Volume 5, Issue 2, April 2018 ISSN: 2348-8565 (Online)
International Journal of Modern Engineering and Research Technology
Website: http://www.ijmert.org Email: editor.ijmert@gmail.com

Probabilistic and Deterministic Verification Approaches of Outsourced Frequent Item Set Mining

Y Mary Sujatha
PG Scholar,
DNR College of Engineering &Technology
Bhimavaram (A.P.) [INDIA]
Email: sujatha.cmm118@gmail.com

B Nanadana Kumar
Assistant Professor
Department of Computer Science and Engineering
DNR College of Engineering &Technology
Bhimavaram (A.P.) [INDIA]
Email: nandankumar007@gmail.com

DDD Suribabu
Head & Associate Professor
Department of Computer Science and Engineering
DNR College of Engineering &Technology
Bhimavaram (A.P.) [INDIA]
Email: dnr.csehod@gmail.com

ABSTRACT

Cloud computing is popularizing the computing paradigm in which data is outsourced to a third-party service provider (server) for data mining. Outsourcing, however, raises a serious security issue: how can the client of weak computational power verify that the server returned correct mining result? In this paper, we focus on the specific task of frequent itemset mining. We consider the server that is potentially untrusted and tries to escape from verification by using its prior knowledge of the outsourced data. We propose efficient probabilistic and deterministic verification approaches to check whether the server has returned correct and complete frequent itemsets. Our probabilistic approach can catch incorrect results with high probability, while our deterministic approach measures the result correctness with 100% certainty. We also design efficient verification methods for both cases that the data and the mining setup are updated. We demonstrate the effectiveness and efficiency of our methods using an extensive set of empirical results on real datasets.

Keywords:—*Cloud computing, data mining as a service, security, result integrity verification.*

I. INTRODUCTION

The increasing ability to generate vast quantities of data presents technical challenges for efficient data mining. Out-sourcing data mining computations to a third-party service provider (server) offers a cost-effective option, especially for data owners (clients) of limited resources. This introduces the data-mining-as-a -service (DMaS) paradigm. Cloud computing provides a natural solution for the DMaS paradigm. A few active industry projects, for example, Google's Prediction APIs and Microsoft's Daytona project, provide cloud-based data mining as a service to users.

In this paper, we focus on frequent itemset mining as the outsourced data mining task. Informally, frequent itemsets refer to a set of data values (e.g., product items) whose number of co-occurrences exceeds a given threshold. Frequent itemset mining has been proven important in many applications such as market data analysis, networking data study, and

human gene association study. Previous research has shown that frequent itemset mining can be computationally intensive, due to the huge search space that is exponential to data size as well as the possible explosive number of discovered frequent itemsets The privacy challenge of outsouced database is two -hold. 1) Sensitive data is stored in cloud, the corresponding private information may be exposed to cloud servers; 2) Besides data privacy, clients' frequent queries will inevitably and gradually reveal some private information on data statistic properties. Thus, data and queries of the outsouced database should be protected against the cloud service provider.

One straightforward approach to mitigate the security risk of privacy leakage is to encrypt the private data and hide the query/access patterns. However, such privacy leakage hasn't been well addressed thoroughly, since OPE is relatively weak to provide sufficient privacy assurance. Some specific purpose cryptology like order preserving encryption (OPE) will expose some private information to the cloud service provider naturally: As it is designed to preserve the order on ciphertexts so that it can be used to conduct range queries, the order information of the data, the statistical properties derived there from, such as the data distribution, and the access pattern will be leaked.

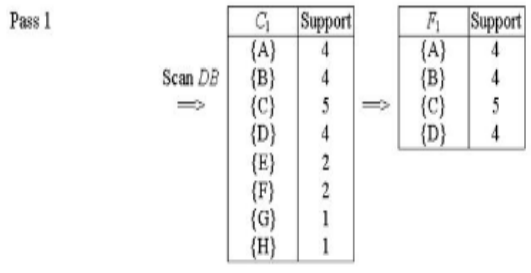

Figure 1 : Dataset Frequent Itemset Distributions.

