MINING OF ASSOCIATION RULES FROM DISTRIBUTED DATA USING MOBILE AGENTS

Gongzhu Hu and Shaozhen Ding
Department of Computer Science, Central Michigan University, Mount Pleasant, MI 48859, U.S.A.

Keywords: Mobile agent, Distributed data mining, Privacy preserving.

Abstract: In this paper, we propose an agent-based approach to mine association rules from data sets that are distributed across multiple locations while preserving the privacy of local data. This approach relies on the local systems to find frequent itemsets that are encrypted and the partial results are carried from site to site. In this way, the privacy of local data is preserved. We present a structural model that includes several types of mobile agents with specific functionalities and communication scheme to accomplish the task. These agents implement the privacy-preserving algorithms for distributed association rule mining.

1 INTRODUCTION

More and more data analysis applications today, including data mining, deal with data that are distributed across multiple locations rather than on a single site. Various techniques have been developed in recent years for distributed applications, such as data warehousing, schema mapping and integration, and mobile agents.

Security has always been a challenge in distributed applications. One of the security issues is privacy-preserving. Although quite a number of approaches have been proposed in the past but few research results have been reported using the mobile agent technique for distributed data mining tasks and preserving privacy.

In this paper, we present an agent-based architecture for distributed association rule mining. The architecture involves several mobile agents, each of which performs some specific functions for the data mining task. These functions implements the existing privacy-preserving distributed data mining algorithms. Our implementation of this architecture and experiments show that the proposed approach produces the correct results and the privacy of data at individual sites are protected while the mobile agents carry the data traveling across the network.

2 DISTRIBUTED MINING OF ASSOCIATION RULES

The Fast Distributed Algorithm (FDM) (Cheung et al., 1996) was one of the approaches dealing with the issue of efficiently mining on the distributed sites. The work in (Kantarcioglu and Clifton, 2004) developed a encryption scheme to achieve privacy preserving for DDM. Our mobile agent method is based on these previous work, hence we briefly review them here.

2.1 Basic Notations

The notations and terminologies for distributed association rule mining are similar to regular (non-distributed) mining, but with extensions to represent the local-vs-global situation.

S_i	site i
s	minimum support threshold
F_k	globally frequent k-itemsets
$F^g_{i(k)}$	globally frequent k-itemsets at site S_i
$F^l_{i(k)}$	Locally frequent k-itemsets in $C_{i(k)}$
$C_{i(k)}$	candidate sets generated from $F^g_{i(k-1)}$
$X.sup_i$	Local support count of itemset X at site S_i

2.2 Globally and Locally Frequent Itemsets

One of the properties related to frequent itemsets is

Hu G. and Ding S. (2009).
MINING OF ASSOCIATION RULES FROM DISTRIBUTED DATA USING MOBILE AGENTS.
In *Proceedings of the International Conference on e-Business*, pages 21-26
DOI: 10.5220/0002231600210026

that every globally frequent itemset must be locally frequent at some sites. If an itemset X is globally frequent and also locally frequent at a site S_i, X is called globally frequent at site S_i. The set of global frequent itemsets at a site will form a basis for the site to generate its own candidate sets.

Another two properties involve an itemset and all of its subsets. First, if an itemset X is locally frequent at site S_i, then all of its subset is also locally frequent at site S_i. Second, if an itemset X is globally frequent at a site S_i, then all of its subsets are also globally frequent at site S_i.

In the FDM algorithm, the locally frequent k-itemsets $F^l_{i(k)}$ and globally frequent k-itemsets $F^g_{i(k)}$ at site S_i can be seen by other sites, and hence need protection, commonly by cryptographic techniques.

2.3 Encryption Scheme

Like other distributed computing tasks, cryptographic techniques should apply to distribute data mining to protect of data privacy. A revised version of FDM was proposed in (Kantarcioglu and Clifton, 2004) that replaced the "support count exchange" step in the original FDM by what are called *Secure Union* and *Secure Sum*.

