Efficient Protocols for Privacy Preserving Matching Against Distributed Datasets

Yingpeng Sang[1], Hong Shen[2], Yasuo Tan[1], and Naixue Xiong[1]

[1] School of Information Science, Japan Advanced Institute of Science and Technology
Asahidai, Nomi, Ishikawa, Japan, 923-1211
{yingpeng, ytan, naixue}@jaist.ac.jp
[2] School of Computer Science, The University of Adelaide
SA 5005, Australia
hong.shen@adelaide.edu.au

Abstract. When datasets are distributed on different sources, finding out matched data while preserving the privacy of the datasets is a widely required task. In this paper, we address two matching problems against the private datasets on N ($N \geq 2$) parties. The first one is the Privacy Preserving Set Intersection (PPSI) problem, in which each party wants to learn the intersection of the N private datasets. The second one is the Privacy Preserving Set Matching (PPSM) problem, in which each party wants to learn whether its elements can be matched in any private set of the other parties. For the two problems we propose efficient protocols based on a threshold cryptosystem which is additive homomorphic. In a comparison with the related work in [18], the computation and communication costs of our PPSI protocol decrease by 81% and 17% respectively, and the computation and communication costs of our PPSM protocol decrease by 80% and 50% respectively. In practical utilities both of our protocols save computation time and communication bandwidth.

Keywords: cryptographic protocol, privacy preservation, distributed database, set intersection, set matching.

1 Introduction

For datasets distributed on different sources, data matching among these sets is always required to gain useful information. Supermarkets need find out the same card numbers which have consuming records in all of their databases, and then provide better service for the card owners. This is a *set intersection* problem among distributed datasets. The tenderees consider that duplicate submission of tenders is a damage of their benefits, so they want to reject those tenderers who have submitted duplicate tenders to any two of them. Such tenderers can be found out by one tenderee by firstly set intersections between his tender set and each set of the others, then a set union on all the intersections. This is a *set matching* problem among distributed datasets.

Privacy may be a critical concern of the data owners, so they are reluctant to directly publish their datasets. Specifically, one supermarket doesn't want other supermarkets to know the card numbers in its database except those in the intersection. One tenderee A even doesn't like another tenderee B to know that it is him that has a matched tender with B. Therefore, there should be some privacy preserving techniques for them to determine the results of set intersection and matching, without the datasets being directly published. In this paper, we address the two related problems: *privacy preserving set intersection*, and *privacy preserving set matching*. Basically, both of them are solved by efficiently constructing and evaluating polynomials whose roots are elements of the set intersection and matching.

Problem Formulation: Suppose there are N ($N \geq 2$) parties, each party P_i ($i = 1, ..., N$) has a set (or multiset) of inputs of size S: $T_i = \{T(i,j)|j = 1, ..., S\}$. We also assume that all T_i for $i = 1, ..., N$ are subsets of a common set $\mathbb{T}$, and $S \ll |\mathbb{T}|$, such that given two arbitrarily selected subsets T_i and $T_{i'}$, The probability that an input $a \in T_i$ equals any input $a' \in T_{i'}$ is negligible (i.e., $\frac{S}{|\mathbb{T}|} \to 0$). In the following we define our two problems:

1) *Privacy Preserving Set Intersection (PPSI):* All parties want to learn the intersection of their private sets, i.e., $TI = T_1 \cap ... \cap T_N$, without gleaning any information other than those computed from a coalition of parties inputs and outputs.
2) *Privacy Preserving Set Matching (PPSM):* Each party P_i wants to learn whether each element of its can be matched in any set of the other parties, i.e., whether each element $T(i,j) \in \bigcup_{i'=1,...,N, i' \neq i}(T_i \cap T_{i'})$, without gleaning any information other than those computed from a coalition of parties inputs and outputs.

Privacy Requirements: Firstly, in both problems, an honest party shouldn't be subject to the dictionary attack, in which an adversary may defraud the honest party of inputs using the common set $\mathbb{T}$. The dictionary attack can be effectively resisted in assumption of $S \ll |\mathbb{T}|$.

What's more, without colluding with the other parties, an adversary-controlled party P_i shouldn't glean the following information:

1) For PPSI, P_i can't know elements on $P_{i'}$ ($i' = 1, ..., N, i' \neq i$) except TI.
2) For PPSM, if $T(i,j) \in \bigcup_{i'=1,...,N, i' \neq i}(T_i \cap T_{i'})$, P_i can't know the specific matching times, i.e., how many parties $T(i,j)$ has matches on, and the matching originations, i.e., which party $T(i,j)$ has a match on.

If P_i is in a coalition of c ($1 \leq c \leq N-1$) adversary-controlled parties, it may get more information than above. We analyze these information in Section 6 of this paper.

Our Contributions: Our main contributions in this paper include:

1) We propose an efficient PPSI protocol, which has lower computation and communication costs than the PPSI protocols in [18] and [8].
2) To our knowledge there hasn't been a direct solution for PPSM. Though a PPSM solution can be derived from the techniques in [18], we propose a

more efficient protocol in the computation and communication costs than the derived one.

The remainder of the paper is organized as follows: Section 2 discusses some related work. Section 3 lists the necessary preliminaries for our protocols. Section 4 and Section 5 propose the PPSI protocol and PPSM protocol respectively. In Section 6 we analyze the security of the two protocols. In Section 7 we compare our two protocols with the related work considering the computation and communication costs. Section 8 concludes the whole paper.

2 Related Work

PPSI and PPSM are specific problems belonging to the general *Secure Multiparty Computation* (SMC) problem. There have been general solutions for the SMC problem ([12], [24]). In general SMC, the function to be computed is represented by a circuit, and every gate of the circuit is privately evaluated. However, when this general solution is used for a specific problem, the large size of the circuit and high cost of evaluating all gates will result in a much less efficient protocol than the non-private protocol for this problem. Therefore, many efficient private protocols for the specific problems have been proposed based on the specific properties of these problems.

PPSI and PPSM can be traced back to the specific problem of private equality test (PET) in two-party case, where each party has a single element and wants to test whether they are equal without publishing the elements. The problem of PET was considered in [1], [4], [19] and [20]. PET solutions can't be simply used for the multi-party cases of PPSI and PPSM, otherwise too much sensitive information will be leaked, e.g., any two parties will know the intersection of their private sets.

A solution for the multi-party case of PPSI was firstly proposed in [8]. The solution is based on evaluating polynomials representing elements in the sets. In [18], another solution for PPSI was proposed, in which each polynomial representing each set is multiplied by a random polynomial which has the same degree with the former polynomial. In this paper, to get a solution with lower costs than [8] and [18], we multiply each polynomial representing each set by a random polynomial which has a low enough degree without compromising the security of the solution. We also multiply the randomized polynomials by a non-singular matrix to improve the correctness of our solution. We will compare our solution for PPSI with [8] and [18] in details in Section 7.

Though there hasn't been a direct solution for the PPSM problem, it can be considered as computing a function $\bigcup_{i'=1,...,N,i\neq i'}(T_i \cap T_{i'})$ on P_i for $i = 1, ..., N$, and can be solved by the techniques of privacy preserving set intersection and set union in [18]. Thus we can derive a solution from [18], and we name it "*Solution D1*" in this paper. In [18] the way to securely construct Solution D1 wasn't provided. Solution D1 also requires high cost. We will compare our solution for PPSM with Solution D1 in Section 7.

Other related privacy preserving problems, such as set cardinality, set disjointness, threshold set union, etc, can be found in [8], [18], [17] and [14]. They are different problems from PPSI and PPSM, thus need different solutions.

3 Preliminaries

3.1 Adversary Model

Generally speaking there are two types of adversaries in SMC, depending on whether they take active steps to disrupt the execution of the protocol, or merely gather information. The latter adversary is referred to as semi-honest (or passive, honest-but-curious); the former one is referred to as malicious (or active). A semi-honest party is assumed to follow the protocol exactly as what is prescribed by the protocol, except that it keeps a record of all its intermediate computations. A malicious party may arbitrarily deviate from the specified protocol, including refusing to participate in the protocol, substituting their local inputs and aborting the protocol prematurely. For the security in the malicious model, a general compiler is given in [11] to force each party to either effectively behave in a semi-honest manner or be detected as cheating.

