An Unconditionally Secure Protocol for Multi-Party Set Intersection*

Ronghua Li[1,2] and Chuankun Wu[1]

[1] State Key Laboratory of Information Security, Institute of Software, Chinese Academy of Sciences, Beijing 100080, P.R. China
[2] Graduate School of Chinese Academy of Sciences, Beijing 100039, P.R. China
{lirh,ckwu}@is.iscas.ac.cn

Abstract. Existing protocols for private set intersection are based on homomorphic public-key encryption and the technique of representing sets as polynomials in the cryptographic model. Based on the ideas of these protocols and the two-dimensional verifiable secret sharing scheme, we propose a protocol for private set intersection in the information-theoretic model. By representing the sets as polynomials, the set intersection problem is converted into the task of computing the common roots of the polynomials. By sharing the coefficients of the polynomials among parties, the common roots can be computed out using the shares. As long as more than $2n/3$ parties are semi-honest, our protocol correctly computes the intersection of n sets, and reveals no other information than what is implied by the intersection and the secrets sets controlled by the active adversary. This is the first specific protocol for private set intersection in the information-theoretic model as far as we know.

Keywords: Secure multi-party computation, privacy-preserving set intersection, unconditional security.

1 Introduction

This paper studies the following problem: n parties each with a secret set want to compute the intersection of these sets without leaking anything else about the secret sets. This problem is a specific case of secure multi-party computation, which is introduced by Yao [10], and extended by Goldreich, Micali and Wigderson [5]. The goal of secure multi-party computation is to design a protocol for the parties each with a secret input to compute securely a public function of their inputs. The protocol should be correct and private. Correctness means that every party believes the correctness of the output. Privacy means that no party can learn more than what is implied by its output and its own input. For the set intersection problem, the inputs are the parties' secret sets and the output is the intersection of these sets (all the parties get the intersection and nothing

* Supported by National Natural Science Foundation of China under Grant NO.90304007 and the National Grand Fundamental Research 973 Program of China under Grant NO.2004CB318004.

about other parties' secret sets). Protocols for private set intersection are useful in online recommendation services, online dating services, medical databases, data mining, etc., as pointed out by Freedman, Nissim and Pinkas [4].

Related work. The set intersection problem is studied in [4,7,8], where several protocols are presented in the cryptographic model based on homomorphic public-key encryption. In the information-theoretic model, many general secure computation protocols are proposed (e.g. [3,1,6,2]). Solutions to the set intersection problem can be derived from these general protocols from the theoretical point of view, however, how to modify the general protocols to solve the set intersection problem is not known.

Our work. This paper presents a protocol for the set intersection problem in the information-theoretic model. The protocol uses the idea of representing a set as a polynomial [4,7,8] and the two-dimensional verifiable secret sharing used in the general unconditionally secure protocols (e.g. [3,2]). Our protocol follows the share-compute-recover paradigm for general constructions in the information-theoretic model. By representing the parties' secret sets as polynomials, the set intersection problem is converted into the task of computing the common roots of these polynomials. Then the coefficients of the polynomials are shared using two-dimensional secret sharing scheme and the common roots of the polynomials are computed out using the shares. This is the first specific protocol for the intersection problem in the information-theoretic model as far as we know.

Assuming that an active adversary corrupts less than $n/3$ parties, the proposed protocol enables n parties each with a secret set to compute privately the intersection of these sets and leaks no other information than what is implied by the intersection and the secret sets controlled by the adversary. The security of the protocol is proved and efficiency is analyzed in Section 4.

The proposed protocol is efficient in terms of communication complexity. The protocol demands 6 rounds of communication and exchanges $\mathcal{O}(n^4k^2)$ elements in E, where k is the size of the secret sets and E is a large finite field. The most efficient general protocol in the information-theoretic model is due to [2]. The protocol uses circuit randomization technique (for details, see[1]) and consists of preparation phase, input phase, and computation phase. If the protocol is applied to the set intersection problem, then at least nk random numbers should be shared in the pre-processing phase, which needs $\mathcal{O}(n^4k)$ elements to be exchanged. As no article considers applying the general unconditionally secure protocols to the set intersection problem before, so we cannot estimate the total communication complexity, thus only a partial comparison is provided here.

