Reducing Communication Cost in a Privacy Preserving Distributed Association Rule Mining

Mafruz Zaman Ashrafi, David Taniar, and Kate Smith

School of Business Systems, Monash University, Clayton, VIC 3800, Australia.
{Mafruz.Ashrafi,David.Taniar,Kate.Smith}@infotech.monash.edu.au

Abstract. Data mining is a process that analyzes voluminous digital data in order to discover hidden but useful patterns from digital data. However, discovery of such hidden patterns has statistical meaning and may often disclose some sensitive information. As a result privacy becomes one of the prime concerns in data mining research community. Since distributed association mining discovers global association rules by combining local models from various distributed sites, breaching data privacy happens more often than it does in centralized environments. In this work we present a methodology that generates global association rules without revealing confidential inputs such as statistical properties of individual sites and yet retains high level of accuracy in resultant rules. One of the important outcomes of the proposed technique is that it reduces the overall communication costs. Performance evaluation of our proposed method shows that it reduces the communication cost significantly when we compare with some well-known distributed association rule mining algorithms. Furthermore, the global rule model generated by the proposed method is based on the exact global support of each itemsets, and hence diminished inconsistency, which indeed occurs when global models are generated from partial support count of an itemset.

1 Introduction

Modern organizations are distributed in various geographical locations. Various business applications used by such organizations normally store their day-to-day data in their corresponding sites. Discovering useful patterns from such organizations using a centralized data mining approach is not always feasible due to a huge network communication cost which is imposed when merging all datasets into a central location. As a result distributed data mining emerges as a new sub-area of research in the data mining domain [1].

Data mining algorithms analyze enormous digital data and discover hidden patterns within the dataset, and hence impose a threat that such a discovery may breach the privacy of data. As a result preserving privacy appears to be a prime concern in the field of data mining [2-7]. Distributed data mining algorithms discover patterns beyond the organization boundary. Hence it threatens the privacy of individual sites more than that of the centralized approach of data mining.

One of the most important fields in distributed data mining is association rule mining. It has attracted a huge attention from numerous research communities. Many interesting and efficient distributed/parallel association rule mining algorithms have

Y. Lee et al. (Eds.): DASFAA 2004, LNCS 2973, pp. 381–392, 2004.

been proposed in the data mining literatures [8, 9]. Most of these algorithms overlooked the privacy issues. However, we believe that privacy should be a main concern of distributed association mining; otherwise the resultant patterns may reveal sensitive information and hence participating sites may loose their businesses. For example, a multi-national company would like to mine its data to find global association rules. However, the laws of individual country may come as an impediment to share global data. Furthermore, privacy does not always means that disclosure of individuals or personal information, rather it may disclosure corporate information or their transaction details. Indeed, such disclosure may allow identifiable information or corporate plans, which may threat corporate business gain.

Association rule mining is one of the most import data mining sub-areas. It discovers all associations with the data that satisfy the user specified minimum support and minimum confidence constraints. There are a number of researches in association rule mining that do not compromise privacy. We can categorize these works into two: (*i*) distortion or randomization approach [2, 3], and (*ii*) secure multi party computation approach [4, 5].

Randomization Approach: The randomization approach intends to discover association rules from randomization datasets of various sites. It focuses on privacy of individual site, and it does not reveal original records of one site to other participating sites. To preserve privacy, transactions are randomized by discarding some items and inserting new items into it [2]. The statistical estimation of original supports and variances given randomized supports allow a central site to adopt the Apriori [10] algorithm to mining frequent itemsets in non-randomized transactions by looking at only randomized ones.

Secure Multiparty Computation: The goal of secure multiparty computation is to build a data mining model from local datasets of various participating sites without revealing individual records of one site to other participating sites. To achieve this, it computes a function $f(x, y)$, where in this case two parties hold their inputs x and y, and at the end all parties know about the result of the function $f(x, y)$ and nothing else.

However, both of these approaches require huge communication costs especially in a distributed context. For example, the randomized approach combines all randomized datasets of participating sites into a centralized site and then adopts the Apriori [1] or any other association rule mining algorithm to find frequent itemsets. Whereas the second approach sends randomized support counts multiple times to other participating sites to generate final association rule mining models. Due to this, both of the approaches incur huge network communication and consequently fade away the intention of distributed association rule mining in the first place.