Based on this architecture, we further propose a series of interaction protocols for a client to conduct numeric-related query over encrypted data from remote cloud servers. The numeric-related query includes common query statements, such as greater than, less than, between, etc..

II. OBJECTIVE

The return erroneous type-1 server that possesses the background knowledge of the outsourced dataset, including the domain of items and their frequency information, and the type-2 server that is aware of the frequency distribution information of both items and transactions, as well as the details of the verification procedure. In this paper, we target at designing verification approaches to catch these two types of servers.

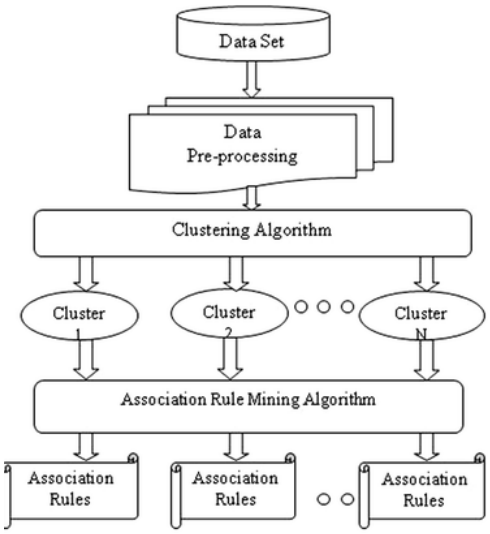

Figure 2: General process of association rules

The key idea of our methods is to construct a set of (in) frequent itemsets from real items, and use these (in) frequent itemsets as evidence to check the integrity of the server's mining result. We remove real items from the original dataset to construct artificial evidence infrequent itemsets (EIs), and insert copies of items that exist in the dataset to construct artificial evidence frequent items (EFs). client uses a set of frequent itemsets as the evidence, and checks whether the server misses any evidence frequent itemset in its returned result. If it does, the incomplete answer by the server is caught with 100% certainty. Other-wise, the

client believes that the answer is complete with a probability.

Construction of Evidence Infrequent Itemsets EIs) Similar to the completeness verification, our basic idea of correctness verification is that the client uses a set of infrequent itemsets as the evidence, and checks whether the server returns any evidence infrequent itemset. Next, we show how to measure the correctness probability guarantee. Similar to the completeness probability guarantee.

III. Process Investigations.

Pick Item Instances for Removal. Given the transactions obtained by each client we need to decide which items in data will be removed. We aim at minimizing the total number of removed items. To address this issue, we put high priority on removing the items that are shared among patterns in AI. Therefore, we do the following to pick the items. First, for each unique item in AI, we count its frequency in AI. Second, we sort the items by their frequency in descending order. Third, we construct the item matrix IM of AI. IM is a u y binary matrix, where u is the number of itemsets in AI, and y is the number of unique items in AI. In the matrix, IM[i;j] = 1 means that the frequent itemset I_i contains the item i_j; otherwise, IM[i;j] = 0. Then we repeat the following procedure on IM. We add the item that corresponds to the first column of IM to the output, update IM by removing all rows i such that IM[i;1] = 1 (i.e., all patterns that contain the item of the largest support), and re-sort the columns in the updated IM by their sum in descending order. We repeat until IM becomes empty. The sequence of the removed columns outputs the items to be picked for removal.

After item removal, by following the property of the downward closure of in frequentness, the client adds all descendant itemsets of the EIs picked by Step 1 that are of non-zero support into the evidence repository R. Furthermore, we are aware that changing frequent itemsets to be infrequent will modify all of its frequent descendants to be infrequent. This will lead to incomplete frequent itemsets even when the server is honest. To solve this problem, for each descendant itemset of AI that is added to R, we count its support and mark it as recoverable if it is frequent.

```
End=6/2/2010 17:53:14
Recorded=255 1201188463 unit7          2 10000
Channel 0=9830 2 3 30000 0 -4 mv
Title=Channel 1
Channel 1=9830 2 3 30000 0 -4 mv
Title=Channel 2
Channel 2=9830 2 3 30000 0 -4 mv
Title=Channel 3
Channel 3=9830 2 3 30000 0 -4 mv
Title=Channel 4
Channel 4=9830 2 3 30000 0 -4 mv
Title=Channel 5
Channel 5=9830 2 3 30000 0 -4 mv
Title=Channel 6
```

Figure 3: Data Set with Priority Rules.

