Pacific Journal of Mathematics for Industry
a SpringerOpen Journal

SURVEY Open Access

Secret computation of purchase history data using somewhat homomorphic encryption

Masaya Yasuda*, Takeshi Shimoyama and Jun Kogure

Abstract

We consider secret computation of purchase history data among two companies of different type of business in order to identify purchase patterns without revealing customer information of each company. Among several privacy-preserving approaches, we focus on homomorphic encryption, which is public-key encryption supporting meaningful computations on encrypted data. In particular, we apply the somewhat homomorphic encryption scheme proposed by Brakerski and Vaikuntanathan (CRYPTO 2011), which can support a limited number of both additions and multiplications over polynomials. The main contribution is to introduce a practical packing method in the scheme to efficiently compute the set intersection of purchase history data over packed ciphertexts. Furthermore, we implemented the scheme for several parameters corresponding to various security levels, and demonstrate the efficiency of our packing method. We hope that this work would give the first practical usage of somewhat homomorphic encryption in marketing analysis.

Keywords: Homomorphic encryption; LWE assumption; Secure inner product; Packing method

1 Introduction

Recently, in Japan, services with a rewards card (or called a loyalty card) commonly used among companies of different type of business, for example, "T-card" and "Ponta-card", have been paid to much attention[a]. As a fascination to use such cards, there are several advantages; For tie-up companies, they can have more opportunities to acquire new customers, and obtain market trend data from companies of different type of business. On the other hand, for customers using such a card, they can collect rewards points from companies of different type of business and bring such points together on the only one card. In particular, the biggest advantage is that tie-up companies can collect market and customers information exceeding the frame of their own type of business, and to use the information for so-called *market basket analysis*, which is one of the marketing analyses in order to identify purchase patterns (e.g., to identify what items tend to be purchased together, sequentially or by seasons).

However, at the same time, some problems would be caused in handling customers information among tie-up companies. For example, when purchase history data are analyzed, it needs to share both customer ID and purchase history data among tie-up companies. In this case, customer information of each company would be revealed to the other companies, and hence some problems related to the customer's privacy might be feared. Furthermore, since purchase history data of each company are directly related to its own sales, the data should be secret to the other companies (see [23], Section 1 for discussion on these issues).

1.1 Application scenario

Looking back on the above issues, we consider the following scenario; "Assume that there are two tie-up companies A and B and they only share their customers ID (e.g., the rewards card number can be used as a customer ID). The two companies A and B have their own customers purchase history data of items X and Y, respectively. Then they would like to know the number of customers who bought both items X and Y in order to identify how much the items are purchased sequentially, without revealing each customer purchase history data to one another".

*Correspondence: yasuda.masaya@jp.fujitsu.com
Fujitsu Laboratories Ltd., 1-1, Kamikodanaka 4-chome, Nakahara-ku, Kawasaki 211-8588, Japan

1.2 Homomorphic encryption

Among currently known privacy-preserving approaches, we apply homomorphic encryption to the application scenario of Section 1.1. At present, there are the following three main types in homomorphic encryption, depending on operations on encrypted data which each type can support.

Additively homomorphic encryption It can support only additions on encrypted data. Paillier scheme [17] and additive ElGamal scheme [7] are typical.

Somewhat homomorphic encryption (SHE) It can support both additions and multiplications on encrypted data, but the number of possible operations is limited. The first construction of such encryption was the BGN scheme [1] based on pairings over elliptic curves. However, the BGN scheme can handle a number of additions but only depth-one multiplications. After Gentry's breakthrough [9,10] of constructing an FHE scheme (see below for FHE), a number of new SHE schemes have been proposed as a building block of FHE, for example, ideal lattices based schemes [9-11], integers based schemes [6,18], and finally learning with errors (LWE) based schemes [2-4]. Unlike the BGN scheme, these schemes can handle additions and multiplications of depth greater than one.

Fully homomorphic encryption (FHE) It can support "any operations" on encrypted data, including the unlimited number of additions and multiplications. In 2009, Gentry in [9,10] proposed a new method to construct an FHE scheme from the SHE scheme based on ideal lattices, whose method is called *bootstrapping*. Currently FHE schemes have some problems mainly including slow performance and the big encrypted data size, and hence FHE is believed to need a long way for practical usage (see [6,11] for their implementation results of "pure" FHE schemes, also [12] for the recent work of implementing a "leveled" FHE scheme).

1.3 Previous work

In the scenario of Section 1.1, assume that two companies A and B have their own customer purchase history data represented by $(x_1, x_2, \ldots, x_m)$ and $(y_1, y_2, \ldots, y_m)$, respectively, where $x_i, y_i \in \{0, 1\}$ denote the purchase history of items X and Y of the customer with ID-i for each $i = 1, \cdots, m$ (let m denote the number of customer's ID). As for the item X, the value $x_i = 1$ (resp. $x_i = 0$) means that the customer with ID-i bought (resp. did not buy) the item X. Under these assumptions, the inner product $\sum_{i=1}^{m} x_i \cdot y_i$ gives our desired result, namely, the number of customers who bought both items X and Y (i.e., the set intersection of purchase history data). In order to evaluate the inner product by homomorphic encryption, we need to apply either an SHE or FHE scheme, and SHE would be suitable in practice. Then, using the SHE scheme based on ideal lattices, the authors in [23] proposed a secret computation model for the application scenario of Section 1.1. In their model, the cloud is assumed to be used as an outsourcing computation resource, and a trusted assayer is involved in order to identify purchase patterns of items X and Y. The flow of their secret computation is as follows (see also Figure 1):