In this paper, we present a methodology that generates global association rules without revealing confidential inputs such as statistical properties of individual sites. One of the important outcomes of the proposed technique is that, it has the ability to minimize a collusion problem, which occurs when two sites on the chain collude to find the exact support of other site. Furthermore, we obtain the global rule model without increasing the overall communication costs at a great extend. Furthermore, it diminishes the reconstruction problem, which is raised when we distort transactions of a dataset by using different randomization techniques.

2 Distributed Data Mining: Background

Distributed Data Mining (DDM) intends to discover rules from different datasets that are distributed across multiple sites and interconnected by a communication network. It tries to avoid the communication cost of combining datasets in a centralized site, which requires huge amounts of network communication. It offers a new technique to discover knowledge or patterns from such loosely coupled distributed datasets and produces global rule models by requiring minimal network communication.

2.1 Distributed Association Rule Mining

Distributed Association Rule Mining (DARM) is a sub-area in distributed data mining. Typically, rules generated by distributed association rule mining algorithms are considered interesting if they satisfy both minimum global support and confidence threshold. In order to find interesting global rules, DARM algorithms generally have two distinct tasks: (*i*) *global support count*, and (*ii*) *global rules generation*.

The prime objective behind DARM is to reduce communication costs in such a way that the overall cost will be less than the cost if we combined datasets of all participating sites into a centralized site. For example, consider there are S_1, S_2, S_3... S_n sites involved in the mining task and each of the site has its own dataset of size D_1, D_2, D_3... D_n. Let C_1, C_2, C_3... C_m is the communication cost incurred after every iteration. Then, we can calculate the total communication cost for generating global frequent itemsets, $G_c = \sum_{i=1}^{m} C_i$, where m is the total number of iterations. Suppose the total communication cost of combining all n number of participating site's datasets into a centralized site, $D_c = \sum_{i=1}^{n} D_i$. Then, $G_c < D_c$ will be the prime property of any DARM algorithms.

2.2 Privacy

DARM algorithms should discover association rules beyond the organization boundary. They form the final rule model by combining various local patterns. For example, suppose there are three sites S_1, S_2 and S_3, and each of them have datasets D_{S1}, D_{S2} and D_{S3}. Suppose A and B are two items having a global support threshold. In order to find rule $A \Rightarrow B$ or $B \Rightarrow A$, we need to aggregate the local support of itemsets AB from all participating sites. When we do such aggregation, all sites learn the exact support count of other sites. However, in many situations, participating sites are reluctant to disclose the exact support of itemset AB to other sites, because support counts of an itemset has a statistical meaning and this gives a threat the privacy.

For the abovementioned reason, we need secure multiparty computation solutions to maintain privacy in DARM [5]. The goal of secure multiparty computation in DARM is to find global support of all itemsets using a function where multiple parties hold their local support counts, and at the end all parties know the global support of all itemsets and nothing else. And finally each participating site uses that global support for rules generation.

2.3 Problem Definition

Consider $X = \{x_1, x_2, x_3 \ldots x_n\}$, $Y = \{y_1, y_2, y_3 \ldots y_n\}$ and $Z = \{z_1, z_2, z_3 \ldots z_n\}$ be support count of candidate *k-itemsets* geographically distributed over three sites such as S_1, S_2 and S_3. In order to generate global frequent *k-itemsets* $F = \lambda\ (X + Y + Z)$, each site needs to send their respective support count of each candidate itemset λ to the other sites. Although broadcasting of support counts does not disclose any information about individual transaction, however it may disclose some valuable information about each site, such as data size, exact support of each itemset, etc., which may subsequently breach the privacy of each site. Hence, the challenge is to find global frequent itemset *F* without revealing λ (i.e. support) of each site.

Furthermore, in a distributed association rule mining context, messages exchange between different sites is considered as one of the main tasks. And this task becomes more expensive when each site broadcasts large numbers of support counts. Indeed, the communication cost of all distributed association rule mining algorithms will increase when privacy of each site is considered as the primary objective. Hence, the problem can be defined as finding all association rules from various distributed sites without revealing support counts of individual site and without increasing the overall communication costs.

