Mining Multiple Large Data Sources

Animesh Adhikari[1], Pralhad Ramachandrarao[2], Bhanu Prasad[3], and Jhimli Adhikari[4]
[1]Department of Computer Science, S. P. Chowgule College, India
[2]Department of Computer Science and Technology, Goa University, India
[3]Department of Computer and Information Sciences, Florida A&M University, USA
[4]Department of Computer Science, Narayan Zantye College, India

Abstract: *Effective data analysis using multiple databases requires highly accurate patterns. Local pattern analysis might extract low quality patterns from multiple large databases. Thus, it is necessary to improve mining multiple databases using local pattern analysis. We present existing specialized as well as generalized techniques for mining multiple large databases. We formalize the idea of multi-database mining using local pattern analysis and propose a new generalized technique for mining multiple large databases. It improves the quality of synthesized global patterns significantly. We conduct experiments on both real and synthetic databases to judge the effectiveness of the proposed technique.*

Keywords: *Multi-database mining, pipelined feedback technique, synthesis of patterns.*

Received December 21, 2008; accepted February 8, 2009

1. Introduction

Due to a liberal economic policy adopted by many countries across the globe, the number of branches of a multi-national company as well as the number of multi-national companies is increasing over time. Moreover, the economies of many countries are growing at a faster rate. As a result the number of multi-branch companies within a country is also increasing. Many of these companies collect a huge amount of data through different branches. Thus, many of them possess multiple databases. Most of the previous pieces of data mining work are based on a single database. Thus, it is necessary to study data mining on multiple databases.

Many large companies operate from a number of branches located at different geographical regions. Each branch collects data continuously and local data get stored locally. Thus, the collection of all branch databases might be large. Many decisions of a multi-branch company are based on data stored over the branches. The challenges involve in making good quality of decisions based on large volume of data that are distributed over the branches. It creates not only risks but also offers opportunities. One of the risks is a significant amount investment on hardware and software to deal with multiple large databases. The goal of this paper is to improve mining multiple large databases.

Based on the number of data sources, patterns in multiple databases could be classified into three categories. They are local patterns, global patterns and patterns that are neither local nor global. A pattern based on a single database is called a local pattern. Local patterns are useful for local data analysis and decision making problems [1, 11]. On the other hand, global patterns are based on all the databases under consideration. They are useful for global data analyses [2, 12] and global decision making problems. In this paper, we propose a new multi-database mining technique, called Pipelined Feedback Technique (PFT), for mining / synthesizing global patterns in multiple databases.

The rest of the paper is organized as follows. We formalize the idea of multi-database mining using local pattern analysis in section 2. In section 3, we discuss existing generalized multi-database mining techniques. Also, we discuss existing specialized multi-database mining techniques in section 4. We propose a new multi-database mining technique for mining multiple databases in section 5. We define error of an experiment in section 6. In section 7, we provide experimental results using both synthetic and real databases.

2. Multi-Database Mining Using Local Pattern Analysis

Consider a large company that deals with multiple large databases. For mining multiple databases, there are three situations viz:

a. Each of the local databases is small, so that a Single Database Mining Technique (SDMT) could mine the union of all databases.
b. At least one of the local databases is large, so that a SDMT could mine every local database, but fail to mine the union of all local databases.
c. At least one of the local databases is very large, so that a SDMT fails to mine every local database.

We face challenges to handle the cases (b) and (c). The challenges posed to us are due to large size of some local databases. The first question comes to our mind whether a traditional data mining technique [4, 6] could provide a good solution in dealing with multiple large databases. To apply a traditional data mining technique one needs to amass all the branch databases together. A traditional data mining technique might not provide a good solution due to the following reasons.

- It might not be suitable as one might have to invest heavily on hardware and software to deal with a large volume of data.
- A single computer might take unreasonable amount of time to mine a huge amount of data.
- It is difficult to identify local patterns if a traditional data mining technique is applied on the collection of local databases.

Thus, a traditional data mining technique might not be suitable in this situation. So, it is a different problem. Hence, it is required to be dealt with in a different way. Zhang *et al*. [14] designed a Multi-Database Mining Technique (MDMT) using local pattern analysis. Multi-database mining using local pattern analysis could be classified into two categories viz., the techniques that analyze local patterns and the techniques that analyze approximate local patterns. A multi-database mining technique using local pattern analysis could be viewed as a two-step process $\tau + \xi$, explained as follows:

- Mine each local database using a SDMT by applying a technique τ (Step 1).
- Synthesize patterns using an algorithm ξ (Step 2).

We use notation MDMT: $\tau + \xi$ to represent a multi-database mining technique using a technique of mining τ and a synthesizing algorithm ξ.
We can apply sampling techniques [10] for taming large volume of data. If an itemset is frequent in a large dataset then it is likely to be frequent in the sampled dataset. Thus, we can mine patterns approximately in a large dataset by analyzing patterns in a representative sampled dataset. There are two categories of multi-database mining techniques viz., specialized and generalized multi-database mining techniques.