2 Preliminaries

2.1 Adversary and Communication Models

Adversary model. A cheating party in a protocol is modelled as an adversary who can corrupt parties. There are two kinds of adversaries. A passive adversary learns the information hold by the corrupted parties, but the corrupted parties still act correctly according to the protocol. An active adversary not only learns the information hold by the corrupted parties but also takes full control of them. The two models are called passive model and active model respectively.

Communication model. There are two communication models in secure computation: cryptographic model and information-theoretic model. The first model assumes that parties are connected by public channels, and parties have bounded computation power (probabilistic polynomial time) and the adversary can see all the messages exchanged among the parties. The second model assumes that there are pair-wise secure channels among parties, and parties have unbounded computation power and the adversary can not learn the messages exchanged between the honest (i.e. uncorrupted) parties.

In the paper, we assume both passive and active adversaries, and we assume the information-theoretic model.

2.2 Polynomial Representation of Sets

Notation. Let E denote a large finite field. $E[x]$ consists of all the polynomials whose coefficients are chosen from E, and the probability that a random polynomial (i.e. the coefficients are chosen randomly from E) in $E[x]$ represents elements of U is negligible.

We use the technique of representing sets as polynomials [4]. Let S be a set of size k. A polynomial of degree k can be constructed: $f(x) = a_0 + a_1x + ... + a_kx^k$, $f(a) = 0$ if and only if $a \in S$, where the coefficients of $f(x)$ are chosen from E.

In [7,8], Kissner and Song convert the set intersection problem into the task of computing the roots of a polynomial by use of the polynomial representation of sets. This paper follows the idea of the protocols for set intersection in [7,8]. We briefly review their idea as follows.

Let $r(x)$ be a random polynomial in $E[x]$. If a is the root of $f(x)$ then a is the root of $f(x)r(x)$ too, that is, $f(x)r(x)$ preservers all the roots of $f(x)$. Let $S_1, ...S_n$ be n sets, and the polynomials $f_1(x), ..., f_n(x)$ represent $S_1, ...S_n$ respectively. Let $r_1(x), ..., r_n(x)$ be n random polynomials in $E[x]$. Then the roots of $f_1(x)r_1(x) + ... + f_n(x)r_n(x)$ represent the intersection of $S_1 \cap ... \cap S_n$.

2.3 Two-Dimensional Verifiable Secret Sharing

Secret sharing is firstly introduced by Shamir [9]. A secret s can be shared with a polynomial of degree t: $f(x)$ such that $f(0) = s$. Let $\alpha_1, ..., \alpha_n$ be public parameters chosen in E. P_i's share is $Piece_i(s) = f(\alpha_i)$, $i = 1, ..., n$. Shamir's secret sharing scheme is extended to two-dimensional secret sharing [3,6]. Each secret value is shared among the players with a polynomial of degree t, and each share is again shared among the parties with a polynomial of degree t. Let $Piece_i$ denote a share, and $Piece_{ij}$ (sometimes written as $Piece_{i,j}$, e.g. $Piece_{i,2k}(s)$) denote a share-share.

Sharing. To share a secret s, a party chooses a random two-dimensional polynomial $p(x, y) = \sum_{i,j=0}^{t} r_{ij}x^iy^j$ such that $p(0, 0) = s$, and sends to P_i the two polynomials $f_i(x) = p(x, \alpha_i), \tilde{f}_i(y) = p(\alpha_i, y)$, $i = 1, ..., n$. P_i's shares are $Piece_i(s) = f_i(0)$, $Piece_{ji}(s) = \tilde{f}_i(\alpha_j) = p(\alpha_i, \alpha_j)$, $j = 1, ..., n$. We say that the party t-shares s or s is t-shared.

Verifying the correctness of sharing. For $i = 1$, ..., n, each party P_j $(j \neq i)$ sends to P_i the share-share $Piece_{ij}(s) = f_j(\alpha_i)$, and P_i checks if the received value is equal to $\tilde{f}_i(\alpha_j)$.

Linear function. Let a, b be two secrets shared among the parties, and r be a constant integer, parties can compute locally the sharing of the new secrets ra and $a + b$.

Multiplication of two secrets. Let a, b be two secrets shared among the parties, the goal is to produce the t-sharing of a new secret $c = ab$ among the parties. The computation procedure develops in two steps: local computation and degree reduction.

