

Chapter 2

Synthesizing Different Extreme Association Rules from Multiple Databases

The model of local pattern analysis provides sound solutions to many multi-database mining problems. In this chapter we discuss different types of extreme association rules in multiple databases viz., heavy association rule, high-frequency association rule, low-frequency association rule, and exceptional association rule. Also, we show, how one can apply the model of local pattern analysis systematically and effectively. For this purpose, we present an extended model of local pattern analysis. The extended model has been applied to mine heavy association rules in multiple databases. Also, we justify why the extended model works more effectively. An algorithm for synthesizing heavy association rule in multiple databases is presented. Furthermore, we show that the algorithm identifies whether a heavy association rule is high-frequency rule or exceptional rule. Experimental results are provided for both synthetic and real-world datasets and a detailed error analysis is carried out. Furthermore, we present a comparative analysis by contrasting the proposed algorithm with some of those reported in the literature. This analysis is completed by taking into consideration the criteria of execution time and average error.

2.1 Introduction

In the [Chap. 1](#), the limitations of using “conventional” data mining technique for mining multiple large databases have been discussed. In many decision support applications, an approximate knowledge stemming from multiple large databases might result in significant savings when being used in decision-making. Hence the model of local pattern analysis (Zhang et al. [2003](#)) used for mining multiple large databases can constitute a viable solution. In this chapter, we show how one could apply the model of local pattern analysis in a systematic and efficient manner for mining different types of extreme association rules in multiple databases.

The analysis of relationships existing among variables is a fundamental task positioned at the heart of many data mining problems. Mining association rules has received a lot of attention to the data mining community. For instance, an

association rule expresses how the purchase of a group of items, called an *itemset*, affects the purchase of another group of items. Association rule mining is based on two measures quantifying the quality of the rules, that is support (*supp*) and confidence (*conf*); see Agrawal et al. (1993). An association rule r in database DB can be expressed symbolically as $X \rightarrow Y$, where X and Y are two itemsets in database DB . It expresses an association between the itemsets X and Y , called the antecedent and consequent of r , respectively. The meaning attached to this type of implication could be clarified as follows. If the items in X are purchased by a customer then the items in Y are likely to be purchased by the same customer at the same time. The interestingness of an association rule could be expressed by its support and confidence. Let E be a Boolean expression defined on the items in DB . Support of E in DB is defined as the fraction of transactions in DB such that the Boolean expression E is true for each of these transactions. We denote the support of E in DB as $supp_a(E, DB)$. The support and confidence of association rule r is expressed as follows:

$$\begin{aligned} supp_a(r, DB) &= supp_a(X \cap Y, DB), \text{ and} \\ conf_a(r, DB) &= supp_a(X \cap Y, DB) / supp_a(X, DB) \end{aligned}$$

Later, we shall be dealing with synthesized support and synthesized confidence of an association rule. Thus, it is required to differentiate between actual support/confidence with synthesized support/confidence of an association rule. The subscript a used in the notation of support/confidence for referring the actual support/confidence of an association rule. On the other hand, the subscript s in the notation of support/confidence is used to refer synthesized support/confidence of an association rule. A synthesized support/confidence of an association rule might depend on the technique applied to synthesizing support/confidence. We present here a technique for synthesizing support and confidence of an association rule in multiple databases. We say that an association rule r in database DB is *interesting* if the following relationships hold

$$\begin{aligned} supp_a(r, DB) &\geq \text{minimum support}(\alpha), \text{ and} \\ conf_a(r, DB) &\geq \text{minimum confidence}(\beta) \end{aligned}$$

The values of the parameters α and β are user-defined. The collection of association rules extracted from a database for the given values of α and β is called a *rulebase*.

In a multi-database mining environment, often one needs to handle multiple large databases. As a result, one may come across various types of patterns. Association rule mining (Agrawal et al. 1993) is an important and popular data mining task. It has many applications to different areas of computing (Zhang and Wu 2011). In this chapter, we are interested in mining association rules in multiple databases that are extreme in some sense. These association rules are induced by different data sources, and thus, these rules are specific to multi-database mining environment. An association rule in multiple databases becomes more interesting if it possesses higher support and higher confidence. This type of association rules

is called heavy association rules (Adhikari and Rao 2008). Sometimes the number of times an association rule gets reported from local databases becomes an interesting issue. In the context of multiple databases, an association rule is called high-frequency rule (Wu and Zhang 2003) if it is extracted from many databases. In this context an association rule is called low-frequency rule (Adhikari and Rao 2008) if it is extracted from a few databases. Some association rules possess high support but have been extracted from a few databases only. These association rules are called exceptional association rules (Adhikari and Rao 2008). Many corporate decisions could be influenced by these types of extreme association rules in multiple databases. Thus, it is important to mine them. In the next section, we present different extreme association rules, and then we present a model of mining such association rules.

The chapter is organized as follows. We discuss some “extreme” types of association rules (Sect. 2.2). In Sect. 2.3, we present the problem formally. An extended model of local pattern analysis is presented in Sect. 2.4. We discuss related work in Sect. 2.5. We present an algorithm for synthesizing different extreme association rules in Sect. 2.6. In this section, we have also defined error of the experiment. We present experimental result in Sect. 2.7. Finally, some conclusions are provided in Sect. 2.8.

2.2 Some Extreme Types of Association Rule in Multiple Databases

Consider a large company with transactions originating from n branches. Let D_i be the database corresponding to the i -th branch of this multi-branch company, $i = 1, 2, \dots, n$. Furthermore let D be the union of all branch databases. First, we define a heavy association rule in a single database. Afterwards, we define a heavy association rule in multiple databases.

Definition 2.1 An association rule r in database DB is heavy if $\text{supp}_a(r, DB) \geq \mu$, and $\text{conf}_a(r, DB) \geq \nu$, where $\mu (>\alpha)$ and $\nu (>\beta)$ are the user-defined thresholds of high-support and high-confidence for identifying heavy association rules in DB , respectively. •

If an association rule is heavy in a local database then it might not be heavy in D . An association rule in D might have different statuses in different local databases. For example, it might be a heavy association rule, or an association rule, or a suggested association rule (defined later), or absent in a local database. Thus, we need to synthesize an association rule for determining its overall status in D . The method of synthesizing an association rule is discussed in Sect. 2.6. After synthesizing an association rule, we get its synthesized support and synthesized confidence in D . Let $\text{supp}_s(r, DB)$ and $\text{conf}_s(r, DB)$ denote synthesized support and synthesized confidence of association rule r in DB , respectively. A heavy association rule in multiple databases is defined as follows:

Definition 2.2 Let D be the union of all local databases. An association rule r in D is heavy if $\text{supp}_s(r, D) \geq \mu$, and $\text{conf}_s(r, D) \geq \nu$, where μ and ν are the user-defined thresholds of high-support and high-confidence used for identifying heavy association rules in D , respectively. •

Apart from synthesized support and synthesized confidence of an association rule, the frequency of an association rule is an important issue in multi-database mining. We define *frequency* of an association rule as the number of extractions of the association rule from different databases. If an association rule is extracted from k out of n databases then the frequency of the association rule is k , $0 \leq k \leq n$. An association rule may be high-frequency rule or, low-frequency rule, or neither high-frequency rule nor low-frequency rule in multiple databases. We could arrive in such a conclusion only if we have user-defined thresholds of low-frequency (γ_1) and high-frequency (γ_2) of an association rule, for $0 < \gamma_1 < \gamma_2 \leq 1$. A low-frequency association rule is extracted from less than $n \times \gamma_1$ databases. On the other hand, a high-frequency association rule is extracted from at least $n \times \gamma_2$ databases. In the context of multi-database mining using local pattern analysis, we define a high-frequency association rule and a low-frequency association rule as follows:

Definition 2.3 Let an association rule be extracted from k out of n databases. Then the association rule is low-frequency rule if $k < n \times \gamma_1$, where γ_1 is the user-defined threshold of low-frequency. •

Definition 2.4 Let an association rule be extracted from k out of n databases. Then the association rule is high-frequency rule if $k \geq n \times \gamma_2$, where γ_2 is the user-defined threshold of high-frequency. •

While synthesizing heavy association rules in multiple databases, it may be worth noting some other attributes of a synthesized association rule. For example, high-frequency, low-frequency, and exceptionality are interesting as well as important attributes of a synthesized association rule. We have already defined high-frequency association rule and low-frequency association rule in multiple databases. We now define an exceptional association rule in multiple databases:

Definition 2.5 A heavy association rule in multiple databases is exceptional if it is a low-frequency rule. •

It may be worth contrasting a heavy association rule, a high-frequency association rule with an exceptional association rule in multiple databases.

- An exceptional association rule is also a heavy association rule.
- A high-frequency association rule is not an exceptional association rule, and vice versa.
- A high-frequency association rule is not necessarily be a heavy association rule.
- There may exist heavy association rules that are neither high-frequency rule nor exceptional rule.

The goal of this chapter is to extract these extreme association rules from multiple databases. For this purpose, we present an extended model of local pattern analysis.

2.3 Problem Statement

In the previous section, we learnt different types of extreme association rules. As discussed in Chap. 1, we have observed some difficulties in extracting different extreme association rules in the union of all branch databases by employing a traditional data mining technique. Therefore, we synthesize different extreme association rules by using patterns in branch databases. Let D be the union of all branch databases. Also, let RB_i and SB_i be the rulebase and suggested rulebase corresponding to database D_i , respectively. An association rule $r \in RB_i$, if $supp_a(r, D_i) \geq \alpha$, and $conf_a(r, D_i) \geq \beta$, $i = 1, 2, \dots, n$. An association rule $r \in SB_i$, if $supp_a(r, D_i) \geq \alpha$, and $conf_a(r, D_i) < \beta$. There is a tendency of a suggested association rule in a database to become an association rule in another database. Apart from the association rules, we also consider the suggested association rules for synthesizing heavy association rules in D . The reasons for considering suggested association rules are given as follows. Firstly, we could synthesize support and confidence of an association rule in D more accurately. Secondly, we could synthesize high-frequency association rules in D more accurately. Thirdly, some experimental results have shown that the number of suggested association rules could be significant for some databases. In general, the accuracy of synthesizing an association rule increases as the number of extractions of the association rule increases. Thus, we consider suggested association rules also in synthesizing heavy association rules in D . In addition, the number of transactions in a database would be required in synthesizing an association rule. We define *size* of database DB as the number of transactions in DB , denoted by $size(DB)$. We state the problem as follows.

Let there be n databases D_1, D_2, \dots, D_n . Let RB_i and SB_i be the set of association rules and suggested association rules in D_i , respectively, $i = 1, 2, \dots, n$. Synthesize heavy association rules in the union of all databases (D) based on RB_i and SB_i , $i = 1, 2, \dots, n$. Also, notify whether each heavy association rule is high-frequency rule or exceptional rule in D .

2.4 An Extended Model of Local Pattern Analysis for Synthesizing Global Patterns

Let D_i be the database corresponding to i -th branch of the organization, $i = 1, 2, \dots, n$. Patterns in multiple databases could be grouped into the following categories based on the number of databases: local patterns, global patterns, and

patterns that are neither local nor global. A pattern based on a branch database is called a *local pattern*. On the other hand, a *global pattern* is based on all databases under consideration. An essence of the extended model of local pattern analysis (Adhikari and Rao 2008) is illustrated in Fig. 2.1. The extended model comes with a set of interfaces and a set of layers. Each interface realizes a set of operations and produces dataset(s) (or, knowledge) based on the dataset(s) available at the next lower layer. There are four interfaces of the proposed model of synthesizing global patterns from local patterns.

Interface 2/1 is concerned with different operations on data realized at the lowest layer. By applying these operations, we come up with a processed database resulting from a local (original) database. These operations are performed on each branch database. Interface 3/2 applies a filtering algorithm to each processed database to separate relevant data from possible outliers. In particular, if we are interested in studying durable items then the transactions containing only non-durable items could be treated as outlier transactions. Different interesting criteria could be set to filter data. This interface supports loading data into the respective data warehouse. 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 fully satisfy the requisite interestingness criteria. The reasons for considering suggested patterns are given as follows. Firstly, by admitting these patterns, we could synthesize patterns more accurately. Secondly, due to the stochastic nature of the transactions, the number of suggested patterns could be significant in some databases. Thirdly, there is a tendency that a suggested pattern of one database could become a local pattern in some other

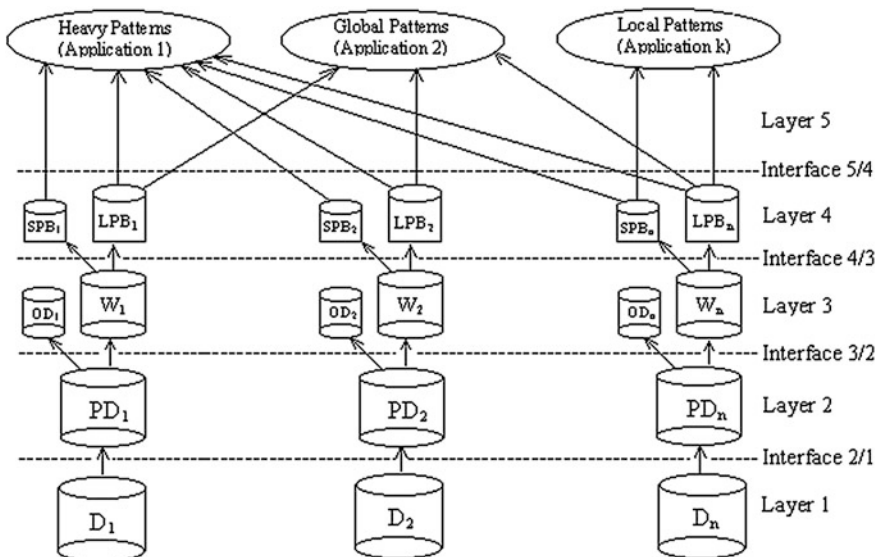


Fig. 2.1 A model of synthesizing global patterns from local patterns in different databases

databases. Thus, the correctness of synthesizing global patterns would increase as the number of local patterns increases. Therefore, the extended model becomes effective in synthesizing non-local patterns. Consider a multi-branch company having n databases. Let LPB_i and SPB_i be the local pattern base and suggested local pattern base corresponding to i -th branch of the organization, respectively, $i = 1, 2, \dots, n$. Interface 5/4 synthesizes global patterns, or analyses local patterns in order to find solutions to many problems.