1. The trusted assayer generates the public key pk and the secret key sk of the SHE scheme, and distributes only the public key pk to the public.
2. Using pk, each company encrypts its own purchase history data $(x_1, \ldots, x_m)$ or $(y_1, \ldots, y_m)$, and sends the encrypted data $(\mathsf{Enc}(x_1), \ldots, \mathsf{Enc}(x_m))$ or $(\mathsf{Enc}(y_1), \ldots, \mathsf{Enc}(y_m))$ with customer's ID to the cloud (using the bit-wise encryption). Since all data are protected by encryption, each company's purchase history data cannot be revealed to one another.
3. In the ascending order of customer's ID, the cloud arranges $(\mathsf{Enc}(x_1), \ldots, \mathsf{Enc}(x_m))$ and $(\mathsf{Enc}(y_1), \ldots, \mathsf{Enc}(y_m))$ as in Figure 1. Then the cloud computes the inner product

$$\mathsf{ct} = \sum_{i=1}^{m} \mathsf{Enc}(x_i) \cdot \mathsf{Enc}(y_i) \tag{1}$$

   on encrypted data (note that ct is the ciphertext of the inner product $\sum_{i=1}^{m} x_i \cdot y_i$ due to homomorphic property), and only sends the encrypted result ct to the assayer. Since the cloud has no the secret key, the cloud cannot learn any information about the purchase history data of each company.
4. Using sk, the assayer decrypts the encrypted result ct to obtain the desired inner product $\sum_{i=1}^{m} x_i \cdot y_i$, which enables the assayer to identify purchase patterns of items X and Y.

1.4 Our contributions

As described in Section 1.3, the authors in [23] adopted the bit-wise encryption in the SHE scheme based on ideal lattices to compute a secure inner product, which would cause difficulty mainly on the encrypted data size (e.g., we need 10,000 ciphertexts for $m = 10,000$ customers). Furthermore, since it requires m times additions and m times

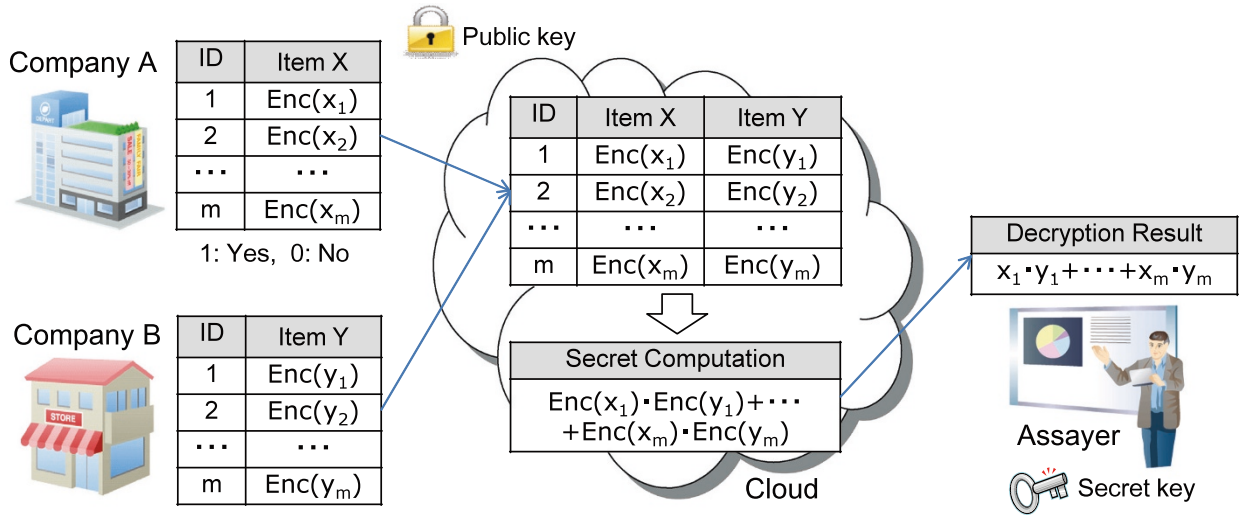

Figure 1 **The secret computation model of purchase history data between two companies proposed in [23].**

multiplications on encrypted data for the secure computation (1), slow performance is also concerned. Then, the aim of this work is to improve their work for reduction of both the performance and the encrypted data size. In the following, we summarize our contributions:

- Unlike [23], we apply the SHE scheme proposed by Brakerski and Vaikuntanathan [4], which is based on a simplified version of the ring-LWE assumption of [14].
- We propose a method in the SHE scheme to pack a vector of certain length into a single ciphertext, which enables to efficiently compute a secure inner product. By this method, we can reduce both the encrypted data size and the performance considerably. Hence the SHE scheme with our packing method could be practically used in various applications.
- To demonstrate the efficiency, we implemented the SHE scheme with our packing method. While the work [22] only implemented the scheme of lattice dimension 2048, this work gives more detailed implementation results for several lattice dimensions 2048, 4096, 8192 and 16384. Furthermore, our implementation is optimized by using inline assembly language in C programs, and hence it gives faster performance than the previous implementation results of [16] in the same scheme.

Remark 1. Our method specializes in the structure of the special ring $\mathbb{Z}[x]/(x^n+1)$, which is used in the construction of the SHE scheme of [4] (see Section 2 below for the construction). Therefore, our packing method can be applied in the scheme based on ideal lattices, and the BGV scheme [2] (the performance and the encrypted data size in these schemes are estimated to be almost the same as in this work). On the other hand, our packing method cannot be applied in the BGN scheme [1] since the scheme is based on pairings over elliptic curves. More specifically, our method needs to use certain polynomial transformations in $\mathbb{Z}[x]/(x^n+1)$, but the BGN scheme cannot support such polynomial transformations. Then we only can use the bit-wise encryption in the BGN scheme for a secure inner product as in Section 1.3. Furthermore, as discussed in [16], the homomorphic multiplication of the BGN scheme is slower than that of the SHE scheme of [4] under almost the same security level (such as 128-bit), and hence the SHE scheme with our packing method is estimated to give much faster performance than the BGN scheme with the bit-wise encryption for a secure inner product.

Basic notation The symbols $\mathbb{Z}$, $\mathbb{Q}$, and $\mathbb{R}$ denote the ring of integers, the field of rational numbers, and the field of real numbers, respectively. For a prime number p, the finite field with p elements is denoted by $\mathbb{F}_p$. For two integers z and d, let $[z]_d$ denote the reduction of z modulo d included in the interval $[-d/2, d/2)$ (the reduction of z modulo d included in the interval $[0,d)$ is denoted by $z \bmod d$ as usual). For a vector $\vec{A} = (a_1, a_2, \ldots, a_n) \in \mathbb{R}^n$, let $||\vec{A}||_\infty$ denote the ∞-norm defined by $\max_i |a_i|$. Let $\langle \vec{A}, \vec{B} \rangle$ denote the inner product of two vectors $\vec{A}$ and $\vec{B}$. Finally, we let $\lg(q)$ denote the logarithm value of an integer q with base 2.