The degree reduction phase involves mainly a sub-protocol: re-share protocol. The re-share protocol allows parties to produce t-shares of a secret given its t'-shares. In the active model, when a party t-shares his share $Piece_i(s)$, he proves that the shared value is indeed $Piece_i(s)$. The details for re-sharing protocol can be found in [6].

Reconstruction. Let s be t-shared among the parties. To reveal the secret s to a designated party, all other parties send their shares of s to the designated party, then the party recovers the secret using Lagrange Interpolation.

Error correction. In the active model, a party may receive wrong shares from corrupted parties when reconstructing a secret. Parties can perform locally error correction to get n correct shares, and recover the secret (see [3] for details).

3 The Proposed Protocol for Private Set Intersection

A protocol in the passive model is constructed first. The protocol in the active model is constructed based on the protocol in the passive model. The two protocols develop similarly except that some verifications and proofs are added to the second protocol.

3.1 Construction in the Passive Model

Let $P_1, ..., P_n$ be n parties connected with pair-wise secure channels. Assume parties are computationally unbounded and a passive adversary is allowed to corrupt $t < n/2$ parties. $P_1, ..., P_n$ have secret sets $S_1, ..., S_n$ respectively. Let k be the size of the sets and $S[j]$ denote the j-th element of S. Then a set S can be represented as a polynomial of degree k in $E[x]$: $f(x) = (x - S[1])...(x - S[k])$=$a_0 + a_1x + ... + a_kx^k$.

The main idea. Each party represents his secret set as a polynomial of k degree. Let $f_1(x) = a_{10} + a_{11}x + ... + a_{1k}x^k$, $f_2(x) = a_{20} + a_{21}x + ... + a_{2k}x^k$,

..., $f_n(x) = a_{n0} + a_{n1}x + ... + a_{nk}x^k$ be the n parties' polynomials respectively. Let $r_1(x)$, ..., $r_n(x)$ be n random polynomials of degree k. If the coefficients of $f_i(x), r_i(x)$, $i = 1, ..., n$, are shared, then the shares of the coefficients of the polynomial $F(x) = f_1(x)r_1(x) + f_2(x)r_2(x) + ... + f_n(x)r_n(x)$ can be computed. Each party publishes his shares of the coefficients of $F(x)$, then every party can recover the polynomial $F(x)$ and evaluate it at each element of his secret set. If the function value equals to zero then the element belongs to the intersection, else the element does not belong to the intersection. Thus the parties compute out the intersection of the sets. The protocol is composed of three phases: input phase, computation phase and output phase.

Input Phase
Step 1. Each party represents his secret set as a polynomial and shares the coefficients of the polynomial among the parties.

- For $i = 1, ..., n$, P_i represents his secret set as a polynomial $f_i(x)$.
- For $i = 1, ..., n$, P_i t-shares the coefficients of his polynomial (e.g. P_i uses $f_{ij}(x)$ to share a_{ij}, $j = 0, ..., k$).

Step 2. The parties produce jointly the t-sharing of n random polynomials of degree k.

- For $i = 1, ..., n$, P_i produces n random polynomials of degree k: $r_{i1}(x) = b_{i10} + b_{i11}x + ... + b_{i1k}x^k$, $r_{i2}(x) = b_{i20} + b_{i21}x + ... + b_{i2k}x^k$, ..., $r_{in}(x) = b_{in0} + b_{in1}x + ... + b_{ink}x^k$.
- For $i = 1, ..., n$, P_i t-shares the coefficients of the n random polynomials among the parties. E.g. P_i uses $r_{ij0}(x)$, $r_{ij1}(x)$, ..., $r_{ijk}(x)$ to share b_{ij0}, b_{ij1},..., b_{ijk} respectively, $j = 1, ..., n$.
- For $i = 1, ..., n$, P_i computes the t-shares of the following n polynomials of degree k $r_j(x) = \sum_{j'=1}^{n} r_{j'1}(x) = w_{j0} + w_{j1}x + ... + w_{jk}x^k$, $j = 1, ..., n$, as below: $Piece_i(w_{j0}) = \sum_{j'=1}^{n} Piece_i(b_{j'10})$, $Piece_i(w_{j1}) = \sum_{j'=1}^{n} Piece_i(b_{j'11})$, ..., $Piece_i(w_{jk}) = \sum_{j'=1}^{n} Piece_i(b_{j'1k})$, $j = 1, ..., n$.

