing any other information (including any element information on oneself and the size of one's own set). This problem corresponds to many privacy computing scenarios in reality: for example, in social networks, two users can compare the proportion of their identical friends without disclosing their specific friend information to calculate social relationship overlap. In the field of health, customers holding private genetic data can confidently interact with public risk gene databases, thereby knowing their probability of contracting a certain disease. This agreement aims to discuss and solve the PSU-CA/PSI-CA problem.

Scheme idea: Based on the above problem description, we know that the application of this protocol may face situations where the number of sets is not equal and the security requirements are very high. In such cases, we pay more attention to the non equilibrium of sets and the corresponding security. Following this idea, we propose using appropriate encoding protocols and semi-homomorphic encryption methods to solve our real-world problems. By using the appropriate data encoding protocols, complex data can be transformed into binary data that are convenient for computation using internationally recognized computer encoding protocols. The detailed operation of the protocol in this paper is shown in Figure 1.

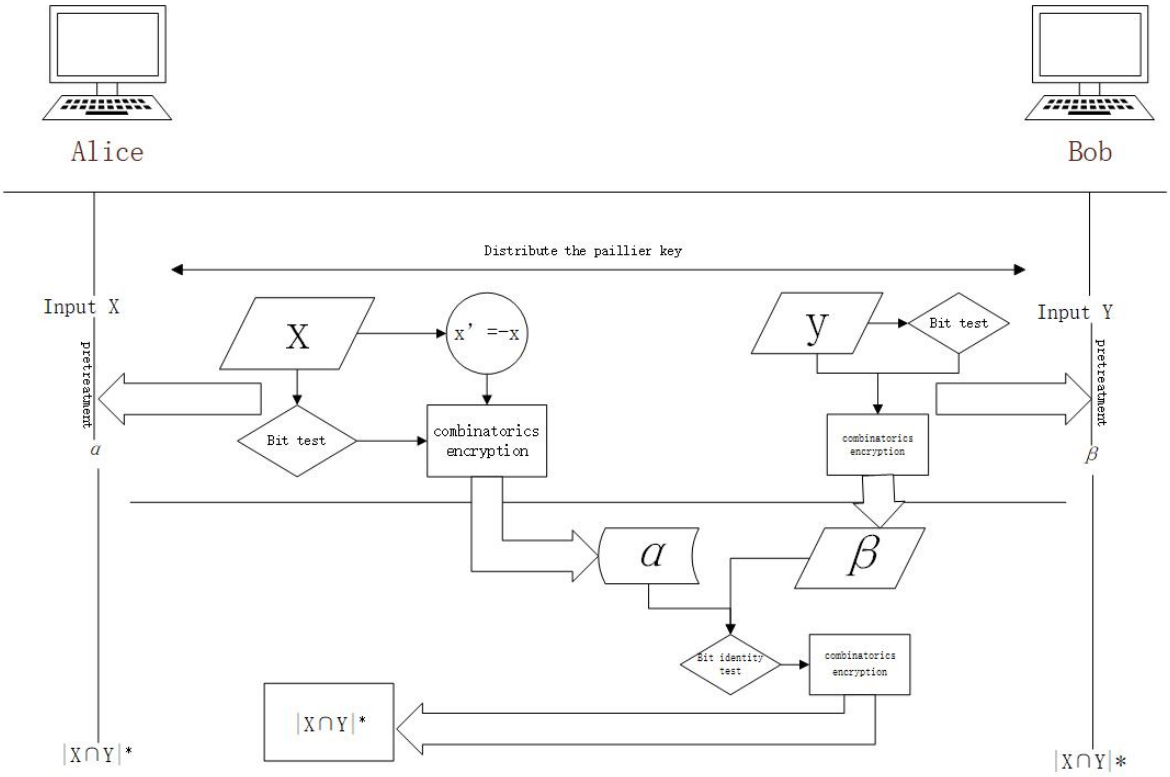


**Figure 1.** Protocol flow chart schemes.

The PSI-CA Construction of This Article

In this section, we will provide protocol construction and related proofs for solving the PSU-CA and PSI-CA problems. Firstly, we provide two protocol constructions (Section 3.1) to solve the PSU-CA and PSI-CA problems, and then provide their correctness proofs (Section 3.2).