2 Somewhat homomorphic encryption

In this section, we briefly review the construction and the correctness of the SHE scheme proposed by Brakerski and Vaikuntanathan [4]. The security of the scheme relies

on the polynomial LWE assumption defined below, which can be regarded as a simplified version of the ring-LWE assumption of Lyubashevsky, Peikert and Regev [14] (see [4], Section 2 for details of the assumption).

Definition 1 (Polynomial LWE assumption)**.** For a security parameter λ, let $f(x) = x^n + 1$ be the cyclotomic polynomial for an integer $n = n(\lambda)$ of 2-power. Let $q = q(\lambda)$ be an integer and set $R = \mathbb{Z}[x]/(f(x))$ and $R_q = R/qR$. Let $\chi = \chi(\lambda)$ be a distribution over R. Then the polynomial LWE assumption $\mathsf{PLWE}_{n,q,\chi}$ is that it is infeasible to distinguish the following two distributions:

1. One samples (a, b) uniformly from $(R_q)^2$.
2. One draws $s \leftarrow \chi$ uniformly and samples (a, b) by sampling $a \leftarrow R_q$ uniformly, $e \leftarrow \chi$ and setting $b = as + e$.

2.1 Construction of the SHE scheme

The following four parameters are needed for the scheme construction:

- n: an integer of 2-power, which defines the base ring $R = \mathbb{Z}[x]/(f(x))$ with the cyclotomic polynomial $f(x) = x^n + 1$ of degree n as in Definition 1. This degree n is often called the *lattice dimension*.
- q: a prime number with $q \equiv 1 \bmod 2n$, which defines the base ring $R_q = \mathbb{F}_q[x]/(f(x))$ of ciphertext space. The condition $q \equiv 1 \bmod 2n$ is not necessary for the scheme construction, but it is required to discuss the provable security [4], Theorem 1.
- t: an integer with $t < q$ to determine a plaintext space $R_t = (\mathbb{Z}/t\mathbb{Z})[x]/(f(x))$ (t is not necessarily prime).
- σ: the parameter to define a discrete Gaussian error distribution $\chi = D_{\mathbb{Z}^n,\sigma}$ with the standard deviation σ, namely, we select each entry in an n-dimensional vector by sampling from a Gaussian distribution $N(0, \sigma)$, and then round it to the nearest integer. In practice, we choose relatively small value such as $\sigma = 4 \sim 8$.

Key generation We first choose an element $R \ni s \leftarrow \chi$, and sample a uniformly random element $a_1 \in R_q$ and an error $R \ni e \leftarrow \chi$. Then set $\mathsf{pk} = (a_0, a_1)$ with $a_0 = -(a_1 s + te)$ as the public key and $\mathsf{sk} = s$ as the secret key.

Encryption For a plaintext $m \in R_t$ and the public key $\mathsf{pk} = (a_0, a_1)$, the encryption samples $R \ni u, f, g \leftarrow \chi$ and computes the "fresh ciphertext" given by

$$\begin{aligned}\mathsf{Enc}(m, \mathsf{pk}) &= (c_0, c_1) \in (R_q)^2 \\ &= (a_0 u + tg + m, a_1 u + tf), \end{aligned} \tag{2}$$

where $m \in R_t$ is considered as an element of R_q in the natural way due to the condition $t < q$.

Homomorphic operations While the above encryption algorithm generates ciphertexts with only two ring elements, the homomorphic multiplication defined below makes the ciphertext length longer. Therefore we need to define homomorphic operations for ciphertexts of any length as follows: Let $\mathsf{ct}' = (c'_0, \ldots, c'_\xi) \in (R_q)^{\xi+1}$, $\mathsf{ct}'' = (c''_0, \ldots, c''_\eta) \in (R_q)^{\eta+1}$ be two ciphertexts. The homomorphic addition "$\dot{+}$" is computed by component-wise addition of ciphertexts, namely, we have

$$\mathsf{ct}' \dot{+} \mathsf{ct}'' = (c'_0 + c''_0, \ldots, c'_{\max(\xi,\eta)} + c''_{\max(\xi,\eta)}),$$

by padding with zero if necessary. Similarly, the homomorphic subtraction is computed by component-wise subtraction. On the other hand, the homomorphic multiplication "$*$" is computed by

$$\mathsf{ct}' * \mathsf{ct}'' = (\hat{c}_0, \ldots, \hat{c}_{\xi+\eta}),$$

where we consider ciphertexts $\mathsf{ct}', \mathsf{ct}''$ as elements of $R_q[z]$ by an embedding map $(R_q)^r \ni (\nu_0, \ldots, \nu_{r-1}) \mapsto \sum_{i=0}^{r-1} \nu_i z^i \in R_q[z]$ for any $r \geq 1$, and compute

$$\sum_{i=0}^{\xi+\eta} \hat{c}_i z^i = \left(\sum_{i=0}^{\xi} c'_i z^i\right) \cdot \left(\sum_{i=0}^{\eta} c''_i z^i\right) \in R_q[z]\,.$$

Decryption For any (fresh or non-fresh) ciphertext $\mathsf{ct}' = (c'_0, \ldots, c'_\xi) \in (R_q)^{\xi+1}$, the decryption with the secret key $\mathsf{sk} = s$ is computed by

$$\mathsf{Dec}(\mathsf{ct}', \mathsf{sk}) = [\tilde{m}]_q \bmod t \in R_t,$$

where $\tilde{m} = \sum_{i=0}^{\xi} c'_i s^i \in R_q$. For the vector $\vec{s} = (1, s, s^2, \ldots)$ (called the *secret key vector*), we can also rewrite

$$\mathsf{Dec}(\mathsf{ct}', \mathsf{sk}) = [\langle \mathsf{ct}', \vec{s} \rangle]_q \bmod t.$$