Computation Phase
Let $f_j(x)r_j(x) = z_{j0} + z_{j1}x + ... + z_{j,2k}x^{2k}$, $j = 1$, ..., n, and $F(x) = f_1(x)r_1(x)$ + $f_2(x)r_2(x)$ +... + $f_n(x)r_n(x)$= $z_0 + z_1x + ... + z_{2k}x^{2k}$.
Step 1. Compute the $2t$-shares of the coefficients of $f_1(x)r_1(x)$, $f_2(x)r_2(x)$, ..., $f_n(x)r_n(x)$.

- For $i = 1, ..., n$, P_i computes locally the following values: $Piece_i(a_{j0})Piece_i$ (w_{j0}), $Piece_i(a_{j0})Piece_i(w_{j1})$, ..., $Piece_i(a_{jk})Piece_i(w_{jk})$, $j = 1, ..., n$.

Step 2. Call the re-sharing protocol, and convert the $2t$-shares of the coefficients of $f_1(x)r_1(x)$, $f_2(x)r_2(x)$, ..., $f_n(x)r_n(x)$ into t-sharing.

- For $i = 1, ..., n$, P_i t-shares the following values: $Piece_i(a_{j0})Piece_i(w_{j0})$, $Piece_i(a_{j0})Piece_i(w_{j1})$, ..., $Piece_i(a_{jk})Piece_i(w_{jk})$, $j = 1, ..., n$.

- For $i = 1, ..., n$, P_i reconstructs the t-shares of $a_{j0}w_{j0}$, ..., $a_{jk}w_{jk}$: $Piece_i(a_{j0}w_{j0})$, $Piece_i(a_{j0}w_{j1})$, ..., $Piece_i(a_{jk}w_{jk})$, $j = 1, ..., n$.
- For $i = 1, ..., n$, P_i computes the t-shares of the coefficients of $f_1(x)r_1(x)$, $f_2(x)r_2(x)$, ..., $f_n(x)r_n(x)$ as below: $Piece_i(z_{j0}) = Piece_i(a_{j0}w_{j0})$, $Piece_i(z_{j1}) = Piece_i(a_{j0}w_{j1}) + Piece_i(a_{j1}w_{j0})$, ..., $Piece_i(z_{j,2k}) = Piece_i(a_{jk}w_{jk})$, $j = 1, ..., n$.

Step 3. Parties compute the t-shares of the $2k + 1$ coefficients of $F(x)$.

- For $i = 1, ..., n$, P_i computes the following $2k + 1$ values: $Piece_i(z_j) = \sum_{j'=1}^{n} Piece_i(z_{j'j})$, $j = 0, ..., 2k$.

Output Phase
Step 1. Each party sends his t-shares of the coefficients of $F(x)$ to all other parties.

Step 2. Find out the intersection.

- For $i = 1, ..., n$, P_i reconstructs the $2k+1$ coefficients of $F(x)$ using $Piece_1(z_j)$, $Piece_2(z_j)$, ..., $Piece_n(z_j)$, $j = 0, ..., 2k$.
- For $i = 1, ..., n$, P_i evaluates $F(x)$ at each element of his set. If the evaluation is zero then the element belongs to the intersection, else the element does not belong to the intersection. All the elements at which the evaluation is zero form the intersection $S_1 \cap ... \cap S_n$.