In this paper we assume the parties are semi-honest, and they may compose any coalition of c ($1 \leq c \leq N-1$) parties $(P_{i_1}, ..., P_{i_c})$. A multi-party protocol is said to *privately compute* a function $\mathfrak{f}$, if whatever a coalition of semi-honest parties can obtain after participating in the protocol could be essentially obtained from the inputs and outputs of these very parties. By [10] and [11], a formal definition of *privacy with respect to semi-honest behavior* is given in the following:

Definition 1. *Let $\mathfrak{f} : (\{0,1\}^*)^m \rightarrow (\{0,1\}^*)^m$ be an m-ary functionality, where $\mathfrak{f}_i(x_1, ..., x_m)$ is the i-th element of $\mathfrak{f}(x_1, ..., x_m)$. For $I = \{i_1, ..., i_c\} \subseteq \{1, ..., m\}$, $\mathfrak{f}_I(x_1, ..., x_m) = \{\mathfrak{f}_{i_1}(x_1, ..., x_m), ..., \mathfrak{f}_{i_c}(x_1, ..., x_m)\}$. Let Π be an m-party protocol for computing $\mathfrak{f}$. The* **view** *of the i-th party (P_i) after participating in an execution of Π on $\overline{x} = (x_1, ..., x_m)$, denoted $VIEW_i^{\Pi}(\overline{x})$, is $(x_i, r, m_1, ..., m_t)$, where r are the random bits generated by P_i, $m_1, ..., m_t$ is a sequence of message received by P_i. For $I = \{i_1, ..., i_c\}$, we let $VIEW_I^{\Pi}(\overline{x}) = (I, VIEW_{i_1}^{\Pi}(\overline{x}), ..., VIEW_{i_c}^{\Pi}(\overline{x}))$.*

We say that Π **Privately Computes** *$\mathfrak{f}$ if there exists a probabilistic polynomial-time (PPT) algorithm, denoted S, such that for every $I \subseteq \{1, ..., m\}$, it holds that*

$$S(I, (x_{i_1}, ..., x_{i_c}), \mathfrak{f}_I(\overline{x}))_{\overline{x} \in (\{0,1\}^*)^m} \equiv^c VIEW_I^{\Pi}(\overline{x})_{\overline{x} \in (\{0,1\}^*)^m} \quad (1)$$

In the definition above, "$\equiv^c$" denotes *computationally indistinguishable*, which is also called *indistinguishable in polynomial time*. Given two ensembles $X = \{X_w\}_{w \in S'}$ and $Y = \{Y_w\}_{w \in S'}$ (S' is a set of strings), they are indistinguishable in polynomial time if for every PPT algorithm D, every positive polynomial $p(\cdot)$, and all sufficiently long $w \in S'$, $|Pr[D(X_w, w) = 1]| - |Pr[D(Y_w, w) = 1]| < \frac{1}{p(|w|)}$.

3.2 Homomorphic Encryption

Our protocols are based on an additive Homomorphic Encryption (HE) scheme. Let ε be a probabilistic encryption scheme. Let M be the message space and C the ciphertext space such that M is a group under operation $\oplus$ and C is a group under operation $\odot$. ε is a $(\oplus, \odot)$-HE scheme if for any instance $E_R(\cdot)$ of the encryption scheme, given $c_1 = E_{r_1}(m_1)$ and $c_2 = E_{r_2}(m_2)$, there exists an r such that $c_1 \odot c_2 = E_{r_1}(m_1) \odot E_{r_2}(m_2) = E_r(m_1 \oplus m_2)$. ε is *additive* when it's a $(+, \odot)$ scheme, and *multiplicative* when it's a $(*, \odot)$ scheme.

The HE scheme in our protocols is also required to support secure (N, N)-threshold decryption. The corresponding secret key is shared by a group of N parties, and the decryption can't be performed by any single party, unless all parties act together.

Thus, we can use Paillier's cryptosystem ([21]) for its following properties: 1) it's an additive homomorphic encryption scheme. Given two encryptions $E(m_1)$ and $E(m_2)$, $E(m_1 + m_2) = E(m_1) \cdot E(m_2)$; 2) given an encryption $E(m)$ and a scalar a, $E(a \cdot m) = E(m)^a$; 3) (N, N)-threshold decryption can be supported (by [6],[7]). In this paper, $\mathcal{N}$ is the RSA-modulus which is the multiplication of two large prime numbers, and $\mathbb{Z}_{\mathcal{N}}$ is the plaintext space of Paillier's cryptosystem.

3.3 Calculations on Encrypted Polynomials

In our protocols, we need do some calculations on encrypted polynomials. For a polynomial $f(x) = \sum_{i=0}^{m} a_i x^i$, we use $E(f(x))$ to denote the sequence of encrypted coefficients $\{E(a_i) | i = 0, ..., m\}$. Given $E(f(x))$, where $E(\cdot)$ is an additive HE scheme (e.g., Paillier), some computations can be made as follows (which have also been used in [8] and [18]):

1) At a value v, we can evaluate $E(f(x))$: $E(f(v)) = E(a_m v^m + a_{m-1} v^{m-1} + ... + a_0) = E(a_m)^{v^m} E(a_{m-1})^{v^{m-1}} \cdots E(a_0)$.
2) Given $E(f(x))$, we can compute $E(c \cdot f(x)) = \{E(a_m)^c, ..., E(a_0)^c\}$.

Table 1. Major Notations in This Paper

Notation	*Definition*
N	Total number of parties
P_i	The i-th party
T_i	The set or multiset on P_i
S	Total number of elements on each party
$T(i,j)$	The j-th element on P_i, $j = 1, ..., S$
c	Total number of colluded parties, $1 \leq c \leq N - 1$
I	The index set of c colluded parties, $\{i_1, ..., i_c\}$
I'	The index set of honest parties, $\{1, ..., N\} \setminus I$
f_i	The polynomial whose roots are elements in T_i. $f_i = \prod_{j=1}^{S}(x - T(i,j))$
$\mathbb{Z}_{\mathcal{N}}$	The plaintext space of Paillier's cryptosystem

3) Given $E(f(x))$ and $E(g(x))$, $g(x) = \sum_{j=0}^{m} b_j x^j$, we can compute $E(f(x) + g(x)) = \{E(a_m)E(b_m), ..., E(a_0)E(b_0)\}$.
4) Given $f(x)$ and $E(g(x))$, we can compute $E(f(x)*g(x))$. Suppose that $g(x) = \sum_{j=0}^{n} b_j x^j$, $f(x) * g(x) = \sum_{k=0}^{m+n} c_k x^k$, then $E(c_k) = E(a_0 b_k + a_1 b_{k-1} + ... + a_k b_0) = E(b_k)^{a_0} \cdots E(b_0)^{a_k}$. a_i or b_j are treated as zero if $i > m$ or $j > n$.

3.4 Notations

The major notations in this paper are listed in Table 1.

4 Protocol for Privacy Preserving Set Intersection

4.1 Main Idea

Our protocol for PPSI is based on evaluating randomized polynomials representing the intersection, which is a similar way with [8] and [18], but achieves lower cost.

Each P_i can compute a polynomial f_i to represent its set T_i: $f_i = (x - T(i,1)) \cdots (x - T(i,S))$. Then it randomizes f_i to be $f_i * \sum_{j=1}^{N} r_{i,j}$ by the help of other parties, in which $r_{i,j}$ is generated by P_j, $r_{i,j} = a_{i,j}x + b_{i,j}$, $a_{i,j}$ and $b_{i,j}$ are uniformly selected from the plaintext space of the threshold HE scheme (for Paillier's scheme, it's $\mathbb{Z}_{\mathcal{N}}$).

The N parties get a polynomial vector $F = (f_1 * \sum_{j=1}^{N} r_{1,j}, ..., f_N * \sum_{j=1}^{N} r_{N,j})$ and compute $G = FR$, in which R is an $N \times N$ nonsingular matrix whose entries R_{uv} $(1 \le u, v \le N)$ are random numbers. The resulting G is another polynomial vector $(g_1, ..., g_N)$ as following:

$$\begin{aligned} g_1 &= f_1 * \sum_{j=1}^{N} r_{1,j} R_{11} + ... + f_N * \sum_{j=1}^{N} r_{N,j} R_{N1} \\ &... \\ g_N &= f_1 * \sum_{j=1}^{N} r_{1,j} R_{1N} + ... + f_N * \sum_{j=1}^{N} r_{N,j} R_{NN} \end{aligned} \tag{2}$$

Then, each P_i evaluates $(g_1, ..., g_N)$ at the element $T(i,j)$. If for $k = 1, ..., N$ $g_k(T(i,j)) = 0$, then P_i determines $T(i,j) \in TI$. The correctness of this determination will be proved in Lemma 1.