Let $\mathsf{ct} = (c_0, c_1)$ be a fresh ciphertext given by (2). Since $a_0 + a_1 s = -te$, we have

$$\begin{aligned}\langle \mathsf{ct}, \vec{s} \rangle &= (a_0 u + tg + m) + s \cdot (a_1 u + tf) \\ &= m + t \cdot (g + sf - ue)\end{aligned}$$

in the ring R_q. If the value $m + t \cdot (g + sf - ue)$ does not wrap around mod q (i.e., all errors $R \ni e, f, g, u \leftarrow \chi$ must be sufficiently small), we have

$$[\langle \mathsf{ct}, \vec{s} \rangle]_q = m + t \cdot (g + sf - ue)$$

in the ring "R" (see also Lemma 2 below for the condition of successful decryption). In this case, we can recover the correct plaintext m by mod t-operation, which shows the decryption mechanism for fresh ciphertexts. Furthermore, for two fresh ciphertexts $\mathsf{ct}_1, \mathsf{ct}_2$, we clearly have

$$\begin{aligned}\langle \mathsf{ct}_1 \dot{+} \mathsf{ct}_2, \vec{s} \rangle &= \langle \mathsf{ct}_1, \vec{s} \rangle + \langle \mathsf{ct}_2, \vec{s} \rangle \\ \langle \mathsf{ct}_1 * \mathsf{ct}_2, \vec{s} \rangle &= \langle \mathsf{ct}_1, \vec{s} \rangle \cdot \langle \mathsf{ct}_2, \vec{s} \rangle\end{aligned}$$

in the ring R_q. These two equations help us to understand the construction and the correctness of homomorphic operations in the encryption scheme, but please refer ([4], Section 1.1) for details. Here we also give a lemma on the "cryptographic security" of the scheme constructed above (see [4] for details).

Lemma 1 (security)**.** *Given* (n, q, t, σ)*, the scheme is provably secure in the sense of IND-CPA under the polynomial LWE assumption* $\mathsf{PLWE}_{n,q,\chi}$ *with* $\chi = D_{\mathbb{Z}^n,\sigma}$ *(see Definition 1 for the definition of* $\mathsf{PLWE}_{n,q,\chi}$*).*

2.2 Correctness of the SHE scheme

By *correctness*, we mean that the decryption can recover the operated result over plaintexts after some homomorphic operations over ciphertexts. For the scheme constructed above, the homomorphic operations over ciphertexts correspond to the ring structure of the plaintext space R_t, namely, we have

- (Addition) $\mathsf{Dec}(\mathsf{ct} \dotplus \mathsf{ct}', \mathsf{sk}) = m + m' \in R_t$, and
- (Multiplication) $\mathsf{Dec}(\mathsf{ct} * \mathsf{ct}', \mathsf{sk}) = m \times m' \in R_t$

for ciphertexts $\mathsf{ct}, \mathsf{ct}'$ corresponding to plaintexts m, m', respectively. However, the scheme merely gives an SHE scheme (not FHE), and its correctness holds under the following condition (see the proof of [16], Lemma 3.3):

Lemma 2 (Condition for successful decryption)**.** *For a ciphertext* ct*, the decryption* $\mathsf{Dec}(\mathsf{ct}, \mathsf{sk})$ *recovers the correct result if* $\langle \mathsf{ct}, \vec{s} \rangle \in R_q$ *does not wrap around mod q, namely, if the condition*

$$||\langle \mathsf{ct}, \vec{s} \rangle||_\infty < \frac{q}{2} \tag{3}$$

is satisfied, where for $a = \sum a_i x^i \in R_q$ *let* $||a||_\infty = \max |a_i|$ *denote the* ∞*-norm of its coefficient representation.*

3 Practical packing method

For reduction of both the encrypted data size and the performance, Lauter, Naehrig and Vaikuntanathan [16] introduce some message encoding techniques in the SHE scheme of [4]. Their main technique is to encode integers in a single ciphertext so that it enables to efficiently compute their sums and products over the integers. Their method first breaks an integer M of at most n bits into a binary vector $(M_0, \ldots, M_{n-1})$, creates a polynomial given by

$$\mathsf{pm}(M) := \sum_{i=0}^{n-1} M_i x^i, \tag{4}$$

and finally encrypts M as

$$\mathsf{ct}_{\mathrm{pack}}(M) := \mathsf{Enc}\left(\mathsf{pm}(M), \mathsf{pk}\right), \tag{5}$$

where $\mathsf{pm}(M)$ is regarded as an element of R_t for sufficiently large t. Note that we clearly have $\mathsf{pm}(M)|_{x=2} = M$ for any integer M of n bits, where $a(x)|_{x=2}$ denotes the value substituted $x = 2$ for a polynomial $a(x)$ (i.e., the value $a(2)$). For two integers M, M' of n bits, the homomorphic addition of $\mathsf{ct}_{\mathrm{pack}}(M)$ and $\mathsf{ct}_{\mathrm{pack}}(M')$ gives the polynomial addition $\mathsf{pm}(M) + \mathsf{pm}(M')$ on encrypted data by the correctness of the encryption scheme, and it also gives the integer addition $M + M'$ since

$$\mathsf{pm}(M) + \mathsf{pm}(M')\big|_{x=2} = M + M'.$$

However, the integer multiplication $M \cdot M'$ causes a problem since the polynomial multiplication $\mathsf{pm}(M) \cdot \mathsf{pm}(M')$ has larger degree than n in general. Their solution to address the problem is to encode integers of at most n/d bits if we need to perform d homomorphic multiplications. Then their method is acceptable in computing multiplications of low depth, such as the standard deviation which can be calculated by depth-one multiplications (in fact, the authors in [16] apply their packing method to compute simple statics such as the mean, the standard deviation, and the logistical regression on encrypted data).

3.1 Our packing method

In contrast to the packing method of [16], we present a new one. Our method is based on [16], and it can be considered as an extension of the method of [16]. Specifically, we give "two types of packed ciphertexts" in order to make use of the ring structure of the plaintext space R_t for a secure inner product over packed ciphertexts. Now let us define our packing method.