In the above protocol, it needs to compute the shares of the coefficients of the multiplication of two polynomials given the shares of the coefficients of these two polynomials. For example, let $f(x)$, $g(x)$ be two polynomials of degree k, whose coefficients are shared among parties, we want to compute the sharing of $h(x) = f(x)g(x)$. Let c_j, $j = 0, ..., k$ be the coefficients of $f(x)$, d_j, $j = 0, ..., k$ be the coefficients of $g(x)$, and let u_j, $j = 0, ..., 2k$ be the coefficients of $h(x)$. We have: $u_0 = c_0d_0$, $u_1 = c_0d_1 + c_1d_0$, ..., $u_k = c_0d_k + c_1d_{k-1} + ... + c_kd_0$, $u_{k+1} = c_1d_k + c_2d_{k-1} + ... + c_kd_1$, ..., $u_{2k} = c_kd_k$. In these $2k + 1$ expressions, there are $(k + 1)(k + 2)$ items of the form c_id_j, and the t-shares of c_id_j can be computed out using the t-shares of c_i, d_j. Then the t-shares of u'_j, $j' = 0, ..., 2k$ can be computed out as below: $Piece_i(u_0) = Piece_i(c_0d_0)$, $Piece_i(u_1) = Piece_i(c_0d_1) + Piece_i(c_1d_0)$, ..., $Piece_i(u_k) = Piece_i(c_0d_k) + Piece_i(c_1d_{k-1}) + ... + Piece_i(c_kd_0)$, $Piecei(u_{k+1}) = Piece_i(c_1d_k) + Piece_i(c_2d_{k-1}) + ... + Piece_i(c_kd_1)$, ..., $Piece_i(u_{2k}) = Piece_i(c_kd_k)$.

3.2 Construction in the Active Model

In the protocol described above, an active adversary can disrupt the security in two ways. One is pointed at [7,8]: the adversary can use zero-polynomial $f(x) = 0$ to replace the the polynomial used to represent a set. The other is that the adversary can send wrong shares to parties when sharing a secret.

If a party uses zero-polynomial to represent his set, then it is equivalent to that the party's set is the union set and he can compute the intersection of other parties' sets. In order to prevent the parties from using zero-polynomial, it is enough to define 1 as the default value of the coefficients of the k-degree items.

In input phase and computation phase, whenever a secret is shared the correctness of sharing should be verified using the share-shares. Additionally, a party should prove that he indeed shares the required value (not a random value) in the re-sharing protocol.

In output phase, each party publishes his shares in order to compute out the coefficients of $F(x)$. For each coefficient of the polynomial $F(x)$, each party gets n shares among which at most t shares are from the adversary (there are at most t wrong shares). In case there are wrong shares, parties can use error correction procedure to correct errors and reconstruct the coefficients correctly.

Using the above methods of preventing cheating activities, the protocol in Section 3.1 can be made secure against an active adversary as below.

Input Phase
Step 1. Each party represents his secret set as a polynomial and shares the coefficients of the polynomial among the parties.

- For $i = 1, ..., n$, P_i represents his secret set as a polynomial $f_i(x)$.
- For $i = 1, ..., n$, P_i two-dimensionally shares the coefficients of $f_i(x)$.
- For $i = 1, ..., n$, P_i checks the correctness of each of the received shares using the corresponding share-shares under the help of other parties.

Step 2. Parties produce jointly the t-sharing of n random polynomials of degree k.

- For $i = 1, ..., n$, P_i chooses randomly n polynomials of degree k: $r_{ij}(x) = b_{ij0} + b_{ij1}x + ... + b_{ijk}x^k$, $j = 1, ..., n$.
- For $i = 1, ..., n$, P_i two-dimensionally t-shares the coefficients b_{ij0}, b_{ij1}, ..., b_{ijk}, $j = 1, ..., n$.
- For $i = 1, ..., n$, P_i verifies the correctness of each of the received shares under the help of other parties.
- For $i = 1, ..., n$, P_i computes the t-shares of the following n polynomials of degree k $r_j(x) = \sum_{j'=1}^{n} r_{j'1}(x) = w_{j0} + w_{j1}x + ... + w_{jk}x^k$, $j = 1, ..., n$, as below: $Piece_i(w_{j0}) = \sum_{j'=1}^{n} Piece_i(b_{j'10})$, $Piece_i(w_{j1}) = \sum_{j'=1}^{n} Piece_i(b_{j'11})$, ..., $Piece_i(w_{jk}) = \sum_{j'=1}^{n} Piece_i(b_{j'1k})$, $j = 1, ..., n$.

Computation Phase
Let $f_j(x)r_j(x) = z_{j0} + z_{j1}x + ... + z_{j,2k}x^{2k}$, $j = 1$, ..., n, and $F(x) = f_1(x)r_1(x) + f_2(x)r_2(x) + ... + f_n(x)r_n(x) = z_0 + z_1x + ... + z_{2k}x^{2k}$.

