International Journal of Multidisciplinary Research and Modern Education (IJMRME)
Impact Factor: 6.725, ISSN (Online): 2454 - 6119
(www.rdmodernresearch.org) Volume 3, Issue 2, 2017

A MODEL APPROACH FOR VERIFYING OUTSOURCED COMPUTATION OF CLOUD FOR FREQUENT ITEM SET MINING

Umesh D. Borse* & Dr. Lalit V. Patil**
Department of Information Technology, Smt. Kashibai Navale College of Engineering, (SKNCOE), Pune, Maharashtra

Cite This Article: Umesh D. Borse & Dr. Lalit V. Patil, "A Model Approach for Verifying Outsourced Computation of Cloud for Frequent Item Set Mining", International Journal of Multidisciplinary Research and Modern Education, Volume 3, Issue 2, Page Number 76-82, 2017.

Abstract:

In recent years, It has been shown that outsourcing the most expensive task in data mining process to a service provider (e.g., cloud) brings several benefits to the data owner such as cost relief and less commitment to storage and computational resources.This introduces Data mining as Service paradigm.An Outsourcing data to the server facing a critical problem of verification, whether the server returned true and complete computation result to the client.The correctness integrity of mining results may be corrupted if the service provider is with random fault or malicious. In the case of frequent item set mining Apriori, Eclat, FP-growth algorithms are used for a vast database which contains a large number of transactions. Merkel hash tree, Bilinear pairing, Artificial Item set Planting (AIP) techniques are used for result verification of outsourced computation. The work introduces a new system for result verification of outsourced frequent item set mining computation. In proposed system tree comparison is the main component used for the verification. Client formed tree from received result of outsourced computation by calculating support value for the frequent item set, and another tree is created for the small part of the outsourced computation. By comparing these two trees verification of the outsourced computation is done.

Key Words: Data Mining as a Service, Frequent Itemset Mining, Tree Matching & Result Verification

1. Introduction:

The current move to cloud computing is increasing the need verification of computations, where a computationally weak client outsources his computation to the powerful cloud. Now a day, IT companies interested in outsourcing the task to the cloud to reduce the size of computing assets and regarding reducing the burden of computation. Frequent item set mining plays important roles in data analysis from a large database and many other significant data mining tasks. Frequent item set mining is a costly task because of huge data size. There for the computationally weak system (Client) delegate data to the computationally more powerful systems (Server). Outsourcing data to the third party service provider brings several benefits to the data owner such as cost relief and less commitment to storage and computational resources. This introduces the Data-Mining as Service (DMaS) paradigm [1].

In such a case, two issues arise. Although outsourcing data to the third party service provider is a viable option to the data owners, still client hesitates to place trust on cloud computing about the integrity of the computed results. The service provider may provide some fake results instead of computing for the correct result. Motivations behind such misbehavior could differ, but saving its own cost and deliberately disrupting the service are two obvious ones that we foresee. The other is confidentiality, or more generally privacy issues.There are several possible reasons to cheat with computation result for the cloud. Computation result can be incorrect if the service provider makes mistakes in the mining process although the service provider is honest. [1] The second issue is about the security or privacy of outsourced computation task if service provider contaminates or tampers computational result[5] or can be access to valuable, sensitive information from it. The next issue is related to revenue generation by cloud server [2]. A cloud would invest fewer resources to improve its revenue while charging for more resources. A cloud server can provide some forged results instead been spending its resources in computing the correct result.

A frequent item set mining is defined as the finding of a regular occurrence of item from a collection of some item sets where support calculated for that item set is greater than or equal to the minimum support threshold. Depending on the occurrence of the items, Frequent Item set (FI) and Infrequent Item set (IFI) can be generated. Frequent item set and infrequent item set are used to find out the frequent use of the particular item for market data analysis, depending upon the frequency of item used demand of production can be controlled for the particular item. The occurrence of particular item from the total transaction data the support is calculated for that item. By using support, a determined tree is generated. To verify the result of outsourced computation tree comparison technique is used, in which two different trees are compared with each other. If trees are matched with each other node to node, then the result of verification is considered as correct.