Definition 2. For a vector $\vec{A} = (A_0, \ldots, A_{n-1})$ of length n, we define two types of packed ciphertexts as follows:

1. As in the equation (4), set

   $$\mathsf{pm}_1(\vec{A}) := \sum_{i=0}^{n-1} A_i x^i.$$

   For sufficiently large t, we consider the above polynomial to be an element of R_t. As well as (5), we then define

   $$\mathsf{ct}^{(1)}_{\mathrm{pack}}(\vec{A}) := \mathsf{Enc}\left(\mathsf{pm}_1(\vec{A}), \mathsf{pk}\right)$$

   as the packed ciphertext of the first type. This type is the same as given in [16].
2. Unlike the first type, set

   $$\mathsf{pm}_2(\vec{A}) := -\sum_{i=0}^{n-1} A_i x^{n-i}.$$

As the second type, we define

$$\mathsf{ct}^{(2)}_{\mathrm{pack}}(\vec{A}) := \mathsf{Enc}\left(\mathsf{pm}_2(\vec{A}), \mathsf{pk}\right).$$

This type is always needed for efficient computation of secure inner product (see Theorem 1 below).

Our packing method can pack a vector of length n into a single ciphertext irrespective of types. Hence, compared to coefficient-wise encryption, our method can reduce the encrypted data size considerably.

3.2 Secure inner product computation

Due to two types of our packing method, we have the following result on secure inner product computation:

Theorem 1 (Secure inner product computation)**.** *For two vectors $\vec{A}, \vec{B}$ of length n, let* ct *denote the ciphertext given by the homomorphic multiplication*

$$\mathsf{ct}^{(1)}_{\mathrm{pack}}(\vec{A}) * \mathsf{ct}^{(2)}_{\mathrm{pack}}(\vec{B}) \tag{6}$$

of two types of packed ciphertexts. Let m_0 denote the constant term of the decryption result $\mathsf{Dec}(\mathsf{ct}, \mathsf{sk}) \in R_t$*. Then we have*

$$m_0 \equiv \left\langle \vec{A}, \vec{B} \right\rangle \bmod t.$$

In other words, the constant term of the decryption result gives the inner product of two vectors $\vec{A}$ and $\vec{B}$ for sufficiently large t.

Proof. Since the homomorphic operations over ciphertexts correspond to the ring structure of the plaintext space R_t (see Section 2.2), the decryption result $\mathsf{Dec}(\mathsf{ct}, \mathsf{sk})$ is equal to the polynomial multiplication $\mathsf{pm}_1(\vec{A}) \times \mathsf{pm}_2(\vec{B})$ in R_t. Set $\vec{A} = (A_0, \ldots, A_{n-1})$ and $\vec{B} = (B_0, \ldots, B_{n-1})$. Then

$$\begin{aligned}
\mathsf{pm}_1(\vec{A}) \times \mathsf{pm}_2(\vec{B}) &= \left(\sum_{i=0}^{n-1} A_i x^i\right) \times \left(-\sum_{j=0}^{n-1} B_j x^{n-j}\right) \\
&= -\sum_{i=0}^{n-1} A_i B_i x^n + (\text{the other terms}) \\
&= \left\langle \vec{A}, \vec{B} \right\rangle + (\text{non-constant terms})
\end{aligned}$$

in R_t since $x^n = -1$. This completes the proof. □

The result of Theorem 1 tells that our packing method enables to compute the inner product of two vectors of length n by "only one homomorphic multiplication" over packed ciphertexts (cf. it needs n homomorphic additions and n homomorphic multiplications in the component-wise encryption to obtain the same inner product). Actually, in Definition 2, we take two types of our packing method so that Theorem 1 holds. Specifically, the key point in the above proof is multiplication between two types of our transformed polynomials $\mathsf{pm}_1(\vec{A})$ and $\mathsf{pm}_2(\vec{B})$ in the ring R_t. Our basic idea is derived from the *convolution* of polynomials, which we encounter in various mathematical fields. More specifically, given two polynomials

$$\begin{cases} a(x) = a_0 + a_1 x + \cdots + a_{n-1} x^{n-1}, \\ b(x) = b_0 + b_1 x + \cdots + b_{n-1} x^{n-1}. \end{cases}$$

The product of $a(x)$ and $b(x)$ is called the convolution. Furthermore, for any $0 \le h \le n-1$, the x^h-coefficient of the convolution is the sum $\sum_{i=0}^{h} a_i b_{h-i}$ of products a_i, b_j for which $i + j = h$. In particular, in the case $h = n-1$, the above sum is similar to our desired inner product $\sum_{i=0}^{n-1} a_i b_i$, and the difference is only the index of b_j's. Then our trick for the inner product is to "re-arrange the coefficients of $b(x)$ by converse order" (see the second polynomial transformation in Definition 2, in which it needs the minus sign in addition due to we have $x^n = -1$ in the special ring $\mathbb{Z}[x]/(x^n + 1)$).

Remark 2 (Privacy enhance technique)**.** While our packing method can give efficient performance for a secure inner product, it gives a decryptor (i.e., the assayer in Figure 1) more information than his desired result and hence it may cause a new privacy issue that the decryptor may know extra information about purchase history data of customers. In fact, the computation (6) gives the inner product in the constant term, but it also includes extra information in the other terms (then the decryptor can know the extra information by decryption). The simplest method to conceal the extra information against the decryptor is to add random data in a ciphertext. Specifically, for a ciphertext $\mathsf{ct} = (c_0, c_1, c_2) \in (R_q)^3$ given by the computation (6) (note that ct has three components generated by one homomorphic multiplication), we first generate a random polynomial

$$r(x) = r_1 x + \cdots + r_{n-1} x^{n-1} \in R$$

and then compute a ciphertext given by

$$\mathsf{ct}' = (c_0 + r(x), c_1, c_2).$$

Then the decryption of the ciphertext ct' includes our desired inner product in the constant term, but the other terms are masked by random information r_i's (then the decryptor can not know extra information).