Secure union means that each site participates in both encryption and decryption of $F^l_{i(k)}$. In other words, each site S_i will hold an encryption-description key pair (e_i, d_i), and $F^l_{i(k)}$ will be encrypted by e_i to generate $F^l_{ei(k)}$. The union of all the $F^l_{ei(k)}$ is then calculated as $F^l_{e(k)} = \bigcup_{i=0}^{n-1} F^l_{ei(k)}$. This value is then decrypted by each site's decryption key d_i and we can get the result as $F^l_{(k)}$. However, we don't know whether the items in $F^l_{(k)}$ are globally supported, in which we need to calculate the sum of the item counts securely. Secure union involves a notion called "commutative encryption" for an encryption model in which all the parties participates in both encryption and decryption. The detailed algorithm for secure union is given in (Kantarcioglu and Clifton, 2004).

Secure sum is an algorithm to get the sum of items' support count in every site in a secure way. Assuming that we have n sites and an item x, the task is to obtain $\sum_{i=1}^{n} x.sup_i$ and protect $x.sup_i$ from other sites. Each site generates a random number r_i added to $x.sup_i$ and sends the sum to the next site and the following sites will do the same. At the last site, we get the $\sum_{i=1}^{n} x.sup_i + \sum_{i=1}^{n} r_i$. This value then travels through all the sites, and the random number r_i is subtracted from the sum at each site. The final result will be $\sum_{i=1}^{n} x.sup_i$ as desired.

The privacy-preserving distributed association rule mining algorithm is separated into two parts: (1) computing $\sum_{i=0}^{n-1} F^l_{i(k)}$, and (2) testing whether $x \in \sum_{i=0}^{n-1} F^l_{i(k)}$ is globally supported.

3 AGENT-BASED MODEL

Mobile agents are the basis of an emerging technology that promises to make it much easier to design, implement and maintain distributed systems (Lange and Oshima, 1998a), (Lange and Oshima, 1998b). Because of the special characteristics, mobile agents are very suitable for distributed data mining. Based on the privacy-preserving method discussed in Section 2, we propose a structural model of six types of agents to work together for the data mining task.

3.1 Agents and their Functionalities

We proposed three objects in our agent framework: *local host*, *agent server* and *agent*. There are many local hosts and agents, and an *agent server* that communicates with the local hosts through agents.

3.2 Agents

We define six types of agents to be used in our scheme as described below.

Encrypt Secure Union Agent (ESUA)

This agent tries to coordinate the n hosts to participate encryption of each $F^l_{i(k)}$ (locally frequent k-itemset at host i). It carries the following necessary information:

a. Mining task description "secure union";
b. Prime number p for the encryption schema;
c. Host address list;
d. An array, P, of (itemset, item_label) pairs, where each item_label is uniformly distributed between 0 and $p-1$ and represents one itemset;
e. Support count s.

Once the host obtains this information from the agent, it will scan the local database and calculate $F^l_{i(k)}$. At the same time, the host randomly generates the encryption and decryption keys (e,d) by using the prime number p. For every $x \in F^l_{i(k)}$ the host searches P to get an integer P_j, and calculates $C_{ij} = P_j^e \bmod(p)$ for encryption. After the agent gets the encrypt result $C_i = \cup_{j=0}^{m} C_{ij}$, where m is the size of $F^l_{i(k)}$, it will perform some internal computation and migrate to the next host. Other hosts accept the array of integers C_i and encrypt it with their own encryption keys. There

are n ESUA agents working in the network at the same time in each itinerary of mining k-itemset, because each host generates one $F^l_{i(k)}$.

Decrypt Secure Union Agent (DSUA)

Once all the ESUAs come back to the Agent Master, the master is able to create $C = \sum_{i=0}^{n-1} C_i$, the union of ciphers in each host, as an array. The DSUA carries the integer array C of C_i's and travels through every host to pursue decryption. The information carried by this agent includes:

a. Mining task description "decrypt union";

b. Host address list;

c. Cipher list C.