The work has been classified as follows Section 2 is dedicated to Related Work, Section 3 detailing of our proposed methodology, Implementation Details and Experimental Setup, Section 4 is dedicated to Result and Discussion, and Section 5 Conclude efforts of the work with some scope for the future extension.

2. Related Work:

This section of related work eventually reveals some facts based on thoughtful analysis of many authors as,

Boxiang Dong, Ruilin Liu [1] Proposes two different approach to verify whether the server has returned correct and complete frequent itemsets result. The first probabilistic approach is designed to find out mining result is correct or not and second deterministic approach intend to see frequent itemset mining result complete or not with maximum probability. The basic idea behind to create frequent and infrequent itemsets is to use those infrequent itemsets as evidence to check server mining result. In this suggested work no fraudulent itemset are used to check the result. Markel Hash Tree and Bilinear Pairing Technique are adopted for these approaches. This proposed methodology has better performance w.r.t time to construct evidence and proof of itemset. This system has a limitation that it is limited for a small set.

Ran Canetti, Ben Riva, Guy N. Rothblum [2] Introduces a protocol for any efficiently computable function, a use of the protocol for the X 86 computation models and a prototype implementation, called Quin. As a fundamental concept of this methodology is outsource the computation to N number of cloud, and then the client gets some result in output. Out of that majority of the same result in output is the correct result of the outsourced computation. The client request for the result of the function f(x) from more than two cloud servers. The each of the servers engages in a protocol with the client, in case they make contradictory claims about f(x). Ultimately the client can determine the valid claim since there is at least one honest server. In this protocol client searches for the difference between the intermediate states of the two server's computations. In this X 86 computational models and Markel Hash Tree is used for computation. In introduced system overhead of the protocol is itself low and having a limitation of using the different operating system.

Bryan Parno, Mariana Raykova, Vinod Vaikuntanathan [3] come up with public verifiable computation in two separate directions. One is Public Delegation, and another one is Public Verifiability, inwhich Public Delegation allows arbitrary parties to submit inputs for delegation. Whereas in Public Verifiability allows random parties (and not just the delegator) to verify the correctness of the results returned by the worker. The verifiable computation strategy with the public delegation and public verifiability have the advantage that this allows random parties to submit inputs for delegation, and verify the correctness of the results returned. Before delegation, the proposed system requires an expensive preprocessing to generate a secret key and evaluation key.

Xiaofeng Chen, Jin Li, Jian Weng [4] recommends the notion of the verifiable database with incremental updates (Inc-VDB).The update algorithm in Inc-VDB is incremental. That is, the client can conveniently compute the new cipher text and the updated tokens with previous values, rather than generate from scratch. Thus, Inc-VDB schemes can lead to high-efficiency gain when the database undergoes for small modifications frequently. Also, recommend a general Inc-VDB framework by incorporating the primitive of vector commitment and the encrypt-then-incremental MAC mode of encryption. Besides, the recommended Inc-VDB scheme supports the public verifiability. Also, introduce a new property called accountability for VDB schemes. If the client detected the tampering of the server, then the client should be able to provide proof to convince the judge about the facts. The author proves that the proposed Inc-VDB scheme satisfies the property of accountability. Bilinear Pairing, Verifiable Database, and Vector commitment scheme is used in proposed methodology. This proposed methodology provides an advantage that the time cost is minimal as compared to the existing scheme. But the system has a limitation that it not applicable to the insertion operation due to the vector commitment.

W. K. Wong, David W. Cheung, Nikos Mamoulis, Ben Kao, Edward Hung [5] Proposed and developed an audit environment, which consists of a result verification method and a database transformation method. Fig 1 shows the architecture diagram of an audit environment. The proposed work is used for a database transformation method and a result verification method.