3 Related Work

There are several numbers of frameworks that have been proposed for maintaining the privacy of association rules [2-7]. However, most of them dealt with sequential or centralized association mining. MASK [2] was proposed for centralized environment to maintain privacy and accuracy of resultant rules. This approach was based on simple probabilistic distortion of user data, employing random numbers generated from a pre-defined distributed function. If we use this algorithm in the context of distributed environment, we need uniform distortion among various sites in order to generate unambiguous rules. However, this uniform distortion may disclose confidential inputs of individual site and may also breach the privacy of data hence not suitable for distributed mining.

Evfimievski et al. [3] provide a randomization technique to preserve privacy of association rules. The authors analyzed this technique in an environment where there are a number of clients connected to a server. Each client sends a set of items to the server where association rules are generated. During the sending process client modifies that set (i.e. items) according to its own randomization policy, hence server is unable to find the exact information about the client. However, this assumption is not suitable for distributed association rule mining because it generates frequent itemsets by aggregating support counts of all clients (i.e. sites). If the randomization policy of each site differs from others, we will not be able to generate the exact support of an itemset. Subsequently, the resultant global frequent itemsets will be erroneous. Furthermore, this technique individually disguises each attribute, and data quality will degrade significantly when number of attributes in a dataset is large.

Atallah et al. [6] introduce a new technique to preserve privacy of sensitive knowledge by hiding out frequent itemsets from large datasets. The authors apply some heuristic to reduce the number of occurrences in such a degree that its support is

below the user specified support threshold. Dasseni et al. [7] extended this work and investigated confidentiality issues of association rule mining. Both work assumes datasets are local and hiding some itemset will not affect the overall performance or mining accuracy. However, in distributed association rule mining, each site has its own dataset and similar kind of assumptions may cause ambiguities in resultant rules.

Vaidya et al. [4] propose a technique to maintain privacy of association rules in vertically partitioned distributed data sources (across two data sources only) where each data site holds some attributes of each transaction. However, if the number of disjoints attributes among the site is high, this technique incurs huge communication costs. Furthermore, this technique worked only for two sites, hence not scalable.

Kantercioglu et al. [5] propose privacy preserving association rule mining for horizontally partitioned data. The authors propose two different protocols: secure union of locally large itemsets and testing support threshold without revealing support counts. The former protocol uses cryptography to encrypt local support count, and therefore, it is not possible to find which itemset belongs to which site. However, it reveals number of itemsets having a common support. The latter protocol adds a random number to each support counts and finds excess supports. Finally, these excess supports are sent to the second site where it learns nothing about the first site actual dataset size or support. The second site adds its excess support and sends the value until it reaches the last site. However, this protocol can raise a collusion problem. For example, site i and $i+2$ in the chain can collude to find the exact excess support of site $i+1$. Furthermore, this protocol only discovers an itemset, which is globally large; not the exact support of an itemset, and each site generates rules based on the local support counts.

In this work, we propose an efficient technique that maintains privacy of distributed association rule mining according to a secure multiparty computation definition [11]. The proposed technique accomplishes to find the exact support of each global frequent itemset without revealing the candidate support counts of individual sites. The proposed technique has the ability to minimize the collusion problem without increasing overall communication costs. Furthermore, it eliminates the reconstruction problem, which is raised when we distort transactions by using different randomization techniques.

4 Proposed Method

In this section, we describe a methodology that maintains privacy of DARM. At first, let us find out the rationale why each site in distributed association rule mining shares its support count of each itemset with all other sites. Firstly, in the context of DARM, each participating site needs to know whether an itemset is globally frequent or not, in order to generate candidate itemsets for the next pass. Without that piece of information, DARM algorithms will not be able to generate global candidate itemsets. Secondly, if any site generates rules based on partial support count of an itemset, the inconsistence problem (i.e. confidence of rules at different sites may vary) will arise. Before embarking on the details of our proposed method, let us discuss the basic notations and assumptions of this framework.