When the host accepts this agent, it decrypts each $C_j \in C$ in the cypher list as $D_j = C_j^d \bmod(p)$, while d is the decryption key and p is the prime number generated at the host. The new array $D = \cup_j D_j$ will be carried by the agent.

Encrypt Sum Agent (ESA)

An Encrypt Sum Agent will carry Rule Set that contains pairs of item_label, count. It travels through all the hosts to obtain the encrypted support count. The information the ESA carries includes:

a. Mining task description "secure sum";

b. Host address list;

c. An array of RuleSet $\{item\_label_j, count_j\}$;

d. A large integer m that satisfies $m > 2|DB|$.

The host generates a random number, R_j, uniformly distributed between 0 to $m-1$ for each RuleSet. At the same time, it scans the local database to calculate the local support count sup_j for each item in the RuleSet as

$$(count_j \leftarrow R_j - s \times |DB_i| + sup_j + count_j) \bmod m. \quad (1)$$

This number will be passed to the next host for a similar operation. There will be one ESA agent working in the process of finding the k frequent itemset.

Decrypt Sum Agent (DSA)

Decrypt Sum Agent carries the array of RuleSet extracted from the returned ESA agent. It travels through all the hosts and let each host subtract the random number they generate when dealing with ESA. The information DSA carries includes:

a. Mining task description "decrypt sum";

b. Host address list;

c. An array of RuleSet $\{item\_label_j, count_j\}$;

Each host applies $count_j \leftarrow count_j - R_j$ and sets the changed RuleSet back to the agent. The agent migrates to the next host for decryption.

Broadcast Agent (BA)

When DSA comes back to the agent master, the globally frequent k-itemsets F_k can be calculated from the decrypted RuleSet. In order to let each host calculate $F^g_{i(k)}$ and $C_{i(k)}$, BA is used to carry F_k to every host. The information BA carries includes:

a. Mining task description "broadcast";

b. F_k.

BA does not request any data from the host; it just notifies host that the current global frequent k-itemset has been calculated and asks the host to prepare $F^g_{i(k)}$ for next iteration. There are n BAs for each of the k iterations. BA agents help to reduce the number of candidate itemsets. When a BA arrives at a local host, the host machine calculates $C_{i(k)}$ based on $F^g_{i(k)}$, while the size of $C_{i(k)}$ is usually smaller than the global candidate k-itemset C_k.

Over Agent (OA)

Once Agent Master found that either all the RuleSets extracted from DSA are not globally supported or the size of $C_{i(k)}$ is 0, it will dispatch this OA agent to notify the hosts that the algorithm has terminated.

3.3 Agent Master (Server)

The agent master has two primary functions: activate an agent server to perform the needed task; create new agents and dispatch them to remote hosts. The actions taken by the agent server is given in Algorithm 1.

Each of the actions is performed according to the encryption scheme described in section 2.3.

3.4 Host

Each host machine performs local association rule mining and the encryption/decryption operations upon the arrival of different types of agents. These operations are based on the description given in section 3.2. For example, the action "Encrypt secure sum agent" looks like this:

Extract $rs \leftarrow A.task.RuleSet$ and
  $m \leftarrow A.task.m$ from A;
Generate a random number array R uniformly
  distributed in $[0, m-1]$;
for $i = 0$ to $rs.size - 1$
  $support\_count \leftarrow$ the support count of
    itemset that matches $rs[i].item\_label$;