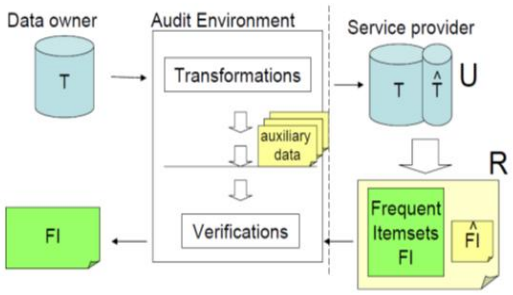

Figure 1: The architecture of the scheme [5].

The primary component verification environment is an Artificial Itemset Planting (AIP) technique. The main component of this audit environment is an artificial itemset planting technique. The approach to solving the integrity problem is to construct an audit environment. Mostly, an audit environment contains (i) a transformation method that transforms an original database to another database.Based on which the service provider will mine and return a mining result. (ii) A set of verification methods that take transformed database as an input and return a deduction of whether transformed database is correctand complete. (iii) Auxiliary data that abetment the verification methods. An impressive property of this approach is the audit environment forms a standalone system. It is self-contained in the sense that the verification process can be done absolutely by using only the auxiliary data that included in the environment. Mainly, the original database need not be accessed during verification. A malignant miner cannot benefit from performing any malicious actions, and thus the returned mining result is both correct and complete with a high confidence, but artificial database takes more time to generate as an original database is large.

In [7] author target on the integrity and verification issue in Uncertain Frequent Itemset mining problem during the outsourcing process, that is how the client verifies the mining results. The author specifically explores and extends the existing work on deterministic frequent itemset out-sourcing verification to the uncertain scenario. For this purpose, the author extends the current outsourcing frequent itemset mining work to an uncertain area with respect to the two popular uncertain frequent itemsets (UFI) definition criteria and the approximate uncertain frequent itemset (UFI) mining methods. Especially, authors construct and enhanced verification scheme with such different uncertain frequent itemset (UFI) definition respectively.

To verify the outsourced computation [8] explains various idea that client can use more than two clouds for the computation instead of using a single cloud for computation. If the client wants the better guarantee of the integrity of outsourced computation, then he can use more cloud. To get the better assurance of the computation result minimum three clouds must be utilized for computation. Now the client can outsource his data to the three different clouds for computation; now all cloud compute the data and returns the result of the computation. Out of that majority of the expected result is the correct output. The proposed idea has the limitation that client has to pay more if the number of clouds used more.

An author first introduces the rational lazy-and-partially-dishonest workers [9] in the outsourcing computation model. A new fair conditional payment scheme for outsourcing computation based on traditional electronic cash systems. The proposed system uses a semi-trusted third party to get the integrity and efficiency. However, In a case of any disputes, the third party is only involved in the protocol. As compared to the other existing solutions the proposed system is more efficient. The third party may collude with one party to obtain profits.

3. Proposed Methodology:

This section illustrates the detailed design and implementation of the proposed system as follows.

A. System Overview:

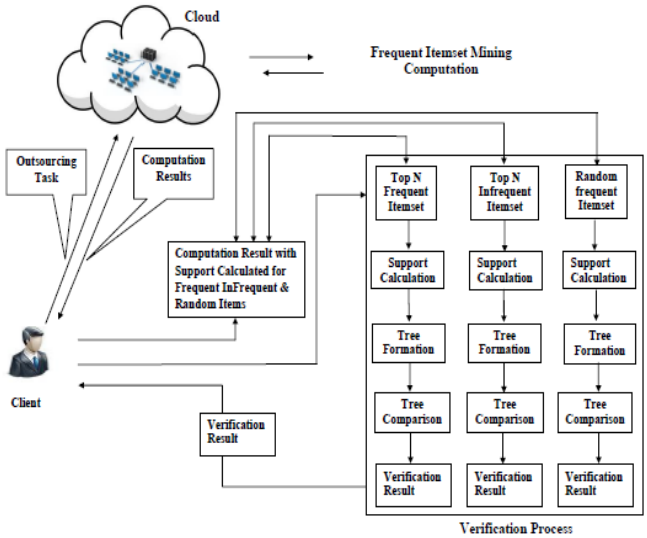

Figure 2: Enriched Framework architecture for verification of outsourced data mining computation.