In the computation of G, to protect the privacy of each f_i, f_i is encrypted by P_i, and the encryption of $f_i * \sum_{j=1}^{N} r_{i,j}$ is also computed. Then each party P_i generates a random matrix R_i so that $R = \prod_{i=1}^{N} R_i$ is nonsingular but no one knows what R is without publishing all R_i. The encryptions of FR_1, FR_1R_2, ..., $FR_1 \cdots R_N$ are computed respectively on P_1, P_2, ..., P_N. Finally, the N parties get the encryption of $G = FR$. After decryption, each P_i knows G, but not $f_{i'}$ for $i' \neq i$.

4.2 The Protocol

Protocol 1: *Protocol for Privacy Preserving Set Intersection*

Inputs: There are N ($N \geq 2$) semi-honest parties. Each party has a private set of S elements, denoted T_i. Each party holds the public key and it's own share of the secret key for the threshold Paillier's cryptosystem.

Output: Each party P_i knows $TI = T_1 \cap ... \cap T_N$.

1) Computing $E(F)$: For $i = 1, ..., N$,

1.1) P_i computes $f_i = (x - T(i,1)) \cdots (x - T(i,S))$, encrypts the coefficients to get $E(f_i)$, and sends $E(f_i)$ to all the other $N-1$ parties.

1.2) on each P_j ($j \neq i$), $r_{i,j}$ is generated as $a_{i,j}x + b_{i,j}$, in which $a_{i,j}$ and $b_{i,j}$ are uniformly selected from $\mathbb{Z}_{\mathcal{N}}$. P_j computes $E(f_i * r_{i,j})$ by computation 4) in Section 3.3, and sends it to P_i.

1.3) P_i also generates $r_{i,i}$ and computes $E(f_i * r_{i,i})$. Then P_i computes $E(f_i * \sum_{j=1}^{N} r_{i,j})$ by computation 3) in Section 3.3, and sends it to P_1.

In the end, P_1 gets $E(F)$ in which $F = (f_1 * \sum_{j=1}^{N} r_{1,j}, ..., f_N * \sum_{j=1}^{N} r_{N,j})$.

2) Computing $E(G)$: For $i = 1, ..., N$,

2.1) P_i generates a nonsingular $N \times N$ matrix R_i which is uniformly distributed over $\mathbb{Z}_N$ (by the method in [22]).

2.2) P_i computes $E(FR_1 \cdots R_i)$ according to computation 2) and 3) in Section 3.3, and sends it to P_{i+1} if $i + 1 \leq N$.

In the end, P_N gets $E(G) = E(F \prod_{i=1}^{N} R_i)$ and sends it to all the other parties.

3) Decryption and Evaluation :

3.1) The parties cooperatively decrypt $E(G)$ and gets $G = F(\prod_{i=1}^{N} R_i)$. Let $R = \prod_{i=1}^{N} R_i$, and $R_{u,v}$ ($1 \leq u, v \leq N$) is the (u,v)-th entry of R, G is a polynomial vector $(g_1, ..., g_N)$ as described in the equation 2) of Section 4.1.

3.2) Every P_i evaluates $T(i,j)$ in G for $j = 1, ..., S$ by computation 1) in Section 3.3. If $G(T(i,j)) = (\ g_1(T(i,j)), ..., g_N(T(i,j))\ ) = (0, ..., 0)$, the $T(i,j) \in\ TI$; otherwise, $T(i,j) \notin\ TI$.

We prove the correctness of Protocol 1 in the following lemma:

Lemma 1. *Protocol 1 is a correct protocol for the PPSI problem.*

Proof: Protocol 1 determines whether $T(i,j) \in TI$ by $G(T(i,j))$. If $T(i,j) \in TI$, $T(i,j)$ is a root of all f_i for $i = 1, ..., N$, then $F(T(i,j)) = (f_1(T(i,j)) \sum_{j=1}^{N} r_{1,j}, ..., f_N(T(i,j)) \sum_{j=1}^{N} r_{N,j}) = (0, ..., 0)$, $G(T(i,j)) = F(T(i,j))R = (0, ..., 0)$. That is, if the evaluation $G(T(i,j)) \neq (0, ..., 0)$, $T(i,j) \notin TI$.

Then we prove that if $G(T(i,j)) = (0, ..., 0)$, overwhelmingly $T(i,j) \in TI$. $G = FR_1 \cdots R_N = F(\prod_{i=1}^{N} R_i) = FR$. Because R_i for $i = 1, ..., N$ are generated to be nonsingular, $R = \prod_{i=1}^{N} R_i$ is also nonsingular. If $G(T(i,j)) = (0, ..., 0)$, a linear system $F(T(i,j))R = (0, ..., 0)$ can be made, and it has only one solution: $F(T(i,j)) = (0, ..., 0)$, i.e.,$f_l(T(i,j)) * \sum_{j=1}^{N} r_{l,j}(T(i,j)) = 0$ for $l = 1, ..., N$.

The coefficients of $r_{l,j}$ are uniformly selected from $\mathbb{Z}_{\mathcal{N}}$. Suppose $\sum_{j=1}^{N} r_{l,j} = a_l x + b_l$, a_l and b_l are also uniformly distributed over $\mathbb{Z}_{\mathcal{N}}$. The probability that any $T(i,j) \in \mathbb{Z}_{\mathcal{N}}$ is a root of $a_l x + b_l$ is $1/\mathcal{N}$. If $\exists T(i,j)$, $\forall l \in \{1, ..., N\}$, $f_l(T(i,j)) * \sum_{j=1}^{N} r_{l,j}(T(i,j)) = 0$, because $f_l(T(i,j))$ must be 0 when $l = i$, so the probability that $\forall l$ $(l \neq i)$ $f_l(T(i,j)) = 0$ is $p = (1 - 1/\mathcal{N})^{N-1}$. N is the number of parties and practically $N \ll \mathcal{N}$. When $\mathcal{N}$ is large enough, $p \to 1$, then overwhelmingly $T(i,j)$ is a root of all f_l and $T(i,j) \in TI$. ■

5 Protocol for Privacy Preserving Set Matching

5.1 Main Idea

The problem of PPSM can be considered as computing a function $\bigcup_{i'=1,...,N, i' \neq i}(T_i \cap T_{i'})$ on P_i for $i = 1, ..., N$. On each P_i the polynomial f_i is computed whose roots are elements in T_i. Then we can use a polynomial $(f_i * \sum_{k=1}^{N} r_{i'k} + f_{i'} * \sum_{k=1}^{N} r'_{i'k})$ to represent elements in $T_i \cap T_{i'}$, in which $r_{i'k} = \sum_{j=0}^{\alpha} a_j x^j$, $r'_{i'k} = \sum_{j=0}^{\alpha} a'_j x^j$. The degrees of $r_{i'k}$ and $r'_{i'k}$ are both α and $\alpha = \lceil \frac{S}{N-1} \rceil$. The coefficients a_j and a'_j for $j = 0, ..., \alpha$ are uniformly selected from the plaintext space of the threshold HE scheme (for Paillier's scheme, it's $\mathbb{Z}_{\mathcal{N}}$).

We can also use the multiplication of these polynomials to represent the elements in the union of all $T_i \cap T_{i'}$ for $i' = 1, ..., N, i' \neq i$. The resulting polynomial is F_i as following:

$$F_i = \prod_{i'=1,...,N, i' \neq i} (f_i * \sum_{k=1}^{N} r_{i'k} + f_{i'} * \sum_{k=1}^{N} r'_{i'k}) \tag{3}$$

The coefficients of F_i should be encrypted in the computations. We can use the evaluation of F_i at $T(i,j)$ to determine whether $T(i,j) \in \bigcup_{i'=1,...,N, i' \neq i}(T_i \cap T_{i'})$. The correctness of the determination will be proved in Lemma 2. For PPSM defined in Section 1, $E(F_i)$ can't be decrypted before evaluations, otherwise P_i will know $T(i,j)$ can be matched by b parties if P_i finds there is a factor $(x - T(i,j))^b$ in F_i, and this will breach the Privacy Requirement 2) in Section 1.