At the lowest layer, all the local databases are retained. We may need to process these databases for the purpose of data mining task. Various data preparation techniques (Pyle 1999)—data preprocessing 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 the original database D_i , $i = 1, 2, \dots, n$. Then we retain all the data that are relevant to the data mining applications. Using a relevance analysis, one could detect outlier data (Last and Kandel 2001) from processed database. A relevance analysis is dependent on the context and varies from one application to another. Let OD_i be the outlier database corresponding to the i -th branch, $i = 1, 2, \dots, n$. Sometimes these databases are also used in some other applications. After removing outliers from the processed database we form data warehouse, and the data present there become ready for data mining task. Let W_i be the data warehouse corresponding to i -th branch. Local patterns for the i -th branch are extracted from W_i , $i = 1, 2, \dots, n$. Finally, the local patterns are forwarded to the central office for synthesizing global patterns, or completing analysis of local patterns. Many data mining applications could be developed based on the local patterns in different databases. In particular, if we are interested in synthesizing global frequent itemsets then an itemset may not be extracted from all the databases. It might be required to estimate the support of a frequent itemset in a database that fails to report it. Thus, in essence, a global frequent itemset synthesized from local frequent itemsets is approximate. 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, one could apply an appropriate sampling technique to reduce the size of the corresponding local database. Otherwise, the database could be partitioned into sub-databases. As a result, the error associated with the results produced in data analysis would increase.

Though the above model introduces many layers and interfaces for synthesizing global patterns, in a real life application, some of these layers might not be fully exploited. In this chapter, we discuss a problem of multi-database mining that uses the above model.

2.5 Related Work

Some applications of multiple large databases have been discussed in Chap. 1. Association rule mining gives rise to interesting association between two itemsets in a database. The notion of association rule was introduced by Agrawal et al.

(1993). The authors have proposed an algorithm to mine frequent itemsets in a database. Many algorithms to extract association rules have been reported in the literature. In what follows, we present a few interesting algorithms for extracting association rules in a database. Agrawal and Srikant (1994) have proposed apriori algorithm that uses breadth-first search strategy to count the supports of itemsets. The algorithm uses an improved candidate generation function, which exploits the downward closure property of support and makes it more efficient than earlier algorithm. Han et al. (2000) have proposed data mining method of FP-growth (frequent pattern growth) which uses an extended prefix-tree (FP-tree) structure to store the database in a compressed form. FP-growth adopts a divide-and-conquer approach to decompose both the mining tasks and databases. It uses a pattern fragment growth method to avoid the costly process of candidate generation and testing. Savasere et al. (1995) have introduced partition algorithm. The database is scanned only twice. In the first scan, the database is partitioned and in each partition support is counted. Then the counts are merged to generate potential frequent itemsets. In the second scan, the potential frequent itemsets are counted to find the actual frequent itemsets.

Existing parallel mining techniques (Agrawal and Shafer 1999; Chatratichat et al. 1997; Cheung et al. 1996) could also be used to mine different extreme association rules in multi-databases. Zeng et al. (2012) surveyed state-of-the-art algorithms and applications in distributed data mining. Zhong et al. (2003) have proposed a theoretical framework for peculiarity oriented mining in multiple data sources. Zhang et al. (2009) have proposed a nonlinear method, named KEMGP, which adopts kernel estimation method for synthesizing global patterns from local patterns. Shang et al. (2008) have proposed an extension to Piatetsky-Shapiro's minimum interestingness condition to mine association rules in multiple databases.

Yi and Zhang (2007) have proposed a privacy-preserving distributed association rule mining protocol based on a semi-trusted mixer model. Rozenberg and Gudes (2006) have presented their work on association rule mining from distributed vertically partitioned data with the goal of preserving the confidentiality of each database. The authors have presented two algorithms for discovering frequent itemsets and for calculating the confidence of the rules.

Zhu and Wu (2007) have proposed a framework DRAMA for discovering patterns from different databases with patterns' relationships satisfying the user-specified constraints. It builds a HFP-tree from multiple databases, and mine patterns from the HFP-tree by integrating users' constraints into the pattern mining process. Wu et al. (2013) have studied the problem of frequent pattern mining without user-specified gap constraints and proposed PMBC to solve the problem of finding patterns without involving user-specified gap requirements. Given a sequence and a support threshold value, PMBC intends to discover all subsequences with their support values equal to or greater than the given threshold value.

Liu et al. (2010) have presented a top-down mining of sequential patterns (TD-Seq) for mining sequential patterns from high-dimensional stock sequence databases. A two-phase mining method has been proposed, in which a top-down transposition-based searching strategy as well as a new support counting method are exploited.

2.6 Synthesizing an Association Rule

The technique of synthesizing heavy association rules is suitable for the real databases, where the trend of the customers' behavior 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. Our estimation procedure captures such trend and estimates the support of a missing association rule in a database. Let $E_1(r, DB)$ be the amount of error in estimating support of a missing association rule r in database DB . Also, let $E_2(r, DB)$ be the level of error in assuming support as 0 for the missing association rule in DB . Then the value of $E_1(r, DB)$ is usually lower than $E_2(r, DB)$. The estimated support and confidence of a missing association rule usually reduce the error of synthesizing heavy association rules in different databases. We would like to estimate the support and confidence of a missing association rule rather assuming it as absent in a database. If an association rule fails to get extracted from database DB , then we assume that DB contributes some amount of support and confidence for the association rule. The support and confidence of an association rule r in database DB satisfy the following inequality:

$$0 \leq \text{supp}_a(r, DB) \leq \text{conf}_a(r, DB) \leq 1 \quad (2.1)$$

At a given $\alpha = \alpha_0$, we observe that the confidence of an association rule r varies over the interval $[\alpha_0, 1]$ as explained in Example 2.1.

Example 2.1 Let $\alpha = 0.333$. Assume that database D_1 contains the following transactions: $\{a1, b1, c1\}$, $\{a1, b1, c1\}$, $\{b2, c2\}$, $\{a2, b3, c3\}$, $\{a3, b4\}$ and $\{c4\}$. The support and confidence of association rule $r: \{a1\} \rightarrow \{b1\}$ in D_1 are 0.333 and 1.0 (highest), respectively. Assume that database D_2 contains the following transactions: $\{a1, b1, c1\}$, $\{a1, b1\}$, $\{a1, c1\}$, $\{a1\}$, $\{a1, b2\}$ and $\{a1, b3\}$. The support and confidence of r in D_2 are 0.333 and 0.333 (lowest), respectively. •

As the support of an association rule is expressed as the lower bound of its confidence, the confidence goes up as support increases. The support of an association rule is distributed over $[0, 1]$. If an association rule is not extracted from a database, then the support falls in $[0, \alpha)$, since the suggested association rules are also considered for synthesizing association rules. We would be interested in estimating the support of such rules. Assume that the association rule $r: \{c\} \rightarrow \{d\}$ has been extracted from m databases, $1 \leq m \leq n$. Without any loss of generality, we assume that the association rule r has been reported from the first m databases. We shall use the average behavior of the customers of the first m branches to estimate the average behavior of the customers in remaining branches. Let $D_{i,j}$ denote the union of databases D_i, D_{i+1}, \dots, D_j , for $1 \leq i \leq j \leq n$. Then, $\text{supp}_a(\{c, d\}, D_{1,m})$ could be viewed as the average behavior of customers of the first m branches for purchasing items c and d together at the same time. Then, $\text{supp}_a(\{c, d\}, D_{1,m})$ is obtained by using the following formula:

$$supp_a(\{c, d\}, D_{1,m}) = \left(\sum_{i=1}^m supp_a(r, D_i) \times size(D_i) \right) / \sum_{i=1}^m size(D_i) \quad (2.2)$$