Algorithm 1: Agent server.

```
1  begin
2      Decide frequent 1-itemsets and a prime
       number p;
3      Assign a item_label for each itemset;
4      Create n ESUA agents for n hosts;
5      for i = 0 to n − 1 do
6          Create one thread to dispatch ESUA[i];
7      end
8      Receive agent A;
9      switch A.description do
10         case "Secure Union": secure union;
11         case "Decrypt Union": decrypt union;
12         case "Secure Sum": secure sum;
13         case "Decrypt Sum": decrypt sum;
14         case "Broadcast": broadcast;
15         case "Over":
16             if all OAs have returned back then
17                 Terminate.
18     end
19 end
```

$$rs[i].count \leftarrow (rs[i].count + R[i] + support\_count - s \times |DB_i|) \bmod m;$$

Pass rs to A and A migrates to next host;

4 EXPERIMENTS AND RESULTS

We have conducted several experiments to apply the proposed agent approach to perform association rule mining on data sets of various sizes. Because the methods of extracting association rules from a frequent itemset with a given support and confidence thresholds are the same with or without using agent approach, we only show the calculation of the frequent itemsets and will not go further to show the association rules.

4.1 Data Set

The data set is a table that records 7985 transactions of the banking services. The task of association rules mining is to find the relationships between different kinds of banking services. In this table, the columns (HMEQC, CKCRD, MMDA, etc.) represent 13 different banking services. This database table uses Boolean value to show whether a banking service exists in a transaction. A sample data is shown in Table 1. The dataset is distributed across three sites with 2034, 2013 and 3938 records, respectively. The items' names are those banking services as the column names in Table 1. An item_label is generated along with the name of each of the banking services. For example, the item_label for HMEQLC is 2373416.

The agent server uses a prime number $p = 5555527$ and the support threshold $s = 5\%$. The encryption and decryption keys at the three sites are (757019, 4119587), (952657, 1364743) and (555557, 3409073), respectively. Due the page limitation, we only show the details of the process for generating the frequent 1-itemsets.

4.2 Frequent 1-Itemsets

The locally frequent 1-itemsets and their support counts at the three site S_1, S_2, and S_3 are shown in Table 2.

The agent server dispatches three ESUAs to every host to perform encrypt secure union. We will briefly show the process for one item, HMEQLC, with its item_label 2373416. According to the algorithm described before, the item_label is encrypted at the three sites as:

$$\begin{aligned} S_1: \ & 2373416^{757019} (\text{mod } 5555527) = 542566 \\ S_2: \ & 542566^{952657} (\text{mod } 5555527) = 3334375 \\ S_3: \ & 3334375^{555557} (\text{mod } 5555527) = 589086 \end{aligned}$$

The three DSUAs will decrypt the secret "item_label" 589086 using their decryption keys as

$$\begin{aligned} S_1: \ & 589086^{4119587} (\text{mod } 5555527) = 4644358 \\ S_2: \ & 4644358^{1364743} (\text{mod } 5555527) = 4213937 \\ S_3: \ & 4213937^{3409073} (\text{mod } 5555527) = 2373416 \end{aligned}$$

We can see that after leaving S_3, the item_label is recovered back to 2373416 that is cast to the item HMEQLC. The other items go through the same encrypt and decrypt process, and the final item_labels map to the correct item names.

In the next step, we use "secure sum" to test whether these locally frequent itemsets are also globally frequent. The agent server sends an ESA (Encrypt Sum Agent) with an integer $m = 20000$ that is larger than $2 \times |DB|$. The count field will be modified securely at each host according to Equation (1). Let's take an example of 2373416 that represents HMEQLC. The initial RuleSet is (item_label, count) = (2373416, 0). With the random numbers for the item HMEQLC at the three sites 11510, 5914, and 18213, the secure sum calculations for the support counts are

Table 1: Transaction data table (a small fraction of 7,985 transactions).