The framework put forward an idea of the verification of outsourced data mining computation of frequent itemset mining. Fig 2 shows enriched framework architecture of outsourced data mining computation. The proposed system architecture depicts the process of integrity verification of outsourced computation of frequent itemset. In which end user outsourced the task to the cloud for frequent itemset mining. Then cloud returns the computation result to the end user. The end user creates power set from the small part of the outsourced task. Then compute for top N frequent, top N infrequent and random itemset by considering support

threshold value. Then calculate the support for the top N frequent, top N infrequent and random itemset. The M tree algorithm is used to form tree by using the support value calculated. The same support value itemsets are clustered at node end. Top frequent, top infrequent and random itemsets are searched in result received from the cloud. After searching for itemset from the result, the support value is calculated for them and tree is formed for the same. Then client compares the both trees to verify the integrity of return result. After comparing both trees if the clusters contain same the top frequent, top infrequent and random itemset generated for small part of a task and received cloud result then computation result is correct. In case few itemset of clusters not matched then it is not verified. In the end, depending upon how many percents of itemsets verified the verification report is generated for the end user.

B. Implementation Details:

Proposed methodology for verification of outsourcing computation of frequent itemsets to cloud is depicted in the above Fig 2. And the steps that are implemented to verify the computational fact of cloud can be elaborately explained with the below mentioned phases.

- ✓ Outsourcing Computational Task: In this step end user that is client creates credentials at the stand alone application at his/ her end. After that enduser outsourcing the task of frequent itemset computation by putting some of the items in the text file. At the the cloud end there is a web application is hosted which eventually provide a panel to the standalone application of the client to outsource the task using the interactive browser.As the client upload the text file it is receives at the web server end that is part of the cloud.
- ✓ Data Preprocessing: This is the task where initail scrutinization of the data is happened using four sub steps like
  - Here in this step symbols wont plays any role in frequent itemset creation. So by discarding them system makes the data more lightweight so that it can perform the further operations of preprocessing easily. Here some special symbols are discarded like, etc..
  - Here in this phase of the step the string that is get rid of the special symbols are now subjected to identify the conjective words in the string. For this process first special symbol freed string is divided on empty spaces and stored in a linear list. Each of the words from this linear list is now checked for the conjective words that are stored in the database or hardcoded in the program. If the word matches with the database list then it is repaced with the empty string.Some of the conjective words are like is, of , the, and , of etc..
  - Stemming Process: Here in this process the items are bringing to their base form so that the complexity of the task can be reduced to some extent. So this process eventually identifies the end string of each of the items, if it matches any protocols of the stemming like – if the string contains "ing" at the end then it is replaced with empty character. And if the string contains "ed" at the end then it also replaced with the empty character or with the suitable string. By doing this we are removing the redundant words to make the process of frequent itemset mining more easier and faster.
  - Tokenization: This is the last phase of the preprocess-ing step where all the stopword removed and stemmed words are stored in the list which eventually gives the index for each of the words.
  - Frequent Itemset and Support Computation: This is the phase where preprocessed list is taken into consideration for frequent itemset creation process. Frequent itemset is created based on the fact of Apriori algorithm along with the power set computation technique. Here by doing this each items are clubbed together with another individual items to produce the frequent patterns that can be shown in the below algorithm 1.