5.2 The Protocol

In the following Protocol 2, each party P_i computes its $E(F_i)$ in $N - 1$ rounds. For example, P_1 firstly computes $E(F_{12}) = E(f_1 * \sum_{k=1}^{N} r_{2k} + f_2 * \sum_{k=1}^{N} r'_{2k})$ in Step 2) of Protocol 2, by summing $E(f_1 * \sum_{k=1}^{N} r_{2k})$ and $E(f_2 * \sum_{k=1}^{N} r'_{2k})$. Then P_1 repeats Step 2), computes $E(F_{13}) = E(F_{12}(f_1 * \sum_{k=1}^{N} r_{3k} + f_3 * \sum_{k=1}^{N} r'_{3k}))$, by summing $E(F_{12} f_1 * \sum_{k=1}^{N} r_{3k})$ and $E(F_{12} f_3 * \sum_{k=1}^{N} r'_{3k})$. After $N - 1$ rounds of Step 2), P_1 gets $E(F_1) = E(F_{1N}) = E((f_1 * \sum_{k=1}^{N} r_{2k} + f_2 * \sum_{k=1}^{N} r'_{2k}) \cdots (f_1 * \sum_{k=1}^{N} r_{Nk} + f_N * \sum_{k=1}^{N} r'_{Nk}))$. Finally, P_1 evaluates each $E(F_1(T(1,j)))$, and decrypts it to see whether it's 0.

Protocol 2: *Protocol for Privacy Preserving Set Matching*

Inputs: There are N ($N \geq 2$) semi-honest parties. Each party has a private set of S elements, denoted T_i. Every party holds the public key and it's own share of the secret key for the threshold Paillier's cryptosystem.

Output: Each party P_i knows whether its tuples belong to $\bigcup_{i'=1,\ldots,N,i'\neq i}(T_i \cap T_{i'})$.

Steps:

1) Each party P_i computes its $f_i = (x - T(i,1)) \cdots (x - T(i,S))$.
2) P_1 initializes $E(F_{11}) = E(1)$, and repeats the following for $j = 2, ..., N$.
   - 2.1) P_1 computes $E(F_{1(j-1)} * f_1)$ and sends it to all the other parties. Each party P_k ($k \neq 1$) randomly chooses r_{jk} as described in Section 5.1, computes $E(F_{1(j-1)} * f_1 * r_{jk})$ by computation 4) in Section 3.3, and sends it back to P_1. P_1 also randomly chooses r_{j1} and computes $E(F_{1(j-1)} * f_1 * \sum_{k=1}^{N} r_{jk})$ by computation 3) in Section 3.3.
   - 2.2) P_1 sends $E(F_{1(j-1)})$ to P_j, P_j computes $E(F_{1(j-1)} * f_j)$, sends it to all the other parties. Each of these parties P_k including P_j randomly chooses r'_{jk}, computes $E(F_{1(j-1)} * f_j * r'_{jk})$ and sends it to P_1. P_1 computes $E(F_{1(j-1)} * f_j * \sum_{k=1}^{N} r'_{jk})$.
   - 2.3) P_1 computes $E(F_{1j}) = E(F_{1(j-1)}(f_1 * \sum_{k=1}^{N} r_{jk} + f_j * \sum_{k=1}^{N} r'_{jk}))$ by summing $E(F_{1(j-1)} * f_1 * \sum_{k=1}^{N} r_{jk})$ and $E(F_{1(j-1)} * f_j * \sum_{k=1}^{N} r'_{jk})$.

   At the end of $j = N$, P_1 gets $E(F_1) = E(F_{1N}) = E(\prod_{j=2}^{N}(f_1 * \sum_{k=1}^{N} r_{jk} + f_j * \sum_{k=1}^{N} r'_{jk}))$.
3) Each P_i other than P_1 repeats Step 2) and gets $E(F_i) = E(\prod_{i'=1\ldots N,i'\neq i} (f_i * \sum_{k=1}^{N} r_{i'k} + f_{i'} * \sum_{k=1}^{N} r'_{i'k}))$.
4) Each P_i evaluates $E(F_i)$ at $T(i,j)$ for $j = 1, ..., S$, using computation 1) in Section 3.3.
5) Each party decrypts $E(F_i(T(i,j)))$ in the collaboration of the other $N-1$ parties for $j = 1, ..., S$. If the evaluation $F_i(T(i,j)) = 0$, $T(i,j)$ has a duplicate on the other parties; otherwise, $T(i,j)$ hasn't any duplicate on the other parties.

Lemma 2. *Protocol 2 is a correct protocol for the PPSM problem.*

Proof: Protocol 2 determines whether $T(i,j) \in \bigcup_{i'=1,\ldots,N,i'\neq i}(T_i \cap T_{i'})$ by the evaluation $F_i(T(i,j))$. If there is a party $P_{i'}$ who has a duplicate of $T(i,j)$, i.e., $T(i,j) \in \bigcup_{i'=1,\ldots,N,i'\neq i}(T_i \cap T_{i'})$, then both $f_{i'}(T(i,j))$ and $f_i(T(i,j))$ are 0, and $f_i(T(i,j)) * \sum_{k=1}^{N} r_{i'k} + f_{i'}(T(i,j)) * \sum_{k=1}^{N} r'_{i'k} = 0$, then $F_i(T(i,j)) = 0$. That is, if $F_i(T(i,j)) \neq 0$, $T(i,j) \notin \bigcup_{i'=1,\ldots,N,i'\neq i}(T_i \cap T_{i'})$.

If $F_i(T(i,j)) = 0$, $T(i,j)$ is a root of at least one factor $(f_i * \sum_{k=1}^{N} r_{i'k} + f_{i'} * \sum_{k=1}^{N} r'_{i'k})$ in F_i. In this factor, $f_i(T(i,j)) = 0$, $\sum_{k=1}^{N} r'_{i'k}$ is a polynomial of degree $\lceil \frac{S}{N-1} \rceil$ uniformly distributed over $\mathbb{Z}_{\mathcal{N}}[x]$. Any $T(i,j) \in \mathbb{Z}_{\mathcal{N}}$ is a root of $\sum_{k=1}^{N} r'_{i'k}$ with probability $1/\mathcal{N}$ (by [13]). When $\mathcal{N}$ is large enough, overwhelmingly $T(i,j)$ is a root of $f_{i'}$, and the corresponding $P_{i'}$ has a duplicate of $T(i,j)$. ■

6 Security Analysis

6.1 Security Analysis on Protocol 1

The Inferred Information by the Definition of PPSI. Suppose there are c colluded parties P_I, $I = \{i_1, ..., i_c\}$. It's unavoidable for P_I to combine their inputs and outputs to infer information. However, by the definition of PPSI in Section 1, they should know no more information than TI in each $T_{i'}$, $\forall i' \in I'$, $I' = \{1, ..., N\} \setminus I$. That is,

6.1.1) On $P_i \in P_I$, if $T(i,j) \in TI$, they know each $T_{i'}$ has $T(i,j)$.
6.1.2) On $P_i \in P_I$, if $T(i,j) \notin TI$, they don't know whether $T(i,j) \in T_{i'}$ for $\forall i' \in I'$.

The Inferred Information after Participating in Protocol 1. In Protocol 1, each P_i gets $G = (g_1, ..., g_N)$, so P_I may infer the roots of $f_{i'}$ for $\forall i' \in I'$ by analyzing the coefficients in G. By the following lemma, we prove that G resists such kind of analysis.

Lemma 3. *In Protocol 1, any P_i in the coalition of c $(1 \le c \le N-1)$ semi-honest parties (P_I) can know no more elements than TI in any $T_{i'}$ for $\forall i' \in I'$.*

Proof: Due to the security of the threshold HE cryptosystem, P_I can't know any information on the plaintexts of the encryptions unless they are decrypted. We use P_i to denote any party in P_I. P_i gets only the decryption of $E(G)$. If $G(T(i,j)) = (0, ..., 0)$, by Lemma 1, P_i knows $T(i,j)$ is a root for all f_l $(l = 1, ..., N)$ and each $T_{i'}$ has $T(i,j)$. This accords with the case 6.1.1).