### *3.1. Protocol Construction*

In the execution of the two protocols, we may consider setting the participants as Alice and Bob, each holding a set of $X$ and $Y$. Our agreement requires both parties to input elements composed of the same set of codes, and the generation process is as follows:

1. Alice and Bob encode their elements according to the same encoding rules, converting the original data into binary data. Please refer to Section 2.4 for the specific conversion methods.
2. Alice and Bob execute an online interaction protocol, ultimately obtaining $|X \cap Y|^*$.

#### 3.1.1. Offline Phase

Alice, as the sender, and Bob, as the receiver, calls the method shown in Section 2.4 for encoding, maps all Alice's data into binary data, and calculates the number of bits corresponding to the binary data. Offline operation is shown in Figure 2, online operation is shown in Figure 3.

pretreament

| Alice | Bob |
| --- | --- |
| $pk = (g, N), sk = \lambda$ | |
| | $\xrightarrow{pk}$ |
| $x_i$ of bite $\lambda_i$ | $y_j$ of bite $\pi_j$ |
| compute $E(-x_i)$ | compute $E(y_j)$ |
| compute $\alpha_i = [\lambda_i, E(-x_i)]$ | compute $\beta_j = [\pi_j, E(y_j)]$ |
| $\alpha = []$ | $\beta = []$ |
| for *i* in range(0,m): | for *j* in range(0,m): |
| $\alpha.append(\alpha_i)$ | $\beta.append(\beta_j)$ |

**Figure 2.** Offline framework.

The meaning of the above image:
Alice
Calculate the number of bites $x_i$ for $\lambda_i$
Encrypt $x_i$ to obtain $E(-x_i)$
Store $\lambda_i$ and $E(-x_i)$ in the list to obtain $\alpha_i = [\lambda_i, E(-x_i)]$
Putting all $\alpha_i$ into a list gives the set $\alpha = [\alpha_1, \alpha_2, \ldots, \alpha_m]$
Bob
Compute $y_j$' bite $\pi_j$
Calculate the number of bites $y_j$ for $\pi_j$
Store $\pi_j$ and $E(y_j)$ in the list to obtain $\beta_i = [\pi_j, E(y_j)]$
Putting all $\beta_j$ into a list yields the list $\beta = [\beta_1, \beta_2, \ldots, \beta_n]$

3.1.2. Interaction Phase

PSI-CA protocol

Input: X,Y

Output: $|X \cup Y|^*, |X \cap Y|^*$

1. Alice sends $\alpha$ to Bob, who judges whether $\lambda_i$ and $\pi_j$ are equal. If they are equal, the corresponding random number $r_t$ is generated, and the calculation is as follows: $v_t = \{\alpha[i][1] + \beta[j][1]\}^{r_t}$ (when $\lambda_i = \pi_j$). $v_t$ is stored in collection V. After all the calculations are complete, Bob sends V to Alice.
2. Alice decrypts all the elements in V and calculates the number of zeros $\omega$. After decrypting them, Alice sends the number of zeros to Bob.

From this, both sides obtain the number of intersects in the set $|X \cap Y|^* = \omega$, and the number of unions of both sides can be obtained using the calculation $|X \cup Y|^* = n + m - \omega$.

**Figure 3.** PSI-CA protocol.

### 3.2. Correctness of the Protocol

According to the protocol, for each element $x_i$ of Alice, the result of encryption with the Paillier system shown in Section 2.2 is:

$$A_i = E(-x_i) = g^{-x_i} {r_i}^N \mathrm{mod} N^2$$

Alice then sends the encrypted data, along with the corresponding data bits $\lambda_i$, to Bob.

Bob encrypts $y_j$ using the public key given by Alice (as shown in the encryption process in Section 2.2 above) and obtains

$$B_j = E(y_j) = g^{y_j} {r_j}^N \mathrm{mod} N^2$$

If $\lambda_i \neq \pi_j$, that means that the number of bits in the same encoding is different: it means that the two data must not be the same and must not be common elements of the two sets.