Assume now the server has passed the verification of MNB nodes, next, the client uses the proof of MNB nodes to prove the completeness of returned frequent itemsets (i.e., each itemset that is not returned must be infrequent). Before we discuss how the client verifies the completeness, we categorize the possible missing frequent itemsets into four types, based on their relationships with the returned frequent itemsets F^S. In particular, consider a frequent itemset I that is not returned by the server.

Table 1: Details of Data Sets.

Data-set	# of trans.	# of items	Avg. trans. length	min-sup	# of freq. itemsets
S1	103	49	10	250	36
S2	104	49	10	250	3854
S3	105	49	10	250	149744
S4	106	49	10	250	3074610
R1	88162	16470	124	50 10	16778 155111
R2	500	100	2.4	5	97
NCDC	500	365	332.9	450	559368361

synthetic datasets S_1;S_2;S_3, and S_4 of various sizes. We also use two real-world datasets named Retail dataset and NCDC dataset[1]. The Retail dataset is available at the Frequent Itemset Mining Dataset Repository[2]. The Retail dataset R_1 contains 88162 transactions and 16470 items. We also construct a small dataset R_2 from the Retail dataset that contains 500 transactions and 100 items. The NCDC dataset comes from National Climatic Data Center of U.S. Department of Commerce. Table I shows the details of the datasets and our mining setup. Among these datasets, the NCDC dataset is a dense dataset, in which most of the transactions are of similar length, and contain > 75% of items; and the R_1 dataset is a sparse dataset in which the transactions are of skewed length distribution. Due to Simulation of malicious actions. We set the error ratio p = 1%; 2%; 5%; 10%, and 20%. For the simulation of incomplete result, we randomly pick p percent of frequent itemsets from the mining result and remove these picked itemsets. For the simulation of incorrect result, we randomly generate p percent of infrequent itemsets and insert them into the result. e measure the performance of proof construction at the server side and verification at the client side and explored various factors that impact the verification performance of our deterministic approach, including various error ratio, frequent itemsets of different lengths, and different database sizes. We set the support threshold on R1 dataset to be 50.

We ran experiments to compare the performance of our probabilistic and deterministic approaches. Table III shows the comparison result on S_3 dataset of various settings. We pick the error ratios of 1%, and vary the probabilistic guarantee threshold from 90% to 100% (probability = 100% corresponds to our deterministic approach). Table III shows the details of the comparison result. In general, the deterministic approach brings higher overhead at the server side than the probabilistic approach. However, this is the sacrifice that we have to pay for higher result integrity guarantee. We also observe that in some cases (marked as N/A in Table III), the probabilistic approach fails as it cannot provide required probabilistic correctness guarantee due to the data distribution. The deterministic approach does not have such limit.

VI. CONCLUSIONS

In this paper, we present two integrity verification approaches for outsourced frequent itemset mining. The probabilistic verification approach constructs evidence (in)frequent itemsets. In particular, we remove a small set of items from the original dataset and insert a small set of artificial transactions into the dataset to construct evidence (in) frequent itemsets. The deterministic approaches requires the server to construct cryptographic proofs of the mining result. The correctness and completeness are measured against the proofs with 100% certainty. Our experiments show the efficiency and effectiveness of our approaches.

REFERENCES:

[1] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules in large databases. In Proceedings of the 20th International Conference on Very Large Data Bases (VLDB), pages 487-499, 1994.

[2] Laszlo Babai, Lance Fortnow, Leonid A. Levin, and Mario Szegedy. Checking computations in polylogarithmic time. In STOC, pages 21–32, 1991.