Remark 3 (Other applications)**.** As well as the secure inner product computation (6), linear combinations of two types of packed ciphertexts can give several meaningful computations such as (see also our related work [21])

- private statistic (e.g., sum and standard deviation),
- statistical analysis (e.g., covariance), and
- distances (e.g., the Hamming distance).

Especially, secure Hamming distance can be applied in privacy-preserving biometrics to measure the similarity of two biometric feature vectors on encrypted data. Please see our previous works [19,21] for secure Hamming distance in SHE schemes (note that the work [19] uses the SHE scheme based on ideal lattices of [11]). Furthermore, our method can be applied to efficient computation of multiple Hamming distance values for secure pattern matching (see [20]).

4 Parameters setting of the scheme

Here we discuss how to choose parameters (n, q, t, σ) of the SHE scheme suitable (maybe not optimal) for the secure inner product (6) over packed ciphertexts, and we give several parameters of more than 80-bit security level. For simplicity, we only consider secure inner product between two binary vectors $\vec{A}, \vec{B}$ of length n. In this case, we can pack each of $\vec{A}$ and $\vec{B}$ into a single ciphertext with our packing method (see the below diagram). In the application scenario of Section 1.1, we assume that the number m of customer's ID is smaller than the lattice dimension n (see our previous work [22] for the case $m > n$), and two binary vectors $\vec{A}$ and $\vec{B}$ represent purchase history data of items X and Y, respectively (note that it is different from the representation of purchase history data in Figure 1).

$$\vec{A} \xrightarrow{\text{packed enc.}} \mathsf{ct}^{(1)}_{\text{pack}}(\vec{A}) \searrow$$

$$\text{ctgiven by (6)} \xrightarrow{\text{dec.}} \langle \vec{A}, \vec{B} \rangle$$

$$\vec{B} \xrightarrow[\text{packed enc.}]{} \mathsf{ct}^{(2)}_{\text{pack}}(\vec{B}) \nearrow$$

4.1 Correctness and security

For the correctness, by Lemma 2, we need to satisfy the condition (3) for the ciphertext ct. Furthermore, it follows from the proof of ([16], Lemma 3.3) that we have

$$\langle \mathsf{ct}, \vec{s} \rangle = \left\langle \mathsf{ct}^{(1)}_{\text{pack}}(\vec{A}), \vec{s} \right\rangle \cdot \left\langle \mathsf{ct}^{(2)}_{\text{pack}}(\vec{B}), \vec{s} \right\rangle$$

in the ring R_q. When we set U to be an upper bound of the ∞-norm size $||\langle \mathsf{ct}', \vec{s} \rangle||_\infty$ for any fresh ciphertext ct', the above equation gives an inequality $|| \langle \mathsf{ct}, \vec{s} \rangle ||_\infty \leq nU^2$ by $||\langle \mathsf{ct}^{(1)}_{\text{pack}}(\vec{A}), \vec{s} \rangle||_\infty, ||\langle \mathsf{ct}^{(2)}_{\text{pack}}(\vec{B}), \vec{s} \rangle||_\infty \leq U$ and the well-known fact

$$||a+b||_\infty \leq ||a||_\infty + ||b||_\infty,\ ||a \cdot b||_\infty \leq n \cdot ||a||_\infty \cdot ||b||_\infty$$

for any two elements $a, b \in R_q$. As in [16], we take U to be the value $2t\sigma^2\sqrt{n}$, which is an experimental estimation. Then we have $||\langle \mathsf{ct}, \vec{s} \rangle||_\infty \leq nU^2 \leq 4n^2t^2\sigma^4$ by the above inequality. Therefore, by the inequality (3), we estimate that the correctness for the ciphertext ct is satisfied if

$$8n^2t^2\sigma^4 < q, \tag{7}$$

which condition gives a lower bound of q for the correctness.

By Lemma 1, the security of the scheme relies on the polynomial LWE assumption $\mathsf{PLWE}_{n,q,\chi}$ in Definition 1. The security analysis in [16] is based on the methodology of Lindner and Peikert [13] for the standard LWE problem. However, we still use their security analysis as in [16]. According to [13], there are two efficient attacks against the general LWE problem (e.g., see [13] for details of the two attacks);

- the distinguishing attack of [15], and
- the decoding attack proposed by [13].

The analysis of [13] shows that the decoding attack is always better than the distinguishing one, but the two attacks seem to have a similar performance for practical advantages such as $\varepsilon = 2^{-32}$ and 2^{-64} (the advantage ε of attackers means the success probability to distinguish the two distributions given in Definition 1). Therefore we only need to consider the security against the distinguishing attack as in [16]. The security of lattice-based cryptographic schemes can be measured by the *root Hermite factor*. According to the analysis of [13], for given parameters (n, q, t, σ), we have a relation between n, q and δ given by

$$c \cdot q/\sigma = 2^{2\sqrt{n \cdot \lg(q) \cdot \lg(\delta)}}, \tag{8}$$

where c is the constant determined by the attack advantage ε ($c \approx \sqrt{\lg(1/\varepsilon)/(\lg 2 \cdot \pi)}$ by [13]), and we here take $c = 3.758$ corresponding to $\varepsilon = 2^{-64}$ (only two values 2^{-32}, 2^{-64} are considered for ε in [16], and we take just one of the two values in this paper).

4.2 Chosen parameters and security levels

As in [16], set $\sigma = 8$ as the standard deviation parameter of the distribution $\chi = D_{\mathbb{Z}^n,\sigma}$ to make the SHE scheme secure against combinatorial style attacks (see also [8]). We also set $t = n$ as the modulus parameter of the plaintext space, which is sufficient for computing the inner product between two binary vectors $\vec{A}$ and $\vec{B}$ of length n. To obtain various security levels, we take four lattice dimension parameters $n = 2048, 4096, 8192$ and 16384 (as the lattice dimension is larger, the security becomes higher). Then for each lattice parameter n, the parameter q is determined by the condition (7), and then the root Hermite factor δ is computed by the relation (8). In Table 1, we show our chosen parameters of the

Table 1 Chosen parameters (n, q, t, σ) of the SHE scheme for the secure inner product (6) over packed ciphertexts (δ denotes the corresponding root Hermite factor, and t_{Adv} the estimated running time of the distinguishing attack given by equation (9))

	n	q	t	σ	δ	t_{Adv}
(i)	2048	61-bit	n	8	1.00499	140-bit
(ii)	4096	65-bit	n	8	1.00266	400-bit
(iii)	8192	69-bit	n	8	1.00141	775-bit
(iv)	16384	73-bit	n	8	1.00075	1554-bit

SHE scheme for the secure inner product (6) over packed ciphertexts.