4.1 Assumptions

Dataset Model: We assume there are N numbers of participating sites and each of them has their own dataset. Each transaction of the datasets has a set of items, and each item is represented by a number. Number of items in a transaction may vary. The dataset of each site may be heterogeneous, but items of each transaction at different sites have the same taxonomy level. Furthermore, each dataset does not have the same number of transactions. So only the aggregate of support count will allow us to identify whether an itemset has global support threshold or not.

Number of Sites: The aim of this work is to find an exact global support of all itemsets without revealing the exact support counts of each participating site. However, when the number of participating sites is equal to two, then it becomes very easy for the both sites to find out the exact support counts of the other site, no matter what kind of secure computation we enforce [5]. Hence, we assume the number of sites participating with this framework is equal to n, where $n > 2$.

To overcome the above problem, one may think the randomization techniques. However, we believe that if different sites distort their respective dataset using a non-uniform randomize faction or add/drop some of the items from the transaction, resultant global rule model will be inconsistent and subsequently diminish the aim of distributed association rule mining in the first place.

Characteristics of each site: Each site participating in distributed association rule mining should possess a minimum level of trust. Ideally it is easier for any distributed association rule mining algorithms to maintain privacy, if all computations are done by a trusted third party. However, this kind of solution is not feasible because of various limitations. Due to this reason, we assume that all participating sites are *semi-honest* sites. A semi-honest site possesses the following characteristics:

- Follow multi-party computation protocols completely,
- Keep record of all intermediate computation, and
- Be capable of deriving additional information using those intermediate records.

4.2 Methodology

We now discuss how we generate global association rules without revealing the exact support counts of participating sites. Our proposed method is based on the following analogy. For example, suppose we have a large real number $N \subseteq R$, which is a sum of two numbers such as N_1 and N_2 where $N_1 \subseteq R$ and $N_2 \subseteq R$, $N_1 \neq 0$ and $N_2 \neq 0$ and R is a real number. If we consider the value of N is known, and N_1 and N_2 are unknown, the value of N_1 or N_2 remains private and secure, until we know the exact value of either N_1 or N_2.

The proposed method uses the abovementioned technique and considers N_1 as an exact support count of an itemset, N_2 as a random number, and N as an obfuscated support count. It has two distinct phases, namely (*i*) *obfuscation* and (*ii*) *de-obfuscation*. In the *obfuscation phase* as shown in the figure 1(a), each support counts of candidate itemset is obfuscated (i.e. an addition of exact support counts and a random number) and is sent to the adjacent site. Each adjacent site then aggregates its obfuscated support count with the receiving support counts, and sends that aggregation to the next site. This sending process continues until it reaches the last

site. When it reaches the last sites, it aggregates its obfuscated support count with the receiving support count. We call this global obfuscated support count.

After computing the global obfuscated support count in the last site, it sends global obfuscated support counts to the first site for de-obfuscation as shown in the figure 1(b). The first site starts the *de-obfuscation phase* by subtracting its own generated random number from each of the global obfuscated support counts and sends to the adjacent site. Each site subtracts their respective random number and sends to the next site till it reaches the last sites. When the last site subtracts its own generated random number from the global obfuscated support count, we found the exact global support of the itemset. Then, all global support counts are checked in order to prune away each non-frequent global itemset. Finally, those global frequent support counts are sent to all other sites and each site generates candidate itemsets for the next iteration. This process continues until there is no more candidate itemsets that can be generated from the previous frequent itemsets. To elaborate this process, let us explain using the following example.