If $\lambda_i = \pi_j$, the two elements are likely to be the same, so then $A_i$ and $B_j$ are added homomorphically (as shown in Section 2.2), i.e.,

$${v_i}' = A_i \times B_j = E(-x_i) \times E(y_j) = E(-x_i + y_j)$$

Bob randomly picks a random number $r_s \in N^*$ for calculation:

$$v_t = ({v_t}')^{r_s} = g^{r_s \times (-x_i + y_j)} (r_i r_j)^{r_s N} \mathrm{mod} N^2$$

And Bob sends V to Alice, who decrypts it and obtains $E({x_i}') = g^{{x_i}'} {r_i}^N \mathrm{mod} N^2$

$$D({v_t}') = \frac{L((v_t' \mathrm{mod} N^2)}{L(g^\lambda \mathrm{mod} N^2)} \mathrm{mod} N = r_s(-x_i + y_j)$$

If $x_i$ and $y_j$ are equal, then $D({v_i}') = 0$.

If $x_i$ and $y_j$ are not equal, then $D({v_t}') \neq 0$.

Therefore, Alice only needs to calculate the value equal to zero in V, that is, the number of the same elements on both sides, and Alice obtains the base number of the intersection of the intersection of the two sides, so the agreement is correct.

### 3.3. Protocol Security

**Theorem 1.** *The privacy intersection cardinality protocol PSI-CA is secure.*

**Proof of Theorem 1.** Under a semi-honest model, this theorem is proved by constructing the simulators $S_1$ and $S_2$ to make Equations (1) and (2) hold, in the protocol PSI-CA

$$view_1^\pi(X, Y) = \{X, \lambda_i, r_i, E(W), f_1(X, Y)\} \quad (1)$$

$$view_2^\pi(X, Y) = \{Y, \pi_j, r_j, r_t, E(A), f_2(X, Y)\} \quad (2)$$

where $X$ and $Y$ are the input from Alice and Bob, $\lambda_i$ is the bite number of $x_i$, $r_i$ is the random number chosen by Alice during encryption, $\pi_j$ is the bite number of $y_j$, and $r_j$ and $r_t$ are the random numbers chosen by Bob after different encryption operations, where $E(A)$ refers to the ciphertext information sent by Alice to Bob. We also have the ciphertext message that Bob sends to Alice, while $f_1(X, Y)$ *and* $f_2(X, Y)$ is the output received by Alice and Bob, respectively.

Firstly, simulator $S_1$ is constructed to simulate $view_1^\pi(X, Y)$; the $S_1$ simulation process is as follows:

1. Accept input $(X, f_1(X, Y))$; based on the values of $f_1(X, Y)$, select set $Y' = \{{y_1}', {y_2}', \ldots, {y_n}'\}$, $f_1(X, Y') = f_1(X, Y)$, and let $X' = \{-x_1, -x_2, \ldots, -x_m\}$.

2. The $S_1$ encryption set $X'$ obtains $E(X') = \{E(-x_1), E(-x_2), \ldots, E(-x_m)\}$ and calculates $v_t{}' = (E(-x_i) \times E(y_j{}'))^{r_s} \bmod N^2$.
3. Where $S_1$ gives the element of the encrypted set $E(W)$ as $v_t{}'(t \in \{1, 2, \ldots, n \times m\})$, to decrypt it, by calculating the number of zero elements, you can judge the number of intersections of the two sides, and obtain the corresponding result. In protocol execution, $view_1^{\pi}(X, Y) = \{X, r_i, E(W), f_1(X, Y)\}$:

$$S_1(X, f_1(X, Y)) = \{X, r_i, E(W'), f_1(X, Y')\}$$

Because $E(W) = \{v_1{}', \ldots, v_2{}', \ldots, v_t{}', \ldots, v_{n\times m}{}'\}(t \in \{1, 2, \ldots, n \times m\})$ is created by Bob based on the $E(X')$ sent by Alice, the $E(Y)$ to which the individual belongs, and the random number $r_s$, although Alice has a private key for decryption, she can only know that the decrypted information is composed of (0,random number). It is impossible to know which ciphertext can be decrypted to obtain 0 or a random number. So, we have $E(W) \stackrel{c}{\equiv} E(W')$, and because $f_1(X, Y) = f_1(X, Y')$, so we have:

$$\{S_1(X, f_1(X, Y))\}_{X.Y} \stackrel{c}{\equiv} \{view_1^{\pi}(X, Y)\}_{X,Y}$$

Secondly, the simulator $S_2$ is constructed to simulate $view_2^{\pi}(X, Y)$. The simulation process of $S_2$ is as follows:

1. Accept input $(Y, f_2(X, Y))$, according to the values of $f_2(X, Y)$, select the set $X' = \{x_1{}', x_2{}', \ldots, x_n{}'\}$, $f_2(X', Y) = f_2(X, Y)$, and let $X^{\alpha} = \{-x_1{}', -x_2{}', \ldots, -x_m{}'\}$.
2. The $S_2$ encrypted set $X^{\alpha}$ gains $E(X^{\alpha}) = \{E(-x_1{}'), E(-x_2{}'), \ldots, E(-x_m{}')\}$ and calculates $v_t{}' = (E(-x_i{}') \times E(y_j{}'))^{r_s} \bmod N^2$.
3. $S_2$ obtains the encryption set $E(A) = E(X^{\alpha})$, $S_2$ decrypts it, and the corresponding result can be obtained by calculating the number of zero elements and judging the number of intersections between the two sides. In the execution of the agreement, $view_2^{\pi}(X, Y) = \{X, r_j, r_s, E(W), f_1(X, Y)\}$, while

$$S_2(X, f_2(X, Y)) = \{X, r_j, r_s, E(A'), f_2(X', Y)\}$$

Since $E(A)$ is encrypted by Alice and Bob has no private key, according to the semantic security of the encryption algorithm, for Bob, $E(A) \stackrel{c}{\equiv} E(A')$. While Bob obtains $\lambda_i$ for the data bit sent by Alice, the probability that Bob can infer Alice's data is $\frac{1}{2^{\lambda_i}}$, and Bob cannot infer the real data by other means. Further, because $f_2(X, Y) = f_2(X', Y)$, hence, $\{S_2(X, f_2(X, Y))\}_{X.Y} \stackrel{c}{\equiv} \{view_2^{\pi}(X, Y)\}_{X,Y}$.

Therefore, the protocol is secure. □

## 4. Discussion

In this section, we will conduct theoretical analysis and specific experiments to compare our protocol with the protocol [23] according to different indicators, in order to demonstrate that our protocol has a good overall performance and is suitable for a wider range of application scenarios.

### *4.1. Theoretical Evaluation*

Table 2 presents a qualitative performance comparison, where m and n represent the sizes of the two sets, respectively. However, according to our research, in fact, n in the [23] protocol depends on the item with the highest number of elements in the two sets.

**Table 2.** Performance analysis.

| Protocol | Time Complexity | Space Complexity | Rounds |
|---|---|---|---|
| The text's | $O(n) \sim O(mn)$ | $O(n) \sim O(mn)$ | 2 |
| Reference [23]'s protocol | $O(n^2)$ | $O(n^2)$ | 6 |

In literature [23], Alice sends Bob the encryption polynomial $E_{pk}(f)$, which is one round; Bob sends Alice the encryption polynomial $E_{pk}(g)$, which is one round; Alice sends Bob the cryptographic $E_{pk}(\varphi_1)$, in which $\varphi_1 = f \times r_1 + g \times r_2$; and Bob sends Alice the encryption polynomial $E_{pk}(\varphi)$, where $\varphi = \varphi_1 + \varphi_2$, $\varphi_2 = f \times s_1 + g \times s_2$, which is one round. Because threshold decryption is used and each participant receives $f \times r + g \times s$, at the end of the protocol, it can be assumed that Alice and Bob send part of their private keys to each other for two rounds. So, the total communication/discussion involves six rounds. More importantly, all participants in the above model finally obtain $f \times r + g \times s$, and the degree of this polynomial is $\max\{|X|, |Y|\}$. Moreover, according to the most advanced complexity-solving polynomial methods, its complexity is difficult to decrease rapidly.

Newton's iterative method: usually has linear convergence and a complexity of about $O(n^2) \sim O(n^3)$, where n is the order of the polynomial.

Dichotomy: has convergence and its complexity is about $O(n \log(M))$, where $n$ is the order of the polynomial and $M$ is the range of values of the polynomial roots.

The Durand–Kerner method is convergent and has a complexity of about $O(kn)$, where $k$ is the number of iterations and $n$ is the order of the polynomial.

Baistow method: usually has quadratic convergence and the complexity is about $O(n^2) \sim O(n^3)$, where n is the order of the polynomial.