3. Generalized Multi-database Mining Techniques

In this section, we discuss existing generalized multi-database mining techniques. These techniques could be used in variety of multi-database mining applications.

3.1. Local Pattern Analysis

Under this model of mining multiple databases, each branch requires to mine its database using a traditional data mining technique. Afterwards, each branch is required to forward the pattern base to the central office. Then the central office could process the pattern bases collected from different branches for synthesizing the global patterns or making some global decisions.

Adhikari and Rao [2] have proposed an extended model of local pattern analysis. The proposed extended model has a set of interfaces and a set of layers. Each interface is a set of operations that produces dataset(s) (or knowledge) based on the dataset(s) at the next lower layer. The functions of the interfaces are described below.

Interface 2/1 applies different operations on data at the lowest layer. By applying these operations, we get a processed database from a local (original) database. These operations are performed on each branch database. Interface 3/2 applies a filtering algorithm on each processed database to separate relevant data from outlier data. In particular, if we are interested in studying the durable items then the transactions containing only non-durable items could be treated as outlier transactions. Interface 4/3 mines local patterns in each local data warehouse. There are two types of local patterns: local patterns and suggested local patterns. A suggested local pattern is close but fails to satisfy the requisite interestingness criteria. The reasons for considering suggested patterns are given as follows. Firstly, one could synthesize patterns more accurately. Secondly, due to the stochastic nature of transactions, the number of suggested patterns could be significant in some databases. Thirdly, there is a tendency that a suggested pattern of one database to become a local pattern in another database. Thus, the correctness of synthesizing global patterns would increase as the number of local patterns increases. Let there are n databases of a multi-branch company. Also, let LPB_i and SPB_i be the local pattern base and suggested local pattern base for the ith branch, respectively, for $i = 1, 2, \ldots, n$. Interface 5/4 synthesizes global patterns or analyses local patterns to meet real life challenges.

Various data preparation techniques [8] like data cleaning, data transformation, data integration, and data reduction are applied to data in the local databases. We get the processed database PD_i corresponding to original database D_i, for $i = 1, 2, \ldots, n$. Then we retain all the data that are relevant to the data mining applications. Using a relevance analysis, one can detect outlier data [7] from processed database. A relevance analysis is dependent on the context and varies from one application to another application. Let OD_i be the outlier database corresponding to the ith branch, for $i = 1, 2, \ldots, n$. After removing outlier data from the processed database we get desired data warehouse, and the data in a data warehouse become ready for data mining task. Let W_i be the data warehouse corresponding to the ith branch, for $i = 1, 2, \ldots, n$. Local patterns for the ith branch are extracted from W_i, for $i = 1, 2, \ldots, n$. Finally, the local patterns are forwarded to the central office for synthesizing global patterns, or analysis of local patterns. Figure 1 illustrates a model of synthesizing global patterns from local patterns in different databases.

In particular, if we are interested in synthesizing global frequent itemsets then an itemset may not get extracted from all the databases. It is required to estimate or ignore the support of an itemset in a database that fails to report it. Thus, a global frequent itemset synthesized from local frequent itemsets is approximate in nature. If any one of the local databases is too large to apply a traditional data mining technique then this model would fail. In this situation, we can apply an appropriate sampling technique to reduce the size of a large local database. Otherwise, the database can be partitioned into sub-databases. As a result, the error of synthesizing a pattern would increase.

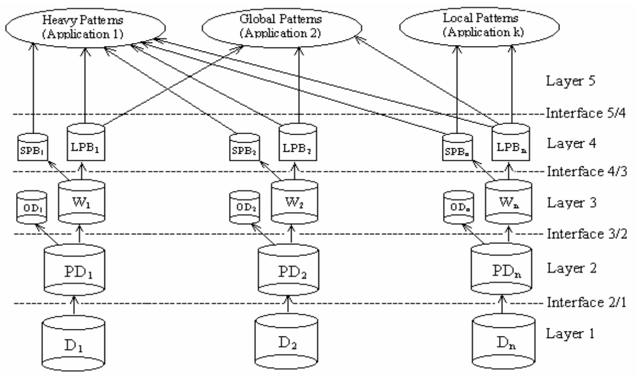


Figure 1. A model of synthesizing global patterns from local patterns in different databases.

Though the above model introduces many layers and interfaces for synthesizing global patterns, but in a real life application, many of these layers and interfaces might be absent. The patterns returned by local pattern analysis are approximate. They might differ considerably from exact global patterns.

3.2. Partition Algorithm

For the purpose of mining multiple databases, one can apply Partition Algorithm (PA) proposed by Savasere *et al.*, [9]. The algorithm is designed for mining a very large database by partitioning. The algorithm works as follows. It scans a database twice. The database is divided into disjoint partitions, where each partition is small enough to fit in memory. In the first scan, the algorithm reads each partition and computes locally frequent itemsets in each partition using apriori algorithm [4]. In the second scan, the algorithm counts the supports of all locally frequent itemsets toward the complete database. In this case, each local database can be considered as a partition. Though partition algorithm mines frequent itemsets in a database exactly, it is an expensive solution to mining multiple large databases, since each database is required to scan twice.