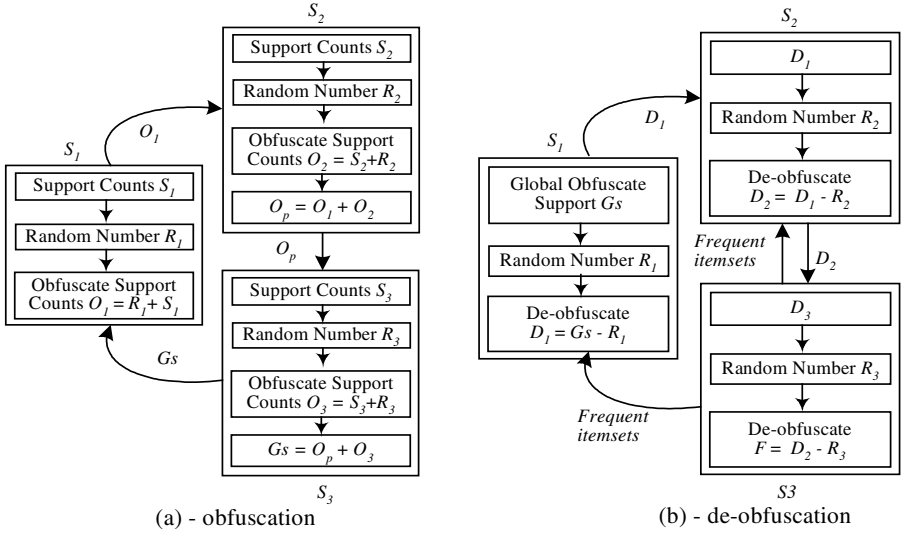

Fig. 1. Proposed methodology

Example: Consider there are three sites; such as S_1, S_2 and S_3 and after the first iteration global candidate *2*-itemsets is equal to {*AB*, *AC* and *BC*}. Let local support counts of those candidates of site S_1 be equal to {*5*, *3* and *4*} at S_2, be equal to {*10*, *9* and *1*} and at S_3, be equal to {*5*, *5* and *1*}. Suppose site S_1, S_2 and S_3 generate a random number $R_1 = 100$, $R_2 = 200$ and $R_3 = 200$ and each site obfuscates its own support count of candidate itemsets by adding a corresponding random number with each support counts. After performing obfuscation consider site S_1 sends it's obfuscated support counts set (i.e. *105*, *103* and *104*) to site S_2. When S_2 receives this obfuscate support counts, it aggregates its own obfuscate support counts {*210*, *209* and *201*} with the receiving support counts. Since each site shares the same candidate itemset, this aggregation operation can be done on the fly. Upon performing these tasks, site S_2 sends obfuscate support counts set {*315*, *312* and *305*} to site S_3 that performs the same task as it does for site S_2 and finishes the obfuscation phase.

In the next phase (i.e. de-obfuscation), site S_3 sends the global obfuscate support counts set {*520*, *517* and *506*} to site S_1 where it subtracts a random number R_1 from each element of that itemsets. However, this subtraction does not reveal any knowledge to site S_1 because R_1 is subtracted from global obfuscate support counts hence candidate itemset support counts of other sites remains hidden to site S_1. After subtracting R_1, site S_1 sends support counts set {*420*, *417* and *406*} to site S_2 that subtracts the random number R_2 and sends support counts to site S_3. After subtracting random number R_3, site S_3 finds the global supports counts {20, 17 and 6} and discover global frequent itemsets. Since the global support counts is an aggregation of all local candidate support counts, hence it is not possible for site S_3 to discover the exact support counts of the other sites (i.e. S_1 and S_2) form that set.

One of the important outcomes of the proposed method is that it minimizes the collusion problem. This is because each site obfuscates candidate support counts with its own random number and that random number is subtracted from global obfuscated support counts on that particular site to perform de-obfuscation. Hence, it requires *n-1* sites of the chain to collude to find out the exact support count of any site S_i.

4.3 Message Optimization

In distributed association rule mining, exchanging messages between different sites is considered as one of the main tasks. Due to this reason, message optimization becomes an integral part of distributed association rule mining algorithms. However, our proposed methods are not able to reduce the message exchange size because it accomplishes the global support counts in two rounds and in each round it exchanges messages. To reduce message exchange size we proposed a further modification of our method where we follow the proposed method to discover frequent itemset of length 1. After that each site uses a function $f(x)$ to obfuscate the support counts by utilizing local support count of those global frequent itemsets, rather than a random number. We use the local support counts of global frequent candidate itemset, since other sites knew the obfuscated support count of an itemset, but not the exact value of it. The function $f(x)$ can be calculated by using the following formula:

$$f(x) = O_S \pm C_S \tag{1}$$

Where, O_S is the exact support count of a *k*-itemset and C_S is the sum of *n* number of local support of *k-1*-itemset. After generating the obfuscated candidate support counts, each site sends the support counts of candidate itemsets to a single site, where the global frequent itemsets for that iteration will be obtained. We refer to the sites that send obfuscated support counts as a *sender* and the sites that generate the global frequent itemsets as *receiver*. For example, if there are three sites participating in the process, two of them will broadcast the obfuscate support counts to the third site. Once the *receiver* site receives an obfuscated local support from different *sender* sites, it aggregate them using the following formula:

$$G_S = \sum_{i=1}^{n} f(x) \pm F_S \tag{2}$$

Where, G_S is the global support count of a *k*-itemset and F_S is the sum of *n* number of global support of *k-1*-itemset. The sum of local support of an itemset is equal to the global support of that itemset, so one can easily prove that $F_S = \sum_{i=1}^{n} C_S$. As a result

when we add or subtract F_s from $\sum_{i=1}^{n} f(x)$, it will give us the exact global support of an itemset.

Table 1. Example

Global Frequent k- Itemset		Local Support at Different sites			Candidate $k+1$ - Itemset		$f(x)$		Global $k+1$- Itemset		
Name	F_S				Name	O_S	C_S	$O_S + C_S$	Name	$f(x)$	G_S
A	150	S_1	A	70	AB	10	120	130	AB	345	60
			B	50	AC	25	105	120			
			C	35	BC	30	85	125			
B	125	S_2	A	40	AB	25	85	110	AC	360	90
			B	45	AC	40	75	115			
			C	35	BC	20	80	100			
C	130	S_3	A	40	AB	25	70	95	BC	355	80
			B	30	AC	25	100	125			
			C	60	BC	10	90	100			

To illustrate the abovementioned procedure, let us consider the example shown in Table 1. There are three sites and after the first iteration it discovers {*A*, *B*, *C*} as the global frequent *1*-itemsets. Then each site generates candidate itemsets and their support counts. In order to obfuscate the support count of each itemset, it uses formula 1 (for each candidate *k*-itemset, C_s is the addition of local support counts of all *k-1*-itemsets, e.g. local support of *A* and *B* for itemset *AB*) and sends those obfuscated support count to the *receiver* sites. Since the *receiver* site receives only the value of $f(x)$, this does not tell the exact support count of an itemset. It is only able to discover the global support of each candidate itemset by using the formula 3. As the result, we will be able to eliminate the de-obfuscation phase of our proposed method and reduce the message exchange size.

5 Performance Evaluation

We have done an extensive performance study on our proposed message reduction techniques to conform our analysis of its effectiveness. The client-server based distributed environment was established in order to evaluate this message optimization technique. Initial evaluation was carried out on four different sites. Each site has a receiving and a sending unit and listens to a specific port in order to send and receive the support counts.

We have also implemented a sequential association-mining algorithm using Java 1.4 and replicate the algorithm to four different sites in order to generates candidate support counts of each site. Each site generates a random number using pseudo random number generator to obfuscate support counts.

We have chosen four real datasets for this evaluation study. Table 2 shows the characteristics of the datasets that are used in this evaluation. It shows the number of items, average size of the each transaction, and the number of transactions of each dataset. Cover Type and Connect-4 dataset are taken from UC Irvine Machine

Learning Dataset Repository [12], whereas BMS-Webview-1 and BMS-Webview-2 are real world datasets containing several months worth of click stream data from an e-commerce web site and are made publicly available by Blue Martini Software [13].

Table 2. Dataset Characteristics

Name	Transaction Size avg.	Number of Distinct Items	Number of Records
Cover Type	55	120	581012
Connect-4	43	130	67557
BMS-WEB-View-1	2	497	59602
BMS-WEB-View-2	5	3340	75512

We divide all datasets into four different partitions and assign them into four different sites. In order to reduce identical transactions among different sites each of these partitioned datasets was generated in such a way that each partition has 75% of the transactions of the original dataset.