And the above method will increase the complexity as the degree of the polynomial increases.

For the protocol in this article, Alice sends Bob encrypted data for one round, and Bob sent Alice encrypted data for one round. So, the total number of communication rounds is two. Moreover, since we determine and solve based on the number of bits of information the data from both parties, this greatly improves the efficiency of encryption and decryption. Using this protocol, the complexity that may be obtained based on different data may not be the same. Although our protocol's efficiency is not currently the highest known, it is optimal for achieving accurate cardinality testing.

### *4.2. Experimental Evaluation*

We have implemented the above protocols separately to compare their specific performance. Both protocols were implemented using the Python language, and the testing platform was equipped with a Core (TM) i7-8750H CPU@2.20 GHz 2.21 GHz Model processor (Intel Corporation, Santa Clara, CA, USA) and 16 GB 1867 MHz DDR3 memory. This test was completed in a LAN network environment with low network latency. For the following two protocols, we implemented them using the Python language. Our implementation is divided into two stages: one is the online communication stage and the other is the offline operation stage. For the above two protocols, we have uniformly ignored the process of generating and interacting with the sender's key, and the length of our key is set to 3072 bits, which can fully ensure our information security. In the offline stage [23], the offline stage mainly deals with polynomial roots and the corresponding encryption and decryption operations. Our protocol mainly handles operations such as encryption, decryption, and scalar multiplication encryption during the offline phase. In the online stage, Ref. [23] mainly obtains encrypted polynomials through the interaction process between both parties and can obtain intersections. However, our protocol mainly matches by specific identifiers, finds approximately identical terms, and sends them to the other party. The specific performance is shown in Table 3.

**Table 3.** Comparison of online time and offline time of different protocols.

| (*X*,*Y*) | Protocol | Online Time (s) | Offline Time (s) | Total Time |
|---|---|---|---|---|
| (100,100) | Text's protocol | 83.39 | 166.78 | 250.17 |
| | Reference [23]'s protocol | 83.33 | 10,021.83 | 10,106.16 |
| (100,300) | Text's protocol | 88.64 | 265.92 | 354.56 |
| | Reference [23]'s protocol | 265.92 | 90,177.28 | 90,265.92 |
| (100,500) | Text's protocol | 73.82 | 295.28 | 369.10 |
| | Reference [23]'s protocol | 369.10 | 250,221.46 | 250,590.56 |

After analyzing the above table, we found that the effect of our protocol in the offline stage is far superior to that of [23]. In order to see the specific changes in the two protocols more directly, we specially drew Figure 4, in which we used the red line to represent the offline calculation stage of our protocol, and the blue line to describe the offline calculation stage [23]. In this experiment, the number of elements in $X$ set is kept constant at 50, and the number of elements in $Y$ set is 1 to 10 times that of $X$, so the $X$-axis represents the ratio of the number of elements in the $X$ and $Y$ sets, and the $Y$-axis represents the corresponding running time. It is obvious from Figure 4 that our protocol efficiency is relatively efficient.