3.3. IdentifyExPattern Algorithm

Zhang *et al.*, [13] have proposed algorithm IdentifyExPattern (IEP) for identifying global exceptional patterns in multi-databases. Every local database is mined separately at Random Order (RO) using a SDMT for synthesizing global exceptional patterns. For identifying global exceptional patterns in multiple databases, the following pattern synthesizing algorithm has been proposed. A pattern in a local database is assumed as zero, if it does not get reported. Let $supp_a(p, DB)$ and $supp_s(p, DB)$ be the actual (i.e, apriori) support and synthesized support of pattern p in database DB, respectively. Let D be the union of all local databases. Then support of pattern p has been synthesized in D based on the following formula:

$$supp_s(p,D) = \frac{1}{num(p)} \sum_{i=1}^{num(p)} \left[supp_a(p,D_i) - \alpha\right] \Big/ \left[1-\alpha\right] \quad (1)$$

where $num(p)$ is the number of databases that report p at a given minimum support level (α). The size (*i.e.*, the number of transactions) of a local database and support of an itemset in a local database are seem to be important parameters for determining the presence of an itemset in a database, since the number of transactions containing the itemset X in a database D_1 is equal to $supp(X, D_1) \times size(D_1)$. The major concern is that the algorithm IEP does not consider the size of a local database to synthesize the global support of a pattern.

3.4. Rule Synthesizing Algorithm

Wu and Zhang [12] have proposed Rule Synthesizing (RS) algorithm for synthesizing high-frequent association rules in multiple databases. Using this technique, every local database is mined separately at Random Order (RO) using a SDMT for synthesizing high-frequent association rules. A pattern in a local database is assumed as zero, if it does not get reported. Based on the association rules in different databases, the authors have estimated weights of different databases. Let w_i be the weight of i-th database, for i = 1, 2, …, n. Without any loss of generality, let the association rule r be extracted from first m databases, for $1 \leq m \leq n$. $supp_a(r, D_i)$ has been assumed as 0, for $i = m + 1, m + 2, …, n$. Then the support of r in D has been synthesized as follows:

$$supp_s(r, D) = w_1 \times supp_a(r, D_1) + \ldots + w_m \times supp_a(r, D_m) \quad (2)$$

Algorithm RS is an indirect approach for synthesizing association rules in multiple databases. Thus, the time complexity of the algorithm is reasonably high. The algorithm executes in $O(n^4 \times maxNosRules \times totalRules^2)$ time, where n, *maxNosRules*, and *totalRules* are the number of data sources, the maximum among the numbers of association rules extracted from different databases, and the total number of association rules in different databases, respectively.

4. Specialized Multi-Database Mining Techniques

For finding solution to a specific application, it might be possible to devise a better multi-database mining technique. In this section, we present two specific multi-database mining techniques.

4.1. Mining Multiple Real Databases

Adhikari and Rao [2] have proposed Association-Rule-Synthesis (ARS) algorithm for synthesizing association rules in multiple real databases. The algorithm uses the model in Figure 1. But, it uses a specific rule synthesizing process explained as follows. For real databases, the trend of the customers' behaviour exhibited in one database is usually present in other databases. In particular, a frequent itemset in one database is usually present in some transactions of other databases even if it does not get extracted. The estimation procedure captures such trend and estimates the support of a missing association rule. Without any loss of generality, let an itemset X be extracted from first m databases, for $1 \leq m \leq n$. Then trend of X in first m databases could be expressed as follows.

$$trend^{1,m}(X \mid \alpha) = \frac{1}{\sum_{i=1}^{m} |D_i|} \times \sum_{i=1}^{m} \left(supp_a(X, D_i) \times |D_i| \right) \quad (3)$$

We can use trend of X in first m databases for synthesizing support of X in D. We estimate support of X in database D_j by $\alpha \times trend^{1,n}(X \mid \alpha)$, for $j = k + 1, k + 2, \ldots, n$. Then the synthesized support of X could be computed as follows.

$$supp_s(X,D) = \frac{trend^{1,m}(X|\alpha)}{\sum_{i=1}^{n} |D_i|} \times \left[(1-\alpha) \times \sum_{i=1}^{m} |D_i| + \alpha \times \sum_{i=1}^{n} |D_i| \right] \quad (4)$$

Association-Rule-Synthesis algorithm might return approximate global patterns.