1) We firstly prove that, if $G(T(i,j)) \neq (0, ..., 0)$, P_i doesn't know whether $T(i,j) \in T_{i'}$ for $\forall i' \in I'$, that is, whether $T(i,j)$ is a root of any $f_{i'}$.
   From the view of P_i, $G = F(\prod_{i \in I} R_i \cdot \prod_{i' \in I'} R_{i'})$, $\prod_{i \in I} R_i$ is generated by P_I, and $\prod_{i' \in I'} R_{i'}$ is generated by $P_{I'}$. P_i doesn't know $\prod_{i' \in I'} R_{i'}$, thus if $G(T(i,j)) \neq (0, ..., 0)$, P_i can't compute $F(T(i,j))$. Then P_i can't know any $f_{i'}(T(i,j))$ and whether $T(i,j) \in T_{i'}$ for $\forall i' \in I'$. This accords with the case 6.1.2).
2) P_i may also analyze the coefficients of a single g_l $(l = 1, ..., N)$. In P_i's view, $g_l = f_{TI}(\mathcal{F}_I + \mathcal{F}_{I'})$, in which f_{TI} is the polynomial whose roots are TI, $\mathcal{F}_I = \sum_{i \in I}(f_i/f_{TI} * \sum_{j=1}^{N} r_{i,j} R_{il})$, and $\mathcal{F}_{I'} = \sum_{i' \in I'}(f_{i'}/f_{TI} * \sum_{j=1}^{N} r_{i',j} R_{i'l})$. We should also prove that P_i can't know $\mathcal{F}_{I'}$, otherwise he will know $\bigcap_{i' \in I'} T_{i'}$ by factoring $\mathcal{F}_{I'}$.
   From the view of P_i, in $\mathcal{F}_I$, $\forall i \in I$, $\sum_{j=1}^{N} r_{i,j} R_{il}$ can be supposed as $b_{i,1}x + b_{i,0}$, in which $b_{i,1}$ and $b_{i,0}$ are random numbers. Given $f_i/f_{TI} = \sum_{k=0}^{S-|TI|} a_{i,k} x^k$, suppose $f_i/f_{TI} * \sum_{j=1}^{N} r_{i,j} R_{il} = \sum_{k=0}^{S-|TI|+1} c_{i,k} x^k$, then $c_{i,k} = a_{i,k} b_{i,0} + a_{i,k-1} b_{i,1}$. Suppose $\mathcal{F}_I = \sum_{k=0}^{S-|TI|+1} e_k x^k$, then $e_k = \sum_{i \in I} c_{i,k}$. Suppose $\mathcal{F}_{I'} = \sum_{k=0}^{S-|TI|+1} e'_k x^k$, then the k-th coefficient of $\mathcal{F}_I + \mathcal{F}_{I'}$: $e''_k = e_k + e'_k = \sum_{i \in I}(a_{i,k} b_{i,0} + a_{i,k-1} b_{i,1}) + e'_k$.

P_i knows all $e_k^{''}$ from g_l/f_{TI}, and all $a_{i,k}$ from f_i/f_{TI}, but doesn't know all $b_{i,1}$, $b_{i,0}$, and e_k'. Thus from $e_k^{''} = \sum_{i \in I}(a_{i,k}b_{i,0} + a_{i,k-1}b_{i,1}) + e_k'$, P_i gets a set of $S - |TI| + 2$ linear equations with $2c + S - |TI| + 2$ unknowns. For $1 \le c \le N-1$, P_i can't compute the solutions for these unknowns. Therefore, P_i can't know e_k' for $k = 0, ..., S - |TI| + 1$, and can't know any root of $\mathcal{F}_{I'}$. In each g_l $(l = 1, ..., N)$, P_i can't know $\mathcal{F}_{I'}$, which makes P_i fail to know any $f_{i'}/f_{TI}$ in $\mathcal{F}_{I'}$.

In sum, in Protocol 1, $P_i \in P_I$ can know no more roots than TI in any $T_{i'}$ for $\forall i' \in I'$. ■

Theorem 1. *Protocol 1 is a privacy preserving protocol for the PPSI problem.*

The proof of this theorem is postponed to the Appendix.

6.2 Security Analysis on Protocol 2

The Inferred Information by the Definition of PPSM. If there is any coalition of c semi-honest parties P_I $(I = \{i_1, ..., i_c\})$, by the definition of PPSM, it's unavoidable for P_i $(\in P_I)$ to infer the following information by combining inputs and outputs of its coalition parties:

6.2.1) if the determination is $T(i,j)$ has a duplicate on the other parties, and P_i knows $T(i,j)$ also has a duplicate on P_I, then it can't know whether there is any duplicate of $T(i,j)$ on the remaining parties $P_{I'}$ $(I' = \{1, ..., N\} \backslash I)$.

6.2.2) if the determination is $T(i,j)$ has a duplicate on the other parties, and P_i knows $T(i,j)$ hasn't any duplicate on P_I, then it knows that $T(i,j)$ must have a duplicate on $P_{I'}$; We denote these $T(i,j)$ on $P_{I'}$ as set $\mathcal{T}$. It's easy to see that $0 \le |\mathcal{T}| \le (N-c)S$.

6.2.3) if the determination is $T(i,j)$ hasn't any duplicate on the other parties, then P_i knows that $T(i,j)$ hasn't any duplicate on $P_{I'}$, that is, $P_{I'}$ can't have $T(i,j)$. We denote these $T(i,j)$ as $\mathcal{T}'$.

Therefore, by the definition P_i $(\in P_I)$ knows $\mathcal{T}$ and $\mathcal{T}'$ as above. We assume this kind of information is harmless.

The Inferred Information after Participating in Protocol 2. In Step 3) of Protocol 1, P_i $(\in P_I)$ can't directly know the coefficients of F_i because they are encrypted. However, P_i knows S pairs of ($T(i,j), F_i(T(i,j))$), and those $f_{i'}$, $r_{i'k}$ and $r_{i'k}'$ generated by its collation parties. Thus, P_i can do an attack by analyzing the coefficients of F_i. In the following lemma, we prove that Protocol 2 is robust against this attack.

Lemma 4. *In Protocol 2, any P_i in the coalition of c $(1 \le c \le N-1)$ semi-honest parties (P_I) can get only the following information:*

1) *the same two sets as $\mathcal{T}$ and $\mathcal{T}'$ in the case 6.2.2) and 6.2.3).*
2) *guessing elements on $P_{I'}$ other than $\mathcal{T}$ and $\mathcal{T}'$, after randomly choosing at least* 1 *numbers.*

Proof: In this proof P_i is any party in P_I. Due to the security of the threshold HE cryptosystem, P_i can't know any information on the plaintexts of the encryptions they receive.

P_i gets S pairs ($T(i,j), F_i(T(i,j))$) by evaluating F_i at $T(i,j)$. Because $f_i(T(i,j)) = 0$, then $F_i(T(i,j))$ becomes $F'_i(T(i,j))$ for which $F'_i = \prod_{i'=1\ldots N, i'\neq i}(f_{i'} * \sum_{k=1}^{N} r'_{i'k})$. If P_i can know all coefficients of F'_i, it can know all roots of F'_i by polynomial factoring, but all coefficients are encrypted. For P_i, there are $(N-1)S$ unknown coefficients in $\prod_{i'=1\ldots N, i'\neq i} f_{i'}$ excluding the leading coefficient ($=1$). Because $\prod_{i'=1\ldots N, i'\neq i}(\sum_{k=1}^{N} r'_{i'k}) = \sum_{j=0}^{\beta} R_j x^j$, $\beta = (N-1)\lceil \frac{S}{N-1} \rceil$, in this part there are at least $S+1$ unknown coefficients. Totally there are at least $(N-1)S + S + 1$ unknown coefficients in F'_i. It's easy to see that P_i can't find a unique F'_i that fits S pairs ($T(i,j), F'_i(T(i,j))$).

However, P_i knows f_{i_t} for $i_t \in I$, and $r'_{i'k}$ generated by P_I. Then in P_i's view, $F'_i = f_{\mathcal{I}} f_{\mathcal{I}'} \prod_{i'=1\ldots N, i'\neq i}(\sum_{k\in I} r'_{i'k} + \sum_{k\in I'} r'_{i'k})$, in which $f_{\mathcal{I}} = \prod_{i_t\in I, i_t\neq i} f_{i_t}$, $f_{\mathcal{I}'} = \prod_{i'_t\in I'} f_{i'_t}$, $\sum_{k\in I} r'_{i'k}$ is generated by P_I, $\sum_{k\in I'} r'_{i'k}$ is generated by $P_{I'}$.

1) if $F'_i(T(i,j)) = 0$, and $f_{\mathcal{I}}(T(i,j)) = 0$, then P_i can't know any root of $f_{\mathcal{I}'}$. This accords with the case 6.2.1).
2) if $F'_i(T(i,j)) = 0$, and $f_{\mathcal{I}}(T(i,j)) \neq 0$, then P_i knows $f_{\mathcal{I}'}(T(i,j)) = 0$. All these $T(i,j)$ are the same as $\mathcal{T}$ in the case 6.2.2).
3) if $F'_i(T(i,j)) \neq 0$, P_i knows $f_{\mathcal{I}'}(T(i,j)) \neq 0$, i.e., $T(i,j)$ isn't one root of $f_{\mathcal{I}'}$. All these $T(i,j)$ are the same as $\mathcal{T}'$ in the case 6.2.3).