According to the state-of-the-art security analysis of Chen and Nguyen [5] for lattice-based cryptographic schemes, a root Hermite factor smaller than $\delta = 1.0050$ is estimated to have more than 80-bit security level with an enough margin. Therefore the four parameters in Table 1 are estimated to have 80-bit security level against the distinguishing attack with advantage $\varepsilon = 2^{-64}$, and also against the more powerful decoding attack (note that as mentioned in Section 4.1 the two attacks has similar performance for $\varepsilon = 2^{-64}$). In particular, note that the parameter (i) in Table 1 is similar as the parameter $(n, q, t, \sigma) = (2048, 58\text{-bit}, 1024, 8)$ with $\delta = 1.0046$ included in [16], Table 1, which is estimated to have more than 120-bit security level according to the security analysis of [13]. On the other hand, the authors in [13] simply estimate the running times for the NTL implementation of the BKZ algorithm, which is one of the most practical lattice reduction algorithms. They also derive a relation of the expected running time t_{Adv} of the distinguishing attack with the root Hermite factor δ by

$$\lg(t_{\mathsf{Adv}}) = \frac{1.8}{\lg(\delta)} - 110. \tag{9}$$

For chosen parameters (i)-(iv) in Table 1, we give the expected running time t_{Adv} computed by the relation (9) also in Table 1. However, their security analysis seems no longer state-of-the-art due to the old NTL implementation. Therefore we remark that the t_{Adv}-data in Table 1 are just at the reference level, but the data tell a rough standard of the security level of each parameter. For example, the parameter (iv) in Table 1 is estimated to have much more than 1000-bit security level against the distinguishing attack with advantage $\varepsilon = 2^{-64}$.

5 Implementation results

For the four parameters (i)-(iv), we implemented the SHE scheme with our packing method for the secure inner product computation (6). Our experiments ran on an Intel Xeon X3480 at 3.07 GHz with 16 GB memory, and we used our own software library using inline assembly language x86_64 in C programs for all computations in the base ring $R_q = \mathbb{F}_q[x]/(x^n + 1)$ of ciphertext space. Our C code was complied using gcc 4.6.0 on Linux. In order to obtain efficient multiplication in R_q, we implemented the Montgomery reduction algorithm for all parameters, and the Karatsuba multiplication algorithm for only the parameter (i), and the multiplication algorithm using the FFT (Fast Fourier Transform) method for (ii)-(iv). Actually, our experiments show that the Karatsuba method is about twice faster than the FFT method for the parameter (i). On the other hand, the FFT method is faster than the Karatsuba method for the parameters (ii)-(iv) (e.g., it is about 5 times faster for the largest parameter (iv)).

In Table 2, we summarize the performances and the sizes of the SHE scheme with our packing method. As described in Section 3.1, our packing method can pack a binary vector of length less than n in a single ciphertext, and the computation (6) enables to compute the inner product between two binary vectors of length n over packed ciphertexts. In the following, we describe details for only the parameter (i) (cf. implementation results of ([16], Section 5) by the computer algebra system MAGMA):

Sizes The size of $\mathsf{pk} = (a_0, a_1) \in R_q^2$ is $2n \cdot \lg(q) \approx 31.2$ KB, and the size of $\mathsf{sk} = s \in R_q$ is $n \cdot \lg(q) \approx 15.6$ KB. A fresh ciphertext has two elements in the ring R_q, and hence its size is $2n \cdot \lg(q) \approx 31.2$ KB. Therefore the size of packed ciphertexts $\mathsf{ct}^{(1)}_{\mathrm{pack}}(\vec{A})$ and $\mathsf{ct}^{(2)}_{\mathrm{pack}}(\vec{B})$ is about 31.2 KB, respectively. In contrast, the non-fresh ciphertext ct given by (6) has three ring elements, and its size is about 46.8 KB. Note that the size of a non-fresh ciphertext depends on the number of its ring elements, whose number can be increased by homomorphic multiplications. However, in this paper, we only consider up to three elements for the computation (6), which can be calculated by only one homomorphic multiplication.

Table 2 The performances and the sizes of the SHE scheme with our packing method for the secure inner product computation (6) (parameters of the SHE scheme are given in Table 1)

Parameters	Packed encryption	Secure inner product	Decryption
(i)	3.65 ms (31.2 KB)	5.31 ms (46.8 KB)	3.47 ms
(ii)	23.03 ms (66.6 KB)	34.34 ms (99.8 KB)	22.17 ms
(iii)	48.07 ms (141.3 KB)	71.25 ms (212.0 KB)	46.35 ms
(iv)	107.25 ms (299.0 KB)	159.45 ms (448.5 KB)	103.94 ms

Performances The key generation (excluding the prime generation) ran in about 1.89 milliseconds (ms), the packed encryption of a binary vector of length less than $n = 2048$ took about 3.65 ms, the secure inner product computation (6) took about 5.31 ms, and finally the decryption took about 3.47 ms.

6 Conclusions

We proposed a new packing method in the SHE scheme based on the polynomial LWE assumption for efficient computation of the inner product over packed ciphertexts, which can be used for the set intersection computation in marketing analysis. According to our implementation, our method enables to compute a secure inner product between two binary vectors of length $n = 2048$ (resp. 4096, 8192, and 16384) in 5.31 ms (resp. 34.34, 71.25, and 159.45 ms). Furthermore, by dividing vectors into blocks of length n, we can pack vectors of length m into $\lceil m/n \rceil$ ciphertexts and compute the inner product over packed ciphertexts. In the case of $n = 2048$ and $m = 10,000$, we need $\lceil m/n \rceil = 5$ packed ciphertexts and it also takes about $5 \times 5.31 = 26.55$ ms for a secure inner product between two binary vectors of length $m = 10,000$. These performances are practical in real life, and hence we hope that the SHE scheme with our packing method would be used in various applications mainly including marketing analysis. However, our packing method can be used between only two parties, and hence our future work is to develop a new method which can be applied to the set intersection computation among more than three parties.