[3] Ran Canetti, Ben Riva, and Guy N. Rothblum. Verifiable computation with two or more clouds. In Workshop on Cryptography and

Security in Clouds, 2011.

[4] Kun-Ta Chuang, Jiun-Long Huang, and Ming-Syan Chen. Power-law relationship and self-similarity in the itemset support distribution: analysis and applications. The VLDB Journal, 17:1121–1141, August 2008.

[5] Rosario Gennaro, Craig Gentry, and Bryan Parno. Non-interactive verifiable computing: outsourcing computation to untrusted workers. In CRYPTO, pages 465–482, 2010.

[6] Fosca Giannotti, Laks V. S. Lakshmanan, Anna Monreale, Dino Pe-dreschi, and Wendy Hui Wang. Privacy-preserving data mining from outsourced databases. In Computers, Privacy and Data Protection, pages 411–426. 2011.

[7] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. SIAM Journal of Computing, 18:186– 208, February 1989.

[8] Hakan Hacigumus, Bala Iyer, Chen Li, and Sharad Mehrotra. Ex-ecuting sql over encrypted data in the database-service-provider model. In SIGMOD, pages 216–227, 2002.

[9] Feifei Li, Marios Hadjieleftheriou, George Kollios, and Leonid Reyzin. Dynamic authenticated index structures for outsourced databases. In SIGMOD, pages 121–132, 2006.

[10] Ruilin Liu, Hui Wang, Anna Monreale, Dino Pedreschi, Fosca Giannotti, and WengeGuo. Audio: An integrity auditing framework of outlier-mining-as-a-service systems. In ECML/PKDD, 2012.

[11] Ian Molloy, Ninghui Li, and Tiancheng Li. On the (in)security and (im)practicality of outsourcing precise association rule mining. In ICDM, pages 872–877, 2009.

[12] Charalampos Papamanthou, Roberto Tamassia, and Nikos Trian-dopoulos. Authenticated hash tables. In CCS, pages 437–448, 2008.

[13] Srinath Setty, Andrew J. Blumberg, and Michael Walfish. Toward practical and unconditional verification of remote computations. In HotOS, 2011.

[14] W. K. Wong, David W. Cheung, Ben Kao, Edward Hung, and Nikos Mamoulis. An audit environment for outsourcing of frequent itemset mining. In PVLDB, volume 2, pages 1162–1172, 2009.

[15] Feida Zhu, Xifeng Yan, Jiawei Han, Philip S. Yu, and Hong Cheng. Mining colossal frequent patterns by core pattern fusion. In ICDE, 2007.

[16] Siavosh Benabbas, Rosario Gennaro, and Yevgeniy Vahlis. Verifiable delegation of computation over large datasets. In CRYPTO, 2011.

[17] Ran Canetti, Ben Riva, and Guy N. Rothblum. Practical delegation of computation using multiple servers. In CCS, 2011.

[18] Dario Fiore and Rosario Gennaro. Publicly verifiable delegation of large polynomials and matrix computations, with applications. In CCS, 2012.

[19] Menezes, Alfred and Vanstone, Scott and Okamoto, Tatsuaki. Re-ducing elliptic curve logarithms to

logarithms in a finite field. In STOC, 1991.

[20] R.C. Merkle Protocols for public key cryptosystems. In Symposium on Security and Privacy, 1980.

[21] Charalampos Papamanthou, Roberto Tamassia, and Nikos Trian-dopoulos. Optimal verification of operations on dynamic sets. In CRYPTO, 2011.

[22] Michael T. Goodrich, Charalampos Papamanthou, Duy Nguyen, Roberto Tamassia, Cristina Videira Lopes, Olga Ohrimenko and Nikos Triandopoulos Efficient Verification of Web-Content Searching Through Authenticated Web Crawlers In PVLDB, volume 5, pages 920–931, 2012

[23] Bryan Parno, Mariana Raykova, and Vinod Vaikuntanathan. How to delegate and verify in public: verifiable computation from attribute-based encryption. In TCC, 2012.

* * * * *