4.2. Mining Multiple Databases for the Purpose of Studying a Set of Items

Adhikari and Rao [3] have proposed a technique for mining patterns of a set of specific items in multiple databases. Many important decisions are based on a set of specific items called the select items. A large section of a local database is irrelevant in providing solution to this problem, since it involves studying select items in multiple databases. Thus, we divide database D_i into FD_i and RD_i, where FD_i and RD_i are called the Forwarded Ddatabase and Remaining Database corresponding to the ith branch respectively, for $i = 1, 2, \ldots, n$. We are interested in the forwarded databases, since every transaction in a forwarded database contains at least one select item. The database FD_i is forwarded to the central office for mining global patterns of select items under consideration, for $i = 1, 2, \ldots, n$. All the local forwarded databases are amassed into a single database FD for the purpose of mining task. The model of mining global patterns of select items could be explained using the following steps:

1. Each branch office constructs the forwarded database and sends it to the central office.
2. Also, each branch extracts patterns from its local database.
3. The central office clubs these forwarded databases into a single database *FD*.
4. A traditional data mining technique could be applied to extract patterns from *FD*.
5. The global patterns of select items could be extracted effectively from local patterns and the patterns extracted from *FD*.

At interface 3/2, we apply an algorithm to partition a local database into two parts viz., forwarded database and remaining database. In the following paragraph, we discuss how to construct FD_i, for $i = 1, 2, \ldots, n$. Initially, FD_i is kept empty. Let T_{ij} be the jth transaction of D_i, for $j = 1, 2, \ldots, |D_i|$. For D_i, a for-loop on j would run for $|D_i|$ times. At the jth iteration, the transaction T_{ij} is tested. If T_{ij} contains at least one of the select items then FD_i is updated by $FD_i \cup \{T_{ij}\}$. At the end of the for-loop on j, FD_i gets constructed.

A traditional data mining algorithm could be applied at the interface 5/4 to extract patterns in *FD*. Let *PB* be the pattern base returned by a traditional data mining algorithm. Since, the database *FD* is not large, one can lower further the values of user-defined inputs, like minimum support, minimum confidence, so that *PB* could contain more patterns of select items. Therefore, we get a better analysis of select items. If we wish to study the association between a select item and other frequent items then the exact support values of other items might not be available in *PB*. Then the central office sends a request to each branch office to forward the details (like support values) of some items that would be required to study the select items. Thus, each branch then applies a traditional mining algorithm (at interface 3/2) on its local database and forwards the details of local patterns requested by the central office. Let LPB_i be the details of ith local pattern base requested by the central office, for $i = 1, 2, \ldots, n$. A global mining application of select items is required to access local patterns and patterns in *PB*. Thus, a global mining application (interface 6/5) can be developed based on the patterns in *PB* and LPB_i, for $i = 1, 2, \ldots, n$. The model of mining global patterns of select items is efficient due to the following reasons:

- We can extract more patterns of select items by lowering further the input parameters like minimum support, minimum confidence, based on the level of data analysis of select items, since *FD* is reasonably small.
- We get the exact global patterns of select items as there is no need of estimating them. Thus, the quality of global patterns is high.

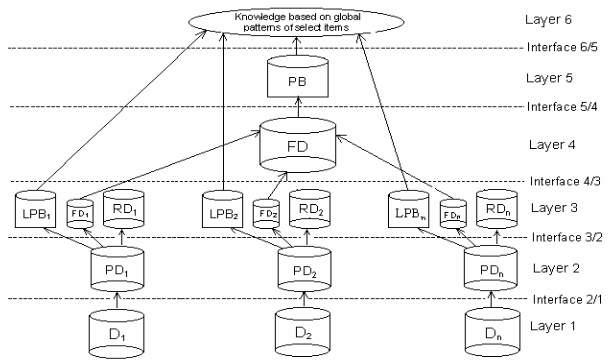


Figure 2. A model of mining global patterns of select items from multiple databases.

5. Mining Multiple Databases Using Pipelined Feedback Technique

Before applying pipelined feedback technique, one needs to prepare data warehouses at different branches of a multi-branch organization. Let W_i be the data warehouse corresponding to the *i*-th branch, for $i = 1, 2, \ldots, n$. Then the local patterns for the i^{th} branch are extracted from W_i, for $i = 1, 2, \ldots, n$. We mine each data warehouse using a SDMT. In Figure 3, we propose a new technique of mining multiple databases.

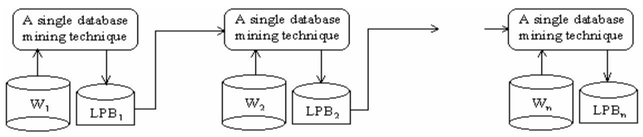

A single database mining technique
W1
LPB1
A single database mining technique
W2
LPB2
A single database mining technique
Wn
LPBn

Figure 3. Pipelined feedback technique of mining multiple databases.