We estimate the support of association rule r for each of the remaining $(n - m)$ databases as follows:

$$supp_s(r, D_{m+1,n}) = \alpha \times supp_a(\{c, d\}, D_{1,m}) \quad (2.3)$$

The number of the transactions containing the itemset $\{c, d\}$ in D_i is $supp_a(r, D_i) \times size(D_i)$, for $i = 1, 2, \dots, m$. The association rule r is not present in D_i , for $i = m + 1, m + 2, \dots, n$. Then the estimated number of the transactions containing the itemset $\{c, d\}$ in D_i is $supp_s(r, D_{m+1,n}) \times size(D_i)$, for $i = m + 1, m + 2, \dots, n$. The estimated support of association rule r in D_i is determined in the form:

$$supp_e(r, D_i) = \begin{cases} supp_a(r, D_i), & \text{for } i = 1, 2, \dots, m \\ supp_s(r, D_{m+1,n}), & \text{for } i = m + 1, m + 2, \dots, n \end{cases} \quad (2.4)$$

Then the synthesized support of association rule r in D is expressed as follows.

$$supp_s(r, D) = \left(\sum_{i=1}^n supp_e(r, D_i) \times size(D_i) \right) / \sum_{i=1}^n size(D_i) \quad (2.5)$$

The confidence of the association rule r depends on the supports of the itemsets $\{c\}$ and $\{c, d\}$. The support of itemset $\{c, d\}$ has been synthesized. Now, we need to synthesize the support of itemset $\{c\}$. Without any loss of generality, let the itemset $\{c\}$ gets extracted from first p databases, for $1 \leq m \leq p \leq n$. The estimated support of frequent itemset $\{c\}$ in D_i is calculated as follows:

$$supp_e(\{c\}, D_i) = \begin{cases} supp_a(\{c\}, D_i), & \text{for } i = 1, 2, \dots, p \\ supp_s(\{c\}, D_{p+1,n}), & \text{for } i = p + 1, p + 2, \dots, n \end{cases} \quad (2.6)$$

Then the synthesized support of itemset $\{c\}$ in D is determined.

$$supp_s(\{c\}, D) = \left(\sum_{i=1}^n supp_e(\{c\}, D_i) \times size(D_i) \right) / \sum_{i=1}^n size(D_i) \quad (2.7)$$

We compute the synthesized confidence of association rule r in D .

$$conf_s(r, D) = supp_s(r, D) / supp_s(\{c\}, D) \quad (2.8)$$

2.6.1 Design of Algorithm

Here we present an algorithm for synthesizing heavy association rules in D . The algorithm also indicates whether a heavy association rule is high-frequency rule or

exceptional rule. Let N and M be the number of association rules and the number of suggested association rules in different local databases, respectively. The association rules and suggested association rules are kept in arrays RB and SB , respectively. An association rule could be described by following attributes: *ant*, *con*, *did*, *supp* and *conf*. The attributes *ant*, *con*, *did*, *supp* and *conf* represent antecedent, consequent, database identification, support, and confidence of a rule, respectively. An attribute x of the i -th association rule of RB is denoted by $RB(i).x$, $i = 1, 2, \dots, |RB|$. All the synthesized association rules are kept in array SR . Each synthesized association rule could be described by following attributes: *ant*, *con*, *did*, *ssupp* and *sconf*. The attributes *ssupp* and *sconf* represent synthesized support and synthesized confidence of a synthesized association rule, respectively. In the context of mining heavy association rules in D , the following additional attributes are also considered: *heavy*, *highFreq*, *lowFreq* and *except*. The attributes *heavy*, *highFreq*, *lowFreq* and *except* are used to indicate whether an association rule is a heavy rule, high-frequency rule, low-frequency rule and exceptional rule in D , respectively. An attribute y of the i -th synthesized association rule of SR is denoted by $SR(i).y$, $i = 1, 2, \dots, |SR|$.

Algorithm 2.1 Synthesize heavy association rules in D . Also, it indicates whether a heavy association rule is high-frequency rule or exceptional rule.

procedure Association-Rule-Synthesis (n , RB , SB , μ , v , $size$, γ_1 , γ_2)

Inputs:

n : number of databases

RB : array of association rules

SB : array of suggested association rules

μ : threshold of high-support for determining heavy association rules

v : threshold of high-confidence for determining heavy association rules

$size$: array of the number of transactions in different databases

γ_1 : threshold of low-frequency for determining low-frequency association rules

γ_2 : threshold of high-frequency for determining high-frequency association rules

Outputs:

Heavy association rules along with their high-frequency and exceptionality statuses

01: copy rules of RB and SB into array R ;

02: sort rules of R based on attributes *ant* and *con*;

03: calculate total number of transactions in all the databases and store it in *totalTrans*;

04: **let** $nSynRules = 1$;

05: **let** $curPos = 1$;

06: **while** ($curPos \leq |R|$) **do**

07: calculate number of occurrences of current rule $R(curPos)$ and store it in $nExtractions$;

08: **let** $SR(nSynRules).highFreq = \text{false}$;

09: **if** $((nExtractions / n) \geq \gamma_2)$ **then**

10: $SR(nSynRules).highFreq = \text{true}$;

11: **end if**

12: **let** $SR(nSynRules).lowFreq = \text{false}$;

13: **if** $((nExtractions / n) < \gamma_1)$ **then**

14: $SR(nSynRules).lowFreq = \text{true}$;

15: **end if**

```

16: calculate  $supp_s(R(curPos), D)$  using formula (2.5);
17: calculate  $conf_s(R(curPos), D)$  using formula (2.8);
18: let  $SR(nSynRules).heavy = \text{false}$ ;
19: if  $((supp_s(SR(nSynRules), D) \geq \mu)$  and  $(conf_s(SR(nSynRules), D) \geq \nu))$  then
20:    $SR(nSynRules).heavy = \text{true}$ ;
21: end if
22: let  $SR(nSynRules).except = \text{false}$ ;
23: if  $((SR(nSynRules)$  is a low-frequency rule) and  $(SR(nSynRules)$  is a heavy rule)) then
24:    $SR(nSynRules).except = \text{true}$ ;
25: end if
26: update index  $curPos$  for processing the next association rule;
27: increase index  $nSynRules$  by 1;
28: end while
29: for each synthesized association rule  $\tau$  in  $SR$  do
30:   if  $\tau$  is heavy then
31:     display  $\tau$  along with its high-frequency and exceptionality statuses;
32:   end if
33: end for
end procedure

```

The above algorithm works as follows. The association rules and suggested association rules are copied into R . All the association rules in R are sorted on the pair of attributes $\{ant, con\}$, so that the same association rule extracted from different databases remains together after sorting. Thus, it would help synthesizing a single association rule at a time. The synthesis process is realized in the while-loop shown in line 6. Based on the number of extractions of an association rule, we could determine its high-frequency and low-frequency statuses. The number of extractions of current association rule has been determined as indicated in line 7. The high-frequency status of current association rule is determined—see lines 8–11. Also, the low-frequency status of current association rule is determined (lines 12–15). We synthesize support and confidence of current association rule based on Eqs. (2.5) and (2.8), respectively. Once the synthesized support and synthesized confidence have been determined, we could identify the heavy and exceptional statuses of current association rule. The heavy status of current association rule is determined using the part of the procedure covered in lines 18–21. Also, the exceptional status of current association rule is determined using lines 22–25. At line 26, we determine the next association rule in R for the synthesizing process. Heavy association rules are displayed along with their high-frequency and exceptionality statuses using lines 29–33. The shaded lines of the pseudo code have been added to report the high-frequency and exceptional statuses of heavy association rules.