Before we compare our proposed privacy preserving (PP) distributed data mining technique with other well-known algorithms such as Count Distribution (CD) and Distributed Mining Association rule (DMA), it is important to know in details how those algorithms broadcast messages. For example, CD algorithm generates support counts at each local site and broadcasts them to all other sites. All sites can then find the global frequent itemsets for that pass. Since each participating site generates support counts from the frequent itemset of previous pass, only support counts of those itemsets will be enough to generate global frequent itemsets of that pass.

DMA algorithm introduces a new optimization technique to reduce message broadcasting costs. For every local large itemset, it assigns a pooling site where it sends a request to check whether that itemset is globally large or not. When a pooling site receives a request, it sends a pooling request to all other remote sites except the originator site. Upon receiving the pooling request, it computes the heavy itemsets and sends them to all other sites. However, when a site sends a pooling request, it needs not only needs to send the support count of that itemset but also the name of that itemset[1]. As a result, it increases the broadcasting cost in some cases. It is worth to mention that both CD and DMA are non-secure parallel/distributed association mining algorithms, meaning that none of them maintain privacy of individual sites' inputs. To ensure privacy, two algorithms, which are extension of DMA, were proposed [7]. However these algorithms took *n* rounds in compare with DMA took a single round in order to generate global support counts and as a result, they exchange more messages than DMA.

Figure 2 depicts the total size of messages (i.e. number of bytes) transmitted by PP, DMA and CD in order to generate global frequent itemsets from different real world datasets. Depending on the characteristics of each dataset we varies support threshold. In order to generate a reasonable number of global frequent itemsets we use very high support threshold for dense dataset and low support threshold for sparse dataset.

The message size was measured by assuming 4 bytes for each support counts and 4 bytes for each candidate itemset name. We keep 4 bytes for each candidate itemset

[1] Please see Example 3 on [4]

name because not only there are large numbers of candidate itemset but also each dataset generates long candidate itemset. Similar kind of assumption is also made other distributed association rule mining algorithms [14].