In PFT, W_1 is mined using a SDMT and local pattern base LPB_1 is extracted. While mining W_2, all the patterns in LPB_1 are extracted irrespective of their values of interestingness measures like, minimum support and minimum confidence. Apart from these patterns, some new patterns that satisfy user-defined threshold values of interestingness measures are also extracted. In general, while mining W_i, all the patterns in W_{i-1} are mined irrespective of their values of interestingness measures, and some new patterns that satisfy user-defined threshold values of interestingness measures, for $i = 2, 3, \ldots, n$. Due to this nature of mining each data warehouse, PFT is called a feedback technique. Thus, $|LPB_{i-1}| \le |LPB_i|$, for $i = 2, 3, \ldots, n$. There are $n!$ arrangements of pipelining for n databases. All the arrangements of data warehouses might not produce the same mining result. If the number of local patterns increases, we get more accurate global patterns and a better analysis of local patterns. An arrangement of data warehouses would produce near optimal result if $|LPB_n|$ is a maximal. Let $size(W_i)$ be the size of W_i (in bytes), for $i = 1, 2, \ldots, n$. We shall follow the following rule of thumb regarding the arrangements of data warehouses for the purpose of mining. The number of patterns in W_i is greater than or equal to the number of patterns in W_{i-1}, if $size(W_i) \ge size(W_{i-1})$, for $i = 2, 3, \ldots, n$. For the purpose of increasing number of local patterns, W_i precedes W_{i-1} in the pipelined arrangement of mining data warehouses if $size(W_i) \ge size(W_{i-1})$, for $i = 2, 3, \ldots, n$. Finally, we analyze the patterns in LPB_1, LPB_2, …, and LPB_n for synthesizing global patterns, or analyzing local patterns.

Let W be the collection of all branch data warehouses. For synthesizing global patterns in W we discuss here a simple pattern synthesizing (SPS) algorithm. Without any loss of generality, let the itemset X be extracted from first m databases, for $1 \le m \le n$. Then synthesized support of X in W could be obtained as follows:

$$supp_s(X, W) = \frac{1}{\sum_{i=1}^{n} |W_i|} \times \sum_{i=1}^{m} [\, supp_a(X, W_i) \times |W_i| \,] \qquad (5)$$

In the following, we propose a new algorithm for mining multiple databases. The algorithm is based on the pipelined feedback technique presented in Figure 4.

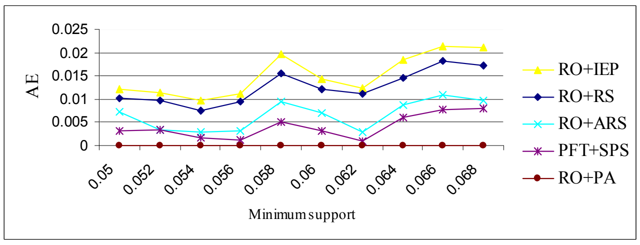


Figure 4. AE vs. α for experiments using dataset *T*.

Algorithm 1: *mine multiple data warehouses using pipelined feedback technique.*

```
procedure PipelinedFeedbackTechnique ($W_1$, $W_2$, ..., $W_n$)
Input: $W_1$, $W_2$, ..., $W_n$
Output: local pattern bases
1  for i = 1 to n do
2     if $W_i$ does not fit in memory then
3        partition $W_i$ into $W_{i1}$, $W_{i2}$, ..., and $Wip_i$ for an
           integer $p_i$;
4     else $W_{i1} = W_i$;
5     end if
6  end for
7  sort data warehouses on size in non-increasing
   order and the data warehouses are renamed as
     $DW_1$, $DW_2$, ..., $DW_N$, where $N = \sum_{i=1}^{n} p_i$;
8  let $LPB_0 = \phi$;
9  for i = 1 to N do
10    mine $DW_i$ using a SDMT with input $LPB_{i-1}$;
```

```
11  end for
12  return $LPB_1, LPB_2, \ldots, LPB_N$;
```

In above algorithm, the usage of LPB_{i-1} during mining DW_i has been explained above. Once a pattern is extracted from a data warehouse, then it also gets extracted from the remaining data warehouses. Thus, the algorithm PipelinedFeedbackTechnique improves synthesized patterns as well as an analysis of local patterns significantly.

6. Error of an Experiment

To evaluate MDMT: PFT+SPS, one needs to measure the amount of error of the experiments. An experiment mines frequent itemsets in multiple databases using PFT, and then synthesizes global patterns using SPS algorithm. One needs to find how the global synthesized support differs from the exact (apriori) support of an itemset.

In PFT, we have $LPB_{i-1} \subseteq LPB_i$, for $i = 2, 3, \ldots, n$. Then, patterns in LPB_i - LPB_{i-1} are generated from databases $D_i, D_{i+1}, \ldots, D_n$. We assume $supp_a(X, D_j) = 0$, for each $X \in LPB_i - LPB_{i-1}$, for $i = 2, 3, \ldots$. Thus, the error of mining X could be defined as follows.

$$\mathrm{E}(X \mid \mathrm{PFT}, \mathrm{SPS}) = \left| supp_a(X, D) - \frac{1}{\sum_{j=1}^{n} |D_j|} \times \sum_{j=\mathrm{i}}^{n} \left[supp_a(X, D_j) \times |D_j| \right] \right|,$$
$$\text{for } \mathrm{X} \in \mathrm{LPB_i} - \mathrm{LPB_{i-1}} \text{ and } \mathrm{i} = 2, 3, \ldots, \mathrm{n}.$$

Also, $E(X|PFT,SPS) = 0$, *for* $X \in LPB_1$. (6)

There are several ways one could define error of an experiment. We have defined following two types of error of an experiment.