Theorem 2.1 *The time complexity of procedure Association-Rule-Synthesis is maximum $\{O((M + N) \times \log(M + N)), O(n \times (M + N))\}$, where N and M are the number of association rules and the number of suggested association rules extracted from n databases.*

Proof The lines 1 and 2 take time in $O(M + N)$ and $O((M + N) \times \log(M + N))$ respectively, since there are $M + N$ rules in different local databases. The while-loop at line 6 repeats maximum $M + N$ times. Line 7 takes $O(n)$ time, since each rule is extracted maximum n number of times. Lines 8–15 take $O(1)$ time. Using formula (2.3), we could calculate the average behavior of customers of the first m databases in $O(m)$ time. Each of lines 16 and 17 takes $O(n)$ time. Lines 18–25 take $O(1)$ time. Line 26 could be executed during execution of line 7. Thus, the time complexity of while-loop 6–28 is $O(n \times (M + N))$. The time complexity of lines 29–33 is $O(M + N)$, since the number of synthesized association rules is less than or equal to $M + N$. Thus, time complexity of procedure *Association-Rule-Synthesis* is $\text{maximum}\{O((M + N) \times \log(M + N)), O(n \times (M + N)), O(M + N)\} = \text{maximum}\{O((M + N) \times \log(M + N)), O(n \times (M + N))\}$. •

Wu and Zhang (2003) have proposed *RuleSynthesizing* algorithm for synthesizing high-frequency association rules in different databases. The algorithm is based on the weights of the different databases. Again, the weight of a database would depend on the association rules extracted from the database. The proposed algorithm executes in $O(n^4 \times \text{maxNosRules} \times \text{totalRules}^2)$ 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. Ramkumar and Srinivasan (2008) have proposed a modification of *RuleSynthesizing* algorithm. In this modified algorithm, the weight of an association rule is based on the size of a database. This assumption seems to be more logical. For synthesizing confidence of an association rule, the authors have described a method which was originally proposed by Adhikari and Rao (2008). Though the time complexity of modified *RuleSynthesizing* algorithm is the same as that of the original *RuleSynthesizing* algorithm, but it reduces the average error in synthesizing an association rule. The algorithm *Association-Rule-Synthesis* could synthesize heavy association rules, high-frequency association rules, and exceptional association rules in $\text{maximum}\{O(\text{totalRules} \times \log(\text{totalRules})), O(n \times \text{totalRules})\}$ time. Thus, algorithm *Association-Rule-Synthesis* takes much less time than the existing algorithms. Moreover, the proposed algorithm is simple and straight forward. We illustrate the performance of the proposed algorithm using the following example.

Example 2.2 Let D_1 , D_2 and D_3 be three databases of sizes 4,000 transactions, 3,290 transactions, and 10,200 transactions, respectively. Let D be the union of the databases D_1 , D_2 , and D_3 . Assume that $\alpha = 0.2$, $\beta = 0.3$, $\gamma_1 = 0.4$, $\gamma_2 = 0.7$, $\mu = 0.3$ and $\nu = 0.4$. The following association rules have been extracted from the given databases. $r_1: \{H\} \rightarrow \{C, G\}$, $r_2: \{C\} \rightarrow \{G\}$, $r_3: \{G\} \rightarrow \{F\}$, $r_4: \{H\} \rightarrow \{E\}$, $r_5: \{A\} \rightarrow \{B\}$. The rulebases are given as follows: $RB_1 = \{r_1, r_2\}$, $SB_1 = \{r_3\}$; $RB_2 = \{r_4\}$, $SB_2 = \{r_1\}$; $RB_3 = \{r_1, r_5\}$, $SB_3 = \{r_2\}$. The supports and confidences of the association rules are given as follows. $\text{supp}_a(r_1, D_1) = 0.22$, $\text{conf}_a(r_1, D_1) = 0.55$; $\text{supp}_a(r_1, D_2) = 0.25$, $\text{conf}_a(r_1, D_2) = 0.29$; $\text{supp}_a(r_1, D_3) = 0.20$, $\text{conf}_a(r_1, D_3) = 0.52$; $\text{supp}_a(r_2, D_1) = 0.69$, $\text{conf}_a(r_2, D_1) = 0.82$; $\text{supp}_a(r_2, D_3) = 0.23$, $\text{conf}_a(r_2, D_3) = 0.28$; $\text{supp}_a(r_3, D_1) = 0.22$, $\text{conf}_a(r_3,$

Table 2.1 Heavy association rules in the union of databases given in Example 2.2

$r: ant \rightarrow con$	Ant	Con	$Supp_s(r, D)$	$Conf_s(r, D)$	Heavy	High frequency	Except
r_2	C	G	0.31	0.66	True	False	False
r_5	A	B	0.57	0.90	True	False	True

$D_1) = 0.29$; $supp_a(r_4, D_2) = 0.40$, $conf_a(r_4, D_2) = 0.45$; $supp_a(r_5, D_3) = 0.86$, $conf_a(r_5, D_3) = 0.92$. Also, let $supp_a(\{A\}, D_3) = 0.90$, $supp_a(\{C\}, D_1) = 0.80$, $supp_a(\{C\}, D_3) = 0.40$, $supp_a(\{G\}, D_1) = 0.29$, $supp_a(\{H\}, D_1) = 0.31$, $supp_a(\{H\}, D_2) = 0.33$, and $supp_a(\{H\}, D_3) = 0.50$. Heavy association rules are presented in Table 2.1.

The association rules r_2 and r_5 have synthesized support greater than or equal to 0.3 and synthesized confidence greater than or equal to 0.4. So, r_2 and r_5 are heavy association rules in D . The association rule r_5 is a exceptional rule, since it is a heavy and low-frequency rule. But the association rule r_2 is neither a high-frequency nor exceptional rule. Though the association rule r_1 is a high-frequency rule but it is not a heavy rule, since $supp_s(r_1, D) = 0.21$ and $conf_s(r_1, D) = 0.48$. •

2.6.2 Error Calculation

To evaluate the proposed technique of synthesizing heavy association rules we have determined the error which has occurred in the experiments. More specifically, the error is expressed relative to the number of transactions, number of items, and the length of a transaction in the databases. Thus the error of an experiment needs to be expressed along with ANT , ALT , and ANI in the given databases, where ANT , ALT and ANI denote the average number of transactions, the average length of a transaction and the average number of items in a database, respectively. There are several ways one could define the error. The proposed definition of error is based on the frequent itemsets generated from heavy association rules. Let $r: \{c\} \rightarrow \{d\}$ be a heavy association rule. Then the frequent itemsets generated from association rule r are $\{c\}$, $\{d\}$, and $\{c, d\}$. Let $\{X_1, X_2, \dots, X_m\}$ be set of frequent itemsets generated from all the heavy association rules in D . We define the following two types of error.

1. Average Error (AE)

$$AE(D, \alpha, \mu, \nu) = \frac{1}{m} \sum_{i=1}^m |supp_a(X_i, D) - supp_s(X_i, D)| \quad (2.9)$$

2. Maximum Error (ME)

$$ME(D, \alpha, \mu, \nu) = maximum\{ |supp_a(X_i, D) - supp_s(X_i, D)|, \quad i = 1, 2, \dots, m \} \quad (2.10)$$

where $\text{supp}_a(X_i, D)$ and $\text{supp}_s(X_i, D)$ are actual support i.e., the support based on apriori algorithm and synthesized support of the itemset X_i in D , respectively. In Example 2.3, we illustrate the behavior of the measures given above.