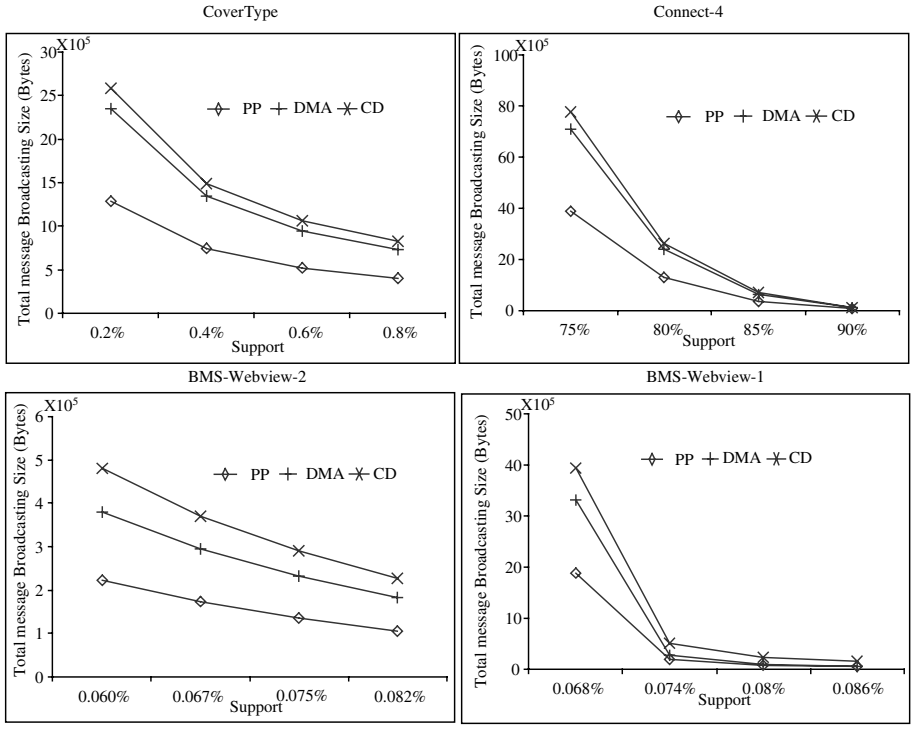

Fig. 2. Comparative total communication costs

From the above comparison between PP, DMA and CD plotted in Figure 2, it is clear PP algorithm exchanges less messages. Indeed, in all cases PP reduces communication cost by 60-80% compared to CD. Using CD algorithm, each site exchanges messages with all other sites after every pass, and hence the message exchange size increases when we increase the number of sites. Nevertheless, PP transmits 25-50% less messages compared with DMA. DMA algorithm exchanges more messages because the polling site sends and receives support counts from remote sites, and further it sends support count of the global frequent itemset to all sites, and consequently increases the message size. In contrast, using our proposed method each site sends its support counts to a single site and receives the global frequent support count from a single site, and hence reduces the number of broadcasting operation. Furthermore, the local pruning technique of DMA effectively works only when different sites have vertically fragmented sparse datasets, but this will not be able to prune significant number of candidate itemsets when each site uses horizontal fragmented datasets.

6 Conclusion

Privacy preserving mining of association rules becomes one of the active research topics in recent years. Maintaining privacy in distributed association rule mining is more difficult than in the centralized approach. In this paper, we propose a methodology, which can efficiently generates distributed association rules without revealing support counts of each site. The proposed method generates rules based on the exact global support of an itemset. The resultant rule model achieved by this method is the same as if one generates it using some of the well-known distributed/parallel algorithms. The proposed method does not distort the original dataset and for this reasons it does not require any further computational costs. Nevertheless the performance evaluation shows that the overall communication cost incurred by our proposed method is less than those of CD and DMA algorithms.

References

1. M. J. Zaki, "Parallel and Distributed Association Mining: A Survey", *IEEE Concurrency*, October-December 1999.
2. S. J. Rizvi and J. R. Haritsa "Maintaining Data Privacy in Association Rule Mining", *In Proc. of* 20*th International Conference on Very Large Databases*, Hong Kong 2002.
3. A. Evfimievski, R. Srikant, R. Agrawal and J. Gehrke, "Privacy Preserving Mining Association Rules", *In Proc. of the SIGKDDD* 2002.
4. J. Vaidya and C. Clifton "Privacy Preserving Association Rule Mining in Vertically Partitioned Data", *In Proc. of ACM SIGKDD*, July 2002.
5. M. Kantercioglu and C. Clifton "Privacy Preserving Distributed Mining of Association Rules on Horizontal Partitioned Data", *In Proc. of DMKD*, 2002.
6. M. Atallah, E. Bertino, A. Elmagarmid, M. Ibrahim and V. Verykios, "Disclosure Limitation of Sensitive Rules", *In Proc. of the KDEX,* November 1999.
7. E. Dasseni, V. S. Verykios, A. K. Elmagarmid and E. Bertino, "Hiding Association Rules by Using Confidence and Support", *Proc. of the Intl. Info. Hiding Workshop(IHW),* April 2001.
8. R. Agrawal and J. C. Shafer, "Parallel Mining of Association Rules", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 8, No. 6, pages. 962-969, December 1996.
9. D. W. Cheung, V. T. Ng, A. W. Fu, and Y. Fu, "Efficient Mining of Association Rules in Distributed Databases", *IEEE TKDE*, 8(6) pages. 911-922, 1996.
10. R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules in Large Database", *Proc. of the 2oth Intl. Conf. on Very Large Databases*, pp. 407-419, 1994.
11. O. Goldriech "Secure Multipart Computation" Working Draft Version 1.3, June 2001.
12. C. L. Blake and C. J. Merz. *UCI Repository of Machine Learning Databases*, University of California, Irvine, www.ics.uci.edu/~mlearn/MLRepository.html, 1998.
13. R. Kohavi,C. Broadley, B Frasca, L. Mason and Z. Zheng "KDD-Cup 2000 organizers report: Peeling the onion", *SIGKDD Explorations*, 2(2):86-98, 2000,. http://www.ecn.purdue.edu/KDDCUP/
14. A. Schuster and R. Wolff, "Communication-Efficient Distributed Mining of Association Rules", *In Proc. ACM SIGMOD*, Santa Barbara.