**哈尔滨工业大学（深圳）**

Harbin Institute of Technology ,Shenzhen

Interim Assessment of the Thesis

for the Master’s Degree

|  |
| --- |
| **Name** Haodong Liu |
| **Entrance Date** 2015.9.10 |
| **Thesis Title** The time series discord discovery |
| under multi-party privacy preserving |
| **Discipline** Computer Science |
| **Supervisor** Chunkai Zhang |
| **Report Date** 2017.3.16 |

|  |
| --- |
| 1. Does the thesis progress according to the research objectives and schedule as stated in the primary report? (at least 100 words)   In my thesis, I propose a solution to solve the time series discord discovery under multi-party privacy preserving, inspired by some related work. My solution uses two non-colluding, semi-honest but untrusted server *C* and *S* and there is no *interaction* between clients. All protocol participants follow the protocol description, but may try to gather information about other parties’ inputs, intermediate results, or overall outputs just by looking at the transcripts. The solution is designed to allow the inputs to be encrypted under diﬀerent independent public keys. proposed the security protocol to ensure the data won’t be leaked in the process of communication or computation, further more we give the brute force algorithm of time series discord discovery and transfer it under multi-party privacy preserving. The correctness of my solution is validated by extensive experimental evaluation. |
| 1. The completed work and its related outcomes (at least 1500 words) .   A **time series** is a series of data points indexed (or listed or graphed) in time order. Most commonly, a time series is a sequence taken at successive equally spaced points in time. Discord refers to a data that are significantly different from the rest of the data. Discord detection is to discover the most unusual pattern in the data by algorithms. Generally speaking, the data is usually generated by a single or a number of system processes, which can reflect the status of the system, when the process is not normal, it will lead to the production of abnormal data. Discord pattern means or contains important information of the system, Time series discords have many uses for data mining, including improving the quality of clustering, data cleaning, summarization, and anomaly detection.  Now we begin by defining the data type of interest.  **non-self match**. Provided a time series *T* of length *n*, *C* is subsequence of *T* and the length is *m (mn)* and its start position is *p*, *M* is another subsequence and their start positions is *q*, when *|p-q| ≥ m*, said subsequence *M* is non-self match of *C*.  **time series discord**. Given time series *T*, the subsequence *A* of length m is said to be the discord subsequence of *T* if *A* has a largest distance to its nearest non-self match. Any subsequence *C* of length m, is the non-self match of *C*, is the non-self match of *A*, it satisfy:  According to the definition we can design the discovery algorithm easily.    Table 1 the process of brute force algorithmto detect the time **s**eries discord   |  | | --- | | **Procedure**：Brute force algorithm  Input: time series T, subsequence length m.  Output: The position of time series T’s discord | | 1. initialize anomaly\_loc and anomaly\_dist;  2. for each p in T  3. initialize current\_dist;  4. for each q in T  5. if p and q is no self match  6. dist(p,q) smaller than current\_dist then update current\_dist;  7. if current\_dist bigger than anomaly\_dist  8. update anomaly\_loc and anomaly\_dist;  9. return anomaly\_loc and anomaly\_dist | |  |   **Additive Homomorphic Cryptosystem:**  Suppose and are two additive homomorphic ciphertexts under the same public key . The additive homomorphic cryptosystem (AHC) has the additive homomorphic property:  (2-1)  and are two plaintext messages, and it have to under the same public key that the property is valid. In this paper, we propose that we can discovery the discord under the multi-party, multi-party usually means multi-key, so how to use the property under multi-key is the key point we should concern.Now, we introduce another additively homomorphic cryptosystem which is variant of the El Gamal cryptosystem by Bresson, Catalano and Pointcheval (BCP). The BCP cryptosystem has a special property which offers two independent decryption mechanisms, secret key and master secret key. Besides the secret key, the master secret key also can decrypt any given ciphertext if the encrypted public key is generate by the protocol. Let’s see the process of BCP cryptosystem in details.  : For a security parameter *κ*, choose a safe-prime RSA-modulus *N = pq* ( and for distinct primes and, respectively) of bitlength *κ*. Pick a random element of order such that mod for . The plaintext space is and the algorithm outputs  *Public parameters* (2-2)  *Master secret* (2-3)  : Pick a random and compute *h*mod . The algorithm outputs  (2-4)  (*m*): Given a plaintext , pick a random and output the ciphertext:  mod mod (2-5)  : Given a ciphertext and secret key , output the plaintext *m*:  (2-6)  : Given a ciphertext , a user’s public key and the master secret MK. Let denote the user’s private key corresponding to . First we compute the :  (2-7)  Where the represent the inverse of *k* modulo *N*. Then compute the *:*  (2-8)  Let represent the inverse of modulo *N* and set The algorithm output the plaintext *m* as:  (2-9)  From that we can understand how BCP cryptosystem offers two independent decryption mechanisms, in part 5 we will use the property to design a construction to solve the multi-key problem. To know more proofs and security information about the BCP cryptosystem, we highly recommend the original paper, see reference [11].  