Trans. id	HMEQC	CKCRD	MMDA	PLOAN	AUTO	ATM	SVG	CD	MTG	IRA	TRUST	CKING	CCRD
513394	0	0	0	0	0	0	0	0	0	0	0	1	0
513414	1	0	1	0	0	1	1	0	0	0	0	1	0
513421	0	0	0	0	1	1	1	0	1	0	0	1	0
513479	0	0	1	0	0	0	0	0	0	0	0	1	0
513660	0	0	0	0	0	1	1	0	0	0	0	1	0
513865	0	0	0	0	0	0	1	0	0	0	0	1	0
513708	0	0	1	0	0	1	0	1	0	0	0	1	0
513822	1	0	0	1	1	0	1	0	0	0	0	1	0
513862	0	0	0	0	0	1	1	0	0	0	0	1	0
513972	0	0	0	0	0	1	1	1	0	1	0	1	0
513983	1	0	0	0	0	1	1	0	0	1	0	1	0
514019	0	0	0	0	0	1	1	0	1	0	0	1	0
514122	0	1	0	0	1	0	1	0	0	0	0	1	1
514126	0	0	1	0	0	1	0	0	0	0	0	1	0
514153	1	0	0	0	0	0	1	1	0	0	1	1	0
514172	0	0	0	0	0	1	0	0	0	0	0	1	0
514472	0	0	0	0	0	1	0	0	0	0	0	1	0
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…

Table 2: Locally frequent 1-itemsets and support counts.

$F^l_{1(1)}$ at S_1		$F^l_{2(1)}$ at S_2		$F^l_{3(1)}$ at S_3	
HMEQLC	335	HMEQLC	334	HMEQLC	667
CKCRD	249	CKCRD	221	CKCRD	442
AUTO	189	AUTO	198	AUTO	355
ATM	787	ATM	791	ATM	1495
SVG	1247	SVG	1255	SVG	2441
CD	476	CD	526	CD	957
MTG	179	MTG	135	MTG	280
IRA	207	IRA	226	IRA	433
CKING	1761	CKING	1735	CKING	3358
CCRD	315	CCRD	31	CCRD	591
		TRUST	109		

S_1: (0 + 11510 + 335 - 2034 $\times$ 0.05) mod 20000
= 11743
S_2: (11743 + 5914 + 334 - 2013 $\times$ 0.05) mod 20000
= 17890
S_3: (17890 + 18213 + 647 - 3938 $\times$ 0.05) mod 20000
= 16553

Then, three DSAs (Decrypt Sum Agent) at the three sites proceed as follows:

$$S_1: (16553 - 11510) \bmod 20000 = 5043$$
$$S_3: (5043 - 5914) \bmod 20000 = 19129$$
$$S_3: (19129 - 18213) \bmod 20000 = 916$$

Since the result support count for the item HMEQLC is 916, which is less than $m/2$, HMEQLC is included in the globally frequent 1-itemsets F^g_1. Similarly, the same procedure applies to the other items in Table 2. Eleven of the 12 items, except (TRUST, 19990) that has a support count 19990 larger than $m/2$, are identified as globally frequent after the process. This result is the same as the case when the *Aprior* algorithm is applied when all transactions are on one site rather than distributed. Note that neither the encrypt sum nor the decrypt sum operation reveals the local information during the process.

Next, the agent server dispatches BA (Broadcast Agent) to each host and lets them generate $F^g_{i(1)}$. As the result, all the 11 globally frequent 1-itemsets are also locally frequent at each host. That is, $F^g_{i(1)} = F^g_1$ for $i = 1,2,3$.

We also calculated frequent 2, 3, and 4-itemsets and the results are also the same as the non-distributed case.

From the global k-itemsets, generation of association rules is straightforward and we do not show the results here.

5 RELATED WORK

A research field referred to as distributed data mining (DDM) that is expected to perform partial analysis of data at individual sites and combine those results to obtain the global results (da Silva et al., 2006).

The privacy-preserving problem in DDM has been a research topic for some time. Early work on this issue can be found in (Clifton and Marks, 1996), for example. Metrics for quantification and measurement of privacy preserving data mining algorithms was proposed in (Agrawal and Aggarwal, 2001). The paper (Rizvi and Haritsa, 2002) presented a scheme to achieve a high degree of privacy and at the same time retain a high level of accuracy in the mining results. A revised version of FDM was proposed in (Kantarcioglu and Clifton, 2004) that included an encryption

scheme to preserve privacy.