Algorithm 1: Frequent Itemset Generation

// Input:Itemset List L= {L_1, L_2, L_3……L_n}
Step 0: Start
Step 1: FOR i=0 to Size of L
Step 2: get i^{th} Element from the List L
Step 3: FOR j=i+1 to Size of L
Step 4: Get the j^{th}Element from the List L
Step 5: fit= i^{th} Element + j^{th} Element (fit= frequent Item set)
Step 6: END Inner FOR
Step 7: END Outer FOR
Step 8: Then fit U fit_k
Step 9: Repeat Step 1 to 8
Step 10: Return fit Set F
Step 11: Stop

Support calculation: This is the process where each of the frequent item set is checked for its presence in the other frequent pattern to get its presence ratio. This can be calculated using the following equation 1.

$$F(s) = (\sum fit \in fit_i) / n \qquad (1)$$

Where,
F(s) is frequent item set Support value
Fit – Frequent itemset
n- Total number of frequent itemsets

This step is carried out at web server's side that is integral part of the cloud.

- ✓ Verification parameter Preparation: This is the phase where client will download the resulted files of outsourced computation of frequent itemsets. This file contains the frequent sets which are associated with their support values that are estimated with the process as mentioned in the prior step. Once this file is downloaded it is subjected for verification process. For verification process client feed the system with original file that was used to outsource the task and outsourced resulted file that is downloaded from the cloud. Once the verification system receives these files as input then it computes the frequent itemset for the original file based on the three facts like most frequent itemsets, most infrequent itemsets and Random itemsets based on the given input from the user.Based on these facts of the original file same frequent itemset are fetched from the cloud's resulted data along with the estimated support value to form respective lists. And now these two lists of cloud data and original data are subjected for verification process.
- ✓ Tree Computation: This step prepares tree based on the support values of the itemsets . Here while creating the tree itself system accumulates all the itemsets which are having same support value to implicitly create a cluster at the tree nodes. Then these clusters are used for the verification process. This Tree creation can be depicted with the following algorithm 2.

Algorithm 2: M Tree Creation

Step 0: Start
Step 1: Create an empty tree as T
Step 2: Create the Root Node for first FrequentitemsetR_n
Step 3: FOR each element of F_v
Step 4: Compare the distance with the root node
Step 5: IF (F_{vi}support <R_n)
Step 6: Add node as left child in T
Step 7: Else
Step 8: Add node as Right child in T
Step 9: End For
Step 10: Stop

- ✓ Verification : This is the last step of the proposed model where it takes the two sets of clusters that are prepared in the prior step one for the original data and another for the outsourced data results. Then each of the respective clusters is extracted from the set based on the support values and they are analyzed for their same number of elements. If both the clusters are having same number of elements that means cloud data is perfect for the given support else there is some problem with the computation results. Based on these facts different cluster sets are been analyzed for Most frequent, Most infrequent and Random frequent elements. And finally a report is generated in the form of ratio that eventually represents the flaws in the computation of the cloud.
- ✓ Experimental Setup: For Experimental setup of our system proposed model uses Java technology for development in both client side and at web server side. Netbeans 6.9.1 is adopted as the development IDE with Swing as a front end. Whereas for designing the web application Dreamweaver 8.0 is used with apache tomcat as a web server and MYSQL 5.5 as Database Server. And the Project is deployed in private cloud computed with the Java plug-in which is developed by using Socket programming of the network interface. For this purpose, the system uses D-Link 4 port which with CAT 6 Ethernet switch.

4. Result and Discussion:

For the real time incorporation of the reduplication system we used one machine of standard configuration with i3 processor and 4 GB RAM. Deployment used java based windows machine with Netbeans as the development IDE. And the application is deployed using Apache tomcat server, which uses the private cloud instance to store the data. System is subjected to many crucial tests for its performance evaluation as stated in the below tests. The given Fig 3 show the graph of support calculated for the items taken as example by cloud and the end user for verification process.