Note that for given under the same , we have  Moreover several different public key which generated by the same public parameters could join together,namely the product  **Prod.pk** = (2-10)  With the formula (2-7), (2-8), (2-9) we know that master secret key also can decrypt the **prod.pk**.  (2-11)  **Ciphertext Refresh:** the ***CR*** algorithm can refresh the ciphertext without changing the original message *m*, by randomly choosing and refreshing the ciphertext as where .  **My Construction:**  The basic system model is shown as follows:  Server C  Server S  Users（clients）  1. Initiate, we use a second untrusted server *S* that acts semi-honestly and that does not collude with any of the other parties. The server *S* runs a setup Init that sets up the system’s public parameters and the master secret key, then storage the master secret key and send the system’s public parameters to server *C*.  2. Server *C* distribute the public parameters to each client in the system, clients can use the cryptosystem’s KeyGen (independently of any other party) to generate their respective pair of public and private keys, and to upload encryptions of their private data to the server *C.*  3. With 3 basic secure protocols: secure addition protocols (**SAP),** secure Multiplication protocols (**SMP**) and secure minimum protocols (**SMINP**). We can realize the addition, multiplication and comparison operation under different public key. With the addition, multiplication and comparison operations, the brute forcealgorithm will beable to be applied under privacy preserving.  4. After all computations are done, client will receive the encrypted output from the server *C*, and it can decrypts locally with its respective private key to get the result.   1. Secure Addition Protocols (**SAP)**   Because of the server S is blind to the original ciphertext, so it can’t let server S directly access the original ciphertext, also can’t let server S know which party or client need the processing ciphertext. Suppose that server *C* need to compute the sum of if x and y was encrypted under the same public key ( ). But now they are under different public key, So the goal of our SAP is to calculate . Prod.pk is defined at formula (2-10).   1. Select two random number ,server *C* calculate 2. Send *X, Y* to server *S*, server *S* decrypt the *X* and *Y* use the master secret key.   Calculate *Z = +*, encrypted *Z*  as , then send it to server *C*   1. Server *C* received *Z*, calculate encrypted *R*  as , calculate 2. secure Multiplication protocols (**SMP**)   The same reason as SAP, given two encrypted number server *C* want to get the product of two number, the goal of our SAP is to calculate .   1. Select two random number ,server *C* calculate 2. Send *X, Y* to server *S*, server *S* decrypt the *X* and *Y* use the master secret key.   Calculate *Z =* , encrypted *Z*  as , then send it to server *C*   1. Server *C* received *Z,* calculate   Before we introduce the **SMINP**, we need to introduce a pre-protocol first, because the **SMINP** is based on this protocol, give a secure less than protocol first.   1. secure less than protocol (**SLT**)   Given two encrypted numbers the goal of SLT protocol is to obtain the encrypted data to show the relationship between the plaintext of the two encrypted data. There is a little trick in here, we must restrict |x| and |y| to be in the range of [0,], where < . If a larger plaintext range is needed, we can simply use a larger N. A larger N implies a broader plaintext range, and therefore, a higher level of security. The description of the **SLT** protocol is as follows:   1. server *C* calculate   server C flips a coin *s* randomly. If *s*=1, calculate  If *s* = 0, calculate  Server *C* choose a random number *r,*  < , and calculate then send to server *S*.   1. Server *S* decrypt the with master secret key, and obtain , server *S* denotes = 1 when > , otherwise = 0. Then server *S* encrypt and send it to server *C.* 2. Once is received, server *C* computes as follows if *s* = 1, do , otherwise server *C* compute   If , it shows , and if , it shows .   1. secure minimum protocols (**SMINP**)   Given two encrypted numbers the goal of **SMINP** is to obtain the .   1. Server *C* and server *S* jointly calculate: 2. Once received, server *C* calculate:   In A,B,C,D we just get the result like , we need to send encrypted result to client, but the client’s secret key can’t decrypt the prod.pk, we need the form like so how to transform the prod.pk to is the last thing we need to do. It is very easy!   1. Transform key(**TransDec**) 2. Server *C* select a random calculate 3. Send *X*  to server S, calculate     For *i* =1 to *n* //n denote n clients  Send every to server *C*   1. Server *C* receive , According to the request client’s public key, calculate.   =  Now, we have **SAP**, **SMP**, **SMIN**, 3 protocols to realize the addition,multiplication, minimum operations under multi-party privacy preserving. So the brute force algorithm can be transfer under multi-party privacy preserving.  **Experiment :**  ***ECG data:*** ECG data is the process of recording the electrical activity of the heart over a period of time using electrodes placed on the skin. These electrodes detect the tiny electrical changes on the skin that arise from the heart muscle's electrophysiologic pattern of depolarizing during each heartbeat. It is a very commonly performed cardiology test [27]. Firstly we consider the single party situation, Figure 1 is the interception of ECG data, the time series contains 650000 data points and there is discord subsequence in the position 41500.    **Figure 1. ECG discord**  Figure 1 shows the ECG discord under the single party, what if we can’t find out the discord subsequence in single party? figure 2 presents the situation under multi-party, we assume that time series A and time series B are two ECG data belongs to hospital A and hospital B, hospital B want to discovery the discord subsequence in time series B, if hospital B just consider their own dataset, call the TSDDSP algorithm. The algorithm denotes the x1 as discord. But in fact, subsequence x1 is normal ECG data, and the data after subsequence x1 are all abnormal, according to the definition, x1 is has the largest NN distance.  But, if we consider time series A, the result will be different, time series A are all normal ECG data, the subsequences in time series A are very similar to x1, call DDMPP algorithm, the algorithm denote the subsequence x2 as discord. Different with TSDDSP algorithm, the DDMPP algorithm is processing under ciphertext, so it need more time. We set the subsequence length is 300, time series A and time series B length is 2000, it cost nearly 48hours to finish the process.    **Figure 2. The ECG data discord under multi-party**  In the experiment we can find the basic protocol and the algorithm we proposed is able to work properly. Although the time cost has a little disappointment, but under the privacy preserving we must sacrifice the time for security. Under the semi-honest model, our system security is reliable. Of course we know that there is definitely plenty ways to speed up our algorithm, this is what we will continue to research. |
| 1. The work to be completed and its schedule .   In current stage, We proposed a time series discord discovery under multi-party privacy preserving. We extend the definition of time series discord to the multi-party domain, and designed a security computation system to realize the algorithm, our solution allow different public key and the algorithm is all parallelizable. The time series discord discovery is a quite important field in time series research, and privacy preserving is a hot topic recently, the extensive experimental evaluation demonstrate that our work have great significance for real-word deployment.  There is also many future work we should focus on.  (1) The algorithm we proposed is for single time series data, but in reality, a lot of time series is multi-dimensional, how to analyze the relationship between multi-dimensional time series and discord discovery is a very significance direction.  (2) In the DDMPP algorithm, we directly use the original time series data to do the mining task. But the time series is huge amount and high dimension, so the suitable choice of representation greatly affects the ease and efficiency of time series data mining. And from the experiment we also know that the efficiency of our algorithm is quite disappointing, there is very large room for improvement. There is great number of time series representations have been introduced, including the Discrete Fourier Transform (DFT) [23], the Discrete Wavelet Transform (DWT) [24], Piecewise Linear, and Piecewise Constant models (PAA) [25], (APCA) [25,26], and Singular Value Decomposition (SVD) [25]. How to realize the time series data representation under privacy preserving is also very important for our research.  (3) we need to parallelize our algorithm on the cloud-side. Nowadays the cloud computing technology developed rapidly. The cost of storage and computation will be unacceptable on the stand-alone model. So a distributed system is necessary under multi-party privacy preserving.  **Schedule:**  March 2017 - May 2017: Propose a new algorithm, and design security protocol  May 2017 –July 2017: coding, implementation and testing, analysis of experimental results.  July 2017 – September 2017: the overall system testing, summary.  September 2017 – December 2017: research conclusion, write thesis, prepare to thesis defence. |
| 1. The existing or expected difficulties and problems.   The encryption algorithm is very complex, and in the experiments we can find out that it will significantly increased the time cost when we transformed the algorithm into multi-party privacy preserving. Discord in the plaintext can be detected immediately, but under ciphertext it need several hours to processing. Of course, we can optimize the algorithm to reduce the time complexity, but the optimization algorithm will lead a more complex operations such as division, square operation. These operations will increase the complexity of algorithm under ciphertext. Therefore, determined how to optimize algorithm and design a reasonable secure protocol is a difficult problem we should concern. |
| 1. The considerations on the possibility of completing the thesis on-time (at least 100 words).   I have done the work of design the basic secure protocol and basic detection algorithm, the experiment proof that the protocol and algorithm is correct. But the complexity is unacceptable, in the next several month the most important thing we should consider is to optimize the detection algorithm and find out a more efficient cryptosystem. Make the system is acceptable both on security and efficiency.  About the distributed system, my lab has built a small distributed system with three hosts, using hadoop+spark. We had done some simple experiment on that, we believe it’s available to transplant the algorithm from local to distributed system.  Overall, I’m sure that my thesis will be completed on time.  **Supervisor’s Signature：** |