Example 2.3 With reference to Example 2.2, $r_2: C \rightarrow G$ and $r_5: A \rightarrow B$ are heavy association rules in D . The frequent itemsets generated between r_2 and r_5 are A, B, C, G, AB and CG . For the purpose of finding the error of an experiment, we need to find the actual supports of the itemsets generated from the heavy association rules. The actual support of an itemset generated from a heavy association rule could be obtained by mining all the databases D_1, D_2 , and D_3 together.

Thus,

$$\begin{aligned} \text{AE}(D, 0.2, 0.3, 0.4) = & \frac{1}{6} \{ |\text{supp}_a(\{A\}, D) - \text{supp}_s(\{A\}, D)| + |\text{supp}_a(\{B\}, D) - \text{supp}_s(\{B\}, D)| \\ & + |\text{supp}_a(\{C\}, D) - \text{supp}_s(\{C\}, D)| + |\text{supp}_a(\{G\}, D) - \text{supp}_s(\{G\}, D)| \\ & + |\text{supp}_a(\{A, B\}, D) - \text{supp}_s(\{A, B\}, D)| + |\text{supp}_a(\{C, G\}, D) - \text{supp}_s(\{C, G\}, D)| \}. \end{aligned}$$

$$\begin{aligned} \text{ME}(D, 0.2, 0.3, 0.4) = & \text{maximum}\{ |\text{supp}_a(\{A\}, D) - \text{supp}_s(\{A\}, D)|, |\text{supp}_a(\{B\}, D) - \text{supp}_s(\{B\}, D)| \\ & |\text{supp}_a(\{C\}, D) - \text{supp}_s(\{C\}, D)|, |\text{supp}_a(\{G\}, D) - \text{supp}_s(\{G\}, D)|, \\ & |\text{supp}_a(\{A, B\}, D) - \text{supp}_s(\{A, B\}, D)|, |\text{supp}_a(\{C, G\}, D) - \text{supp}_s(\{C, G\}, D)| \}. \bullet \end{aligned}$$

2.7 Experiments

We have carried out several experiments to study the effectiveness of the approach presented in this chapter. We present the experimental results using three real databases. The database *retail* (Frequent itemset mining dataset repository 2004) is obtained from an anonymous Belgian retail supermarket store. The databases *BMS-Web-Wiew-1* and *BMS-Web-Wiew-2* can be found from KDD CUP 2000 (Frequent itemset mining dataset repository 2004). We present some characteristics of these databases in Table 2.2. We use notation *DB*, *NT*, *AFI*, *ALT* and *NI* to denote a database, the number of transactions, the average frequency of an item, the average length of a transaction and the number of items in the database, respectively.

Each of the above databases is divided into 10 subsets for the purpose of carrying out experiments. The databases obtained from *retail*, *BMS-Web-Wiew-1* and *BMS-Web-Wiew-2* are named as R_i , B_{1i} and B_{2i} respectively, $i = 0, 1, \dots, 9$.

Table 2.2 Dataset characteristics

Dataset	<i>NT</i>	<i>ALT</i>	<i>AFI</i>	<i>NI</i>
<i>Retail</i>	88,162	11.31	99.67	10,000
<i>BMS-Web-Wiew-1</i>	1,49,639	2.00	155.71	1,922
<i>BMS-Web-Wiew-2</i>	3,58,278	2.00	7165.56	100

Table 2.3 Branch database characteristics

<i>DB</i>	<i>NT</i>	<i>ALT</i>	<i>AFI</i>	<i>NI</i>	<i>DB</i>	<i>NT</i>	<i>ALT</i>	<i>AFI</i>	<i>NI</i>
R_0	9,000	11.24	12.07	8,384	R_5	9,000	10.86	16.71	5,847
R_1	9,000	11.21	12.27	8,225	R_6	9,000	11.20	17.42	5,788
R_2	9,000	11.34	14.60	6,990	R_7	9,000	11.16	17.35	5,788
R_3	9,000	11.49	16.66	6,206	R_8	9,000	12.00	18.69	5,777
R_4	9,000	10.96	16.04	6,148	R_9	7,162	11.69	15.35	5,456
B_{10}	14,000	2.00	14.94	1,874	B_{15}	14,000	2.00	280.00	100
B_{11}	14,000	2.00	280.00	100	B_{16}	14,000	2.00	280.00	100
B_{12}	14,000	2.00	280.00	100	B_{17}	14,000	2.00	280.00	100
B_{13}	14,000	2.00	280.00	100	B_{18}	14,000	2.00	280.00	100
B_{14}	14,000	2.00	280.00	100	B_{19}	23,639	2.00	472.78	100
B_{20}	35,827	2.00	1326.93	54	B_{25}	35,827	2.00	716.54	100
B_{21}	35,827	2.00	1326.93	54	B_{26}	35,827	2.00	716.54	100
B_{22}	35,827	2.00	716.54	100	B_{27}	35,827	2.00	716.54	100
B_{23}	35,827	2.00	716.54	100	B_{28}	35,827	2.00	716.54	100
B_{24}	35,827	2.00	716.54	100	B_{29}	35,835	2.00	716.70	100

The databases R_j and B_{ij} are called branch databases, $i = 1, 2$, and $j = 0, 1, \dots, 9$. Some characteristics of the branch databases are presented in Table 2.3.

The results of the three experiments using Algorithm 2.1 are presented in Table 2.4. The choice of different parameters is an important issue. We have selected different values of α and β for different databases. But, they are kept the same for branch databases obtained from the same database. For example, α and β are the same for branch databases R_i , for $i = 0, 1, \dots, 9$.

Table 2.4 First five heavy association rules reported from different databases (sorted in non-increasing order on synthesized support)

Data base	α	β	μ	ν	Heavy assoc rules	Syn supp	Syn conf	High freq	Exceptional
$\cup_{i=0}^9 R_i$	0.05	0.2	0.1	0.5	$\{48\} \rightarrow \{39\}$	0.33	0.68	Yes	No
					$\{39\} \rightarrow \{48\}$	0.33	0.56	Yes	No
					$\{41\} \rightarrow \{39\}$	0.13	0.63	Yes	No
					$\{38\} \rightarrow \{39\}$	0.12	0.66	Yes	No
					$\{41\} \rightarrow \{48\}$	0.10	0.51	Yes	No
$\cup_{i=0}^9 B_{1i}$	0.01	0.2	0.007	0.1	$\{1\} \rightarrow \{5\}$	0.01	0.13	No	No
					$\{5\} \rightarrow \{1\}$	0.01	0.11	No	No
					$\{7\} \rightarrow \{5\}$	0.01	0.12	No	No
					$\{5\} \rightarrow \{7\}$	0.01	0.11	No	No
					$\{3\} \rightarrow \{5\}$	0.01	0.12	No	No
$\cup_{i=0}^9 B_{2i}$	0.006	0.01	0.01	0.1	$\{3\} \rightarrow \{1\}$	0.02	0.14	Yes	No
					$\{1\} \rightarrow \{3\}$	0.02	0.14	Yes	No
					$\{7\} \rightarrow \{1\}$	0.02	0.14	Yes	No
					$\{1\} \rightarrow \{7\}$	0.02	0.14	Yes	No
					$\{5\} \rightarrow \{1\}$	0.02	0.14	Yes	No

2.7.1 Results of Experimental Studies

After mining a branch database from a group of branch databases using a reasonably low values α and β , one could fix α and β for the purpose data mining task. If α and β are smaller, then synthesized support and synthesized confidence values are closer to their actual values. Thus, the synthesized association rules are closer to the true association rules in multiple databases.