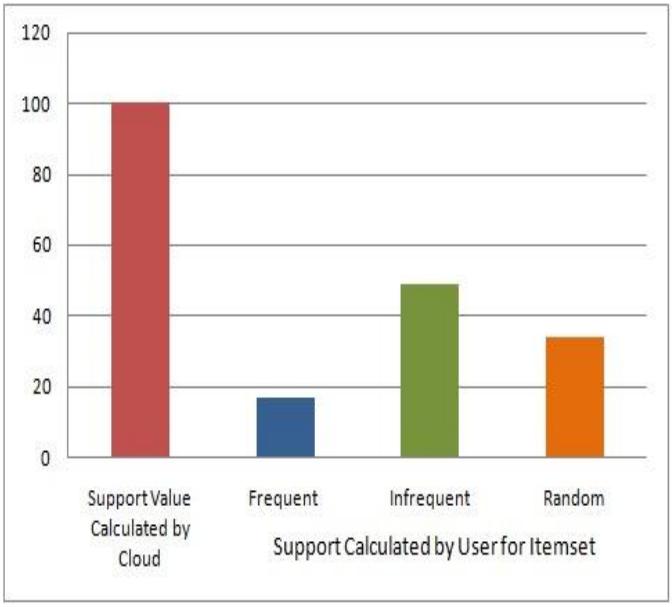

Figure 3: Support Values Calculated by Cloud & User.

Our validation technique is compared with that of cross validation method that is stated in [10]. [10] Uses the optimized apriori algorithm for validation of frequent itemsets. When apriori validation method is compared with our approach of tree based technique for the normalized high support values for the performance time parameter we got the results that are tabulated in the table 1.

Table 1: Performance time of Apriori Cross Validation Optimized Approach and Tree Pattern Approach

Support Value	Apriori Cross Validation Optimized Approach [10](In Seconds)	Tree Pattern Approach (In Seconds)
0.6	23	27
0.5	38	35
0.4	47	41
0.3	81	78
0.2	115	98

In [10], it is mentioned that Apriori cross validation optimized approach is performing better for support values. So our tree pattern is optimized for the higher and lower values of the support and compared with that of prior one. The plot in Fig2 indicates the enhancement of the performance from our proposed model over [10].

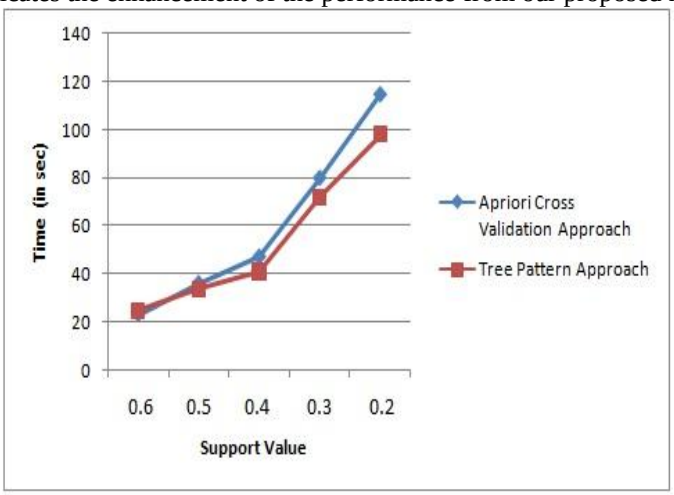

Figure 2: Performance Time Comparison of Apriori Cross Validation Optimized Approach V/s Tree Pattern Approach

5. Conclusion and Future Scope:

The work analyses all the possibilities of errors from the cloud for the task of the frequent item set computation by considering different sets of frequent item sets. Tree matching technique eases the process of verification and thereby generates a valid report of the same.The given result comparison graph of systems show that the proposed work has verify the outsourced computation of cloud for the frequent itemset mining with

better the performance and more efficiently. The system can be enlarge for the future by deploying it in the distributed system to the huge frequent itemset computationand verification act.