Step 1. Compute the $2t$-shares of the coefficients of $f_1(x)r_1(x)$, $f_2(x)r_2(x)$, ..., $f_n(x)r_n(x)$.

- For $i = 1, ..., n$, P_i computes locally the following values: $Piece_i(a_{j0})Piece_i(w_{j0})$, $Piece_i(a_{j0})Piece_i(w_{j1})$, ..., $Piece_i(a_{jk})Piece_i(w_{jk})$, $j = 1, ..., n$.

Step 2. Call the re-sharing protocol, and convert the $2t$-shares of the coefficients of $f_1(x)r_1(x)$, $f_2(x)r_2(x)$, ..., $f_n(x)r_n(x)$ into the t-shares.

- For $i = 1, ..., n$, P_i two-dimensionally t-shares the values: $Piece_i(a_{j0})Piece_i(w_{j0})$, $Piece_i(a_{j0})Piece_i(w_{j1})$, ..., $Piece_i(a_{jk})Piece_i(w_{jk})$, $j = 1, ..., n$.
- For $i = 1, ..., n$, P_i checks the correctness of each of the received shares using the corresponding share-shares under the help of other parties.
- For $i = 1, ..., n$, P_i proves that the shared values are indeed the multiplication of his shares: $Piece_i(a_{j0})Piece_i(w_{j0})$, $Piece_i(a_{j0})Piece_i(w_{j1})$, ..., $Piece_i(a_{jk})Piece_i(w_{jk})$, $j = 1, ..., n$.
- For $i = 1, ..., n$, P_i reconstructs the t-shares of $a_{j0}w_{j0}$, ..., $a_{jk}w_{jk}$: $Piece_i(a_{j0}w_{j0})$, $Piece_i(a_{j0}w_{j1})$, ..., $Piece_i(a_{jk}w_{jk})$, $j = 1, ..., n$.
- For $i = 1, ..., n$, P_i computes the t-shares of the coefficients of $f_1(x)r_1(x)$, $f_2(x)r_2(x)$, ..., $f_n(x)r_n(x)$ as below: $Piece_i(z_{j0}) = Piece_i(a_{j0}w_{j0})$, $Piece_i(z_{j1}) = Piece_i(a_{j0}w_{j1}) + Piece_i(a_{j1}w_{j0})$, ..., $Piece_i(z_{j,2k}) = Piece_i(a_{jk}w_{jk})$, $j = 1, ..., n$.

Step 3. Parties compute the t-shares of the $2k + 1$ coefficients of $F(x)$.

- For $i = 1, ..., n$, P_i computes the following $2k + 1$ values: $Piece_i(z_j) = \sum_{j'=1}^{n} Piece_i(z_{j'j})$, $j = 0, ..., 2k$.

Output Phase
Step 1. Each party sends his t-shares of the coefficients of $F(x)$ to all other parties.

Step 2. Find out the intersection.

- For $i = 1, ..., n$, P_i performs the error correction procedure to get n correct shares for each of the $2k + 1$ coefficients of $F(x)$.
- For $i = 1, ..., n$, P_i reconstructs the $2k + 1$ coefficients of $F(x)$ using the correct shares.
- For $i = 1, ..., n$, P_i evaluates $F(x)$ at each element of his set. If the evaluation is zero then the element belongs to the intersection, else the element does not belong to the intersection. All the elements at which the evaluation is zero form the intersection $S_1 \cap ... \cap S_n$.

4 Security and Efficiency Analysis of the Proposal

4.1 Correctness and Security

Theorem 1. *In the passive model, suppose at most $t < n/2$ parties collude, the protocol in Section 3.1 is a solution to the private set intersection problem.*

Proof. The correctness of the protocol is based on the polynomial $F(x) = f_1(x)r_1(x) + f_2(x)r_2(x) + ... + f_n(x)r_n(x)$, and the privacy is based on the randomness of $r_1(x)$, $r_2(x)$, ..., $r_n(x)$ and the properties of the secret sharing scheme.

Correctness. Firstly, all the parties get the correct polynomial $F(x)$. When the input phase ends, all the coefficients of $f_1(x)$, ..., $f_n(x)$, $r_1(x)$, ..., $r_n(x)$ are shared correctly among the parties. When the computation phase ends, the coefficients of $F(x)$ are shared correctly among the parties. In the output phase, each party publishes his shares and all the party can compute out the coefficients of $F(x)$, which means that, all the parties learn the polynomial $F(x)$.