Mobile agent as one of the models for distributed applications has been used for data mining tasks. Several agent-based data mining methods were developed, such as creating an accurate global model using a modified decision tree algorithm (Baik et al., 2005), and a bidding mobile agent scheme (Peng et al., 2005) to achieve privacy. The paper (Cartrysse and van der Lubbe, 2004) addressed the privacy problems in agent technology and offered several solutions.

6 CONCLUSIONS

Distributed data mining is one of the distributed applications for which there are potential risks of leaking data privacy when distribute sites communicate to obtain global knowledge. In this paper, we proposed an agent-based approach to address this problem to mine association rules securely from data resides across multiple sites. The privacy preserving characteristics of this approach relies on the encryption and decryption techniques that are applied to the calculation of the union of the frequent k-itemsets and the sum of the support counts. In the proposed method, several types of agents are used to perform the encryption and decryption of the *secure union* and *secure sum* operations. In an experiment with about 8,000 transactions, the result (globally frequent k-itemset) by our approach applied to the data distributed cross three sites is the same as the result that would be obtained from the *Aprior* algorithm with the same data reside on a single host. And, the data carried by the agents are scrambled, indistinguishable and only being encrypted and decrypted when all the hosts participate.

Although our agent system is capable of securely computing the frequent itemsets, there are areas that need further study such as system stability (e.g. recover from single site crash) and security improvement (e.g. trustworthiness of the agent server).

REFERENCES

Agrawal, D. and Aggarwal, C. C. (2001). On the design and quantification of privacy preserving data mining algorithms. In *Proceedings of the 20th ACM SIGMOD-SIGACT-SIGART symposium on Principles of Database*, pages 247–255. ACM.

Baik, S. W., Bala, J., and Cho, J. S. (2005). Agent based distributed data mining. In *Parallel and Distributed Computing: Applications and Technologies*, volume 3320 of *Lecture Notes in Computer Science*, pages 42–45. Springer.

Cartrysse, K. and van der Lubbe, J. C. A. (2004). Privacy in mobile agents. In *IEEE First Symposium on Multi-Agent Security and Survivability*, pages 73–82. IEEE Computer Society.

Cheung, D. W.-L., Ng, V. T. Y., Fu, A. W.-C., and Fu, Y. (1996). Efficient mining of association rules in distributed databases. *IEEE Transactions on Knowledge and Data Engineering*, 8(6):911–922.

Clifton, C. and Marks, D. (1996). Security and privacy implications of data mining. In *ACM SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery*, pages 15–19.

da Silva, J. C., Klusch, M., Lodi, S., and Moro, G. (2006). Privacy-preserving agent-based distributed data clustering. *Web Intelligence and Agent Systems*, 4(2):221–238.

Kantarcioglu, M. and Clifton, C. (2004). Privacy-preserving distributed mining of association rules on horizontally partitioned data. *IEEE Transactions on Knowledge and Data Engineering*, 16(9):1026–1037.

Lange, D. B. and Oshima, M. (1998a). Mobile agents with Java: The Aglet API. *World Wide Web*, 1(3):111–121.

Lange, D. B. and Oshima, M. (1998b). *Programming and Deploying Java Mobile Agents Aglets*. Addison-Wesley Longman Publishing.

Peng, K., Dawson1, E., Nieto1, J. G., Okamoto1, E., and Lpez, J. (2005). A novel method to maintain privacy in mobile agent applications. In *Cryptology and Network Security*, volume 3810 of *Lecture Notes in Computer Science*, pages 247–260. Springer.

Rizvi, S. J. and Haritsa, J. R. (2002). Maintaining data privacy in association rule mining. In *Proceedings of the 28th International Conference on Very Large Data Bases*, pages 682–693. ACM.