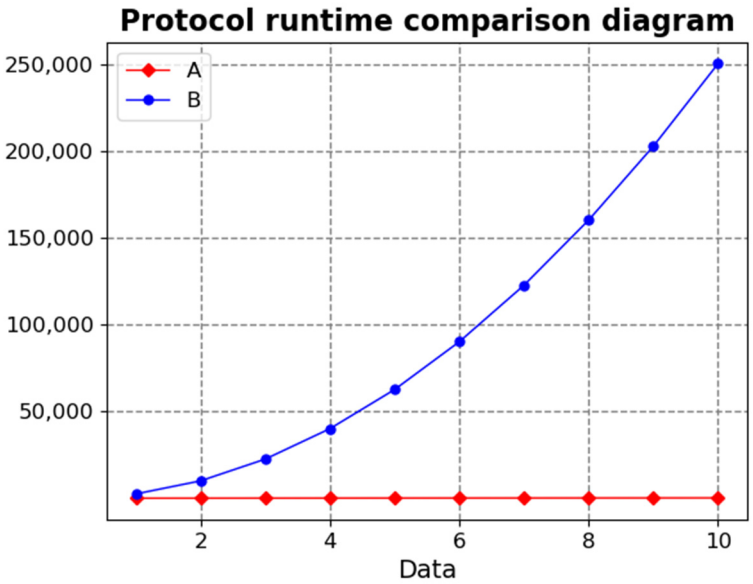


**Figure 4.** Protocol runtime comparison diagram.

Our protocol has been tested and it was found that when our protocol is faced with an imbalance of two elements, the effect of offline computing is far better than the equilibrium situation. Moreover, after some modifications to our protocol, we can quickly find the intersection elements. We only need to add a new guide to Alice's element in the pre-processing stage, and we can change to a new encryption, which can only be seen by ourselves, and send the element to Bob. When Bob calculates the element, it is only necessary to form a list with the guide to participate in the operation, and send it to Alice. Then, Alice calculates the 0 element, and can find the corresponding set intersection element through the corresponding guide of the 0 element.

## 5. Conclusions

This article proposes a new perfect computing protocol to solve the PSI-CA/PSU-CA problem and proves its security in a semi-honest model. It uses Paillier semi-homomorphic encryption technology, and compared with the most advanced protocol A, this protocol has fewer constant rounds of communication and a lower computational complexity in

the offline stage, and has a wide range of application scenarios. In addition, the protocol has more room for optimization in the future, and the computing efficiency of the protocol should be improved on the premise of ensuring correctness. Can the protocol be combined with other advanced technologies to improve the computing efficiency?

**Author Contributions:** Methodology, T.M.; software, T.M., P.Z. and Q.Z.; validation, P.Z. and Q.Z.; writing—original draft preparation, T.M. and J.Z.; writing—review and editing, P.Z.; visualization, J.W.; funding acquisition, T.M. and J.W. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was supported by the Major Science and Technology Projects of Longmen Laboratory (No. 231100220300), the National Natural Science Foundation of China (No. 62102134), the Key Scientific Research Project in Colleges and Universities of Henan Province of China (No. 21A510003, 23A520046, and 23A413005) and the Key Science and Technology Project of Henan Province of China (No. 222102210053, 232102210130, and 232102210138).

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** The data presented in this study are available on request from the corresponding author. The data are not publicly available due to privacy.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References

1. Micali, S.; Goldreich, O.; Wigderson, A. How to play any mental game. In Proceedings of the Nineteenth ACM symposium on Theory of Computing, STOC, New York, NY, USA, 1 January 1987; ACM: New York, NY, USA, 1987; pp. 218–229.
2. Yao, A.C. Protocols for secure computations. In Proceedings of the 23rd Annual Symposium on Foundations of Computer Science (sfcs 1982), Washington, DC, USA, 3–5 November 1982; pp. 160–164.
3. Aggarwal, G.; Mishra, N.; Pinkas, B. Secure computation of the *k*th-ranked element. In *EUROCRYPT'04, LNCS*; Springer: Berlin/Heidelberg, Germany, 2004; Volume 3027, pp. 40–55.
4. Kiayias, A.; Mitrofanova, A. Testing disjointness of private datasets. In Proceedings of the International Conference on Financial Cryptography and Data Security, Roseau, Dominica, 28 February–3 March 2005; Springer: Berlin/Heidelberg, Germany, 2005; pp. 109–124.
5. Sang, Y.; Shen, H. Efficient and secure protocols for privacy-preserving set operations. *ACM Trans. Inf. Syst. Secur. TISSEC* **2009**, *13*, 1–35. [CrossRef]
6. Hohenberger, S.; Weis, S.A. Honest-verifier private disjointness testing without random oracles. In Proceedings of the Privacy Enhancing Technologies: 6th International Workshop, PET 2006, Cambridge, UK, 28–30 June 2006; Revised Selected Papers 6; Springer: Berlin/Heidelberg, Germany, 2006; pp. 277–294.
7. Frikken, K. Privacy-preserving set union. In Proceedings of the Applied Cryptography and Network Security: 5th International Conference, ACNS 2007, Zhuhai, China, 5–8 June 2007; Proceedings 5; Springer: Berlin/Heidelberg, Germany, 2007; pp. 237–252.
8. Hazay, C.; Nissim, K. Efficient set operations in the presence of malicious adversaries. In Proceedings of the Public Key Cryptography–PKC 2010, 13th International Conference on Practice and Theory in Public Key Cryptography, Paris, France, 26–28 May 2010; Proceedings 13; Springer: Berlin/Heidelberg, Germany, 2010; pp. 312–331.
9. Egert, R.; Fischlin, M.; Gens, D.; Jacob, S.; Senker, M.; Tillmanns, J. Privately computing set-union and set-intersection cardinality via bloom filters. In Proceedings of the Information Security and Privacy: 20th Australasian Conference, ACISP 2015, Brisbane, QLD, Australia, 29 June–1 July 2015; Proceedings 20; Springer International Publishing: Cham, Switzerland, 2015; pp. 413–430.
10. Dong, C.; Loukides, G. Approximating Private Set Union/Intersection Cardinity with Logarithmic Complexity. *IEEE Trans. Inf. Forensics Secur.* **2017**, *12*, 2792–2806. [CrossRef]
11. 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, Prague, Czech Republic, 2–6 May 1999; Springer: Berlin/Heidelberg, Germany, 1999; pp. 223–238.
12. Pinkas, B.; Schneider, T.; Tkachenko, O.; Yanai, A. Efficient circuit-based PSI with linear communication. In *Advances in Cryptology–EUROCRYPT 2019, Proceedings of the 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, 19–23 May 2019*; Proceedings, Part III 38; Springer International Publishing: Cham, Switzerland, 2019; pp. 122–153.
13. Pinkas, B.; Schneider, T.; Weinert, C.; Wieder, U. Efficient circuit-based PSI via cuckoo hashing. In Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, 29 April–3 May 2018; Springer International Publishing: Cham, Switzerland, 2018; pp. 125–157.