Secondly, all the parties learn the correct intersection $S_1 \cap ... \cap S_n$ from $F(x)$. If the element a belongs to the intersection $S_1 \cap ... \cap S_n$, then $f_1(a) = 0$, ..., $f_n(a) = 0$ and $f_1(a)r_1(a) = 0$, ..., $f_n(a)r_n(a) = 0$, and $F(a) = f_1(a)r_1(a) + ... + f_n(a)r_n(a) = 0$. It is to say that, if a belongs to $S_1 \cap ... \cap S_n$, then each party learns $F(a) = 0$ when evaluating $F(x)$ at the element a. If a does not belong to $S_1 \cap ... \cap S_n$, then at least there is one set, i.e., S_i does not include a and $f_i(a) \neq 0$), $i \in \{1, ..., n\}$, so the probability that $f_i(a)r_i(a) = 0$ is negligible (remember that the probability that $r_i(x)$ represents the elements of sets is negligible) and the probability that $F(a) = f_1(a)r_1(a) + ... + f_n(a)r_n(a) = 0$ is negligible. Thus if a does not belong to $S_1 \cap ... \cap S_n$, then the probability that a party whose set includes a determines wrongly that $a \in S_1 \cap ... \cap S_n$ is negligible. This completes the correctness proof.

Privacy. Before output phase, the protocol leaks no information about coefficients of the polynomials, or equivalently, the secret sets. In computation phase, the parties need to reconstruct some secrets when running the re-sharing protocol. In the re-sharing protocol, a party is allowed to reconstruct $Piece_i(c)$, a share of the multiplication of two shared secrets, say a, b. As there are at most t parties collude and t parties cannot recover the shared secrets, so the facts that parties reconstruct the $Piece_i(c)$s does not leak anything information about c, not even to say the secrets a or b.

In output phase, the parties only learn $S_1 \cap ... \cap S_n$ from $F(x)$. The polynomials $r_1(x)$, ..., $r_n(x)$ are random due to the fact that at least $n - t$ parties honestly choose random polynomials from $E(x)$ when jointly producing n random polynomials. The randomness of $r_1(x), ..., r_n(x)$ hides the elements at which the evaluation of $F(x)$ is nonzero. This completes the privacy proof. □

Theorem 2. *In the active model, suppose at most $t < n/3$ parties collude, the protocol in Section 3.2 is a solution to the private set intersection problem.*

Proof. An active adversary cannot disrupt the privacy of the protocol due to two reasons. First, less than $t + 1$ colluding parties cannot recover a shared secret before output phase according to the properties of verifiable secret sharing scheme. Second, the protocol allows at most t parties to collude. So an active who corrupts t parties cannot recover the shared secrets. The proof of privacy is similar to that of Theorem 1, and is omitted here.

Correctness. An active adversary cannot affects the correctness of the protocol. In input phase, the coefficients of $f_1(x), ..., f_n(x)$ are correctly t-shared, since verification of correct sharing is performed whenever a secret is shared. Similarly, the coefficients of the random polynomials $r_1(x), ..., r_n(x)$ are correctly t-shared among the parties. In the computation phase, the parties communicate only when the re-sharing protocol is recalled. In the re-sharing protocol, when a party, say P_i, re-shares his share, all parties jointly verify the correctness of sharing, and P_i is asked to prove that the shared value is indeed the shares that he holds.

So the computation phase is correct. The correctness of the output phase is similar to the proof of Theorem 1 and is omitted here. □

4.2 Efficiency

The protocol only uses simple operation like addition and multiplication, so we only consider the communication complexity of the protocol. The communication complexity is measured in elements in E.

The protocol in the passive model requires 3 rounds of communication. In input phase, the sharing in step 1 and step 2 can be executed in parallel in 1 round. In computation phase, the re-sharing of secrets needs 1 round of communication. In the output phase, parties publish their shares, which requires 1 round of communication.

The communication complexity of the protocol in the passive model is $\mathcal{O}(n^3k^2)$. The communication complexity is dominated by computation phase, where each party needs to share $n(k+1)(k+2)$ secret values and the total communication is $\mathcal{O}(n^3k^2)$.