6. Acknowledgement:

I would like to express my deep and sincere gratitude to my research guide, Dr. L.V. Patil, Ph.D (Computer Sci), Professor (IT Dept), Smt. Kashibai Navale College of Engineering, Savitribai Phule Pune University, Pune, for providing valuable guidance throughout this research. His vision and motivation have deeply inspired me. It was a great privilege and honor to work and study under his guidance.

7. References:

1. Boxiang Dong, Ruilin Liu, Hui (Wendy) Wang, "Trust But Verify:Verifying Result Correctness of Outsourced Frequent Itemset Mining in Data-mining-as-a-service Paradigm" IEEE Transactions 10.1109/ TSC.2015. 24363872015.
2. Ran Canetti, Ben Riva, Guy N. Rothblum, "Practical Delegation of Computation using Multiple Servers" ACM 978-1-4503-0948-6/11/10 October 17–21, 2011.
3. Bryan Parno, Mariana Raykova, Vinod Vaikunta Nathan, "How to Delegate and Verify in Public Verifiable Computation from Attribute-Based Encryption" International Association for Cryptologic Research LNCS 7194, pp. 422–439, 2012.
4. Xiaofeng Chen, Jin Li, JianWeng, Jianfeng Ma, and Wenjing Lou, "Verifiable Computation over Large Database with Incremental Updates" European Symposium on Research in Computer Security (ESORICS 2014), LNCS8712, Springer, pp. 148-162, 2014.
5. W. K. Wong, David W. Cheung, Edward Hung, "An Audit Environment for Outsourcing of Frequent Itemset Mining" VLDB Endowment, August 2428, ACM 2009.
6. Zahra Farzanyar, Nick Cercone, "Efficient Mining of Frequent Itemsets in Social Network Data based on MapReduce Framework" ACM 978-1-4503-2240-9 /13/08 August 25-29, 2013.
7. Qiwei Lu, Wenchao Huang, Yan Xiong, and Xudong Gong, "Integrity Verification for Outsourcing Uncertain Frequent Itemset Mining" arXiv:1307. 2991v2 [cs.DB] 12 Jul 2013.
8. Ran Canetti, Ben Riva, Guy Rothblum, "Verifiable Computation with Two or More Clouds" Workshop on Cryptography and Security in Clouds, 2011.
9. Xiaofeng Chen, Jin Li, and Willy Susilo, Senior Member, IEEE "Efficient Fair Conditional Payments for Outsourcing Computations" IEEE Transactions on Information Forensics and Security, Vol. 7, No. 6, December 2012.
10. Predrag Stanisic, Savo Tomovic "Cross Validation Method in Frequent Itemset Mining" IEEEEmbedded Computing (MECO), 2012 Mediterranean Conference on19-21 June 2012, 978-1-4673-2366-6, 16 August 2012.
11. Rosario Gennaro, Craig Gentry, Bryan Parno,"Non-interactive Verifiable Computing: Outsourcing Computation to Untrusted Workers"International Association for Cryptologic Research, CRYPTO 2010, LNCS 6223, pp. 465–482, 2010.
12. Ruilin Liu, Hui (Wendy) Wang, Anna Monreale, Dino Pedreschi, Fosca Giannotti, Wenge Guo4,"AUDIO: An Integrity Auditing Framework of Outlier-Mining-as-a-Service Systems"Volume 7524Computer Science pp 1-18, Springer-Verlag Berlin 2012.
13. Suqing Lin, Rui Zhang, Hui Ma, and Mingsheng Wang, "Revisiting Attribute-Based Encryption With Verifiable Outsourced Decryption" IEE Transactions On Information Forensics And Security, Vol. 10, No. 10, October 2015
14. Xiaofeng Chen, Jin Li, and Willy Susilo, Senior Member, IEEE "Efficient Fair Conditional Payments for Outsourcing Computations" IEEE Transactions On Information Forensics And Security, Vol. 7, No. 6, December 2012.