Suppose in 2), all P_I know C_1 roots of $f_{\mathcal{I}'}$, then $0 \leq C_1 \leq (N-c)S$. Suppose in 3), P_i knows C_2 pairs ($T(i,j), F'_i(T(i,j))$) for F'_i, then $0 \leq C_2 \leq S$. Because P_i knows $f_{\mathcal{I}}(T(i,j))$, P_i can know $C_1 + C_2$ evaluations of F''_i: $F''_i = F'_i / f_{\mathcal{I}} = f_{\mathcal{I}'} \prod_{i'=1\ldots N, i'\neq i}(\sum_{k\in I} r'_{i'k} + \sum_{k\in I'} r'_{i'k})$. For P_i, there are $(N-c)S$ unknown coefficients in $f_{\mathcal{I}'}$ excluding the leading coefficient ($=1$). In $\prod_{i'=1\ldots N, i'\neq i}(\sum_{k\in I} r'_{i'k} + \sum_{k\in I'} r'_{i'k}) = \sum_{j=0}^{\beta} R'_j x^j$, $\beta = (N-1)\lceil \frac{S}{N-1} \rceil$, there are at least $S+1$ unknown coefficients. Therefore P_i still needs to arbitrarily guess $t = (N-c)S + S + 1 - (C_1 + C_2)$ coefficients in F''_i. It's easy to see that $t \geq 1$. That is, P_i should guess at least 1 random number before inferring other roots in $f_{\mathcal{I}'}$ than $\mathcal{T}$ and $\mathcal{T}'$. ■

Theorem 2. *Protocol 2 is a privacy preserving protocol for the PPSM problem.*

The proof of this theorem is postponed to the Appendix.

7 Comparisons with Related Work

7.1 Comparisons for Protocol 1

Complexity of Protocol 1

1) *Computation Cost*: Each Paillier's encryption and decryption requires a cost of $2lg\mathcal{N}$ modular multiplications (mod $\mathcal{N}^2$). Each exponentiation has the same cost with the encryption. We compare our protocol with other related

work regarding their computation cost on encryptions and multiplications of ciphertexts, and consider modular multiplication (mod $\mathcal{N}^2$) as a basic computation.

Thus, for each party in Protocol 1, the total encryptions are $(S+2)(N-1)^2-2$, and the total multiplications of ciphertexts are $(S+2)(N^2+2N-3)$. Then the total computation cost for each party is $2((S+2)(N-1)^2-2)lg\mathcal{N}+(S+2)(N^2+2N-3)$ modular multiplications.

2) *Communication Cost*: The length of each encryption is $2lg\mathcal{N}$. Then in Protocol 1, the total communication cost among all parties is $2N(N-1)(4S+5)lg\mathcal{N}$ bits.

Speeding up techniques can be employed in Protocol 1. If all parties ensure that there is a coalition of c $(1 \leq c \leq N-1)$ semi-honest parties, in Step 1) of Protocol 1 each $E(f_i)$ can be randomized as $E(f_i * \sum_{j=1}^{c+1} r_{i,j})$ by sending $E(f_i)$ to c parties. In Step 2) $E(G)$ can be computed as $E(F\prod_{i=1}^{c+1} R_i)$. What's more, in Step 1) the iterations $i = 1, ..., N$ can be made in parallel. Then the computation cost is $2(c(S+2)(N-1)-2)lg\mathcal{N}+c(S+2)(N+3)$ modular multiplications. The communication cost is $2cN(4S+5)lg\mathcal{N}$ bits.

Kissner's Protocol. In Kissner's protocol for PPSI ([18]), a single polynomial $F=\sum_{l=1}^{N} f_l * \sum_{k=1}^{N} r_{l,k}$ is constructed and evaluated on each $T(i,j)$. f_l is a polynomial representing elements on P_l, $r_{l,k}$ is a polynomial uniformly selected by P_k and has the same degree with f_l. In this protocol, it's easy to see that $T(i,j) \in TI$ is a sufficient condition for the evaluation $F(T(i,j))=0$, but $F(T(i,j))=0$ is not even a sufficient condition for $\forall l \in \{1,...,N\}$ $f_l(T(i,j)) * \sum_{k=1}^{N} r_{l,k}(T(i,j))=0$. In Lemma 1 we have proved that if $\forall l \in \{1,...,N\}$ $f_l(T(i,j)) * \sum_{k=1}^{N} r_{l,k}(T(i,j))=0$, the probability that $T(i,j) \in TI$ is $(1-1/\mathcal{N})^{N-1}$. Therefore, in Kissner's protocol, if $F(T(i,j))=0$, the probability that $T(i,j) \in TI$ is less than the probability achieved by our Protocol 1.

The major cost of this protocol is on computing the encrypted F. It's also based on Paillier's cryptosystem. The computation cost for each party and communication cost among all parties are shown in Table 2.

Freedman's Protocol. Freedman's protocol for PPSI ([8]) is quite different from ours and [18]'s. In their protocol, each party P_i $(i=1,...,N-1)$ sends the encrypted polynomial f_i representing its elements to P_N. P_N evaluates its elements $T(N,j)$ for $j=1,...,S$ on all these polynomials, randomizes the evaluations and sends them to all the other parties. These parties decrypt and combine the evaluations to determine whether $T(i,j) \in TI$. In this protocol each party also generates a random matrix, but the matrices are used in a different way from our Protocol 1 for they aren't full rank and not for multiplications. The XOR of each row of the matrices is required to be zero, and they are used to randomize the decryptions on each party.

The major cost of this protocol is on the evaluations of encrypted polynomials at all elements of P_N. The protocol is also based on Paillier's cryptosystem. The average computation cost for each party and communication cost among all parties are shown in Table 2. In [8] only the protocol for the semi-honest model is given.

Comparisons of 3 protocols. From Table 2, the computation costs of Protocol 1, protocols in [18] and [8] are respectively $O(cSNlg\mathcal{N})$, $O(cS^2lg\mathcal{N})$, $O(S(S+N)lg\mathcal{N})$ modular multiplications. Practically the size of a set, S, may be much larger than the number of parties, N. Then it's easy to see that Protocol 1 is more efficient in computation than [18] and [8], and more efficient in communication than [18].

For a quantitative analysis, we conservatively set $S = 20$, $N = 5$, and set $c = 3$, $lg\mathcal{N} = 1024$, then Protocol 1 saves 81% and 63% computation costs, 17% and 20% communication costs in comparison with [18] and [8]. We also notice that if $c = 4$, i.e., all of the N parties are semi-honest, then the communication cost in Protocol 1 will increase by 6% in comparison with [8]. Thus Protocol 1 can utilize the knowledge on honest relationships among some of the N parties to reduce the communication cost.

Table 2. Comparisons of solutions for the PPSI problem

	Computation Cost	*Communication Cost*
Our Protocol 1	$2(c(S+2)(N-1)-2)lg\mathcal{N}+c(S+2)(N+3)$	$2cN(4S+5)lg\mathcal{N}$
Protocol in [18]	$2(c(S+1)^2+5S+3)lg\mathcal{N}+c(S^2+4S+2)$	$2cN(5S+2)lg\mathcal{N}$
Protocol in [8]	$((S+1)(S+2)+3S(N-1)-1)2lg\mathcal{N}+S(S+1)$	$10S(N-1)^2lg\mathcal{N}$

7.2 Comparisons for Protocol 2

Complexity of Protocol 2. In Protocol 2, on each party the computation on encryptions and multiplications of ciphertexts requires $2N^2S^2lg\mathcal{N} + N^3S$ modular multiplications (mod $\mathcal{N}^2$). The communication cost among all parties is $4N(N-1)^2Slg\mathcal{N}$ bits.

The complexity of Protocol 2 can be reduced if all parties ensure that there may be a coalition of c ($1 \le c \le N-1$) parties. In Step 2.1), P_1 can send $E(F_{1(j-1)} * f_1)$ to c parties; in Step 2.2), P_j can send $E(F_{1(j-1)} * f_j)$ to c parties; At the end of Step 2), P_1 gets $E(F_1) = E(\prod_{j=2}^{N}(f_1 * \sum_{k=1}^{c+1} r_{jk} + f_j * \sum_{k=1}^{c+1} r'_{jk}))$. The iterations in Step 3) can also be made in parallel with Step 2). Then the computation cost is $2cNS^2lg\mathcal{N} + cN^2S$ modular multiplications, and the communication cost is $4c(N-1)^2Slg\mathcal{N}$ bits.