1. Average Error (AE)

$$AE(D,\alpha) = \frac{1}{|LPB_1 + \sum_{i=2}^{n}(LPB_i - LPB_{i-1})|} \times \sum_{X \in [LPB_1 + \sum_{i=2}^{n}(LPB_i - LPB_{i-1})]} E(X|PFT,SPS) \quad (7)$$

2. Maximum Error (ME)

$$ME(D,\alpha) = maximum\{ E(X|PFT,SPS), \; for\ X \in \{LPB_1 + \sum_{i=2}^{n}(LPB_i - LBP_{i-1})\}\} \quad (8)$$

$supp_a(X_i, D)$ is obtained by mining D using a traditional data mining technique, for $i = 1, 2, \ldots, m$. $supp_s(X_i, D)$ is obtained by SPS, for $i = 1, 2, \ldots, m$.

7. Experiments

We have carried out several experiments to study the effectiveness of the proposed technique. All the experiments have been implemented on a 2.8 GHz Pentium D dual core processor with 512 MB of memory using visual C++ (version 6.0) software. We present experimental results using synthetic database *T10I4D100K* (*T*) [5] and two real databases Retail (*R*) [5] and BMS-Web-Wiew-1 (*B*) [5]. The databases *random500* (*R1*) and *random1000* (*R2*) are generated synthetically for the purpose of conducting experiments. We present some characteristics of these databases in Table 1.

Table 1. Database characteristics.

D	*N T*	*ALT*	*AFI*	*NI*
T	1,00,000	11.10228	1276.12413	870
R	88,162	11.30575	99.67380	10000
B	1,49,639	2.00000	155.71176	1922
R1	10,000	6.47000	109.40000	500
R2	10,000	12.48560	111.85785	1000

Let *NT*, *ALT*, *AFI*, and *NI* denote the number of transactions, average length of a transaction, average frequency of an item, and number of items in database, respectively. The error of synthesizing itemset in multiple databases is relative to the following parameters: the number of transactions, the number of items, and the length of transactions in the given databases. If the number of transactions in a database increases the error of synthesizing itemsets increases, provided other two parameters remain constant. If the length of transactions of a database increase the error of synthesizing itemsets is likely to increase, provided other two parameters remain constant. Lastly, if the number of items increases the error of synthesizing itemsets is likely to decrease, provided other two parameters remain constant.

Each of the above databases is divided into 10 databases for the purpose of carrying out experiments. The databases obtained from *T*, *R*, *B*, *R1*, *R2* are named as T_i, R_i, B_i, $R1_i$, $R2_i$ respectively, for $i = 0, 1, \ldots, 9$. The databases T_i, R_i, B_i, $R1_i$, $R2_i$ are called input DataBases (*DB*s), for $i = 0, 1, \ldots, 9$. Some characteristics of these input databases are presented in the Table 2. In Table 3, we present some outputs for the purpose of showing that the proposed technique improves significantly the mining results. Also, we have performed experiments using other MDMTs on these databases for the purpose of comparing with MDMT: PFT+SPS. Each of the Figures 4, 5, 6, 7 and 8 shows average error against different αs. From these figures, one could conclude that AE normally increases as α increases. The number of databases reporting a pattern decreases as α increases. Thus, the AE of synthesizing patterns normally increases as α increases. Figures 5 to 8 show that MDMT: PFT+SPS produces more accurate mining result among all the techniques that scan each database only once.

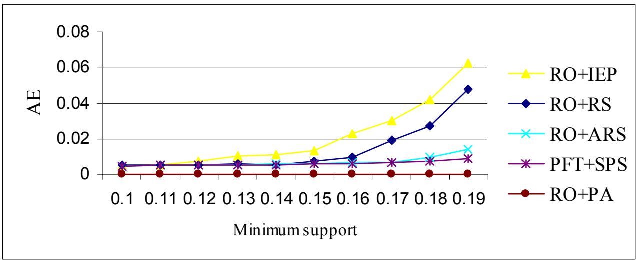


Figure 5. AE vs. α for experiments using dataset *R*.

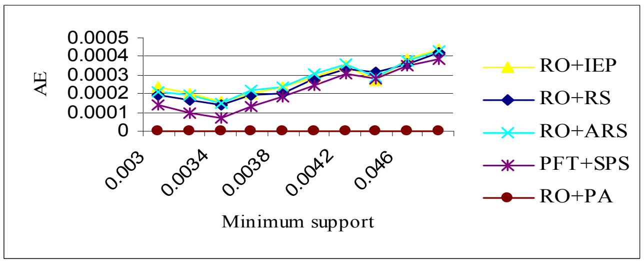


Figure 7. AE vs. α for experiments using dataset *R1*.