The choice of the values of μ and ν are context dependent. Also if μ and ν are kept fixed then some databases might not report heavy association rules, while other databases might report many heavy association rules. While generating association rule one could estimate the average synthesized support and confidence based on the generated association rules. Thus, it gives an idea of thresholds for high-support and high-confidence for synthesizing heavy association rules in different databases. Also, the choice of γ_1 and γ_2 are also context dependent. It has been found that “reasonable” values of γ_1 and γ_2 could lie in the interval $[0.3, 0.4]$ and $[0.6, 0.7]$, respectively. Given these findings, we have taken $\gamma_1 = 0.35$, and $\gamma_2 = 0.60$ for synthesizing heavy association rules.

The experiments conducted on the three databases have resulted in no exceptional association rule. Normally, exceptional association rules are rare. Also, we have not found any association rule which is a heavy rule as well as high-frequency rule in multiple databases obtained from *BMS-Web-Wiew-1*.

In many applications, the suggested association rules are significant. While synthesizing the association rules from different databases we might need to consider the suggested association rules for the correctness of synthesizing association rules. We have observed that the number of suggested association rules in the set of databases $\{R_0, R_1, \dots, R_9\}$ and $\{B_{10}, B_{11}, \dots, B_{19}\}$ are significant. But, the set of databases $\{B_{20}, B_{21}, \dots, B_{29}\}$ do not generate any suggested association rule. We present the number of association rules and the number of suggested association rules for different experiments in Table 2.5.

The error of synthesizing association rules in a database 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 database increases, the error of synthesizing association rules also increases, provided other two parameters remain constant. If the lengths of transactions of a database increase, the error of synthesizing association rules is likely to increase, provided that two other parameters remain constant. Lastly, if the number of items

Table 2.5 Number of association rules and suggested association rules extracted from multiple databases

Database	α	β	Number of association rules (N)	Number of suggested association rules (M)	$M/(N + M)$
$\cup_{i=0}^9 R_i$	0.05	0.2	821	519	0.39
$\cup_{i=0}^9 B_{1i}$	0.01	0.2	50	96	0.66
$\cup_{i=0}^9 B_{2i}$	0.006	0.01	792	0	0

Table 2.6 Error of synthesizing different extreme association rules

Database	α	β	μ	ν	(AE, ANT, ALT, ANI)	(ME, ANT, ALT, ANI)
$\cup_{i=0}^9 R_i$	0.05	0.2	0.1	0.5	(0.00, 8816.2, 11.31, 5882.1)	(0.00, 8816.2, 11.31, 5882.1)
$\cup_{i=0}^9 B_{1i}$	0.01	0.2	0.007	0.1	(0.00, 14963.9, 2.0, 277.4)	(0.00, 14963.9, 2.0, 277.4)
$\cup_{i=0}^9 B_{2i}$	0.006	0.01	0.01	0.1	(0.000118, 35827.8, 2.0, 90.8)	(0.00, 35827.8, 2.0, 90.8)

increases, then the error of synthesizing association rules is likely to decrease, provided that two other parameters remain constant. Thus, the error needs to be reported along with the *ANT*, *ALT* and *ANI* for the given databases. The obtained results are presented in Table 2.6.

2.7.2 Comparison with Existing Algorithm

In this section, we make a detailed comparison among the part of the proposed algorithm that synthesizes only high-frequency association rules, *RuleSynthesizing* algorithm (Wu and Zhang 2003) and *Modified RuleSynthesizing* algorithm (Ramkumar and Srivinasan 2008). Let the part of the proposed algorithm be *High-Frequency-Rule-Synthesis* used for synthesizing (only) high-frequency association rules in different databases. We conduct experiments for comparing these algorithms. We compare them on the basis of the following two criteria, namely average error and execution time.

2.7.2.1 Analysis of Average Error

The definitions of average error and maximum error given above and those proposed by Wu and Zhang (2003) are similar and use the same set of synthesized frequent itemsets. However the methods of synthesizing frequent itemsets for these two approaches are different. Thus, the value of error incurred in these two approaches might differ. In *RuleSynthesizing* algorithm, if an itemset fails to get extracted from a database then the support of the itemset is assumed to be 0. But in *Association-Rule-Synthesis* algorithm, if an itemset fails to get extracted from a database then the support of the itemset is estimated. The synthesized support of an itemset in the union of databases in these two approaches might be different. As the number of databases increases the relative presence of a rule normally decreases. The error of synthesizing an association rule normally increases. The AE reported in the experiment is likely to increase if the number of databases increases. We observe such phenomenon in Figs. 2.2 and 2.3.

The proposed algorithm follows a direct approach in identifying high-frequency association rules as opposed to the *RuleSynthesizing* and *Modified RuleSynthesizing* algorithms. In Figs. 2.2 and 2.3, we observe that AE of an experiment conducted using *High-Frequency-Rule-Synthesis* algorithm is less than that of

Fig. 2.2 AE versus the number of databases from *retail* at $(\alpha, \beta, \gamma) = (0.05, 0.2, 0.6)$

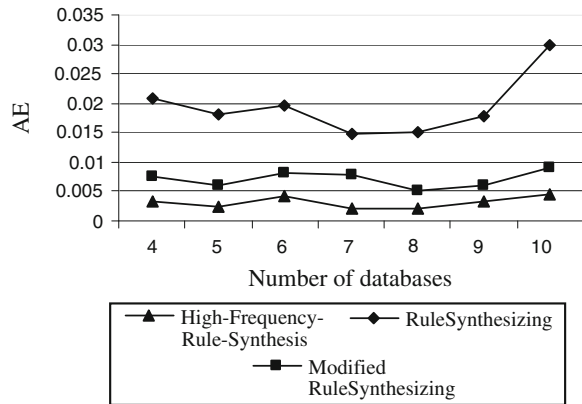
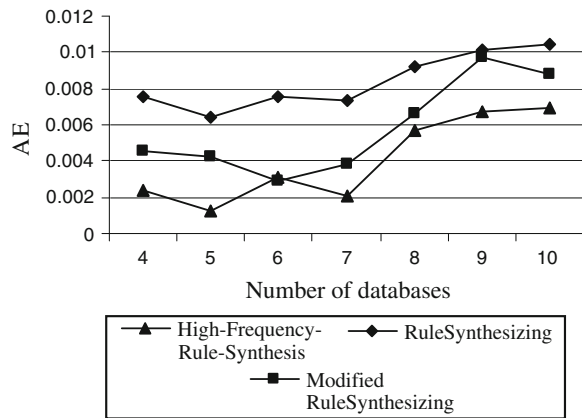


Fig. 2.3 AE versus the number of databases from *BMS-Web-Wiew-1* at $(\alpha, \beta, \gamma) = (0.005, 0.1, 0.3)$



RuleSynthesizing algorithm. But the *Modified RuleSynthesizing* algorithm improves the accuracy of synthesizing an association rule as compared to *RuleSynthesizing* algorithm. It remains less accurate when compared to the accuracy of the *High-Frequency-Rule-Synthesis* algorithm.