Endnote

[a]This is a full version paper of the work [22] presented at the Forum on Information Technology (FIT2013).

Acknowledgements
The authors would like to thank the anonymous reviewers for their useful and helpful comments to improve the paper.

Received: 19 March 2014 Revised: 2 April 2014 Accepted: 3 April 2014
Published online: 14 October 2014

References

1. Boneh, D., Goh, EJ., Nissim, K.: Evaluating 2-DNF formulas on ciphertexts Theory of Cryptography–TCC 2005. In: Lecture Notes in Computer Science, pp. 325–341. Springer, (2005)
2. Brakerski, Z., Gentry, C., Vaikuntanathan, V.: (Leveled) fully homomorphic encryption without bootstrapping. In: Innovations in Theoretical Computer Science–ITCS 2012, pp. 309–325. ACM, (2012)
3. Brakerski, Z., Vaikuntanathan, V.: Efficient fully homomorphic encryption from (standard) LWE. In: Foundations of Computer Science–FOCS 2011, pp. 97–106. IEEE, (2011)
4. Brakerski, Z., Vaikuntanathan, V.: Fully homomorphic encryption from ring-LWE and security for key dependent messages. In: Advances in Cryptology–CRYPTO 2011. Lecture Notes in Computer Science, pp. 505–524. Springer, (2011)
5. Chen, Y., Nguyen, P., BKZ 2.0: better lattice security estimates. In: Advances in Cryptology–ASIACRYPT 2011. Lecture Notes in Computer Science, pp. 1–20. Springer, (2011)
6. Coron, JS., Mandal, A., Naccache, D., Tibouchi, M.: Fully homomorphic encryption over the integers with shorter public keys. In: Advances in Cryptology–CRYPTO 2011 Lecture Notes in Computer Science, pp. 487–504. Springer, (2011)
7. Cramer, R., Shoup, V., Schoenmakers, B.: A secure and optimally efficient multi-authority election scheme. In: Advances in Cryptology–EUROCRYPT 1997. Lecture Notes in Computer Science, pp. 103–118. Springer, (1997)
8. Damgård, I., Pasto, V., Smart, N., Zakarias, S.: Multiparty computation from somewhat homomorphic encryption. In: Advances in Cryptology–CRYPTO 2012. Lecture Notes in Computer Science, pp. 643–662. Springer, (2012)
9. Gentry, C.: A fully homomorphic encryption scheme. PhD thesis, Stanford University (2009)
10. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Symposium on Theory of Computing–STOC 2009, pp. 169–178. ACM, (2009)
11. Gentry, C., Halevi, S.: Implementing Gentry's fully-homomorphic encryption scheme. In: Advances in Cryptology–EUROCRYPT 2011. Lecture Notes in Computer Science, pp. 129–148. Springer, (2011)
12. Gentry, C., Halevi, S., Smart, N.: Homomorphic evaluation of the AES circuit, pp. 850–867. Springer, (2012)
13. Lindner, R., Peikert, C.: Better key sizes (and attacks) for LWE-based encryption. In: RSA Conference on Topics in Cryptology–CT-RSA 2011. Lecture Notes in Computer Science, pp. 319–339. Springer, (2011)
14. Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. In: Advances in Cryptology–EUROCRYPT 2010. Lecture Notes in Computer Science, pp. 1–23. Springer, (2010)
15. Micciancio, D., Regev, O.: Worst-case to average-case reduction based on Gaussian measures. SIAM J. Comput. **37**(1), 267–302 (2007)
16. Naehrig, M., Lauter, K., Vaikuntanathan, V.: Can homomorphic encryption be practical? In: Proceedings of the 3rd ACM workshop on Cloud computing security workshop–CCSW 2011, pp. 113–124. ACM, (2011)
17. Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In: Advances in Cryptology–EUROCRYPT 1999. Lecture Notes in Computer Science, pp. 223–238. Springer, (1999)
18. van Dijk, M., Gentry, C., Halevi, S., Vaikuntanathan, V.: Fully homomorphic encryption over the integers. In: Advances in Cryptology–EUROCRYPT 2010 Lecture Notes in Computer Science, pp. 24–43. Springer, (2010)
19. Yasuda, M., Shimoyama, T., Kogure, J., Yokoyama, K., Koshiba, T.: Packed homomorphic encryption based on ideal lattices and its application to biometrics. In: Modern Cryptography and Security Engineering–MoCrySEn 2013. Lecture Notes in Computer Science, pp. 55–74. Springer, (2013)
20. Yasuda, M., Shimoyama, T., Kogure, J., Yokoyama, K., Koshiba, T.: Secure pattern matching using somewhat homomorphic encryption. In: Proceedings of the 5th ACM workshop on Cloud computing security–CCSW 2013, pp. 65–76. ACM, (2013)
21. Yasuda, M., Shimoyama, T., Kogure, J., Yokoyama, K., Koshiba, T.: Practical packing method in somewhat homomorphic encryption. In: Data Privacy Management and Autonomous Spontaneous Security. Lecture Notes in Computer Science, pp. 34–50. Springer, (2014)
22. Yasuda, M., Shimoyama, T., Yokoyama, K., Kogure, J.: A customer information analysis between enterprises using homomorphic encryption (in Japanese). In: Forum on Information Technology–FIT 2013, pp. 15–22. IPSJ, (2013)
23. Yasuda, M., Yajima, J., Shimoyama, T., Kogure, J.: Secret totalization of purchase histories of companies in cloud (in Japanese). In: 29th Symposium on Cryptography and Information Security–SCIS 2012 number 3D-2. IEICE, (2012)

doi:10.1186/s40736-014-0005-x
Cite this article as: Yasuda *et al.*: **Secret computation of purchase history data using somewhat homomorphic encryption.** *Pacific Journal of Mathematics for Industry* 2014 **6**:5.