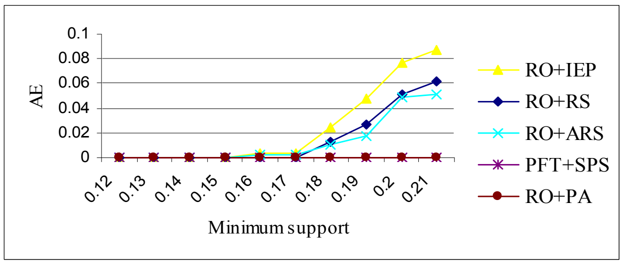


Figure 6. AE vs. α for experiments using dataset *B*.

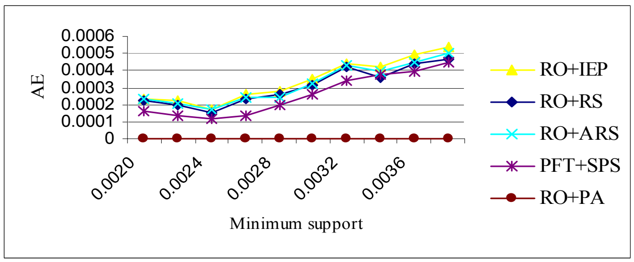


Figure 8. AE vs. α for experiments using dataset *R2*.

Table 2. Input database characteristics.

DB	NT	ALT	AFI	NI	DB	NT	ALT	AFI	NI
T_0	10000	11.0550	127.6559	866	T_5	10000	11.1391	128.6270	866
T_1	10000	11.1333	128.4118	867	T_6	10000	11.1078	128.5625	864
T_2	10000	11.0670	127.6471	867	T_7	10000	11.0984	128.4538	864
T_3	10000	11.1226	128.4365	866	T_8	10000	11.0815	128.5557	862
T_4	10000	11.1367	128.7480	865	T_9	10000	11.0814	128.1087	865
R_0	9000	11.2439	12.07001	8384	R_5	9000	10.8558	16.7098	5847
R_1	9000	11.2092	12.26541	8225	R_6	9000	11.2001	17.4155	5788
R_2	9000	11.3367	14.59657	6990	R_7	9000	11.1551	17.3455	5788
R_3	9000	11.4898	16.66259	6206	R_8	9000	11.9971	18.6903	5777
R_4	9000	10.9568	16.03953	6148	R_9	7162	11.6920	15.3479	5456
B_0	14000	2.0000	14.94130	1874	B_5	14000	2.0000	280.0000	100
B_1	14000	2.0000	280.0000	100	B_6	14000	2.0000	280.0000	100
B_2	14000	2.0000	280.0000	100	B_7	14000	2.0000	280.0000	100
B_3	14000	2.0000	280.0000	100	B_8	14000	2.0000	280.0000	100
B_4	14000	2.0000	280.0000	100	B_9	23639	2.0000	472.7800	100
$R1_0$	1000	6.3670	10.7340	500	$R1_5$	1000	6.3380	10.6760	500
$R1_1$	1000	6.5020	11.0040	500	$R1_6$	1000	6.6240	11.2480	500
$R1_2$	1000	6.4020	10.8040	500	$R1_7$	1000	6.4150	10.8300	500
$R1_3$	1000	6.5230	11.0460	500	$R1_8$	1000	6.5790	11.1580	500
$R1_4$	1000	6.2980	10.5960	500	$R1_9$	1000	6.6520	11.3040	500
$R2_0$	1000	6.4210	5.4347	996	$R2_5$	1000	6.4440	5.4553	997
$R2_1$	1000	6.4140	5.4351	995	$R2_6$	1000	6.4770	5.4949	996
$R2_2$	1000	6.5560	5.5798	995	$R2_7$	1000	6.4770	5.4884	997
$R2_3$	1000	6.5290	5.5370	998	$R2_8$	1000	6.5380	5.5572	996
$R2_4$	1000	6.5000	5.5418	991	$R2_9$	1000	6.5000	5.5597	988

Table 3. Error of the experiments at given α.

Database	T10I4D100K		retail		BMS-Web-Wiew-1		random500		random1000	
α	0.05		0.11		0.19		0.005		0.004	
Error type	AE	ME	AE	ME	AE	ME	AE	ME	AE	ME
MDMT: RO+IEP	0.0122	0.0373	0.0052	0.0583	0.0052	0.0583	0.0074	0.0095	0.0057	0.0076
MDMT: RO+RS	0.0102	0.0361	0.0050	0.0576	0.0050	0.0576	0.0056	0.0078	0.0041	0.0050
MDMT: RO+ARS	0.0072	0.0360	0.0049	0.0573	0.0049	0.0573	0.0058	0.0070	0.049	0.0062
MDMT: PFM+SPS	0.0032	0.0359	0.0048	0.0573	0.0048	0.0573	0.0026	0.0045	0.0027	0.0043
MDMT: RO+PA	0	0	0	0	0	0	0	0	0	0

8. Conclusions

In this paper, we discuss existing generalized as well as specialized multi-database mining techniques. For a particular problem, one technique is more suitable than others. Thus, one needs to study the details of each multi-database mining technique, so that one can select the right technique for solving a particular problem. We formalize the idea of multi-database mining using local pattern analysis by considering it as a two-step process. We propose here a new technique for mining multiple large databases. It improves significantly the accuracy of mining multiple databases as compared to the existing techniques that scan each database only once. MDMT: PFT+SPS is effective and promising. The proposed technique could also be used for mining a large database by dividing it into sub-databases.