2.7.2.2 Analysis of Execution Time

We have also completed experiments to study the execution time by varying the number of databases. The number of synthesized frequent itemsets increases as the number of databases increases. The execution time increases with the increase of number of databases. We observe this phenomenon in Figs. 2.4 and 2.5. However, more significant differences are noted with the increase in the number of databases.

The time complexities of *RuleSynthesizing* and *Modified RuleSynthesizing* algorithms is the same. When the number of databases are less the *RuleSynthesizing*

Fig. 2.4 Execution time versus the number of databases from *retail* at $(\alpha, \beta, \gamma) = (0.05, 0.2, 0.6)$

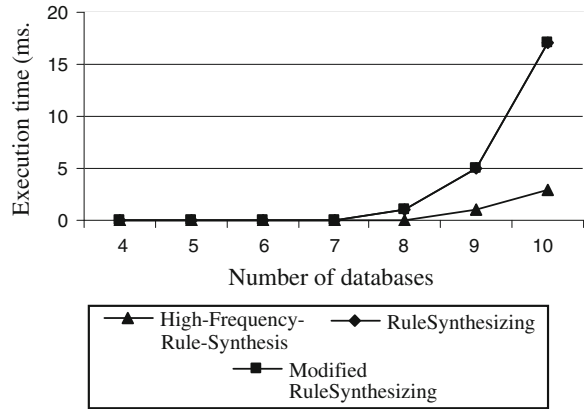
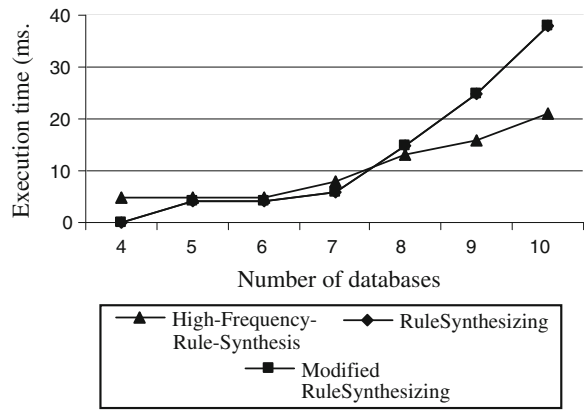


Fig. 2.5 Execution time versus the number of databases from *BMS-Web-Wiew-1* at $(\alpha, \beta, \gamma) = (0.005, 0.1, 0.3)$



and *Modified RuleSynthesizing* algorithms might be faster than *High-Frequency-Rule-Synthesizing* algorithm. As the number of databases increases, *High-Frequency-Rule-Synthesizing* algorithm works faster than both *RuleSynthesizing* and *Modified RuleSynthesizing* algorithms.

2.8 Conclusions

The extended model of local pattern analysis enables us to develop useful multi-database mining applications. Although it exhibits many layers and interfaces, this general model can come with many variations. In particular, some of these layers might not be present when developing a certain application. Synthesizing heavy association rule is an important component of a multi-database mining system. In this chapter, we have presented three extreme types of association rules present in

multiple databases viz., heavy association rules, high-frequency association rules, and exceptional association rules. The introduced algorithm referred to as the *Association-Rule-Synthesis* is used to synthesize these extreme association rules in multiple databases.

References

- Adhikari A, Rao PR (2008) Synthesizing heavy association rules from different real data sources. *Pattern Recogn Lett* 29(1):59–71
- Agrawal R, Shafer J (1999) Parallel mining of association rules. *IEEE Trans Knowl Data Eng* 8(6):962–969
- Agrawal R, Srikant R (1994) Fast algorithms for mining association rules. In: *Proceedings of international conference on very large data bases*, pp 487–499
- Agrawal R, Imielinski T, Swami A (1993) Mining association rules between sets of items in large databases. In: *Proceedings of ACM SIGMOD conference*, pp 207–216
- Chatrattichat J, Darlington J, Ghanem M, Guo Y, Hüning H, Köhler M, Sutiwaraphun J, To HW, Yang D (1997) Large scale data mining: Challenges, and responses. In: *Proceedings of the third international conference on knowledge discovery and data mining*, pp 143–146
- Cheung D, Ng V, Fu A, Fu Y (1996) Efficient mining of association rules in distributed databases. *IEEE Trans Knowl Data Eng* 8(6):911–922
- Frequent itemset mining dataset repository (2004) <http://fimi.cs.helsinki.fi/data>
- Han J, Pei J, Yiwen Y (2000) Mining frequent patterns without candidate generation. In: *Proceedings of ACM SIGMOD conference on management of data*, pp 1–12
- Last M, Kandel A (2001) Automated detection of outliers in real-world data. In: *Proceedings of the second international conference on intelligent technologies*, pp 292–301
- Liu H, Lin F, He J, Cai Y (2010) New approach for the sequential pattern mining of high-dimensional sequence databases. *Decis Support Syst* 50(1):270–280
- Pyle D (1999) *Data preparation for data mining*. Morgan Kufmann, San Francisco
- Ramkumar T, Srivinasan R (2008) Modified algorithms for synthesizing high-frequency rules from different data sources. *Knowl Inf Syst* 17(3):313–334
- Rozenberg B, Gudes E (2006) Association rules mining in vertically partitioned databases. *Data Knowl Eng* 59(2):378–396
- Savasere A, Omiecinski E, Navathe S (1995) An efficient algorithm for mining association rules in large databases. In: *Proceedings of the 21st international conference on very large data bases*, pp 432–443
- Shang S, Dong X, Li J, Zhao Y (2008) Mining positive and negative association rules in multi-database based on minimum interestingness. In: *Proceedings of the 2008 international conference on intelligent computation technology and automation*, pp 791–794
- Wu X, Zhang S (2003) Synthesizing high-frequency rules from different data sources. *IEEE Trans Knowl Data Eng* 14(2):353–367
- Wu X, Zhu X, He Y, Abdullah N, Arslan AN (2013) PMBC: Pattern mining from biological sequences with wildcard constraints. *Comp Bio Med* 43(5):481–492
- Yi X, Zhang Y (2007) Privacy-preserving distributed association rule mining via semi-trusted mixer. *Data Knowl Eng* 63(2):550–567
- Zeng L, Li L, Duan L, Lü K, Shi Z, Wang M, Wu W, Luo P (2012) Distributed data mining: a survey. *Inf Technol Manage* 13(4):403–409
- Zhang S, Wu X (2011) *Fundamentals of association rules in data mining and knowledge discovery*. Wiley Interdisc Rev: Data Min Knowl Discovery 1(2):97–116
- Zhang S, Wu X, Zhang C (2003) Multi-database mining. *IEEE Comput Intell Bull* 2(1):5–13

- Zhang S, You X, Jin Z, Wu X (2009) Mining globally interesting patterns from multiple databases using kernel estimation. *Expert Syst Appl: An Int J* 36(8):10863–10869
- Zhong N, Yao YYY, Ohshima M (2003) Peculiarity oriented multidatabase mining. *IEEE Trans Knowl Data Eng* 15(4):952–960
- Zhu X, Wu X (2007) Discovering relational patterns across multiple databases. In: *Proceedings of ICDE*, pp 726–735

Data Analysis and Pattern Recognition in Multiple
Databases

Adhikari, A.; Adhikari, J.; Pedrycz, W.

2014, XV, 238 p. 97 illus., Hardcover

ISBN: 978-3-319-03409-6