Solution D1 Derived from [18]. The main idea of the private set intersection protocol in [18] is to plus the randomized polynomials representing the data sets, and their private set union protocol is mainly to multiply the polynomials representing the data sets. Therefore, Solution D1 (as described in Section 2) should firstly compute $(f_i * \sum_{k=1}^{N} r_{i'k} + f_{i'} * \sum_{k=1}^{N} r'_{i'k})$ for $T_i \cap T_{i'}$ for $i' = 1, ..., N$, $i' \neq i$, then compute $F_i = \prod_{i'=1...N, i'\neq i}(f_i * \sum_{k=1}^{N} r_{i'k} + f_{i'} * \sum_{k=1}^{N} r'_{i'k})$

for $\bigcup_{i'=1\ldots N, i' \neq i}(T_i \cap T_{i'})$, and evaluate it. In [18], the way to privately compute the encryption of F_i wasn't provided, and all $r_{i'k}$ and $r'_{i'k}$ are randomly chosen polynomials with degree S. Because $r_{i'k}$ and $r'_{i'k}$ have the same degree with f_i and f'_i, Solution D1 needs a high cost to compute the encrypted polynomial multiplications and evaluations. The computation and communication costs of Solution D1 are shown in Table 3.

Comparisons. From Table 3, it's easy to see that Protocol 2 is more efficient in computation and communication than Solution D1. Suppose $N = 5$, $c = 4$, $S = 20$, $lg\mathcal{N} = 1024$, then Protocol 2 saves 80% computation cost and 50% communication cost.

Table 3. Comparisons of solutions for the PPSM problem

	Computation Cost	*Communication Cost*
Our Protocol 2	$2cNS^2lg\mathcal{N} + cN^2S$	$4c(N-1)^2Slg\mathcal{N}$
Solution D1	$2cN^2S^2lg\mathcal{N} + 2cN^2S$	$8c(N-1)^2Slg\mathcal{N}$

8 Conclusions and Open Problems

We address two problems in privacy preserving matching against distributed datasets: Privacy Preserving Set Intersection (PPSI) and Privacy Preserving Set Matching (PPSM) among N parties. The two problems are solved by constructing polynomials representing elements in the set intersection and set matching, and evaluating the polynomials to determine whether an element is in the set intersection and set matching, without publishing the datasets on each party. The security of the two protocols are proved assuming there is a coalition of c $(1 \leq c \leq N-1)$ semi-honest parties. In comparison with related work in [18] and [8], our two protocols have less computation and communication costs.

In the future, we will extend the two protocols in the semi-honest model to the malicious model employing some zero-knowledge proofs. We will also utilize the two protocols to protect the privacy in some practical problems, e.g., internet congestion control ([23]).

In the problem formulation of Section 1, we have assumed that the size of each party's set (S) is much less than the size of the common set $\mathbb{T}$ to prevent the dictionary attack. There are many applications fitting for this assumption, e.g., the intersection among the sets of credit card numbers. It's well known that the common set $\mathbb{T}$ for credit card numbers is large enough, so that given a suitable S, the probability that an adversary arbitrarily chooses a number which equals any number of the honest party is $\frac{S}{|\mathbb{T}|} \to 0$. However, there are also some applications where $\frac{S}{|\mathbb{T}|}$ can't be negligible. Then how can the dictionary attack be prevented? In these cases, our two protocols can effectively resist the semi-honest behaviors of the adversary, and be extended to resist the malicious behaviors, but it's also an important problem to prevent the adversary from defrauding the honest party of inputs using the common set $\mathbb{T}$ when $\frac{S}{|\mathbb{T}|}$ isn't negligible.

Acknowledgments. This research is conducted as a program for the "Fostering Talent in Emergent Research Fields" in Special Coordination Funds for Promoting Science and Technology by Ministry of Education, Culture, Sports, Science and Technology, Japan. We also acknowledge the anonymous reviewers for their helpful suggestions on improving the quality of this paper, and all the other people who contribute to this paper.

References

1. F. Boudot, B. Schoenmakers and J. Traor'e, "A Fair and Efficient Solution to the Socialist Millionaires' Problem", in *Discrete Applied Mathematics*, 111(1-2), pp. 23-36, 2001.
2. R. Cramer, I. Damgard, and J. Nielsen, "Multiparty Computation from Threshold Homomorphic Encryption", in *Advances in Cryptology - EUROCRYPT 2001*, LNCS, Springer, vol. 2045, pp. 280-300, 2001.
3. W. Du and M. Attalah, "Protocols for Secure Remote Database Access with Approximate Matching", in *Proc. of the 7th ACM CCS, the 1st Workshop on Security and Privacy in E-commerce*, 2000.
4. R. Fagin, M. Naor, and P. Winkler, "Comparing Information without Leaking It", in *Communications of the ACM*, 39(5): 77-85, 1996.
5. J. Feigenbaum, Y. Ishai, T. Malkin, K. Nissim, M. Strauss, and R. Wright, "Secure Multiparty Computation of Approximations", in *Proc. of the 28th International Colloquium on Automata, Languages and Programming (ICALP 2001)*, pp. 927-938, 2001.
6. P. Fouque, G. Poupard and J. Stern, "Sharing Decryption in the Context of Voting or Lotteries", in *Proc. of the 4th International Conference on Financial Cryptography*, pp. 90 - 104, 2000.
7. P. Fouque and D. Pointcheval, "Threshold Cryptosystems Secure against Chosen-ciphertext Attacks", in *Proc. of Asiacrypt 2001*, pp. 351 - 368, 2001.
8. M. Freedman, K. Nissim and B. Pinkas, "Efficient Private Matching and Set Intersection", in *Proc. of Eurocrypt '04*, LNCS, Springer, vol. 3027, pp. 1 - 19, 2004.
9. B. Goethals, S. Laur, H. Lipmaa, and T. Mielikainen, "On Secure Scalar Product Computation for Privacy-Preserving Data Mining", in *Proc. of ICISC*, 2004.
10. O. Goldreich, "Foundations of Cryptography: Volume 1, Basic Tools", Cambridge University Press, 2001.
11. O. Goldreich, "Foundations of Cryptography: Volume 2, Basic Applications", Cambridge University Press, 2004.
12. O. Goldreich, S. Micali, and A. Wigderson, "How to Play Any Mental Game", in *Proc. of 19th STOC*, pp. 218-229, 1987.
13. T. Hartman and R. Raz, "On the Distribution of the Number of Roots of Polynomials and Explicit Weak Designs", in *Random Structures and Algorithms*, Vol. 23 (3), pp. 235 - 263, 2003.
14. S. Hohenberger and S. A. Weis, "Honest-Verifier Private Disjointness Testing without Random Oracles", in *Workshop on Privacy Enhancing Technologies (PET)*, 2006.
15. P. Indyk and D. Woodruff, "Polylogarithmic Private Approximations and Efficient Matching", in *Proc. of the Third Theory of Cryptography Conference (TCC 2006)*, LNCS, Springer, vol. 3876, pp. 245-264, 2006.

16. M. Jakobsson, A. Juels, "Mix and Match: Secure Function Evaluation via Ciphertexts", in *ASIACRYPT 2000*, pp 162-177, 2000.
17. A. Kiayias and A. Mitrofanova, "Testing disjointness of private datasets", in *Proc. of Financial Cryptography (FC 2005)*, LNCS, Springer, vol. 3570, pp. 109C124, 2005.
18. L. Kissner and D. Song, "Privacy-Preserving Set Operations", in *Advances in Cryptology - CRYPTO 2005*, LNCS, Springer, vol.3621, pp. 241-257, 2005.
19. H. Lipmaa, "Verifiable Homomorphic Oblivious Transfer and private Equality Test", in *Advances in Cryptography ASIACRYPT 2003*, pp. 416-433, 2003.
20. M. Naor and B. Pinkas, "Oblivious Transfer and Polynomial Evaluation", in *Proc. of the 31st Annual ACM Symposium on Theory of Computing*, pp. 245-254, 1999.
21. P. Paillier, "Public-key Cryptosystems based on Composite Degree Residuosity Classes", in *Proc. of Asiacrypt 2000*, pp. 573-584, 2000.
22. D. Randall, "Efficient Generation of Random Nonsingular Matrices", in *Random Structures and Algorithms*, vol. 4(1), pp. 111-118, 1993.
23. N. Xiong, X. Defago, X. Jia, Y. Yang, and Y. He, "Design and Analysis of a Self-tuning Proportional and Integral Controller for Active Queue Management Routers to support TCP Flows", in *Proc. of IEEE INFOCOM*, 2006.
24. A.C. Yao, "Protocols for Secure Computations", in *Proc. of the 23rd Annual IEEE Symposium on Foundations of Computer Science*, pp. 160 - 164, 1982.