In the active model, the protocol requires 6 rounds of communication. In input phase, sharing of secrets requires 1 round and verification of correct sharing requires 1 round. In computation phase, re-sharing of shares needs 1 round, verification of correct sharing needs 1 round, and proof that the shared value is indeed the required share needs 1 round. In output phase, publishing the shares needs 1 round.

The communication complexity of the protocol in the active model is $\mathcal{O}(n^4k^2)$. The communication complexity is dominated by computation phase. In computation phase, each party re-shares $n \times (k+1)^2$ secret values, which needs $n \times (k+1)(k+2) \times n^2$ elements to be sent. So the communication complexity of the computation phase is $\mathcal{O}(n^4k^2)$.

5 Conclusion

This paper considers the private set intersection problem in the information-theoretic model. We adopt the technique of polynomial representation of sets used in the previous protocols in the cryptographic model. By representing sets as polynomials, the set intersection problem is converted into the problem of computing the common roots of polynomials. This paper follows the share-compute-recover paradigm for the general protocols in the information-theoretic model, and presents a protocol based on the two-dimensional secret sharing scheme. The protocol consists of input phase, computation phase, and output phase. The protocol allows an active adversary to corrupt $t < n/3$ parties and demands 6 rounds of communication and $\mathcal{O}(n^4k^2)$ elements in a large finite field to be exchanged. It is interesting to consider whether the protocol can be improved with the "dispute control" technique [2] to allow $t < n/2$ colluding parties. In the information-theoretic model, other problems like cardinality set intersection, set union, etc. are also worthy of consideration.

Acknowledgments

The first author would like to show gratitude to Sujing Zhou for the helpful comments on the writing of this paper.

References

1. Beaver, D.: Efficient Multiparty Protocols Using Circuit Randomization. In: Feigenbaum, J. (ed.): Advances in Cryptology - CRYPTO'91. Lecture Notes in Computer Science, Vol. 576. Springer-Verlag Berlin Heidelberg (1992) 420–432
2. Beerliová-Trubíniová, Z., Hirt, M.: Efficient Multi-Party Computation with Dispute Control. In: Halevi, S., Rabin, T. (eds.): the third Theory of Cryptography Conference (TCC 2006). Lecture Notes in Computer Science, Vol. 3876. Springer-Verlag Berlin Heidelberg (2006) 305–328
3. Ben-Or, M., Goldwasser, S., Wigderson, A.: Completeness Theorems for Non-Cryptographic Fault-Tolerant Distributed Computation. In: 20th Annual Symposium on the Theory of Computing (STOC'88). ACM Press (1988) 1–10
4. Freedman, M. J., Nissim, K., Pinkas, B.: Efficient Private Matching and Set Intersection. In: Cachin, C., Camenisch, J. (eds.): Advances in Cryptology - EUROCRYPT'04. Lecture Notes in Computer Science, Vol. 3027. Springer-Verlag Berlin Heidelberg (2004) 1–19
5. Goldreich, O., Micali, S., Wigderson, A.: How to Play Any Mental Game — A Completeness Theorem for Protocols with Honest Majority. In: 19th ACM Symposium on the Theory of Computing (STOC'87). ACM Press (1987) 218–229
6. Hirt, M., Maurer, U., Przydatek, B.: Efficient Secure Multi-Party Computation. In: Okamoto, T. (ed.): Advances in Cryptology - ASIACRYPT'00. Lecture Notes in Computer Science, Vol. 1976. Springer-Verlag Berlin Heidelberg (2000) 143–161
7. Kissner, L., Song, D.: Privacy-Preserving Set Operations. In: Shoup, V. (ed.): Advances in Cryptology - CRYPTO'05. Lecture Notes in Computer Science, Vol. 3621. Springer-Verlag Berlin Heidelberg (2005) 241–257
8. Kissner, L., Song, D.: Private and Threshold Set-Intersection. Technical Report CMU-CS-05-113, Carnegie Mellon University, 2005.
9. Shamir, A.: How to Share a Secret. Communications of the ACM, Vol. 22. ACM Press (1977) 612–613
10. Yao, A. C.: Protocols for Secure Computations. In: 23rd IEEE Symposium on Foundations of Computer Science (FOCS'82). IEEE Computer Security (1982) 160–164