References

[1] Adhikari A. and Rao P., "Efficient Clustering of Databases Induced by Local Patterns," *Decision Support Systems,* vol. 44, no. 4, pp. 925-943, 2008.

[2] Adhikari A. and Rao P., "Synthesizing Heavy Association Rules from Different Real Data Sources," *Pattern Recognition Letters*, vol. 29, no. 1, pp. 59-71, 2008.

[3] Adhikari A. and Rao P., "Study of Select Items in Multiple Databases by Grouping," *in Proceedings of 3rd Indian International Conference on Artificial Intelligence*, pp. 1699-1718, 2007.

[4] Agrawal R. and Srikant R., "Fast Algorithms for Mining Association Rules," *in Proceedings of Very Large Data Bases*, pp. 487-499, Santiago, Chile, 1994.

[5] Frequent itemset mining dataset repository, http://fimi.cs.helsinki.fi/data/.

[6] Han J., Pei J., and Yin Y., "Mining Frequent Patterns Without Candidate Generation," *in Proceedings of SIGMOD*, pp. 1-12, Dallas, Texas, USA, 2000.

[7] Last M. and Kandel A., "Automated Detection of Outliers in Real-World Data," *in Proceedings of the Second International Conference on Intelligent Technologies*, pp. 292-301, 2001.

[8] Pyle D., *Data Preparation for Data Mining*, Morgan Kufmann, San Francisco, 1999.

[9] Savasere A., Omiecinski E., and Navathe S., "An Efficient Algorithm for Mining Association Rules in Large Databases," *in Proceedings of Very Large Data Bases*, pp. 432-443, 1995.

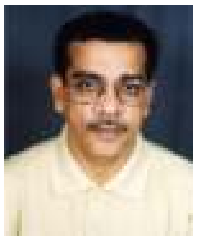
[10] Toivonen H., "Sampling Large Databases for Association Rules," *in Proceedings of the 22th International Conference on Very Large Data Bases*, pp. 134-145, San Francisco, CA, USA, 1996.

[11] Wu X., Zhang C., and Zhang S., "Database Classification for Multi-Database Mining," *Information Systems*, vol. 30, no. 1, pp. 71-88, 2005.

[12] Wu X. and Zhang S., "Synthesizing High-Frequency Rules from Different Data Sources," *IEEE Transactions on Knowledge and Data Engineering*, vol. 15, no. 2, pp. 353-367, 2003.

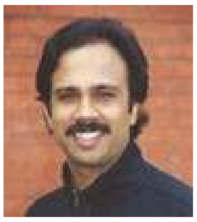
[13] Zhang C., Liu M., Nie W., and Zhang S., "Identifying Global Exceptional Patterns in Multi-Database Mining," *IEEE Computational Intelligence Bulletin*, vol. 3, no. 1, pp. 19-24, 2004.

[14] Zhang S., Wu X., and Zhang C., "Multi-Database Mining," *IEEE Computational Intelligence Bulletin*, vol. 2, no. 1, pp. 5-13, 2003.



Animesh Adhikari is a lecturer in the Department of Computer Science, S P Chowgule College, India. In June, 2008, he has submitted doctoral dissertation in the Department of Computer Science and Technology, Goa University, India. He received Master of technology in computer science from Indian Statistical Institute, India. His areas of interest include data mining and knowledge discovery, decision support systems, database systems, and artificial intelligence.

Pralhad Ramachandrarao is a professor in the Department of Computer Science and Technology, Goa University, India. He received his PhD degree from Indian Institute of Technology, Mumbai, India. His areas of interest include graph theory, data mining and knowledge discovery, and data warehousing.



Bhanu Prasad received Master of technology and PhD degrees in computer science, from Andhra University and Indian Institute of Technology Madras, respectively. Currently he is working as a faculty member in the Department of Computer and Information Sciences at Florida A&M University in Tallahassee, Florida, USA. His research interests include artificial intelligence with a special focus on knowledge representation, reasoning, and product recommending systems.

Jhimli Adhikari is a lecturer in the Department of Computer Science in Narayan Zantye College, Bicholim, Goa. She received Master of computer application from Jadavpur University, Kolkata, India. Currently, she is a PhD student at the Department of Computer Science and Technology, Goa University, India.