Appendix: Proofs of Theorems

Theorem 1. *Protocol 1 is a privacy preserving protocol for the PPSI problem.*

Proof: By the definition of PPSI, we actually should compute a multi-party function $\mathfrak{f}$: $\mathfrak{f}(T_1, ..., T_N) = \mathfrak{f}(\overline{T}) = \{PPSI(T(i,j))|T(i,j) \in T_i, i = 1, ..., N, j = 1, ..., S\}$, with the i-th element $\mathfrak{f}_i(\overline{T}) = \{PPSI(T(i,j))|T(i,j) \in T_i, j = 1, ..., S\}$ for the party P_i, where $PPSI(T(i,j)) = 1$ if $T(i,j) \in TI$, and $PPSI(T(i,j)) = 0$ if $T(i,j) \notin TI$.

Given any coalition of c ($c \leq N - 1$) semi-honest parties indexed by $I = \{i_1, ..., i_c\}$, their views after participating in Protocol 1 are denoted by $VIEW_I^{\Pi}(\overline{T}) = (I, VIEW_{i_1}^{\Pi}(\overline{T}), ..., VIEW_{i_c}^{\Pi}(\overline{T}))$. We also let $\mathfrak{f}_I(\overline{T}) = (\mathfrak{f}_{i_1}(\overline{T}), ..., \mathfrak{f}_{i_c}(\overline{T}))$. From the definition in Section 3.1, we have to show that there exists a PPT algorithm S such that $S(I, (T_{i_1}, ..., T_{i_c}), \mathfrak{f}_I(\overline{T}))$ and $VIEW_I^{\Pi}(\overline{T})$ are computationally indistinguishable.

$VIEW_I^{\Pi}(\overline{T}) = \{V_1, V_2, V_3, V_4\}$: 1) V_1 is $I = \{i_1, ..., i_c\}$. 2)V_2 are $T_{i_1}, ..., T_{i_c}$. 3)V_3 are $E(G)$ and the intermediate encryptions received by P_I. 4)V_4 are $G(T(i_t, j))$ for any $i_t \in I$. With these views, the coalition can do the following two types of analysis:

1) *Cryptanalysis on* (V_1, V_2, V_3): Due to the semantic security of the threshold HE cryptosystem, P_i can't gain extra information from the encryptions in V_3. That is, supposing V_3 has s encryptions, with only negligible probability, P_i can distinguish V_3 and $\mathcal{ER}_1 = (E(r_1), ..., E(r_s))$ by randomly choosing $\mathcal{R}_1 = (r_1, ..., r_s)$ over the plaintext space of the HE scheme. Thus, $(V_1, V_2, V_3) \equiv^c (V_1, V_2, \mathcal{R}_1, \mathcal{ER}_1)$.

2) *Roots analysis on* (V_1, V_2, V_4): From Lemma 3, P_I can't know roots other than TI in any $f_{i'}$ for $\forall i' \in I'$. Thus, $V_4 = (\mathcal{A}, TI)$. $\mathcal{A} = \{a_{i_t,j} | i_t \in \{i_1, ..., i_c\}, j = 1, ..., S\}$ in which $a_{i_t,j} = 1$ if $G(T(i_t, j)) = (0, ..., 0)$, and $a_{i_t,j} = 0$ otherwise.

In sum, $VIEW_I^{\Pi}(\overline{T}) \equiv^c (V_1, V_2, \mathcal{R}_1, \mathcal{ER}_1, \mathcal{A}, TI)$.

$\mathfrak{f}_I(\overline{T})$ also equals $(\mathcal{A}, TI)$ by the analysis of the cases 6.1.1) and 6.1.2) in Section 6.1. Let $\mathcal{R}'_1 = \{r'_i | i = 1, ..., s\}$ are randomly chosen by P_I, and $\mathcal{ER}'_1$ are the encryptions of the sequence in $\mathcal{R}'_1$, then $S(I, (T_{i_1}, ..., T_{i_c}), \mathfrak{f}_I(\overline{T})) \equiv^c (I, (T_{i_1}, ..., T_{i_c}), \mathcal{A}, TI, \mathcal{R}'_1, \mathcal{ER}'_1) \equiv^c (V_1, V_2, \mathcal{A}, TI, \mathcal{R}_1, \mathcal{ER}_1) \equiv^c VIEW_I^{\Pi}(\overline{T})$. Then Protocol 1 privately computes PPSI against the coalition of any c ($c \leq N - 1$) semi-honest parties. ■

Theorem 2. *Protocol 2 is a privacy preserving protocol for the PPSM problem.*

Proof: By the definition of PPSM, we actually should compute a multi-party function $\mathfrak{f}$: $\mathfrak{f}(T_1, ..., T_N) = \mathfrak{f}(\overline{T})$, with the i-th element $\mathfrak{f}_i(\overline{T}) = \{PPSM(T(i, j)) | T(i, j) \in T_i, j = 1, ..., S\}$ for the party P_i, where $PPSM(T(i, j)) = 1$ if $T(i, j) \in \bigcup_{i'=1,...,N, i' \neq i}(T_i \cap T_{i'})$, and $PPSM(T(i, j)) = 0$ otherwise. Given any coalition of c ($c \leq N - 1$) semi-honest parties indexed by $I = \{i_1, ..., i_c\}$, their views after participating in Protocol 2 are $VIEW_I^{\Pi}(\overline{T})$. We also let $\mathfrak{f}_I(\overline{T}) = (\mathfrak{f}_{i_1}(\overline{T}), ..., \mathfrak{f}_{i_c}(\overline{T}))$. We show that there exists a PPT algorithm S such that $S(I, (T_{i_1}, ..., T_{i_c}), \mathfrak{f}_I(\overline{T}))$ and $VIEW_I^{\Pi}(\overline{T})$ are computationally indistinguishable.

$VIEW_I^{\Pi}(\overline{T}) = \{V_1, V_2, V_3, V_4\}$: 1) V_1 is $I = \{i_1, ..., i_c\}$. 2)V_2 are $T_{i_1}, ..., T_{i_c}$. 3)V_3 are $E(F_i)$ and the intermediate encryptions received by P_I. 4)V_4 are $F_{i_t}(T(i_t, j))$ for any $i_t \in I$. The coalition can do the following analysis:

1) *Cryptanalysis on* (V_1, V_2, V_3): Due to the semantic security of the threshold HE cryptosystem, supposing V_3 has s encryptions, with only negligible probability, P_i can distinguish V_3 and $\mathcal{ER}_1 = (E(r_1), ..., E(r_s))$ by randomly choosing $\mathcal{R}_1 = (r_1, ..., r_s)$. Thus, $(V_1, V_2, V_3) \equiv^c (V_1, V_2, \mathcal{R}_1, \mathcal{ER}_1)$.
2) *Roots analysis on* (V_1, V_2, V_4): From Lemma 4, $V_4 = (\mathcal{A}, \mathcal{T}, \mathcal{T}', \mathcal{R}_2)$. $\mathcal{A} = \{a_{i_t}^j | i_t \in \{i_1, ..., i_c\}, j = 1, ..., S\}$ in which $a_{i_t}^j = 1$ if $F_{i_t}(T(i_t, j)) = 0$, and $a_{i_t}^j = 0$ otherwise. $\mathcal{R}_2 = \{R_i | i = 1, ..., t\}$, in which R_i is a random number guessed by P_i, $t \geq 1$.

In sum, $VIEW_I^{\Pi}(\overline{T}) \equiv^c (V_1, V_2, \mathcal{R}_1, \mathcal{ER}_1, \mathcal{A}, \mathcal{T}, \mathcal{T}', \mathcal{R}_2)$. $\mathfrak{f}_I(\overline{T}) = (\mathcal{A}, \mathcal{T}, \mathcal{T}')$ by the analysis of the cases 6.2.1), 6.2.2) and 6.2.3). Let $\mathcal{R}'_1 = \{r'_i | i = 1, ..., s\}$, $\mathcal{R}'_2 = \{R'_i | i = 1, ..., t\}$ are randomly chosen by P_I, and $\mathcal{ER}'_1$ are the encryptions of the sequence in $\mathcal{R}'_1$. Then $S(I, (T_{i_1}, ..., T_{i_c}), \mathfrak{f}_I(\overline{T})) = (I, (T_{i_1}, ..., T_{i_c}), \mathcal{A}, \mathcal{T}, \mathcal{T}', \mathcal{R}'_1, \mathcal{ER}'_1, \mathcal{R}'_2) \equiv^c VIEW_I^{\Pi}(\overline{T})$. Then Protocol 2 privately computes PPSM against the coalition of any c ($c \leq N - 1$) semi-honest parties. ■