14. Huang, Y.; Evans, D.; Katz, J. Private set intersection: Are garbled circuits better than custom protocols? In Proceedings of the 19th Network and Distributed Security Symposium, San Diego, CA, USA, 5–8 February 2012.
15. Asokan, N.; Dmitrienko, A.; Nagy, M.; Reshetova, E.; Sadeghi, A.R.; Schneider, T.; Stelle, S. Crowdshare: Secure mobile resource sharing. In Proceedings of the Applied Cryptography and Network Security: 11th International Conference, ACNS 2013, Banff, AB, Canada, 25–28 June 2013; Proceedings 11; Springer: Berlin/Heidelberg, Germany, 2013; pp. 432–440.
16. Bogdanov, D.; Laur, S.; Willemson, J. Sharemind: A framework for fast privacy-preserving computations. In Proceedings of the Computer Security-ESORICS 2008, 13th European Symposium on Research in Computer Security, Málaga, Spain, 6–8 October 2008; Proceedings 13; Springer: Berlin/Heidelberg, Germany, 2008; pp. 192–206.
17. Demmler, D.; Schneider, T.; Zohner, M. ABY-A framework for efficient mixed-protocol secure two-party computation. In Proceedings of the Network and Distributed System Security (NDSS) Symposium, San Diego, CA, USA, 8–11 February 2015.
18. Hazay, C.; Scholl, P.; Soria-Vazquez, E. Low cost constant round MPC combining BMR and oblivious transfer. *J. Cryptol.* **2020**, *33*, 1732–1786. [CrossRef]
19. Keller, M. MP-SPDZ: A versatile framework for multi-party computation. In Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security, Virtual, 9–13 November 2020; pp. 1575–1590.
20. Mohassel, P.; Rindal, P. ABY3: A mixed protocol framework for machine learning. In Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, Toronto, ON, Canada, 15–19 October 2018; pp. 35–52.
21. Chaudhari, H.; Rachuri, R.; Suresh, A. Trident: Efficient 4pc framework for privacy preserving machine learning. *arXiv* **2019**, arXiv:1912.02631.
22. Braun, L.; Demmler, D.; Schneider, T.; Tkachenko, O. Motion—A framework for mixed-protocol multi-party computation. *ACM Trans. Priv. Secur.* **2022**, *25*, 1–35. [CrossRef]
23. Kissner, L.; Song, D. Privacy-Preserving Set Operations. In *Advances in Cryptology, Proceedings of the Annual International Cryptology Conference CRYPTO 2005, Santa Barbara, CA, USA, 14–18 August 2005*; Lecture Notes in Computer, Science; Shoup, V., Ed.; Springer: Berlin/Heidelberg, Germany, 2005; Volume 3621. [CrossRef]